

**8월 18일 세미나**

# 개요

근본적인 질문 : **딥러닝을 사용하여** 미래의 데이터를 예측할 수 있을까?

**(조건이 맞는다면 가능하다.)**

(조건 : 이전의 데이터가 미래의 데이터와 연관성이 존재해야 한다.)

즉 sin 그래프나 cos 그래프를 교육하고 시간이 지남에 따라서  
해당 그래프가 어떻게 진행될지 예측이 가능하다.

# 개요

시계열을 예측하는.. 딥러닝 모델?..

1. CNN,
2. Simple RNN, 3. LSTM, 4. GRU,
5. Attention, 6. transformer

-> 학습하는 모델에 따른 성능차이를 나타냄.

최근에 나온 모델일수록 항상 더 좋은 성능을 내는 것은 아님을 유념.

처리 프로세서 : 데이터 전 처리 - 모델 구축 - 학습 - 평가

# 체크 요소

시계열 예측을 하기 위해서 반드시 체크해야 할 요소

- Train 데이터가 골고루 분포하여 존재하는가 (데이터의 분포)

개와 고양이의 분포에서

ex) train data로 강아지의 데이터가 20개 고양이 데이터 2개가 들어온다면  
-> 분류하라.

결과 : 강아지 분류의 정확도 >> 고양이의 정확도

- 데이터가 정제된 데이터인가

곡선 길에서 정답을 찾는 것보다.  
직선 길에서 정답을 찾는 것이 효과가 좋다.

- Train data와 Test data가 연관성이 존재하는가

이전 데이터를 사용하여 교육이 잘되더라도  
미래의 데이터와 연관성이 없다면 불가능하다.  
Ex) 로또 번호 예측기

# 체크 요소

시계열 예측을 하기 위해서 반드시 체크해야 할 요소

- 모델의 구성은 올바르게 되었는가.

모델의 하이퍼 파라미터의 구성이 최적화 되어 있는가.

optimizer, loss function은 올바르게 설정이 되어 있는가.

- 학습을 반복 수가 적절한 횟수를 지니고 있는가. (오버 피팅 방지)

테스트 데이터에만 너무 많은 교육을 해버린 경우 실제 예측력은 떨어진다.

- 예측된 데이터들이 한쪽으로 치우쳐서 결과를 내지는 않았는가

상승과 하락의 예측 확률을 확인하며,  
실제로 예측하는 것이 제대로 이루어 졌는지 확인이 필요하다.

# 체크 요소

- Train 데이터가 골고루 분포하여 존재하는가 (데이터의 분포)
  - 데이터가 정제된 데이터 인가
  - Train data와 Test data가 연관성이 존재하는가
  - 모델의 구성은 올바르게 되었는가.
- 학습을 반복 수가 적절한 횟수를 지니고 있는가. (오버 피팅 방지)
- 예측된 데이터들이 한쪽으로 치우쳐서 결과를 내지는 않았는가

-> 추가 데이터 분석 측면에서 해석

# 정리

Q. Train 데이터가 골고루 분포하여 존재하는가.

우상향 혹은 우하향하는 주가의 경우는 데이터가 골고루 존재하지 않음  
(데이터의 80%는 상향 데이터의 20%는 하향으로 교육될 수 있음)

상승과 하락이 빈번하게 발생하는 그래프를 대상으로 해야함 학습이 가능함

Q. 데이터가 정제된 데이터 인가.

A. 입력되는 데이터는 여러가지이지만 주가 데이터는 보통  
Open, Close, Change, Low, High, Volume 등의 데이터가 존재합니다.

이들 중 Change의 경우 Close와 Open을 통해서 결정되기 때문에, 입력의 선별이 필요함

# 정리

Q. 이전의 데이터와 미래의 데이터가 연관성이 존재하는가.

A. 사실상 없을 것이다. 하지만 주가를 예측하는 다수의 논문들은 지난 데이터를 이용하여 예측하는 경우가 많이 존재하였고, 이를 통한 결과도 내는 것을 확인할 수 있다.

따라서 지난 주가 데이터로 미래의 주가 데이터를 예측한다는 것은  
아래의 가정을 세우고, 진행하는 것입니다.

가정 : 미래의 주가 데이터와 이전 주가 데이터는 연관성이 존재한다.  
따라서 이전 데이터를 통하여 미래를 예측 가능할 것이다



# 정리

Q. 모델은 어떤 모델을 선별해야 할 것 인가.

1. CNN,
2. Simple RNN, 3. LSTM, 4. GRU,
5. Attention, 6. transformer

많은 모델의 성능을 비교하는 경우가 많았으며,

그 중 대부분은 CNN과 LSTM을 결합한 모델이  
제일 성능평가가 좋았다고 합니다.

CNN과 LSTM을 섞는 것이 왜 좋은가?

# CNN

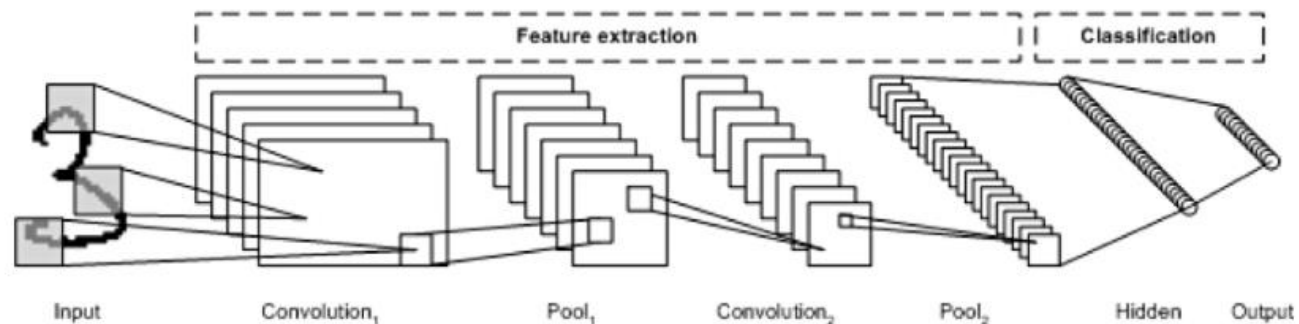
우선.

**CNN모델이란** : 특징을 검출해주는 딥러닝 기법입니다.

보통은 영상처리에서 분류 모델을 처리할 때 사용되며,

ex) 강아지나 고양이를 사진을  
입력으로 줄 시 각 사진의 특징을 뽑아줍니다.

들어오는 입력데이터가 강아지 사진의 특징을 가지고 있다면  
강아지로 판별합니다.



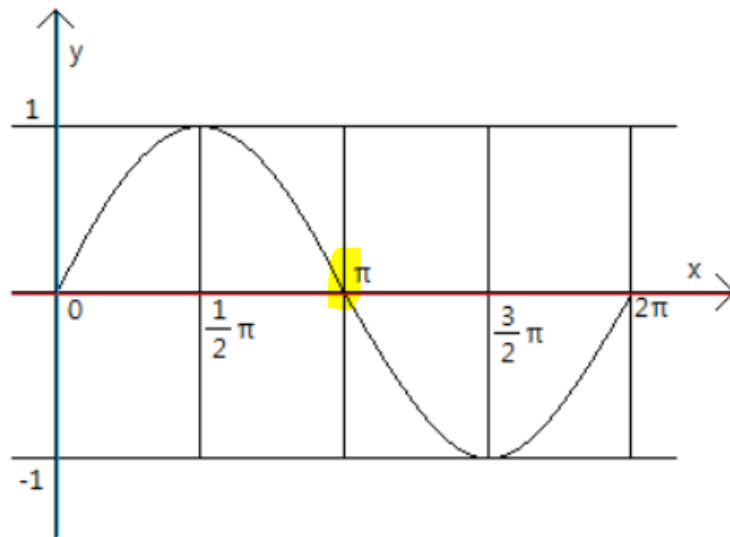
# LSTM

**LSTM 모델 이란** : 시계열 데이터의 흐름을 파악할 때 사용합니다.

들어오는 순서에 관련 있는 데이터를 교육하기 위해서 사용됩니다.

sin, cos 그래프를 그릴 때, 들어오는 순서에 따라서 그려야 할 방향이 달라집니다.

따라서 이러한 순서가 존재하는 데이터를 학습할 땐 LSTM이라고 하는 모델을 사용합니다.



# 정리

따라서

CNN + LSTM 이란?

“시계열 데이터의 특징을 뽑아서 순서에 맞게 교육을 시키겠다.”의 의미이며,  
이렇게 한다면 기존 CNN 혹은 LSTM보다 성능이 좋은 경우가 많았기 때문에  
CNN과 LSTM의 결합모델을 사용하기 시작했습니다.


Q. 현재까지 나와있는 주가를 분석하는 논문들은 어떤 형태인가요.  
해당 주제로 의미있는 결과를 내기 위해서는 어떻게 해야 할까요.


A. 보통의 해당 주제 논문들은 2가지의 분류로 나뉘게 됩니다.

# 정리

추가데이터를 가지고 무언가를 증명하고자 한다면??

1. 동일한 데이터에 다양한 모델을 결합하여,  
해당 모델이 가장 좋은 성능을 낼 수 있다. 라는 실험과 증명이 필요합니다.

 **applied  
sciences**



Article

## Predicting the Trend of Stock Market Index Using the Hybrid Neural Network Based on Multiple Time Scale Feature Learning

Yaping Hao \* and Qiang Gao

School of Electronic and Information Engineering, Beihang University, Beijing 100191, China;  
gaoqiang@buaa.edu.cn  
\* Correspondence: haoyaping@buaa.edu.cn; Tel.: +86-138-1096-8583

Received: 25 April 2020; Accepted: 4 June 2020; Published: 7 June 2020


 check for  
updates

Table 2. Comparisons with existing models in accuracy.

Forecast Horizon	Model	Accuracy (%)
One week	Simplistic Model	54.82
	SVM	61.98
	LSTM	65.05
	CNN	59.34
	Multiple Pipeline Model	63.30
	NFNN	65.93
	The proposed model	<b>66.59</b>
One month	Simplistic Model	56.92
	SVM	70.91
	LSTM	71.59
	CNN	67.95
	Multiple Pipeline Model	72.05
	MFNN	72.27
	The proposed model	<b>74.55</b>

# 정리

따라서 추가데이터를 가지고 무언가를 증명하고자 한다면??

## 2. 추가 데이터의 경우

“이전의 데이터와 미래의 데이터가 연관성이 존재하는가”에 관한 확신이 없으므로,

모델에 해당 추가 데이터(섹션)와 관련된 다른 데이터들을 가지고 와서,  
데이터가 결합한 경우 모델의 정확도가 올라가는지 검사하는 경우도 존재합니다.

Ex) 여행 관련 섹션의 추가의 상승하락을 예측할 때, 환율데이터를 넣어서  
loss 값이 낮아지는지 검사

또한 하루 뒤의 데이터와 연관성이 있는 데이터  
일주일 뒤의 데이터와 연관성이 있는 데이터  
한달 뒤의 데이터와 연관성이 있는 데이터

한가지의 추가 데이터라도 예측기간에 따른 필요 데이터가 다를 수 있기 때문에,  
방향성이 방대하다.

# 정리

따라서 주가데이터를 가지고 무언가를 증명하고자 한다면??

Journal of Digital Convergence  
Vol. 18, No. 7, pp. 223-228, 2020

ISSN 2713-6434 / eISSN 2713-6442  
<https://doi.org/10.14400/JDC.2020.18.7.223>

A study on stock price prediction system based on text mining  
method using LSTM and stock market news

Sunghyuck Hong  
Professor, Baekseok University, Division of ICT

LSTM과 증시 뉴스를 활용한 텍스트 마이닝 기법 기반  
주가 예측시스템 연구

홍성혁  
백석대학교 ICT학부 교수

<http://www.jsebs.org>  
ISSN: 2288-3908

The Journal of Society for e-Business Studies  
Vol.25, No.4, November 2020, pp.61-75  
<https://doi.org/10.7838/jsebs.2020.25.4.061>

SNS감성 분석을 이용한 주가 방향성 예측:  
네이버 주식토론방 데이터를 이용하여

Stock Price Prediction Using Sentiment Analysis:  
from "Stock Discussion Room" in Naver

김명진(Myongjin Kim)\*, 류지혜(Jihye Ryu)\*\*,  
차동호(Dongho Cha)\*\*\*, 심민규(Min Kyu Sim)\*\*\*\*



STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING  
*Stat., Optim. Inf. Comput.*, Vol. 9, June 2021, pp 268-287.  
Published online in International Academic Press ([www.IAPress.org](http://www.IAPress.org))

Stock Price Predictions with LSTM Neural Networks and Twitter Sentiment

Marah-Lisane Thormann<sup>1</sup>, Jan Farchmin<sup>1</sup>, Christoph Weisser<sup>1,3</sup>, René-Marcel Kruse<sup>2</sup>, Benjamin  
Säfken<sup>2,3</sup>, Alexander Silbersdorf<sup>1,3</sup>

# 정리

Q. 아직 발표전이지만, 서울대학교 강유 교수님의 인터뷰를 따온다면 아래와 같습니다.

강유 교수팀의 주가 예측 AI가 주목받는 이유는 특별한 AI 기술이 아닌 메인 아이디어에 있다. 정확한 주가를 맞추는 것이 아닌 주가 상승 혹은 하락 여부만 예측하는 것을 목표로 정한 점을 먼저 꼽을 수 있다. 특정 기업 주가 예측을 위해 다른 여러 종목 기업 데이터를 함께 활용한 것도 중요한 아이디어였다는 설명이다.

A. 주가 데이터(한정된 정보)를 가지고 최대한 예측을 잘하는 좋은 모델도 중요하지만, 데이터의 상관관계가 존재하는 다른 데이터들을 가지고 오는 것이 loss 값을 낮출 수 있는 더 좋은 방법론입니다.



# 개념

데이터 관의 상관관계를 파악하기 위해서는 경제 상황을 볼 수 있어야 했습니다.  
환율 데이터와 코스피와의 상관관계 및 기업에서는 어떤 역할을 가지고 있는지.

## 한가지의 예시

수출이 많이 이루어지는 기업에서는 환율에 대한 리스크를 항상 가지고 있습니다.  
환율이 떨어진다면, 기업입장에서는 떨어졌을 때 파는 것이 직접적인 손해로 다가오기 때문입니다.  
이를 환리스크라고 표현합니다.

대부분의 기업에서는 따라서 환리스크를 관리하며 회사의 경영을 이끌어 가고 있습니다.  
저희 나라에서 유명한 기업 중 환리스크에 대한 관리가 다른 곳은 한곳이 존재합니다.

# 개념

삼성전자의 경우 다음과 같습니다.

내추럴 헤지라고 하는 방법을 사용하여, 환율에 따른 리스크를 최소한으로 줄이지만,  
반도체 부분은 영향이 존재한다고 하였습니다.

-> 환율 데이터를 포함하여 교육 시켜본다면 어떤 결과를 낼 수 있을까

삼성전자의 분기당 영업이익이 다시 10조원 밑으로 줄었다.  
4분기는 세트부분의 계절적 요인과 원달러 환율급락이 부정적 영향을 끼쳤다.  
인위적인 환헤지를 하지 않는 삼성전자 특성상 환차손 위험이 그대로 반영된 탓이다.

다만 벌어들이는 외화로 수입대금을 결제하는 일명 '내추럴헤지(Natural Hedge)'  
방식이 보편화 돼 있어 단기 환율변동의 영향은 크지 않을 것으로 전망된다.

삼성전자 관계자는 "4분기에 마케팅비가 몰리는 세트부분의 계절적 요인과 환율변동 영향이 있다"며 "특히 반도체, DS부분은 거의 달러를 쓰기 때문에 환율급락의 영향이 있었다"고 설명했다.

# TEST

## -> 환율 데이터를 포함하여 교육 시켜본다면 어떤 결과를 낼 수 있을까 Month data 결과 비교

```
Epoch 00004: val_loss improved from 0.68709 to 0.68623, saving model to ckeckpointer.ckpt
Epoch 5/100
6/6 [=====] - 1s 218ms/step - loss: 0.6926 - accuracy: 0.5039 - val_loss: 0.6879 - val_accuracy: 0.7441

Epoch 00005: val_loss did not improve from 0.68623
Epoch 6/100
6/6 [=====] - 1s 227ms/step - loss: 0.6920 - accuracy: 0.5039 - val_loss: 0.6896 - val_accuracy: 0.7441

Epoch 00006: val_loss did not improve from 0.68623
Epoch 7/100
6/6 [=====] - 1s 248ms/step - loss: 0.6910 - accuracy: 0.5039 - val_loss: 0.6909 - val_accuracy: 0.8081

Epoch 00007: val_loss did not improve from 0.68623
Epoch 8/100
6/6 [=====] - 1s 228ms/step - loss: 0.6895 - accuracy: 0.5039 - val_loss: 0.6921 - val_accuracy: 0.6633

Epoch 00008: val_loss did not improve from 0.68623
Epoch 9/100
6/6 [=====] - 1s 239ms/step - loss: 0.6870 - accuracy: 0.5039 - val_loss: 0.6957 - val_accuracy: 0.5084

Epoch 00009: val_loss did not improve from 0.68623
Epoch 10/100
6/6 [=====] - 1s 229ms/step - loss: 0.6836 - accuracy: 0.5039 - val_loss: 0.7041 - val_accuracy: 0.4276

Epoch 00010: val_loss did not improve from 0.68623
Epoch 11/100
6/6 [=====] - 1s 228ms/step - loss: 0.6807 - accuracy: 0.5039 - val_loss: 0.7183 - val_accuracy: 0.2593

Epoch 00011: val_loss did not improve from 0.68623
Epoch 12/100
6/6 [=====] - 1s 228ms/step - loss: 0.6772 - accuracy: 0.5039 - val_loss: 0.7490 - val_accuracy: 0.2559

Epoch 00012: val_loss did not improve from 0.68623
Epoch 13/100
6/6 [=====] - 1s 227ms/step - loss: 0.6733 - accuracy: 0.6051 - val_loss: 0.8170 - val_accuracy: 0.2559

Epoch 00013: val_loss did not improve from 0.68623
Epoch 14/100
6/6 [=====] - 1s 220ms/step - loss: 0.6695 - accuracy: 0.6449 - val_loss: 0.9385 - val_accuracy: 0.2559

Epoch 00014: val_loss did not improve from 0.68623
Epoch 15/100
6/6 [=====] - 1s 220ms/step - loss: 0.6645 - accuracy: 0.6566 - val_loss: 1.1378 - val_accuracy: 0.2559
```

```
Epoch 00088: val_loss did not improve from 0.60306
Epoch 89/100
6/6 [=====] - 2s 348ms/step - loss: 0.6849 - accuracy: 0.6031 - val_loss: 0.6552 - val_accuracy: 0.7003

Epoch 00089: val_loss did not improve from 0.60306
Epoch 90/100
6/6 [=====] - 2s 347ms/step - loss: 0.6848 - accuracy: 0.6012 - val_loss: 0.6544 - val_accuracy: 0.6801

Epoch 00090: val_loss did not improve from 0.60306
Epoch 91/100
6/6 [=====] - 2s 340ms/step - loss: 0.6846 - accuracy: 0.6021 - val_loss: 0.6537 - val_accuracy: 0.6667

Epoch 00091: val_loss did not improve from 0.60306
Epoch 92/100
6/6 [=====] - 2s 336ms/step - loss: 0.6843 - accuracy: 0.6031 - val_loss: 0.6535 - val_accuracy: 0.6566

Epoch 00092: val_loss did not improve from 0.60306
Epoch 93/100
6/6 [=====] - 2s 340ms/step - loss: 0.6840 - accuracy: 0.6021 - val_loss: 0.6535 - val_accuracy: 0.6330

Epoch 00093: val_loss did not improve from 0.60306
Epoch 94/100
6/6 [=====] - 2s 331ms/step - loss: 0.6838 - accuracy: 0.6002 - val_loss: 0.6535 - val_accuracy: 0.6263

Epoch 00094: val_loss did not improve from 0.60306
Epoch 95/100
6/6 [=====] - 2s 313ms/step - loss: 0.6837 - accuracy: 0.6002 - val_loss: 0.6547 - val_accuracy: 0.6061

Epoch 00095: val_loss did not improve from 0.60306
Epoch 96/100
6/6 [=====] - 2s 317ms/step - loss: 0.6835 - accuracy: 0.5992 - val_loss: 0.6583 - val_accuracy: 0.5892

Epoch 00096: val_loss did not improve from 0.60306
Epoch 97/100
6/6 [=====] - 2s 317ms/step - loss: 0.6835 - accuracy: 0.5992 - val_loss: 0.6583 - val_accuracy: 0.5892
```

# 정리

-> 환율 데이터를 포함하여 교육 시켜본다면 어떤 결과를 낼 수 있을까  
Month data 결과 비교

```
----- test evaluate -----
4/4 [=====] - 1s 78ms/step - loss: 0.6917 - accuracy: 0.5727
----- test label 데이터 뽑기 -----
(653, 1)
   예측  정답
0  0.505624  1
1  0.505750  1
2  0.505688  1
3  0.505718  1
4  0.505718  1
..      ..
648 0.503612  1
649 0.503613  1
650 0.503509  1
651 0.503560  1
652 0.503565  1

[653 rows x 2 columns]
-----각 상황의 카운터-----
Up_count : 653
Up_label_count = 374
Down_count = 0
Down_label_count = 279
-----각 상황의 예측 정확도-----
Up evaluate rate = 0.5727411944869831
```

```
----- test evaluate -----
4/4 [=====] - 1s 107ms/step - loss: 0.6531 - accuracy: 0.5988
----- test label 데이터 뽑기 -----
   예측  정답
0  0.495611  1
1  0.495680  1
2  0.495677  1
3  0.495656  1
4  0.495643  1
..      ..
648 0.495439  1
649 0.495337  1
650 0.495305  1
651 0.495250  1
652 0.495253  1

[653 rows x 2 columns]
-----각 상황의 카운터-----
Up_count : 590
Up_label_count = 374
Down_count = 63
Down_label_count = 279
-----각 상황의 예측 정확도-----
Up evaluate rate = 0.5949152542372881
Down evaluate rate = 0.6349206349206349
```

Week data와 day data는 큰 차이를 보이지 못해서(두개 다 학습이 어려움), 생략

# 정리

실제 테스트의 대한 이야기

한 섹터와 연관성이 있는 데이터들..

실험을 해봤던 것은

여행 섹터의 주가 데이터 교육 vs 여행 섹터 주가 데이터 + 환율데이터

전체가 영향력을 가지고 있는 것은 아니지만,

2개의 주가는 환율 데이터를 대입하였을 때 더 좋은 loss 값을 나타내는 것을 확인 할 수 있었습니다.

전체 여행 섹터 중에서 -> 7가지 중 거의 2가지 정도만 결과가 달라져서 큰 결과 값이 아니라고  
생각되어서 방향을 조금 틀게 되었습니다.

# 정리

현재 상황 : 주가 데이터를 예측하는 논문들은 많이 나오고 있으며,  
텍스트 마이닝 기법을 사용하여 예측하는 것, 뉴스를 통해 예측하는 것 혹은  
본인이 조합한 모델을 통해 좋은 성능을 얻는 것 등이 있습니다.

그래서 저는 암호화폐 쪽으로 눈을 돌려서 보려고 합니다.

암호화폐의 지난 데이터가 미래에 영향을 미친다는 확신은 없지만,

다른 암호화폐들은 비트코인이라는 대장주의 흐름에 따라가기 때문입니다.

따라서 비트코인의 주가 데이터를 다른 암호화폐에 같이 교육을 시켜 결과를 테스트 할 예정입니다.

# 암호화폐 그래프

## 비트코인 BTC / KRW



## 리플 XRP / KRW



## 이더리움 ETH / KRW



## 라이트코인 LTC / KRW



# 교육 환경

Model : CNN + LSTM

모델 : 암호화폐의 데이터 vs 암호화폐의 데이터 + 비트 코인 데이터

모델 1 데이터의 입력 값

1. data Close, 2. data Volume, 3. data Change

모델 2 데이터의 입력 값

1. data Close, 2. data Volume, 3. data Change  
4. Bitcoin Close 5. Bitcoin Volume, 6.Bitcoin Change

데이터 1 : 이더리움

데이터 2 : 리플

(데이터 3 : 라이트 코인

데이터 4 : 스텔라 달러)

각각 week, month 데이터로 비교



# 교육 환경

이더리움 : 2017 ~ 2020  
(2016 4월에 상장됨)

행 이름 설정 완료						
	Change	Close	Volume	Change_USD	Close_USD	Volume_USD
Date						
2017-06-05	-0,0063	293750	106390,0	0,0415	3188000	18120,0
2017-06-06	0,0689	314000	269510,0	0,0634	3390000	32900,0
2017-06-07	-0,0261	305800	154300,0	-0,0425	3246000	18750,0
2017-06-08	-0,0041	304550	112370,0	0,0083	3273000	22070,0
2017-06-09	0,0573	322000	203290,0	-0,0024	3265000	16940,0
...	...	...	...	...	...	...
2019-12-28	0,0110	147600	8510,0	0,0039	8425000	400,0
2019-12-29	0,0447	154200	24300,0	0,0077	8490000	900,0
2019-12-30	-0,0175	151500	8310,0	-0,0176	8341000	480,0
2019-12-31	-0,0152	149200	13700,0	-0,0040	8308000	550,0
2020-01-01	0,0101	150700	4920,0	-0,0004	8305000	320,0

Train : 70%  
validation : 10%  
test : 20%

리플 : 2018 ~ 2020  
(2017년 5월에 상장됨)

행 이름 설정 완료						
	Change	Close	Volume	Change_USD	Close_USD	Volume_USD
Date						
2018-01-01	0,0322	2849,0	124660000,0	-0,0288	18655000	10240,0
2018-01-02	0,0579	3014,0	135750000,0	0,0634	19838000	17460,0
2018-01-03	0,2326	3715,0	147030000,0	0,0293	20419000	15670,0
2018-01-04	0,1386	4230,0	107960000,0	0,0567	21576000	14730,0
2018-01-05	-0,0279	4112,0	112730000,0	0,1599	25026000	25290,0
...	...	...	...	...	...	...
2019-12-28	0,0127	222,5	4380000,0	0,0039	8425000	400,0
2019-12-29	0,0144	225,7	7260000,0	0,0077	8490000	900,0
2019-12-30	-0,0151	222,3	3320000,0	-0,0176	8341000	480,0
2019-12-31	0,0009	222,5	5420000,0	-0,0040	8308000	550,0
2020-01-01	0,0004	222,6	3490000,0	-0,0004	8305000	320,0

Data : 지난 50일 데이터

Mini Batch size = 100

Callback : val\_loss

Loss : binary cross entropy

Optimizer : Adam

라이트코인 : 2018 ~ 2020

# 이더리움 week data

```
----- test evaluate -----
xoch 87/200
'3 [=====] - 2s 707ms/step - loss: 0.7393 - accuracy: 0.4846 - val_loss: 0.6752 - val_accuracy: 0.6095

xoch 00087: val_loss improved from 0.69201 to 0.67524, saving model to ckeckpointer1.ckpt
xoch 88/200
'3 [=====] - 2s 673ms/step - loss: 1.0885 - accuracy: 0.3480 - val_loss: 0.7342 - val_accuracy: 0.6095

xoch 00088: val_loss did not improve from 0.67524
xoch 89/200
'3 [=====] - 2s 677ms/step - loss: 0.9018 - accuracy: 0.5286 - val_loss: 0.6869 - val_accuracy: 0.6095

xoch 00089: val_loss did not improve from 0.67524
xoch 90/200
```

-----각 상황의 카운터-----

```
Up_count : 0
Up_label_count = 198
Down_count = 381
Down_label_count = 183
```

-----각 상황의 예측 정확도-----

```
Down evaluate rate = 0.48031496062992124
```

```
----- test evaluate -----
Epoch 30/200
3/3 [=====] - 2s 1s/step - loss: 0.6296 - accuracy: 0.6828 - val_loss: 0.6935 - val_accuracy: 0.5858

Epoch 00030: val_loss did not improve from 0.69238
Epoch 31/200
3/3 [=====] - 2s 1s/step - loss: 0.6265 - accuracy: 0.6872 - val_loss: 0.6748 - val_accuracy: 0.6095

Epoch 00031: val_loss improved from 0.69238 to 0.67480, saving model to ckeckpointer1.ckpt
Epoch 32/200
3/3 [=====] - 2s 1s/step - loss: 0.7407 - accuracy: 0.5419 - val_loss: 0.7272 - val_accuracy: 0.5503

Epoch 00032: val_loss did not improve from 0.67480
Epoch 33/200
```

```
----- test evaluate -----
4/4 [=====] - 2s 115ms/step - loss: 0.7151 - accuracy: 0.5748
----- test label 데이터 뽑기 -----
```

```
----- test evaluate -----
-----각 상황의 카운터-----
```

```
Up_count : 184
Up_label_count = 198
Down_count = 197
Down_label_count = 183
```

-----각 상황의 예측 정확도-----

```
Up evaluate rate = 0.5978260869565217
Down evaluate rate = 0.5532994923857868
```

# 이더리움 month data

```
3/3 [=====] - 2s 668ms/step - loss: 0.7700 - accuracy: 0.5849 - val_loss: 0.4451 - val_accuracy: 0.8377
```

```
Epoch 00048: val_loss improved from 0.62823 to 0.44513, saving model to ckeckpointer1.ckpt
```

```
Epoch 49/200
```

```
3/3 [=====] - 2s 662ms/step - loss: 0.9697 - accuracy: 0.3962 - val_loss: 0.7657 - val_accuracy: 0.1623
```

```
Epoch 00049: val_loss did not improve from 0.44513
```

```
Epoch 50/200
```

```
3/3 [=====] - 2s 677ms/step - loss: 0.8677 - accuracy: 0.6368 - val_loss: 0.7429 - val_accuracy: 0.1623
```

```
Epoch 00050: val_loss did not improve from 0.44513
```

```
Epoch 51/200
```

```
-----각 상황의 카운터-----
```

```
Up_count : 0
```

```
Up_label_count = 177
```

```
Down_count = 366
```

```
Down_label_count = 189
```

```
-----각 상황의 예측 정확도-----
```

```
Down evaluate rate = 0.5163934426229508
```

```
Epoch 00016: val_loss did not improve from 0.68720
```

```
Epoch 17/200
```

```
3/3 [=====] - 3s 1s/step - loss: 0.6275 - accuracy: 0.7264 - val_loss: 1.2316 - val_accuracy: 0.2792
```

```
Epoch 00017: val_loss did not improve from 0.68720
```

```
Epoch 18/200
```

```
3/3 [=====] - 3s 1s/step - loss: 0.5701 - accuracy: 0.7547 - val_loss: 0.4556 - val_accuracy: 0.8377
```

```
Epoch 00018: val_loss improved from 0.68720 to 0.45564, saving model to ckeckpointer1.ckpt
```

```
Epoch 19/200
```

```
3/3 [=====] - 3s 1s/step - loss: 0.5159 - accuracy: 0.7925 - val_loss: 0.4516 - val_accuracy: 0.8377
```

```
Epoch 00019: val_loss improved from 0.45564 to 0.45162, saving model to ckeckpointer1.ckpt
```

```
Epoch 20/200
```

```
3/3 [=====] - 3s 1s/step - loss: 0.5370 - accuracy: 0.7736 - val_loss: 0.5963 - val_accuracy: 0.7403
```

```
Epoch 00020: val_loss did not improve from 0.45162
```

```
Epoch 21/200
```

```
3/3 [=====] - 3s 1s/step - loss: 0.5502 - accuracy: 0.7689 - val_loss: 0.6643 - val_accuracy: 0.7013
```

```
----- test evaluate -----
```

```
4/4 [=====] - 3s 110ms/step - loss: 0.5945 - accuracy: 0.7322
```

```
-----각 상황의 카운터-----
```

```
Up_count : 205
```

```
Up_label_count = 177
```

```
Down_count = 161
```

```
Down_label_count = 189
```

```
-----각 상황의 예측 정확도-----
```

```
Up evaluate rate = 0.6926829268292682
```

```
Down evaluate rate = 0.782608695652174
```

# 리플 week data

```
Epoch 00170: val_loss improved from 0.63146 to 0.63131, saving model to ckckpointer1.ckpt
Epoch 171/200
2/2 [=====] - 2s 2s/step - loss: 0.6257 - accuracy: 0.6707 - val_loss: 0.6310 - val_accuracy: 0.6723

Epoch 00171: val_loss improved from 0.63131 to 0.63105, saving model to ckckpointer1.ckpt
Epoch 172/200
2/2 [=====] - 2s 1s/step - loss: 0.6253 - accuracy: 0.6707 - val_loss: 0.6305 - val_accuracy: 0.6723

Epoch 00172: val_loss improved from 0.63105 to 0.63053, saving model to ckckpointer1.ckpt
Epoch 173/200
2/2 [=====] - 2s 1s/step - loss: 0.6860 - accuracy: 0.6707 - val_loss: 0.6393 - val_accuracy: 0.6723

Epoch 00173: val_loss did not improve from 0.63053
Epoch 174/200
2/2 [=====] - 2s 1s/step - loss: 0.6301 - accuracy: 0.6707 - val_loss: 0.6468 - val_accuracy: 0.6723

Epoch 00174: val_loss did not improve from 0.63053
Epoch 175/200
```

```
Epoch 00051: val_loss did not improve from 0.63154
Epoch 52/100
2/2 [=====] - 2s 2s/step - loss: 0.6115 - accuracy: 0.6707 - val_loss: 0.6481 - val_accuracy: 0.6723

Epoch 00052: val_loss did not improve from 0.63154
Epoch 53/100
2/2 [=====] - 2s 2s/step - loss: 0.6100 - accuracy: 0.6707 - val_loss: 0.6497 - val_accuracy: 0.6723

Epoch 00053: val_loss did not improve from 0.63154
Epoch 54/100
2/2 [=====] - 2s 2s/step - loss: 0.6116 - accuracy: 0.6707 - val_loss: 0.6307 - val_accuracy: 0.6723

Epoch 00054: val_loss improved from 0.63154 to 0.63066, saving model to ckckpointer1.ckpt
```

```
----- test evaluate -----
3/3 [=====] - 1s 90ms/step - loss: 0.6841 - accuracy: 0.5845
```

```
-----각 상황의 카운터-----
Up_count : 0
Up_label_count = 118
Down_count = 284
Down_label_count = 166
-----각 상황의 예측 정확도-----
Down evaluate rate = 0.5845070422535211
```

# 리플 month data

```
Epoch 00146: val_loss improved from 0.65660 to 0.65458, saving model to ckeckpointer1.ckpt
Epoch 147/170
2/2 [=====] - 2s 1s/step - loss: 0.5486 - accuracy: 0.7852 - val_loss: 0.6549 - val_accuracy: 0.6346
```

```
Epoch 00147: val_loss did not improve from 0.65458
Epoch 148/170
2/2 [=====] - 2s 1s/step - loss: 0.5551 - accuracy: 0.7852 - val_loss: 0.6545 - val_accuracy: 0.6346
```

```
Epoch 00148: val_loss improved from 0.65458 to 0.65450, saving model to ckeckpointer1.ckpt
Epoch 149/170
2/2 [=====] - 2s 1s/step - loss: 0.5569 - accuracy: 0.7852 - val_loss: 0.6542 - val_accuracy: 0.6346
```

```
Epoch 00149: val_loss improved from 0.65450 to 0.65418, saving model to ckeckpointer1.ckpt
Epoch 150/170
2/2 [=====] - 2s 1s/step - loss: 0.5523 - accuracy: 0.7852 - val_loss: 0.6547 - val_accuracy: 0.6346
```

```
Epoch 00150: val_loss did not improve from 0.65418
Epoch 151/170
2/2 [=====] - 2s 1s/step - loss: 0.5438 - accuracy: 0.7852 - val_loss: 0.6568 - val_accuracy: 0.6346
```

```
Epoch 00151: val_loss did not improve from 0.65418
Epoch 152/170
```

```
----- test evaluate -----
3/3 [=====] - 1s 82ms/step - loss: 0.6517 - accuracy: 0.6431
```

```
-----각 상황의 카운터-----
```

```
Up_count : 0
Up_label_count = 96
Down_count = 269
Down_label_count = 173
```

```
-----각 상황의 예측 정확도-----
```

```
Down evaluate rate = 0.6431226765799256
```

```
Epoch 00157: val_loss did not improve from 0.63557
Epoch 158/170
2/2 [=====] - 2s 2s/step - loss: 0.3797 - accuracy: 0.7852 - val_loss: 0.6347 - val_accuracy: 0.5481
```

```
Epoch 00158: val_loss improved from 0.63557 to 0.63468, saving model to ckeckpointer1.ckpt
Epoch 159/170
2/2 [=====] - 2s 2s/step - loss: 0.3816 - accuracy: 0.7852 - val_loss: 0.6372 - val_accuracy: 0.5481
```

```
----- test evaluate -----
3/3 [=====] - 2s 89ms/step - loss: 0.6402 - accuracy: 0.7435
```

```
-----각 상황의 카운터-----
```

```
Up_count : 71
Up_label_count = 96
Down_count = 198
Down_label_count = 173
```

```
-----각 상황의 예측 정확도-----
```

```
Up evaluate rate = 0.6901408450704225
Down evaluate rate = 0.7626262626262627
```

# 라이트 코인 week data

```
Epoch 00121: val_loss did not improve from 0.65056
Epoch 122/200
2/2 [=====] - 2s 1s/step - loss: 0.6455 - accuracy: 0.6341 - val_loss: 0.6506 - val_accuracy: 0.6387

Epoch 00122: val_loss did not improve from 0.65056
Epoch 123/200
2/2 [=====] - 2s 1s/step - loss: 0.6445 - accuracy: 0.6341 - val_loss: 0.6501 - val_accuracy: 0.6387

Epoch 00123: val_loss improved from 0.65056 to 0.65015, saving model to ckpointer1.ckpt
Epoch 124/200
2/2 [=====] - 2s 1s/step - loss: 0.6426 - accuracy: 0.6341 - val_loss: 0.6504 - val_accuracy: 0.6387

Epoch 00124: val_loss did not improve from 0.65015
Epoch 125/200
2/2 [=====] - 2s 1s/step - loss: 0.6513 - accuracy: 0.6341 - val_loss: 0.6591 - val_accuracy: 0.6387

Epoch 00125: val_loss did not improve from 0.65015
Epoch 126/200
2/2 [=====] - 2s 1s/step - loss: 0.6516 - accuracy: 0.6341 - val_loss: 0.6636 - val_accuracy: 0.6387
```

```
----- test evaluate -----
3/3 [=====] - 1s 105ms/step - loss: 0.6830 - accuracy: 0.5599
```

-----각 상황의 카운터-----

```
Up_count : 0
Up_label_count = 125
Down_count = 284
Down_label_count = 159
```

-----각 상황의 예측 정확도-----

```
Down evaluate rate = 0.5598591549295775
```

```
Epoch 00041: val_loss did not improve from 0.62539
Epoch 42/100
2/2 [=====] - 2s 2s/step - loss: 0.6334 - accuracy: 0.6341 - val_loss: 0.6313 - val_accuracy: 0.6387

Epoch 00042: val_loss did not improve from 0.62539
Epoch 43/100
2/2 [=====] - 2s 2s/step - loss: 0.6314 - accuracy: 0.6341 - val_loss: 0.6236 - val_accuracy: 0.6387

Epoch 00043: val_loss improved from 0.62539 to 0.62363, saving model to ckpointer1.ckpt
Epoch 44/100
2/2 [=====] - 2s 2s/step - loss: 0.6346 - accuracy: 0.6341 - val_loss: 0.6295 - val_accuracy: 0.6387

Epoch 00044: val_loss did not improve from 0.62363
Epoch 45/100
2/2 [=====] - 2s 2s/step - loss: 0.6316 - accuracy: 0.6341 - val_loss: 0.6332 - val_accuracy: 0.6807

Epoch 00045: val_loss did not improve from 0.62363
Epoch 46/100
2/2 [=====] - 2s 2s/step - loss: 0.6309 - accuracy: 0.6341 - val_loss: 0.6325 - val_accuracy: 0.7059
```

```
----- test evaluate -----
3/3 [=====] - 2s 103ms/step - loss: 0.7272 - accuracy: 0.6092
```

-----각 상황의 카운터-----

```
Up_count : 98
Up_label_count = 125
Down_count = 186
Down_label_count = 159
```

-----각 상황의 예측 정확도-----

```
Up evaluate rate = 0.5714285714285714
Down evaluate rate = 0.6290322580645161
```

# 라이트코인 month data

```
Epoch 00197: val_loss did not improve from 0.57412
Epoch 198/200
2/2 [=====] - 2s 1s/step - loss: 0.5336 - accuracy: 0.7383 - val_loss: 0.5728 - val_accuracy: 0.7115
```

```
Epoch 00198: val_loss improved from 0.57412 to 0.57282, saving model to ckckpointer1.ckpt
Epoch 199/200
2/2 [=====] - 1s 1s/step - loss: 0.5313 - accuracy: 0.7383 - val_loss: 0.5708 - val_accuracy: 0.7115
```

```
Epoch 00199: val_loss improved from 0.57282 to 0.57084, saving model to ckckpointer1.ckpt
Epoch 200/200
2/2 [=====] - 2s 1s/step - loss: 0.5283 - accuracy: 0.7383 - val_loss: 0.5679 - val_accuracy: 0.7115
```

```
Epoch 00200: val_loss improved from 0.57084 to 0.56790, saving model to ckckpointer1.ckpt
```

----- test evaluate -----

```
3/3 [=====] - 1s 116ms/step - loss: 0.6620 - accuracy: 0.6394
```

-----각 상황의 카운터-----

```
Up_count : 0
Up_label_count = 97
Down_count = 269
Down_label_count = 172
```

-----각 상황의 예측 정확도-----

```
Down evaluate rate = 0.6394052044609665
```

```
Epoch 00075: val_loss did not improve from 0.46417
Epoch 76/200
2/2 [=====] - 2s 2s/step - loss: 0.4573 - accuracy: 0.7383 - val_loss: 0.5155 - val_accuracy: 0.6923
```

```
Epoch 00076: val_loss did not improve from 0.46417
Epoch 77/200
2/2 [=====] - 2s 2s/step - loss: 0.4679 - accuracy: 0.7383 - val_loss: 0.4482 - val_accuracy: 0.7500
```

```
Epoch 00077: val_loss improved from 0.46417 to 0.44816, saving model to ckckpointer1.ckpt
Epoch 78/200
2/2 [=====] - 2s 2s/step - loss: 0.7760 - accuracy: 0.7383 - val_loss: 0.5603 - val_accuracy: 0.7115
```

```
Epoch 00078: val_loss did not improve from 0.44816
Epoch 79/200
2/2 [=====] - 2s 2s/step - loss: 0.5124 - accuracy: 0.7383 - val_loss: 0.5824 - val_accuracy: 0.7115
```

```
Epoch 00079: val_loss did not improve from 0.44816
Epoch 80/200
2/2 [=====] - 2s 2s/step - loss: 0.5377 - accuracy: 0.7383 - val_loss: 0.5897 - val_accuracy: 0.7115
```

----- test evaluate -----

```
3/3 [=====] - 2s 117ms/step - loss: 0.8768 - accuracy: 0.7100
```

-----각 상황의 카운터-----

```
Up_count : 77
Up_label_count = 97
Down_count = 192
Down_label_count = 172
```

-----각 상황의 예측 정확도-----

```
Up evaluate rate = 0.6233766233766234
Down evaluate rate = 0.7447916666666666
```

# 결론

암호화폐 ( 리플과 이더리움, 라이트 코인 )를 테스트 하였을 때  
Week data or month data에서 Loss값이 줄어드는 것을 확인할 수 있습니다.

암호화폐 시장에서

비트코인을 이용하여, 이더리움, 리플, 라이트 코인의

week, month data의 상승 하락을 예측하는 정확도는

단일 비교를 하였을 때보다 높은 것을 확인할 수 있습니다.