금융 챗봇
“Hi 고슴도치”

CONTENTS

01 프로젝트 동기

02 프로젝트 목표

03 데이터 소스 및 적용기술

04 프로젝트 일정 및 역할분담

금융 챗봇 “Hi 고슴도치”

주식투자를 하기 위해 많은 정보가 필요하지만 원하는 정보를 바로바로 찾아보기는 쉽지 않다. 챗봇을 통해 필요한 정보를 간단한 대화로 바로 볼 수 있도록 종목, 테마 정보와 AI모델을 통한 추세와 예측 정보를 추천시스템과 함께 제공하려고 한다.

01.
프로젝트 동기

뉴스핌

대신증권, 금융챗봇 '벤자민' 받은 질문 100만건 돌파

[서울=뉴스핌] 장봄이 기자= 2017년 대신증권이 도입한 금융챗봇 '벤자민'이 받은 고객 질문이 100만건을 돌파했다. 대신증권은 답변 정확도와 고객...

2020. 2. 26.

천지일보

[비즈라이프 은행특집] 비대면 시대, 은행권의 선택은 'AI' 서비스

국민은행의 '챗봇 비비'는 AI 기술을 활용해 쉽고 간단하게 채팅으로 상담할 수 있도록 만들어진 인공지능 금융 챗봇 서비스다.

2021. 4. 26.

한국경제

국민銀 "AI 챗봇과 금융상담 해보세요"

챗봇 비비의 최대 장점은 개인화된 상담 서비스다. 소비자가 모바일 금융 앱에서 비비에 '내 계좌 조회해 줄래?'와 같은 문장을 입력하면 계좌 조회뿐...

2021. 10. 5.

아주경제

신한은행 AI 금융비서 챗봇 오로라, 더 똑똑해진다

신한은행 AI 금융비서 챗봇 '오로라', 더 똑똑해진다 ... 신한은행은 인공지능(AI) 챗봇 서비스인 '오로라(Orora)'를 고도화했다고 25일 밝혔다. 오로라는...

2022. 2. 25.

02.
프로젝트 목표

1. 데이터베이스(DB)구축을 통해 종목의 정보를 바로 제공할 수 있게 한다.

2. 뉴스, 공시, 재무데이터 등을 사용하여 모델을 구축하고 종목, 테마의 상승 또는 하락을 예측한다.

3. DB 정보와 예측 결과를 챗봇을 통해 구현한다.

02.
프로젝트 목표

삼성전자 주가 얼마야?

삼성전자 주가는 68,000원입니다.

지금 뜨는 테마가 뭐야?

반도체, 철강, 엔터... 입니다.

반도체 테마 종목 추천해줘

삼성전자, Sk하이닉스, 동부하이텍 ... 입니다.

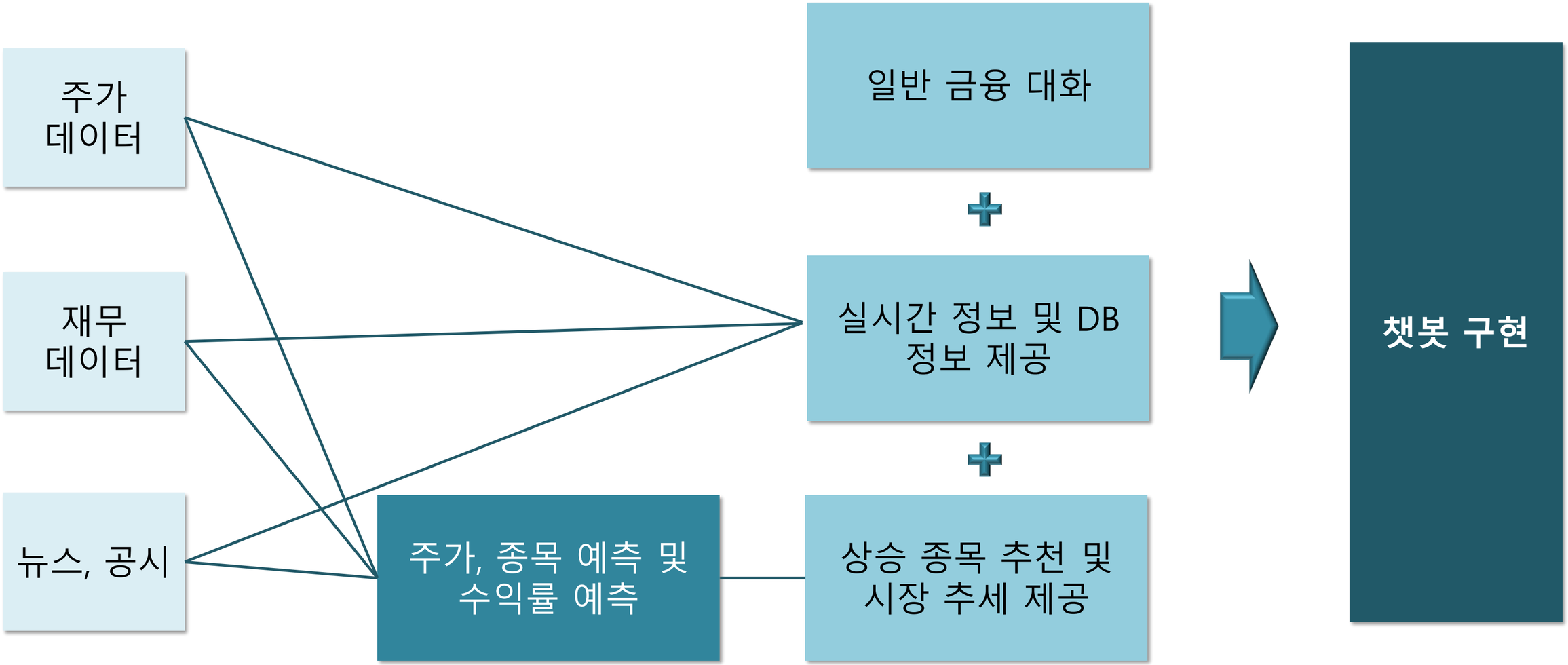
삼성전자 PER 비율이 어떻게 돼?

10.45배 입니다.

삼성전자 추세가 어떨까?

한 달뒤 약 3% 오를 추세입니다.

02.
프로젝트 Frame Work



03.
데이터 소스 및 적용기술

데이터 수집 및 전처리

Requests, aws, Selenium, pandas, jupyter, MySQL, Beautifulsoup, python Programming, NumPy

분석 모델 구현

Keras, scikit learn, PyTorch, PROPHET, NLTK, KoNLPy, docker, TensorFlow

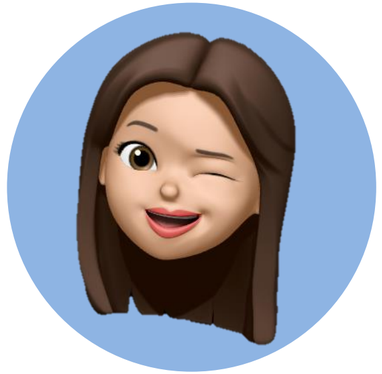
결과 및 시각화

matplotlib, seaborn, Streamlit, Telegram

04.
프로젝트 일정 및 역할분담

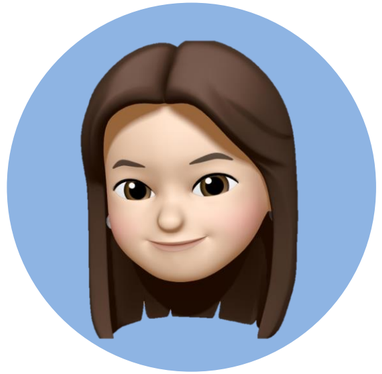
프로세스			2022-05-25 ~ 2022-06-01	2022-06-02 ~ 2022-06-09	2022-06-10 ~ 2022-06-17	2022-06-18 ~ 2022-06-27
DB구축 및 문헌 조사	재무데이터, 주가데이터 수집					
	뉴스, 공시 크롤링					
	수익모델 문헌 조사					
주식,테마 예측모델 구축	개별종목	재무제표DB ML, DL				
		재무제표DB 퀀트				
		뉴스,공시 NLP,				
	테마	코스피 추세, 업종 강세 추세				
		테마 종목 추천				
챗봇 구현	챗봇 모델 학습	KoGPT finetuning				
		Aws DB 연동, 실시간 뉴스, 정보 UI 연동				
결과 정리						

04.
프로젝트 일정 및 역할분담



이지혜

김원중

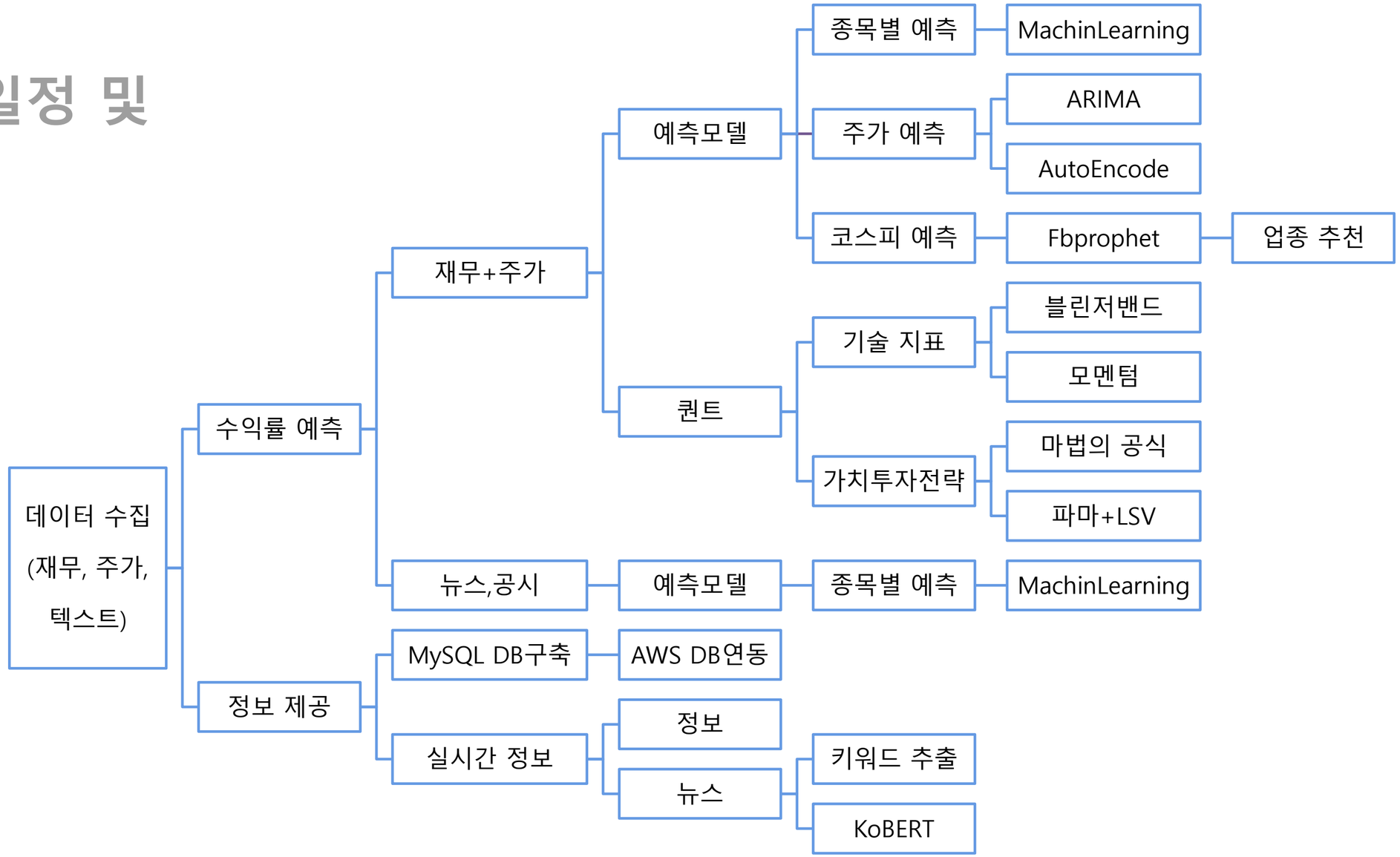


유일심

프로세스 진행

04.

프로젝트 일정 및 역할분담



04.

데이터수집 및 전처리

➢ 데이터 수집

- ✓ 공공데이터 DART에서 공시데이터 수집
- ✓ OpenDart, KRX API를 이용해 주가, 재무데이터 수집
- ✓ 네이버 뉴스 크롤링 수집 5년치(2016.01.01~2022.03.31)

➡ 각 수집한 데이터 전처리

DART

KRX 한국거래소

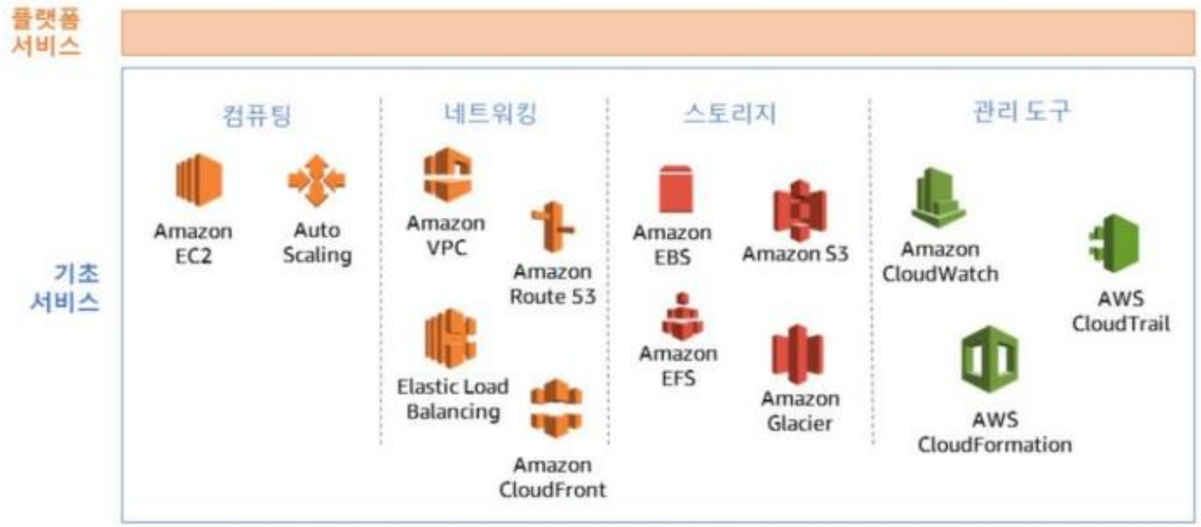
NAVER

04.
AWS DB구축

➢ AWS

✓ 아마존닷컴에서 개발한 클라우드 컴퓨팅 플랫폼
현재 소규모 법인(회사) 및 개인 을 포함한 다양한 사용자들이 사용하고 있으며,
클라우드 컴퓨팅의 장점을 이용하기 위해 많은 거대 기업에서도 활용

✓ 장점
- 저렴한 비용, 속도 및 민첩성 개선, 민첩성, 즉각적 융통성

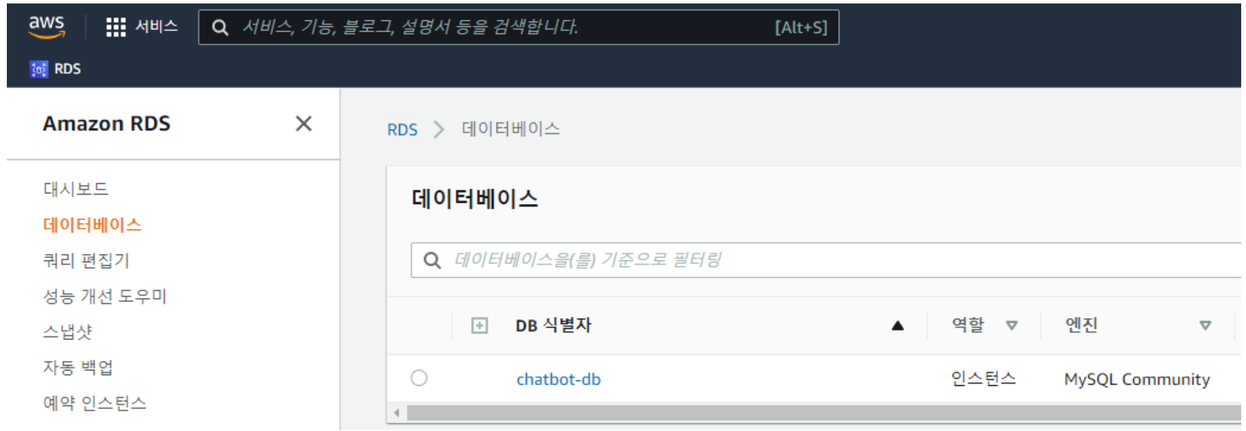


04.

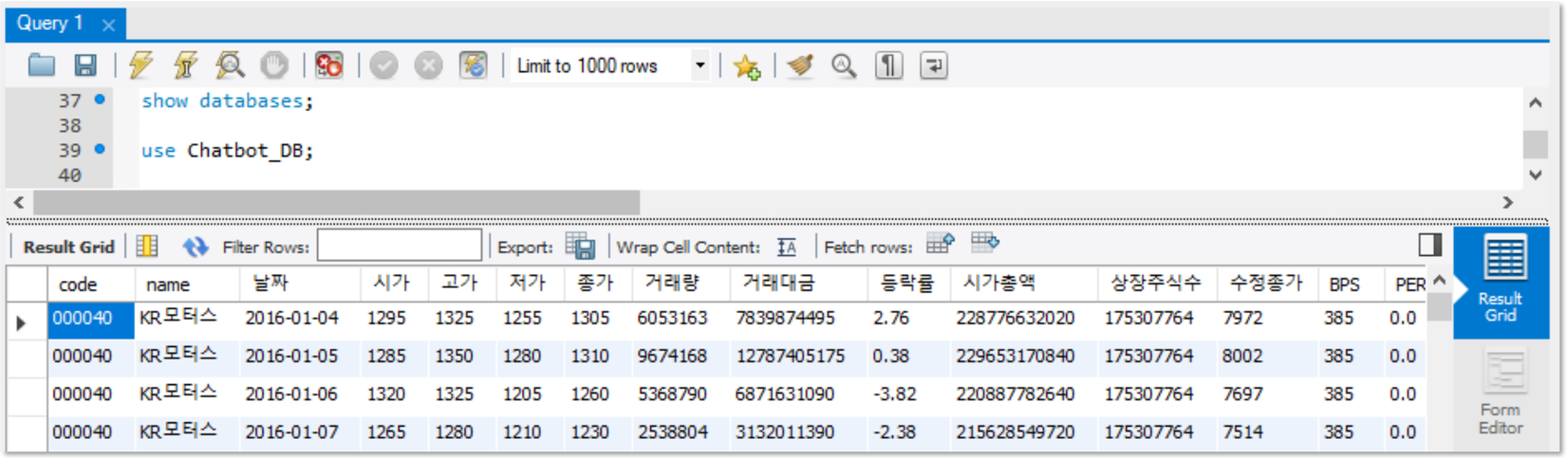
AWS DB구축

AWS에서
데이터베이스를 생성하여
주가+재무의 DB를 구축

- AWS



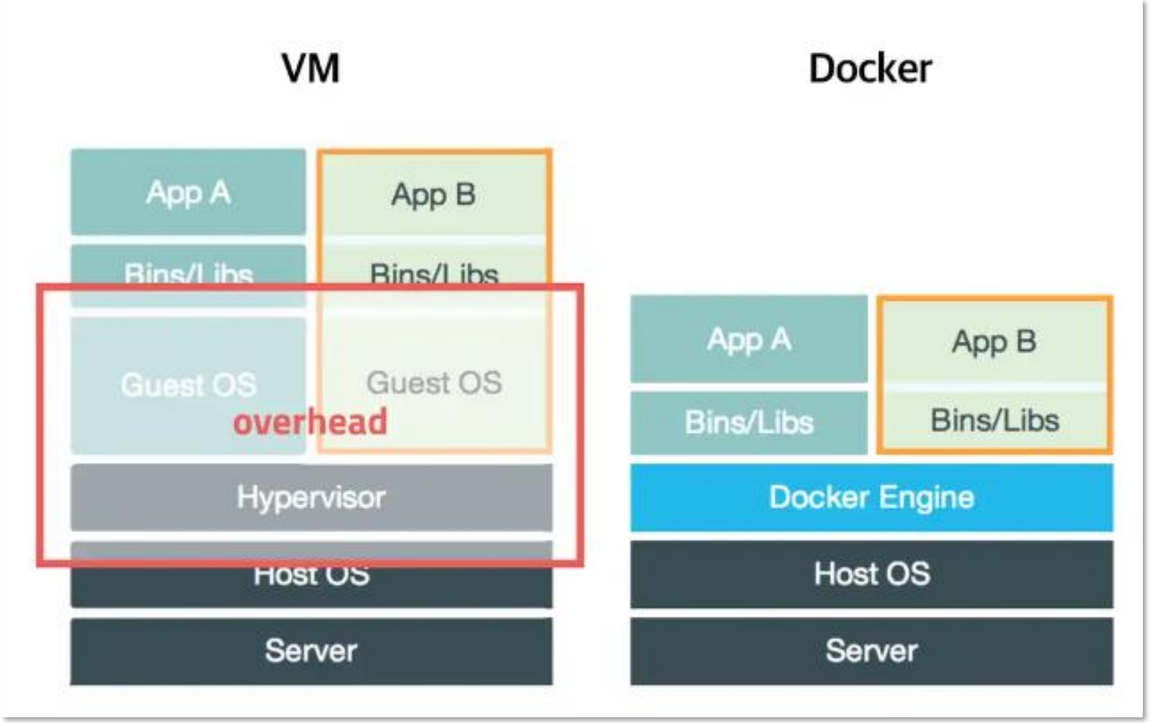
- MySQL Workbench



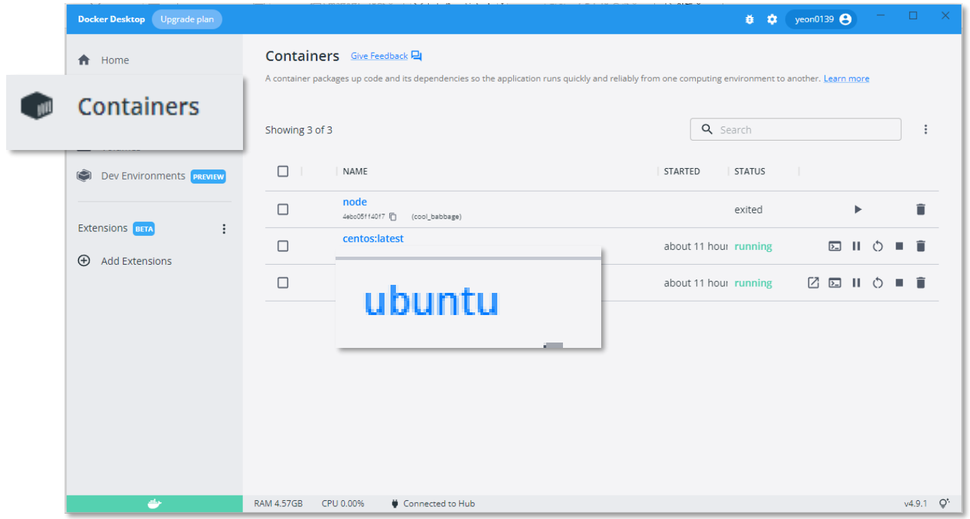
04.
Docker

➢ docker

✓ 리눅스의 응용 프로그램들을 프로세스 격리 기술들을 사용해 컨테이너로 실행하고 관리하는 오픈 소스 가상화 플랫폼으로 Docker를 사용하면 환경에 구애받지 않고 애플리케이션을 신속하게 배포 및 확장할 수 있으며 코드가 문제없이 실행



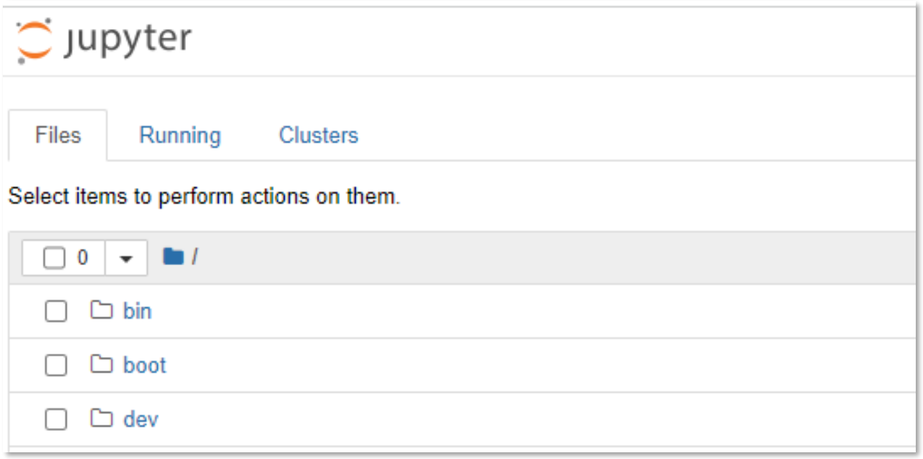
04.
Docker



- Docker GUI

Doker에서 ubuntu 컨테이너를
실행하여 jupyter notebook 환경 구축

```
root@48de7e522bfc:/# jupyter notebook --ip='0.0.0.0' --port=8888 --allow-root
[I 13:06:14.852 NotebookApp] Writing notebook server cookie secret to /root/.local/
[I 13:06:15.013 NotebookApp] Serving notebooks from local directory: /
[I 13:06:15.013 NotebookApp] Jupyter Notebook 6.4.8 is running at:
[I 13:06:15.013 NotebookApp] http://48de7e522bfc:8888/?token=8647390f937e2096f54bbe
[I 13:06:15.013 NotebookApp]  or http://127.0.0.1:8888/?token=8647390f937e2096f54bb
[I 13:06:15.013 NotebookApp] Use Control-C to stop this server and shut down all ke
[W 13:06:15.016 NotebookApp] No web browser found: could not locate runnable browse
[C 13:06:15.016 NotebookApp]
```



- Ubuntu를 이용한 Jupyter Notebook

04.

ARIMA모델을 이용한 시리얼데이터 이해

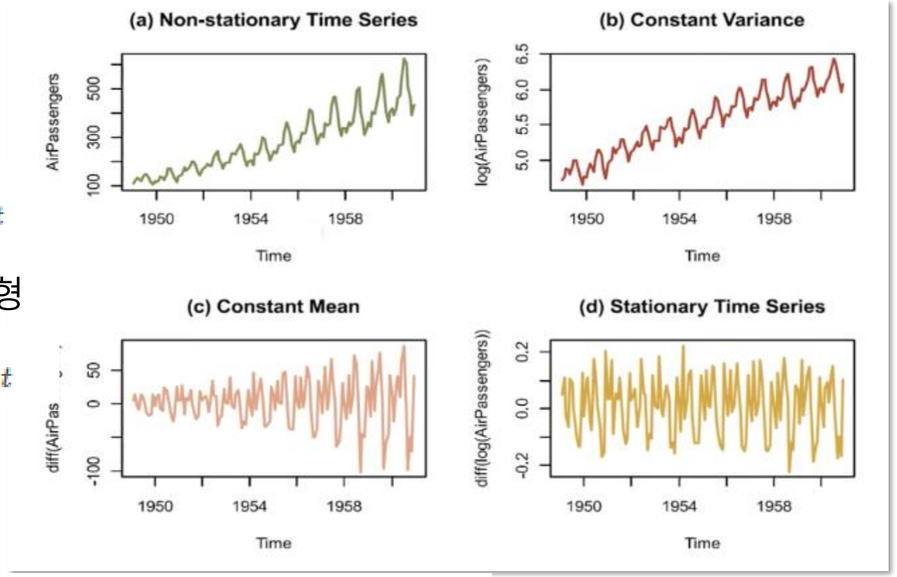
➢ ARIMA

✓ AR(Autoregression) 모형과 MA(Moving Average)모형을 합친 모형 ARIMA 모형은 시계열 데이터의 정상성(평균, 분산이 시간에 따라 일정한 성질)을 가정한다

✓ 정상성을 나타내지 않는 데이터를 정상 시계열로 변환

1.변동폭이 일정하지 않은 경우 -> 로그 변환

2.추세, 계절성이 존재하지 않는 경우 -> 차분(differncing, yt - yt_{-1})



✓ AR 모형 : 자기상관성을 시계열 모형으로 구성

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \ldots + \phi_p y_{t-p} + \varepsilon_t$$

✓ MA 모형 : 예측 오차를 이용하여 예측하는 모형

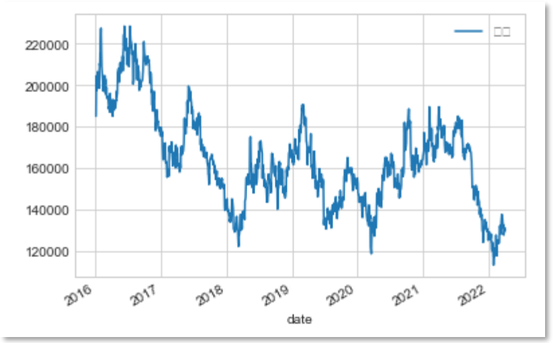
$$y_t = c + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \ldots + \theta_q \varepsilon_{t-q} + \varepsilon_t$$

✓ ARIMA 모형 : d차 차분한 데이터에 위 AR(p) 모형과 MA(q) 모형을 합친 모형

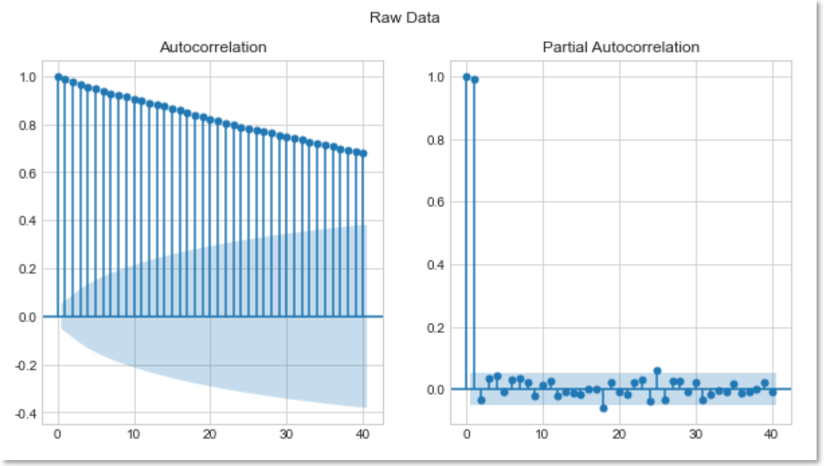
$$y_t' = c + \phi_1 y_{t-1}' + \phi_2 y_{t-2}' + \ldots + \phi_p y_{t-p}' + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \ldots + \theta_q \varepsilon_{t-q} + \varepsilon_t$$

04.

ARIMA모델을 이용한 시리얼데이터 이해

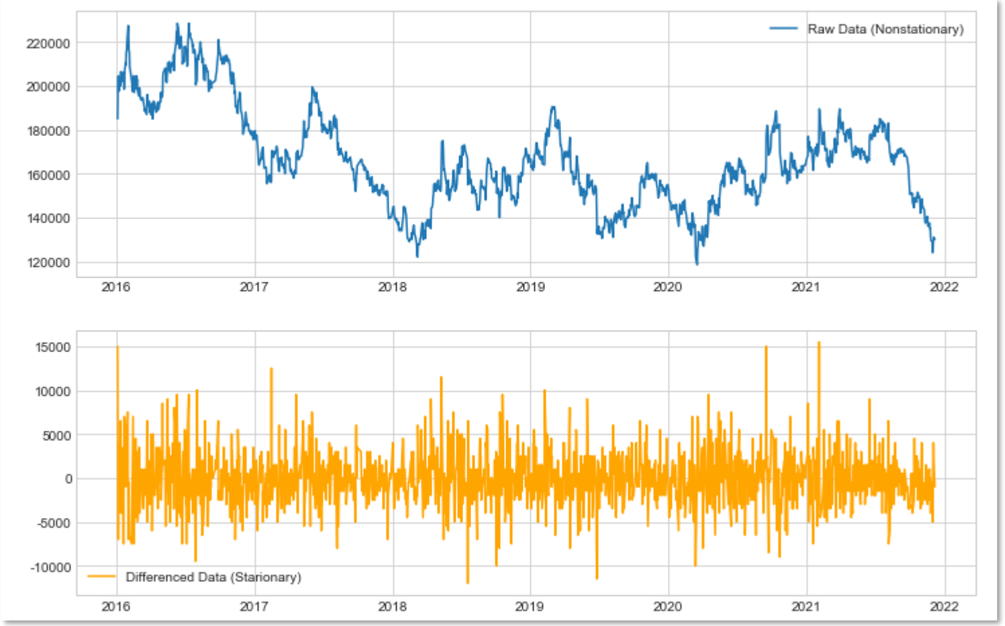


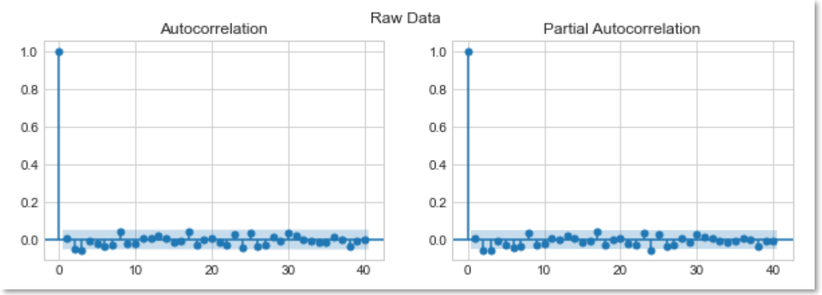
✓ 종가기준 그래프



✓ ACF와 PACF 그래프

✓ 종가그래프와 차분한 그래프





✓ 차분(differencing)한 ACF와 PACF 그래프

차분을 통해
Nonstationary한 데이터를
Stationary한 데이터로 변환

04.

ARIMA모델을 이용한 시리얼데이터 이해

```python
auto_arima_model = auto_arima(stock_data_train, start_p=1, start_q=1,
                              max_p=3, max_q=3, seasonal=False,
                              d=1,
                              trace=True,
                              error_action='ignore',
                              suppress_warnings=True,
                              stepwise=False)
```

```
ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=27564.511, Time=0.03 sec
ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=27549.206, Time=0.07 sec
ARIMA(0,1,2)(0,0,0)[0] intercept   : AIC=27551.040, Time=0.09 sec
ARIMA(0,1,3)(0,0,0)[0] intercept   : AIC=27542.641, Time=0.15 sec
ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=27549.089, Time=0.07 sec
ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=27551.156, Time=0.32 sec
ARIMA(1,1,2)(0,0,0)[0] intercept   : AIC=27553.242, Time=0.34 sec
ARIMA(1,1,3)(0,0,0)[0] intercept   : AIC=27543.778, Time=0.66 sec
ARIMA(2,1,0)(0,0,0)[0] intercept   : AIC=27551.104, Time=0.09 sec
ARIMA(2,1,1)(0,0,0)[0] intercept   : AIC=27553.222, Time=0.15 sec
ARIMA(2,1,2)(0,0,0)[0] intercept   : AIC=27539.579, Time=1.12 sec
ARIMA(2,1,3)(0,0,0)[0] intercept   : AIC=27537.940, Time=1.38 sec
ARIMA(3,1,0)(0,0,0)[0] intercept   : AIC=27543.984, Time=0.12 sec
ARIMA(3,1,1)(0,0,0)[0] intercept   : AIC=27544.673, Time=0.40 sec
ARIMA(3,1,2)(0,0,0)[0] intercept   : AIC=27538.035, Time=1.30 sec

Best model:  ARIMA(2,1,3)(0,0,0)[0] intercept
Total fit time: 6.303 seconds
```

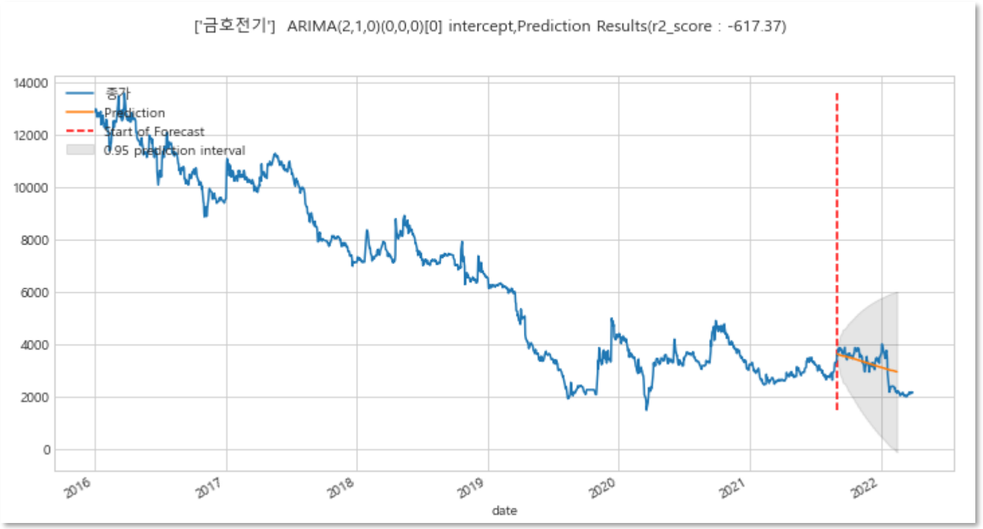
```python
auto_arima_model.summary()
```

SARIMAX Results

Dep. Variable:	y	No. Observations:	1457
Model:	SARIMAX(2, 1, 3)	Log Likelihood	-13761.970
Date:	Wed, 08 Jun 2022	AIC	27537.940
Time:	13:06:49	BIC	27574.925
Sample:	0	HQIC	27551.739
	- 1457		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	-61.6317	228.765	-0.269	0.788	-510.003	386.740
ar.L1	-0.7835	0.041	-19.110	0.000	-0.864	-0.703
ar.L2	-0.8845	0.042	-20.869	0.000	-0.968	-0.801
ma.L1	0.8496	0.043	19.962	0.000	0.766	0.933
ma.L2	0.9515	0.043	22.171	0.000	0.867	1.036
ma.L3	0.0378	0.015	2.589	0.010	0.009	0.066
sigma2	9.448e+06	0.029	3.25e+08	0.000	9.45e+06	9.45e+06

Ljung-Box (L1) (Q):	3.77	Jarque-Bera (JB):	160.48
Prob(Q):	0.05	Prob(JB):	0.00
Heteroskedasticity (H):	0.95	Skew:	0.30
Prob(H) (two-sided):	0.58	Kurtosis:	4.51



Auto_arima를 이용하여
Best model을 뽑아서
종목당 주가를 예측

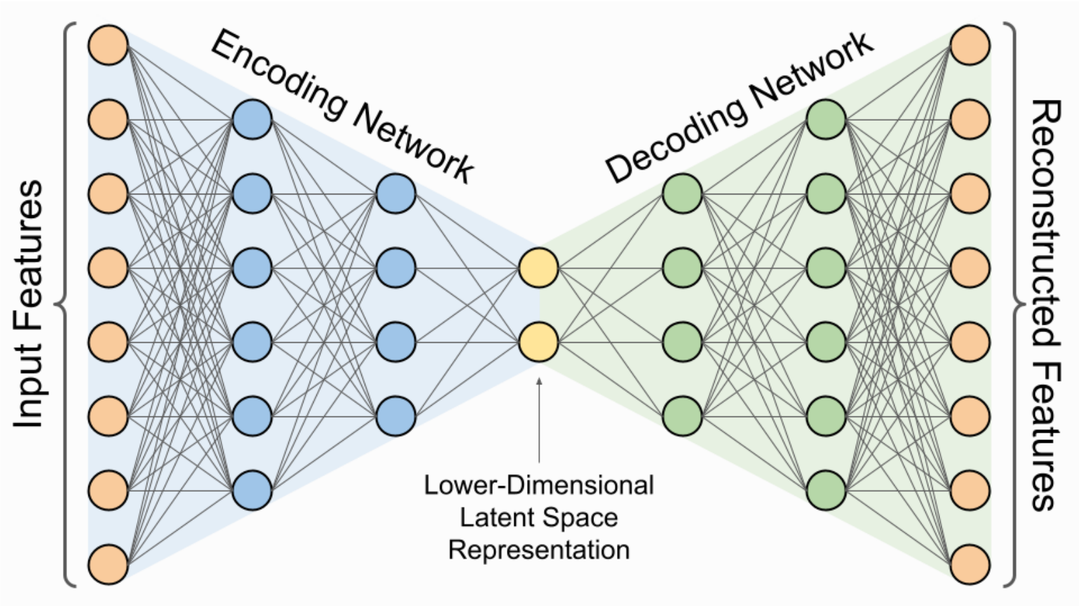
04.

AutoEncoder 모델를 이용한 종목별 주가예측

➢ Autoencoder

- ✓ 차원 축소 등을 위해 표현학습(Representation Learning)또는 특징학습(Feature Learning)을 비지도 학습의 형태로 학습하는 신경망
- ✓ 이상탐지 : 데이터를 통해 이상 패턴(이상징후), 개체 등을 찾아내는 것을 의미
- ✓ Anomaly Detection : Normal sampl과 Abnomal sample을 구별해내는 문제로 시계열 데이터의 일정 주기로 일정 패턴을 보이다가 이상치를 보이는 부분이 있는데 이런 이상치를 Anomaly or Abnoraml이라 하고, 이를 찾아내는 것을 Anomaly Detection이라 한다.

=> Autoencoder의 특성이 입력 데이터의 가장 중요한 특징을 학습하는 것이므로, noise가 아닌 주요 특징에 대해서만 학습하게 된다.



04.

AutoEncoder 모델를 이용한 종목별 주가예측

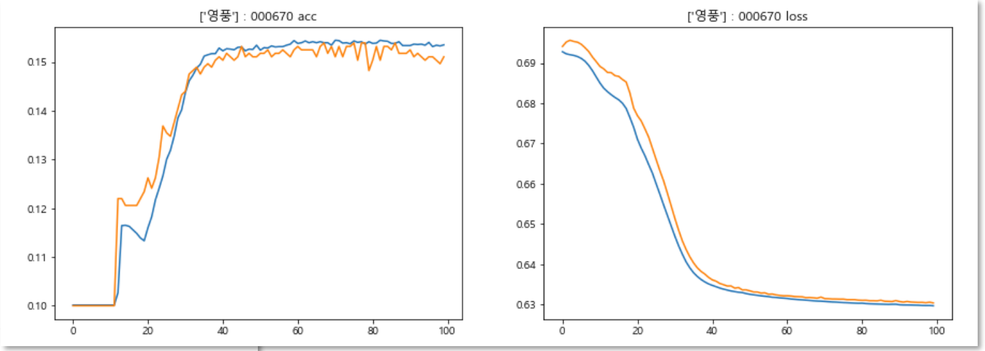
```python
autoencoder = Sequential()

autoencoder.add(Conv1D(16, 3, input_shape=(10,1), padding='same', activation='relu'))
autoencoder.add(MaxPooling1D(2, padding='same'))
autoencoder.add(Conv1D(1, 3, activation='relu', padding='same'))
autoencoder.add(MaxPooling1D(2, padding='same'))


autoencoder.add(Conv1D(1, 3, padding='same', activation='relu'))
autoencoder.add(UpSampling1D(2))
autoencoder.add(Conv1D(16, 2, activation='relu'))
autoencoder.add(UpSampling1D(2))
autoencoder.add(Conv1D(1, 3, activation='sigmoid', padding='same'))

autoencoder.summary()
```

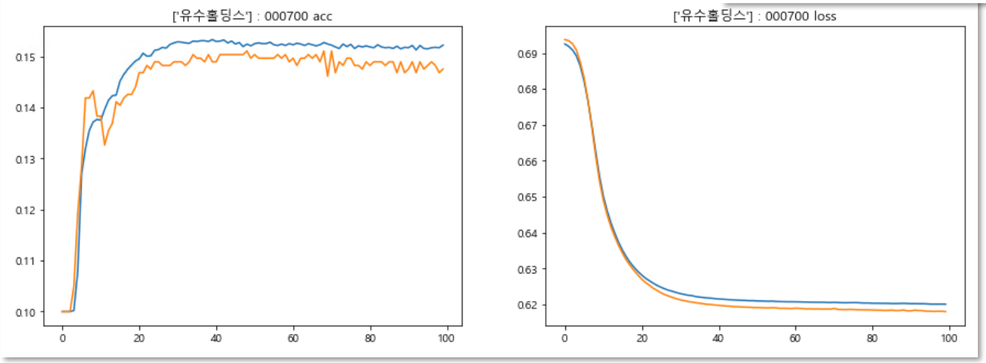
```python
autoencoder.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

```python
history = autoencoder.fit(x_train, x_train, epochs=100, batch_size=100, validation_data=(x_test, x_test))
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d (Conv1D)             (None, 10, 16)            64

 max_pooling1d (MaxPooling1D  (None, 5, 16)            0
 )

 conv1d_1 (Conv1D)           (None, 5, 1)              49

 max_pooling1d_1 (MaxPooling  (None, 3, 1)             0
 1D)

 conv1d_2 (Conv1D)           (None, 3, 1)              4

 up_sampling1d (UpSampling1D  (None, 6, 1)             0
 )

 conv1d_3 (Conv1D)           (None, 5, 16)             48

 up_sampling1d_1 (UpSampling  (None, 10, 16)           0
 1D)

 conv1d_4 (Conv1D)           (None, 10, 1)             49

=================================================================
Total params: 214
Trainable params: 214
Non-trainable params: 0
_________________________________________________________________
```

종목별로 autoencoder의
CNN 모델로 주가예측





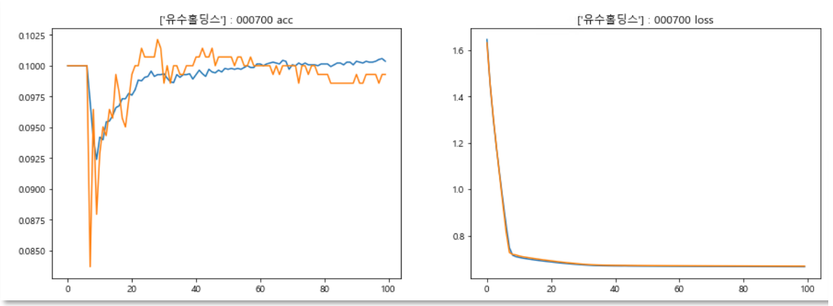
04.

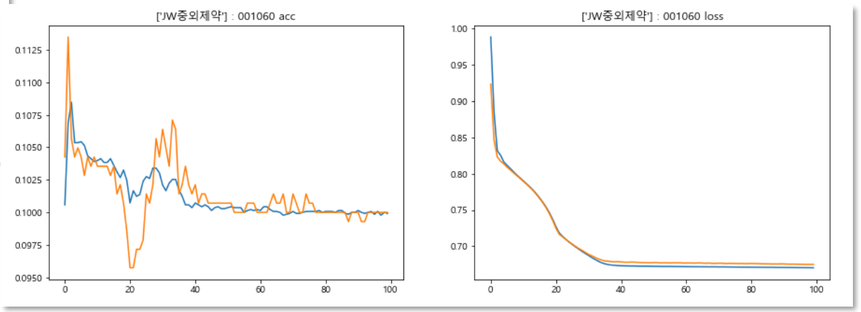
AutoEncoder 모델를 이용한 종목별 주가예측

```python
model = models.Sequential()
model.add(layers.LSTM(encoding_dim, activation='relu', input_shape=(10, 1)))
model.add(layers.RepeatVector(10))
model.add(layers.LSTM(1, activation='relu', return_sequences=True))
# model.add(layers.TimeDistributed(layers.Dense(1)))

model.summary()
```

```
WARNING:tensorflow:Layer lstm_12 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as f
allback when running on GPU.
WARNING:tensorflow:Layer lstm_13 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as f
allback when running on GPU.
Model: "sequential_497"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_12 (LSTM)              (None, 3)                 60

 repeat_vector_6 (RepeatVect  (None, 10, 3)            0
 or)

 lstm_13 (LSTM)              (None, 10, 1)             20

=================================================================
Total params: 80
Trainable params: 80
Non-trainable params: 0
_________________________________________________________________
```

종목별로 autoencoder의 LSTM 모델로 주가예측 (binary)



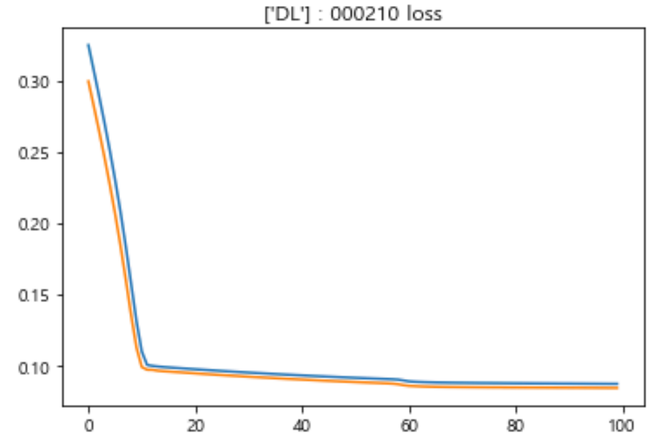


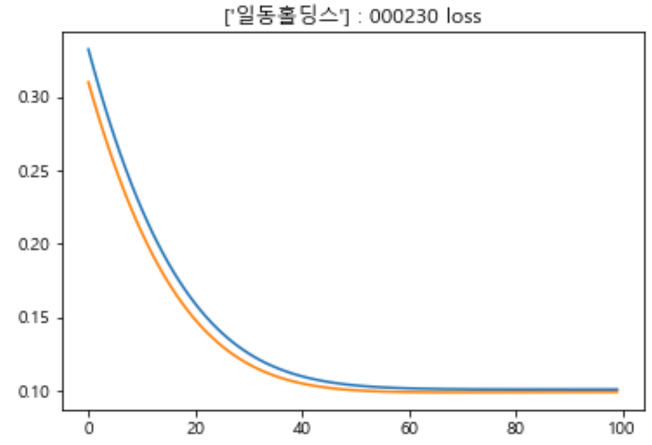
04.

AutoEncoder 모델를 이용한 종목별 주가예측

```python
model = models.Sequential()
model.add(layers.LSTM(encoding_dim, activation='relu', input_shape=(10, 1)))
model.add(layers.RepeatVector(10))
model.add(layers.LSTM(1, activation='relu', return_sequences=True))
model.add(layers.TimeDistributed(Dense(1)))

model.summary()
```

```
WARNING:tensorflow:Layer lstm_14 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU kernel as f
allback when running on GPU.
WARNING:tensorflow:Layer lstm_15 will not use cuDNN kernels since it doesn't meet the criteria. It w
allback when running on GPU.
Model: "sequential_498"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_14 (LSTM)              (None, 3)                 60

 repeat_vector_7 (RepeatVect  (None, 10, 3)            0
 or)

 lstm_15 (LSTM)              (None, 10, 1)             20

 time_distributed_5 (TimeDis  (None, 10, 1)            2
 tributed)

=================================================================
Total params: 82
Trainable params: 82
Non-trainable params: 0
_________________________________________________________________
```





종목별로 autoencoder의 LSTM 모델로 주가예측 (MSE)

04.

MachinLeaning을 이용한 종목별 상승예측 (주가+재무)

1, 3, 6개월로 수익기간을 정하고,
5, 10, 15 persent의 수익률을
기준으로 Target을 지정

			시가총액	...	부채총계	자본총계	매출액	영업이익	법인세차감전 순이익	당기순이익	수익률	5per	10per	15per
			3161746791000	...	5.279987e+11	1.395241e+12	2.764778e+11	1.990684e+10	6.575533e+10	5.127185e+10	8.0	1	0	0
			3200780702000	...	5.279987e+11	1.395241e+12	2.764778e+11	1.990684e+10	6.575533e+10	5.127185e+10	5.0	0	0	0
			3184051883000	...	5.279987e+11	1.395241e+12	2.764778e+11	1.990684e+10	6.575533e+10	5.127185e+10	8.0	1	0	0
813625500	1.05	100	3217509521000	...	5.279987e+11	1.395241e+12	2.764778e+11	1.990684e+10	6.575533e+10	5.127185e+10	7.0	1	0	0
279629500	3.99	100	3345763800000	...	5.279987e+11	1.395241e+12	2.764778e+11	1.990684e+10	6.575533e+10	5.127185e+10	3.0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

```python
rfc = RandomForestClassifier()
svc = SVC(probability=True)
lr = LogisticRegression()
xg = XGBClassifier()
lbgm = LGBMClassifier()

model = [rfc, svc, lr, xg, lbgm]

for m in model:
    m.fit(x_train, y_train)
    pred = m.predict(x_test)
    prob = m.predict_proba(x_test)
    print(m)
    print(classification_report(y_test, pred))
    print('-'*50)
    print('roc_auc : ',roc_auc_score(y_test, prob[:
    print('='*100)
```

```
RandomForestClassifier()
              precision    recall  f1-score   support

           0       0.95      0.95      0.95       182
           1       0.89      0.88      0.88        81

    accuracy                           0.93       263
   macro avg       0.92      0.91      0.91       263
weighted avg       0.93      0.93      0.93       263

--------------------------------------------------
roc_auc :  0.9707638040971374
====================================================
SVC(probability=True)
              precision    recall  f1-score   support

           0       0.85      0.86      0.85       182
           1       0.67      0.65      0.66        81

    accuracy                           0.79       263
   macro avg       0.76      0.76      0.76       263
weighted avg       0.79      0.79      0.79       263

--------------------------------------------------
roc_auc :  0.8630443630443629
====================================================
LogisticRegression()
              precision    recall  f1-score   support

           0       0.91      0.91      0.91       182
           1       0.80      0.79      0.80        81

    accuracy                           0.87       263
   macro avg       0.85      0.85      0.85       263
weighted avg       0.87      0.87      0.87       263

--------------------------------------------------
roc_auc :  0.9158187491520825
====================================================
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
              gamma=0, gpu_id=-1, importance_type=None,
              interaction_constraints='', learning_rate=0.300000012,
              max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
              monotone_constraints='()', n_estimators=100, n_jobs=8,
              num_parallel_tree=1, predictor='auto', random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=None)
              precision    recall  f1-score   support

           0       0.92      0.96      0.94       182
           1       0.89      0.81      0.85        81

    accuracy                           0.91       263
   macro avg       0.91      0.89      0.89       263
weighted avg       0.91      0.91      0.91       263

--------------------------------------------------
roc_auc :  0.9765635598968933
====================================================
LGBMClassifier()
              precision    recall  f1-score   support

           0       0.93      0.96      0.94       182
           1       0.89      0.84      0.87        81

    accuracy                           0.92       263
   macro avg       0.91      0.90      0.90       263
weighted avg       0.92      0.92      0.92       263

--------------------------------------------------
roc_auc :  0.9758852258852259
====================================================
```

머신러닝의 여러 모델을
실행시켜서 데이터에
맞는 모델 선정

04.

MachinLeaning을 이용한 종목별 상승예측 (주가+재무)

✓ 1개월 5%

```
pd.read_csv('c:/Users/bitcamp/Desktop/final_data/모델정리/종목별
```

	code	name	precision	f1_score	accuracy	roc_auc
32	990	['DB하이텍']	0.941176	0.421053	0.603604	0.752608
98	3480	['한진중공업홀딩스']	0.928571	0.742857	0.918919	0.910847
454	112610	['씨에스윈드']	0.923077	0.631579	0.873874	0.840698
415	79430	['현대리바트']	0.916667	0.687500	0.909910	0.934066
85	3000	['부광약품']	0.916667	0.785714	0.945946	0.967105
...	...	...	...	...	...	...

✓ 1개월 10%

```
pd.read_csv('c:/Users/bitcamp/Desktop/final_data/모델정리/종
```

	code	name	precision	f1_score	accuracy	roc_auc
75	2710	['TCC스틸']	0.956522	0.758621	0.873874	0.746241
451	105840	['우진']	0.928571	0.481481	0.747748	0.763732
129	4770	['써니전자']	0.920000	0.686567	0.810811	0.904589
415	79430	['현대리바트']	0.916667	0.846154	0.963964	0.966863
378	47810	['한국항공우주']	0.909091	0.444444	0.774775	0.750382
...	...	...	...	...	...	...

✓ 1개월 15%

```
pd.read_csv('c:/Users/bitcamp/Desktop/final_data/모델정리/종
```

	code	name	precision	f1_score	accuracy	roc_auc
32	990	['DB하이텍']	0.916667	0.468085	0.774775	0.797744
378	47810	['한국항공우주']	0.888889	0.484848	0.846847	0.780412
266	13580	['계룡건설산업']	0.857143	0.521739	0.900901	0.830263
279	15260	['에이엔피']	0.833333	0.322581	0.810811	0.657674
188	7610	['선도전기']	0.785714	0.846154	0.963964	0.975589
...	...	...	...	...	...	...
232	10950	['S-Oil']	0.017544	0.034483	0.495495	0.700000

✓ 3개월 15%

```
pd.read_csv('c:/Users/bitcamp/Desktop/final_data/모델정리/종목별
```

	code	name	precision	f1_score	accuracy	roc_auc
219	9810	['플레이그램']	0.963636	0.762590	0.733871	0.855506
222	10060	['OCI']	0.916667	0.733333	0.935484	0.760482
184	7340	['디티알오토모티브']	0.880000	0.354839	0.354839	0.657374
408	74610	['이엔플러스']	0.848485	0.767123	0.862903	0.932143
420	84010	['대한제강']	0.837209	0.911392	0.943548	0.971749
...	...	...	...	...	...	...
460	123700	['에스제이엔']	0.012821	0.025316	0.379032	0.971545

✓ 6개월 15%

```
pd.read_csv('c:/Users/bitcamp/Desktop/final_data/모델정리/종목
```

	code	name	precision	f1_score	accuracy	roc_auc
75	2710	['TCC스틸']	0.978495	0.866667	0.779528	0.827778
230	10770	['평화홀딩스']	0.952381	0.851064	0.944882	0.988385
312	23150	['MH에탄올']	0.950000	0.904762	0.968504	0.904978
277	14990	['인디에프']	0.944444	0.790698	0.929134	0.830784
106	3780	['진양산업']	0.942029	0.726257	0.614173	0.720588
...	...	...	...	...	...	...
372	42700	['한미반도체']	0.023622	0.046154	0.023622	0.690860

04.

MachinLeaning을 이용한 종목별 상승예측 (뉴스)

1 뉴스빈도
- 2 키워드 빈도
  - 2.1 전처리: 특수문자제거-> 띄어쓰기 -> 명사토큰화 -> 불용어 제거
    - 2.1.1 특수문자 제거:clean_text()
    - 2.1.2 띄어쓰기 : spacing()
  - 2.2 명사 토큰화: han.nouns
  - 2.3 불용어 제거: remove_stwords
  - 2.4 한글자 이상인 단어만

5년치 뉴스제목 데이터를
Count-vectorize 시켜
머신러닝에 사용

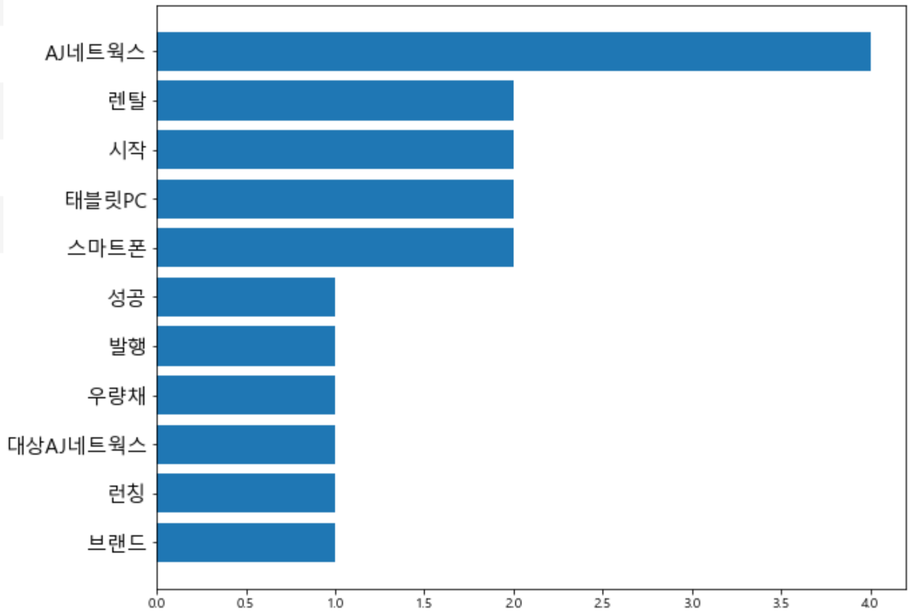
날짜	뉴스제목	뉴스
2016-01-04	AJ네트웍스, SKB 등에업고 '고리사채급 위약금 폭탄'	[AJ네트웍스, 사채급, 위약금, 폭탄]
2016-01-05	AJ네트웍스, 스마트폰, 태블릿PC 렌탈사업 시작	[AJ네트웍스, 스마트폰, 태블릿PC, 렌탈, 사업, 시작]
2016-01-06	AJ네트웍스, 스마트폰, 태블릿PC 등 Mobile Device 렌탈 사업 시작	[AJ네트웍스, 스마트폰, 태블릿PC, 렌탈, 사업, 시작]
2016-01-13	AJ네트웍스, 파렛트및 고소장비 렌탈 신규 서비스 브랜드 런칭	[AJ네트웍스, 파렛트, 고소장비, 렌탈, 신규, 서비스, 브랜드, 런칭]
2016-01-22	대상·AJ네트웍스 등 비우량채도 잇달아 발행 성공	[대상, AJ네트웍스, 우량채, 발행, 성공]

04.
MachinLeaning을 이용한 종목별 상승예측 (뉴스)

Out [124] :

날짜	등에업	고리사채급	위약금	폭탄	파렛트및	고소장비	브랜드	런칭	대상AJ네트웍스	우량채	...	도로와철도운송	마감	현재	시황	상승	상위	상승률	종목	업종	AJ네트웍스
2016-01-01	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2016-01-02	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2016-01-03	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2016-01-04	1	1	1	1	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	1
2016-01-05	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2022-03-27	0	0	0	0	0	0															
2022-03-28	0	0	0	0	0	0															
2022-03-29	0	0	0	0	0	0															
2022-03-30	0	0	0	0	0	0															
2022-03-31	0	0	0	0	0	0															

2282 rows × 1044 columns



5년치 뉴스제목 데이터를
Count-vectorize 시켜
머신러닝에 사용

04.

MachinLeaning을 이용한 종목별 상승예측 (뉴스)

4.1.3 3개월 15%

```
df_result = pd.read_csv(f'./data/model_result_test/machine_model3_63일_0.15.csv',index_col=0)
df_result = df_result[(df_result['precision']>0.5) &(df_result['precision'] != 1)]
df_result = df_result.sort_values(by='precision',ascending=False)
df_result.drop_duplicates(subset='회사이름',inplace=True)
df_result
```

executed in 28ms, finished 14:55:47 2022-06-23

	회사이름	모델이름	accuracy	precision	recall	roc_auc	모델주소
112	코리아써키트	CatBoostClassifier	0.297521	0.909091	0.106383	0.454492	./data/machine_model3_3개월_0.15/코리아써키트_CatBoostClassifier.pkl
145	후성	CatBoostClassifier	0.628099	0.833333	0.102041	0.592404	./data/machine_model3_3개월_0.15/후성_CatBoostClassifier.pkl
12	LG이노텍	RandomForestClassifier	0.380165	0.800000	0.142857	0.456725	./data/machine_model3_3개월_0.15/LG이노텍_RandomForestClassifier.pkl
14	LIG넥스원	XGBClassifier	0.347107	0.750000	0.037037	0.560185	./data/machine_model3_3개월_0.15/LIG넥스원_XGBClassifier.pkl
32	대웅제약	RandomForestClassifier	0.876033	0.666667	0.125000	0.607143	./data/ma
45	두산	CatBoostClassifier	0.603306	0.666667	0.040816	0.480159	
51	디티알오토모티브	CatBoostClassifier	0.173554	0.666667	0.019802	0.491584	./data/machir
89	아세아시멘트	XGBClassifier	0.702479	0.666667	0.054054	0.534266	./dat
23	TCC스틸	XGBClassifier	0.330579	0.571429	0.048780	0.436210	

모델예측 500종목 중 precision이
0.5보다 큰 종목만 선출
->3개월 5% 19종목, 6개월 5% 5종목
->3개월 10% 5종목, 6개월 5% 7종목
-> 3개월 15% 9종목, 6개월 15% 4종목 선출

4.1.6 6개월 15%

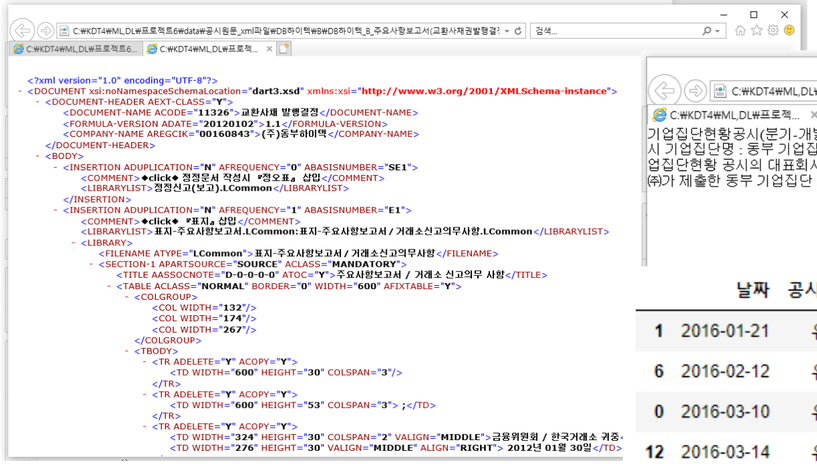
```
df_result = pd.read_csv(f'./data/model_result_test/machine_model3_126일_0.15.csv',index_col=0)
df_result = df_result[(df_result['precision']>0.5) &(df_result['precision'] != 1)]
df_result = df_result.sort_values(by='precision',ascending=False)
df_result.drop_duplicates(subset='회사이름',inplace=True)
df_result
```

executed in 19ms, finished 14:56:30 2022-06-23

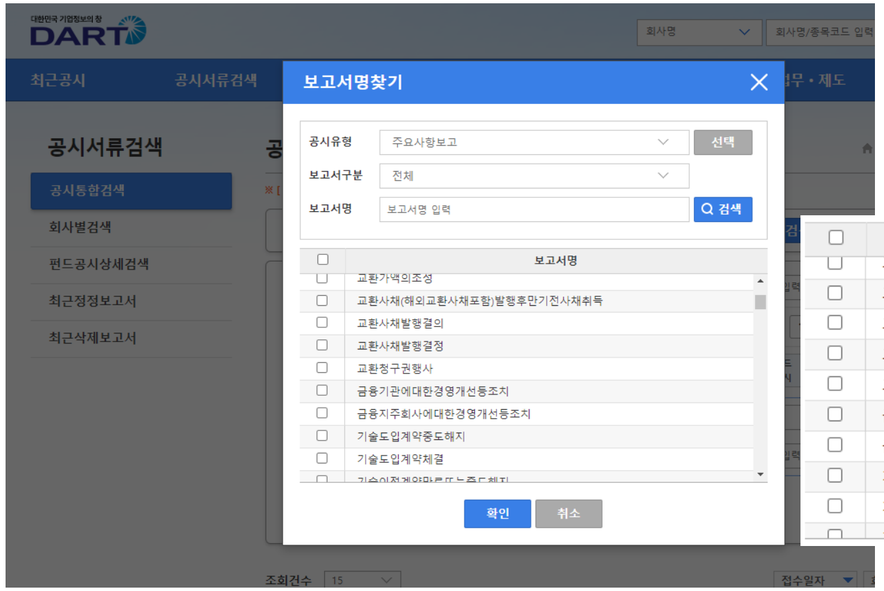
	회사이름	모델이름	accuracy	precision	recall	roc_auc	모델주소
37	TCC스틸	CatBoostClassifier	0.198347	0.952381	0.172414	0.293103	./data/machine_model3_6개월_0.15/TCC스틸_CatBoostClassifier.pkl
122	진양산업	CatBoostClassifier	0.148760	0.875000	0.064220	0.753823	./data/machine_model3_6개월_0.15/진양산업_CatBoostClassifier.pkl
116	일진머티리얼즈	CatBoostClassifier	0.528926	0.859649	0.500000	0.531943	./data/machine_model3_6개월_0.15/일진머티리얼즈_CatBoostClassifier.pkl
112	이수화학	RandomForestClassifier	0.504132	0.571429	0.065574	0.509973	./data/machine_model3_6개월_0.15/이수화학_RandomForestClassifier.pkl

04.

MachinLeaning을 이용한 종목별 상승예측 (공시)



	날짜	공시회사	공시보고서명
1	2016-01-21	유 SK	주요사항보고서(회사합병결정) (자회사의 주요경영사항)
6	2016-02-12	유 SK	기타안내사항(안내공시)
0	2016-03-10	유 SK	[기재정정]매출액또는손익구조30%(대규모법인은15%)이상변경
12	2016-03-14	유 SK	[기재정정]매출액또는손익구조30%(대규모법인은15%)이상변경 (자회사의 주요경영사항)
4	2016-05-10	유 SK	[기재정정]해외증권거래소등에신고한사업보고서등의국내신고 (자회사의 주요경영사항)
...	...	...	...
4	2022-04-28	유 SK	현금・현물배당결정 (자회사의 주요경영사항)
3	2022-04-29	유 SK	해외증권거래소등에신고한사업보고서등의국내신고 (자회사의 주요경영사항)
2	2022-05-16	유 SK	분기보고서 (2022.03)
1	2022-05-24	유 SK	타법인주식및출자증권취득결정 (자회사의 주요경영사항)
0	2022-05-26	유 SK	증권신고서(채무증권)



1. 파싱되는 문서와 안되는 문서가 공존
2. 보고서 형식이 너무 다양하여 중요한 텍스트 추출 불가

04.

Fb-prophet모델을 이용한 코스피지수예측

➢ Prophet

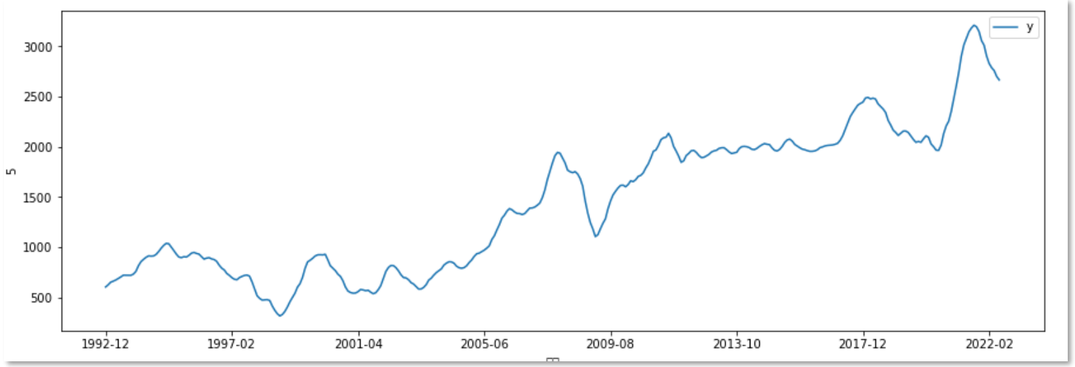
- ✓ 페이스북이 만든 시계열 예측 라이브러리
- ✓ ARIMA 같은 시계열 모델은 시간에 종속적인 구조를 가지는 반면 Prophet은 종속적이지 않고 Curve Fitting으로 문제를 해결

- ✓ 장점
  1. 학습 속도가 빠르고, 빈 구간을 interpolate하지 않아도 된다
  2. 직관적으로 이해할 수 있는 파라미터를 통해 모형을 쉽게 조정이 가능하다

$$y(t) = g(t) + s(t) + h(t) + e_t$$

04.

Fb-prophet모델을 이용한 코스피지수예측



코스피 지수 데이터를
Prophet으로 예측

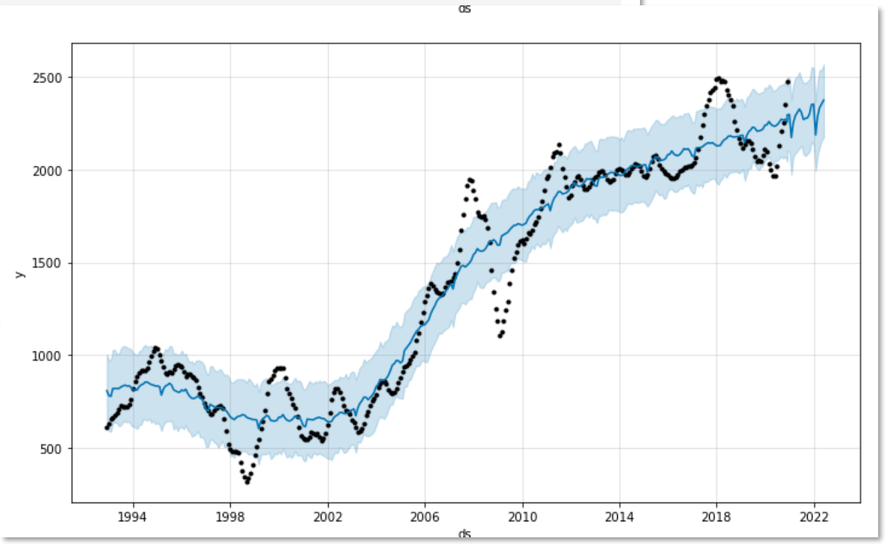
```
df_prophet = Prophet(changepoint_prior_scale=0.15, daily_seasonality=True)
df_prophet.fit(train)
```

```
INFO:fbprophet:Disabling weekly seasonality. Run prophet with weekly_seasonality=Tru
```

```
<fbprophet.forecaster.Prophet at 0x1c1dcbd8708>
```

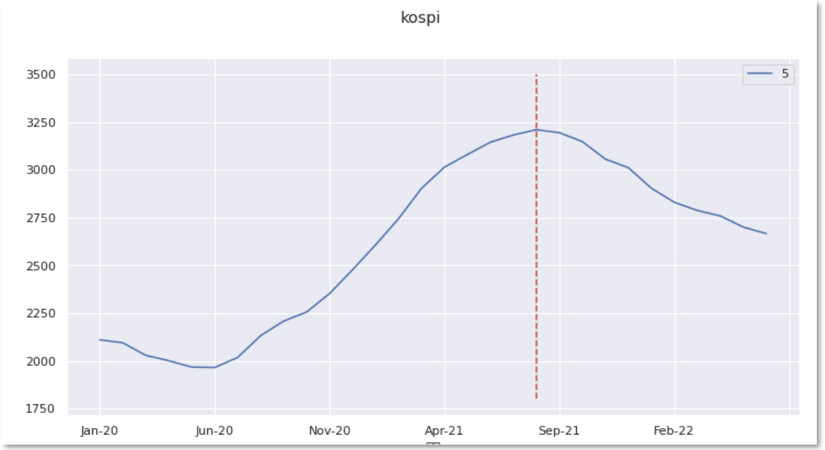
```
df_forecast
```

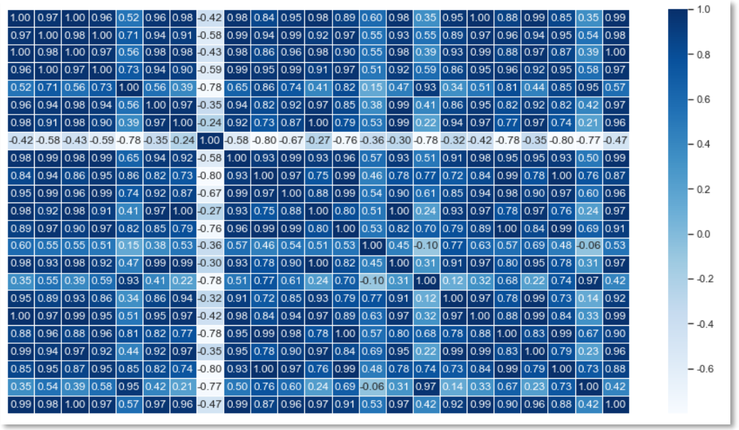
	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_terms_lower	additive_terms_upper	daily	da
0	1992-12	646.278622	588.365143	1021.355853	646.278622	646.278622					
1	1993-01	647.745092	593.961060	982.804905	647.745092	647.745092					
2	1993-02	649.211562	567.028202	971.990183	649.211562	649.211562					
3	1993-03	650.536116	608.496439	1016.767860	650.536116	650.536116					
4	1993-04	652.002587	613.960848	1012.884745	652.002587	652.002587					
...	...	...	...	...	...	...	...				
351	2022-	2146.251938	1993.000661	2380.409109	2129.355845	2163.708857					

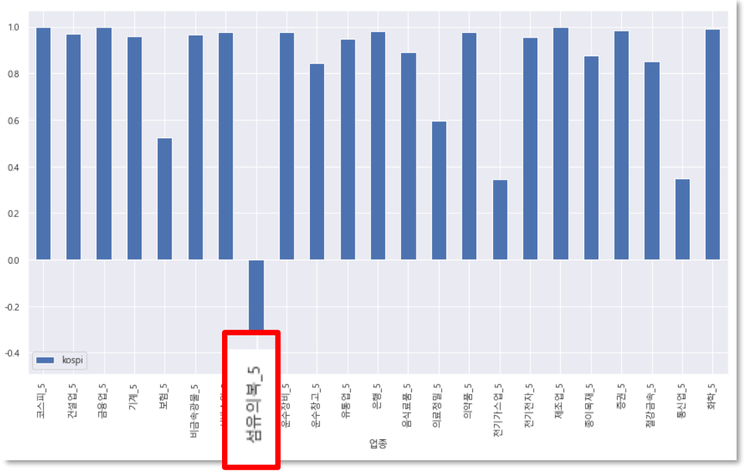


04.

코스피 지수대비 업종추천







코스피 지수를 파악하여
강세장과 약세장의 시점을 분리하고
업종과의 상관계수를 구한다

```
kospi_corr_df.sort_values(by=['kospi'])
```

0]:

	업종	kospi
7	섬유의복_5	-0.419609
15	전기가스업_5	0.346384
21	통신업_5	0.349331
4	보험_5	0.524969
13	의료정밀_5	0.596333
9	운수창고_5	0.843766
20	철강금속_5	0.849781
18	종이목재_5	0.875944
12	음식료품_5	0.890343
10	유통업_5	0.948501
16	전기전자_5	0.954856
3	기계_5	0.960248
5	비금속광물_5	0.964626
1	건설업_5	0.971464
14	의약품_5	0.977247
8	운수장비_5	0.978528
6	서비스업_5	0.978629
11	은행_5	0.979900
19	증권_5	0.985212
22	화학_5	0.992914
2	금융업_5	0.997373
17	제조업_5	0.998766
0	코스피_5	1.000000

04.

퀀트 투자 전략의 비교

➢ 퀀트 투자 전략

- ✓ 기술지표 – 볼린저밴드, 절대 모멘텀
- ✓ 가치투자전략 – 마법의 공식, FamaLSV

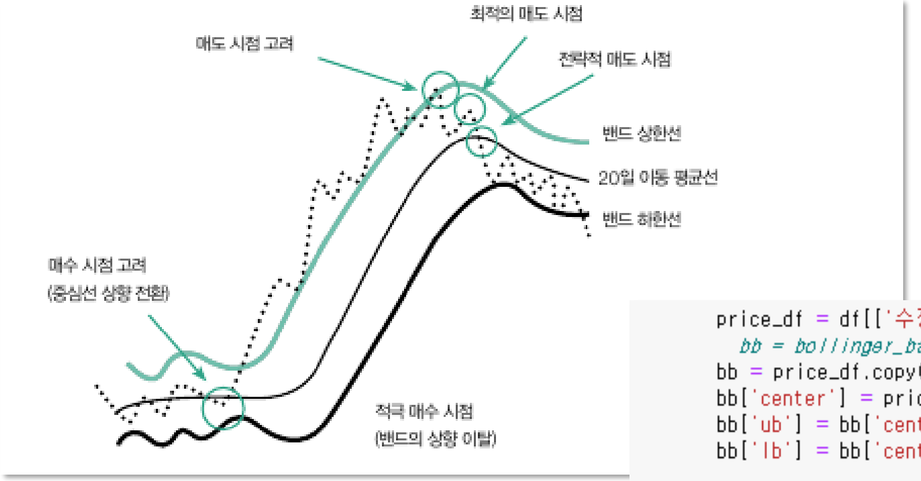
- ✓ 각각의 전략들은 3,6,12개월 리밸런싱 전략으로 나누어지며 현재시장과 비슷한 상황의 시점들의 벤치마크대비 오버퍼폼이 가장 많이 나온 전략들을 추천해준다.
- ✓ 기술지표의 매도 시그널을 잡지 않은 이유는 수수료 등의 핸디캡을 없애고
- ✓ 가치투자전략들과 비교를 할 수 있게 만들기 위해 똑같은 리밸런싱 전략을 구축하였다.

04.

퀀트 투자 전략의 비교

➢ 볼린저 밴드

상 하위 밴드는 20일 이동 평균선의 +- 2 *20일 이동 표준편차를 사용한다.
밴드의 상향이탈시점에서
매수 시그널을 잡는다.




```
price_df = df[['수정종가','종목명']]
  bb = bollinger_band(price_df,20,2)
bb = price_df.copy()
bb['center'] = price_df['수정종가'].rolling(20).mean() #중앙 이동평균선
bb['ub'] = bb['center'] + 2 * price_df['수정종가'].rolling(20).std() # 상단 밴드
bb['lb'] = bb['center'] - 2 * price_df['수정종가'].rolling(20).std() # 하단 밴드

book = bb[['수정종가','종목명']].copy()
book['trade'] = ''

for i in book.index:
    if bb.loc[i, 'lb'] > bb.loc[i, '수정종가']:
        book.loc[i, 'trade'] = 'buy'
bb_lis.append(book[book['trade'] == 'buy'])
```

date	수정종가	종목명	trade
2016-05-18	6933	40	buy
2016-05-25	6536	40	buy
2016-08-09	6597	40	buy
2016-09-09	6475	40	buy
2016-10-17	6353	40	buy
...	...	...	...

수정종가_x	종목명_x
889.0	40.0
15950.0	50.0
37450.0	80.0
58400.0	100.0
129500.0	120.0

04.

퀀트 투자 전략의 비교

➢ 절대 모멘텀

최근 1년동안 수익률이 양수면
매수하는 시그널을 잡는다.

```
month_last_df.set_index(['date'],inplace=True)
month_last_df['BF_1M_Adj Close'] = month_last_df.shift(1)['수정종가']
month_last_df['BF_12M_Adj Close'] = month_last_df.shift(12)['수정종가']
month_last_df.fillna(0, inplace=True)
```

```
for x in month_last_df.index:
    signal = ''
    # 절대 모멘텀을 계산한다.
    momentum_index = month_last_df.loc[x,'BF_1M_Adj Close'] / month
    # 절대 모멘텀 지표 True / False를 판단한다.
    flag = True if ((momentum_index > 0.0) and (momentum_index !=
    else False ₩
    and True
    if flag :
        signal = 'buy ' + ticker # 절대 모멘텀 지표가 Positive이면
    print('날짜 : ',x,' 모멘텀 인덱스 : ',momentum_index, 'flag : '
    book.loc[x:,'trade'] = signal
```

종목명	수정종가	STD_YM
40	889	2022-03
50	15950	2022-03
80	37450	2022-03
100	58400	2022-03
120	129500	2022-03

상대 모멘텀

절대 모멘텀

CODE	AAPL	AMZN	BND	GDX	GLD	GM	MSFT	SLV	SPY	USM	USO	WMT
Date												
2010-01-2	0.997635	0.936595	1.011682	0.853490	0.965027	NaN	0.910501	0.922809	0.867588	0.870922	0.885026	0.985248
2010-02-2	1.050789	0.996046	1.009869	1.022124	1.009868	NaN	1.013871	0.962274	1.015404	1.004918	1.057190	1.011032
2010-03-3	1.124458	1.060172	0.996720	0.994848	0.985614	NaN	1.086004	1.062925	1.049976	1.117472	1.050587	1.037344
2010-04-3	1.106454	1.040134	1.011023	1.088577	1.046254	NaN	1.047325	1.041048	1.008574	0.996684	1.002182	0.966681
2010-05-2	0.984445	0.912503	1.011821	0.986599	1.027218	NaN	0.939808	0.970827	0.908786	0.986954	0.821632	0.946283
2010-06-3	0.984345	0.886593	1.013782	1.043164	1.014791	NaN	0.858766	1.010544	0.964485	1.008331	1.023508	0.944030
2010-07-3	1.033294	1.062455	1.008130	0.971768	0.986757	NaN	1.114421	1.066027	1.079083	1.142059	1.073145	1.038958
2010-08-3	0.928395	1.039644	1.015888	1.115713	1.099804	NaN	0.896134	1.054009	0.933990	0.887947	0.875000	0.969679
2010-09-3	1.133504	1.165448	1.009156	1.056278	1.051114	NaN	1.024668	1.123845	1.057911	1.075322	1.057360	1.045213
2010-10-2	1.065343	1.078846	1.004006	1.012368	1.032760	NaN	1.083828	1.116397	1.033854	1.007590	0.987089	1.015180
2010-11-1	1.022914	1.078853	0.992210	1.039664	1.020531	1.000293	0.949331	1.140008	0.998662	0.976328	1.006423	0.990949
2010-12-3	1.019469	1.019541	0.995089	1.026821	1.024671	1.059804	1.071813	1.085321	1.044635	1.061198	1.040387	0.991367
2011-01-3	1.029504	0.920855	1.002121	0.887280	0.941087	0.984620	0.991065	0.913609	1.012030	0.987839	0.968732	1.027676
2011-02-2	1.023708	1.006856	1.004877	1.079103	1.052446	0.919880	0.955238	1.168016	1.016434	1.007252	1.029419	0.922778
2011-03-3	0.997713	1.065090	0.999082	0.988467	0.998786	0.941730	0.970566	1.085622	1.017078	1.063623	1.052371	1.008622
2011-04-2	1.076166	1.087046	1.013383	1.040659	1.094812	0.990126	1.017289	1.271839	1.024934	0.947105	1.043865	1.054871
2011-05-3	1.054476	0.977653	1.011315	0.967232	0.994881	0.988552	0.981057	0.877889	0.990310	1.009193	0.901402	1.009909
2011-06-3	0.971523	1.042838	0.991449	0.956545	0.973818	1.004100	1.084265	0.946573	1.005720	0.982416	0.942308	0.978687
2011-07-2	1.127563	1.062199	1.015886	1.058616	1.062182	0.809167	1.053036	1.177273	0.963725	0.984172	1.007268	0.985050
2011-08-2	0.999956	0.972483	1.013266	1.094840	1.126837	0.856074	0.981587	1.056412	0.948060	0.988795	0.927668	1.018145
2011-09-2	1.030761	1.017351	1.002183	0.870779	0.888627	0.878240	0.949638	0.713475	0.949428	0.925753	0.884527	0.985755
2011-10-3	1.080566	1.007218	0.995476	1.080441	1.039637	1.131188	1.085818	1.133944	1.141838	1.089761	1.201749	1.091609
2011-11-3	0.963910	0.906801	0.992026	1.032473	1.019430	0.912359	0.991638	0.989793	1.024508	1.130592	1.101704	1.047484
2011-12-3	1.044003	0.878101	1.012132	0.855331	0.896009	0.967080	1.026899	0.945574	1.010610	1.015237	0.987030	1.025985
2012-01-3	1.110038	1.086075	1.009488	1.045442	1.085877	1.141082	1.135100	1.119987	1.028941	1.054483	0.952883	1.017073
2012-02-2	1.189966	1.001282	1.001432	0.975491	0.968820	1.067706	1.048883	1.023803	1.034348	0.922781	1.054703	0.950143

진입일 : 2020-06-30 long 진입가격 : 157500

날짜 : 2020-06-30 모멘텀 인덱스 : 0.1970260223048328 flag : True signal : buy 000120.csv

04.

퀀트 투자 전략의 비교

➢ 마법의 공식

개별종목을 시가총액순으로 나열한 다음 일정금액 이상의 종목을 기준으로 설정함
이익 수익률 지표가 마이너스를 보인 종목은 제거함
자본 수익률과 이익수익률 두가지 지표를 기준으로 순위를 매겨 종목을 선별함.

```python
a = result_rank.mask(result_rank < 0, np.nan)
s_value_mask_rank2 = a.rank(ascending=True, na_option='bottom')
result_rank = s_value_mask_rank2

result_rank = result_rank.where(result_rank <= 10, 0)  # 합산 순위 필터링
result_rank = result_rank.mask(result_rank > 0, 1)  # 순위 제거

mf_df = df_2022.loc[result_rank > 0,['종목명','시가총액']].copy() # 선택된 종목 데이터프레임
# mf_stock_list = df_2022.loc[result_rank > 0, '종목명'].values # 선택된 종목명 추출
```

```python
per_rank.head()
```

```
date
2016-01-04     24.0
2016-01-04    154.0
2016-01-04    127.0
2016-01-04     24.0
2016-01-04     60.0
Name: PER, dtype: float64
```

```python
roa_rank.head()
```

```
date
2016-01-04    150.0
2016-01-04    125.0
2016-01-04     79.0
2016-01-04     68.0
2016-01-04    110.0
Name: ROA, dtype: float64
```

date	종목명	시가총액
2016-01-04	700.0	1.955740e+11
2016-01-04	990.0	6.278048e+11
2016-01-04	1470.0	3.494667e+10
2016-01-04	5720.0	3.963565e+11
2016-01-04	10060.0	1.733849e+12
2016-01-04	10600.0	3.507473e+10
2016-01-04	23150.0	6.352911e+10
2016-01-04	93370.0	5.081986e+11
2016-01-04	123700.0	1.004955e+11
2016-01-04	210540.0	6.636067e+10

1. 투자에 사용할 자금과 투자 대상 기업 규모를 설정한다.
2. 마법공식에 따른 순위를 나열한다.
   예를 들어, 코스피 200에서 자본수익률이 높은 기업에 대한 순위를 매기고, 동시에 이익수익률이 높은 기업을 순서대로 나열한다.
3. 자본 수익률 순위와 이익 수익률 순위를 더한다.
   예를 들면, 자본 수익률 순위 1위와 이익 수익률 순위 4위의 총합은 5이다.
   더한 값의 등수가 낮은 순으로 순위를 매긴다.
4. 등수가 가장 낮은 5~7개 기업을 매수한다.
   처음 투자 기간 1년 동안은 투자 금액의 20~30%만 매수한다.
5. 나머지 자금을 2~3개월마다 위 과정을 반복해 예정 투자금의 100%를 사용해 매수한다.
6. 매수가 완료된 주식을 1년 동안 보유한 후 매도한다.
7. 매도 이후 위 과정을 계속해서 반복한다.

04.

퀀트 투자 전략의 비교

➢ Fama+LSV

```
df_2016 = df.loc[date]
df_2016['PSR'] = df_2016['시가총액'] / df_2016['매출액']
# df_2018['시총_rank'] = df_2018.loc['2018-01-04':,'시가총액'].rank()
# df_2018['pbr_rank'] = df_2018.loc['2018-01-04':,'PBR'].rank()
# df_2018['psr_rank'] = df_2018.loc['2018-01-04':,'PSR'].rank()
df_2016['시총_rank'] = df_2016['시가총액'].rank()
df_2016['pbr_rank'] = df_2016['PBR'].rank()
df_2016['psr_rank'] = df_2016['PSR'].rank()

df_2016['total_rank'] = df_2016['pbr_rank'] + df_2016['psr_rank'] + df_2016['시총_rank']
```

저 PSR + 저 PBR + 저 시총으로 순위를 매겨 종목선별

pbr_rank	psr_rank	total_rank
20.0	42.0	87.0
38.0	15.0	105.0
23.5	56.0	111.5
9.5	2.0	122.5
9.5	29.0	122.5

```
famaLSV('2022-01-03')
```

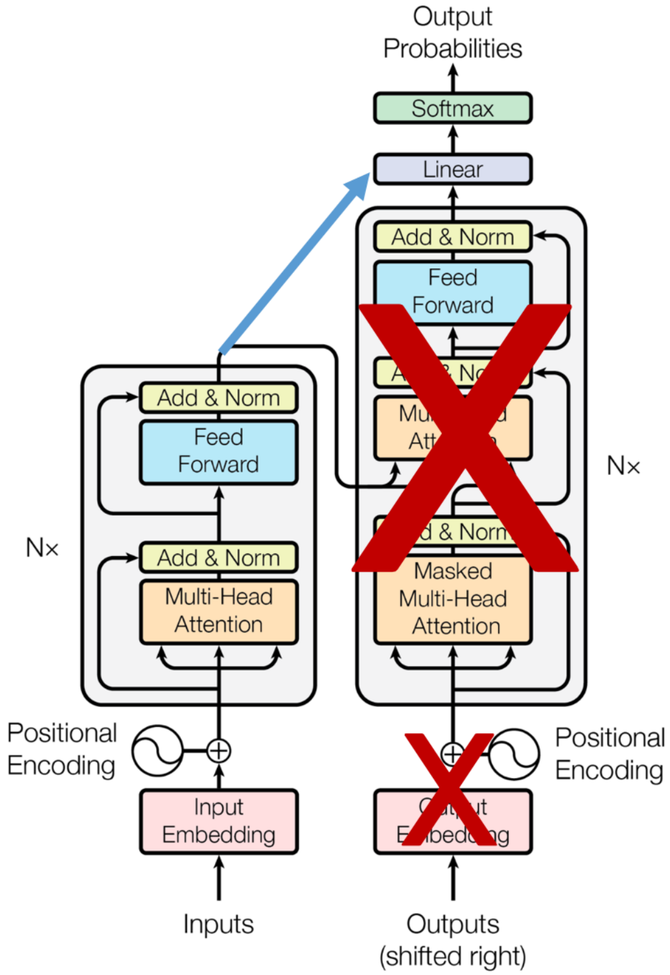
```
[['SUN&L'], ['한국프랜지'], ['케이비아이동국실업'], ['세이브존I&C'], ['무림페이퍼']]
```

04.

BERT로 뉴스 호재악재 분류

➢ BERT(Bidirectional Encoder Representations from Transformers)

✓ 양방향 마스크 언어모델로 문장의 의미를 추출하는데 강점



04.

BERT로 뉴스 호재악재 분류

➢ Fama+LSV

```python
MODEL_NAME = "klue/bert-base"
model = TFBertForSequenceClassification.from_pretrained(MODEL_NAME, num_labels=3, from_pt=True)
tokenizer = BertTokenizer.from_pretrained(MODEL_NAME)
```

executed in 11.4s, finished 13:04:57 2022-06-07

finance_data.csv

- 언어 : 영어, 한국어
- 출처 : Finance Phrase Bank (Malo et al., 2014)
- 데이터 건수 : 4,846건

영어 데이터

- Finance Phrase Bank (Malo et al., 2014)
- 금융 뉴스 데이터에서 4,840여개의 문장 추출
- 16명의 전문지식을 갖춘 연구자들에 의해 수동 라벨링하여 만들었음.
- 감정 라벨 : positive, neutral, negative

한국어 데이터

- 위 데이터를 한국어로 번역 및 검수한 데이터.

```python
data = pd.read_csv('./data/finance_data.csv')
print('총 샘플의 수 :',len(data))
data.head()
```

executed in 39ms, finished 16:55:59 2022-06-24

총 샘플의 수 : 4846

	labels	sentence	kor_sentence
0	neutral	According to Gran, the company has no plans to move all production to Russia, although that is w...	Gran에 따르면, 그 회사는 회사가 성장하고 있는 곳이지만, 모든 생산을 러시아로 옮길 계획이 없다고 한다.
1	neutral	Technopolis plans to develop in stages an area of no less than 100,000 square meters in order to...	테크노폴리스는 컴퓨터 기술과 통신 분야에서 일하는 회사들을 유치하기 위해 10만 평방미터 이상의 면적을 단계적으로 개발할 계획이라고 성명은 밝혔다.
2	negative	The international electronic industry company Elcoteq has laid off tens of employees from its Ta...	국제 전자산업 회사인 엘코텍은 탈린 공장에서 수십 명의 직원을 해고했으며, 이전의 해고와는 달리 회사는 사무직 직원 수를 줄였다고 일간 포스티메스가 보도했다.
3	positive	With the new production plant the company would increase its capacity to meet the expected incre...	새로운 생산공장으로 인해 회사는 예상되는 수요 증가를 충족시킬 수 있는 능력을 증가시키고 원자재 사용을 개선하여 생산 수익성을 높일 것이다.
4	positive	According to the company's updated strategy for the years 2009-2012, Basware targets a long-term...	2009-2012년 회사의 업데이트된 전략에 따르면, Basware는 20% - 40% 범위의 장기적인 순매출 성장을 목표로 하고 있으며, 영업이익률은 순매출액의 10% -...

04.

BERT로 뉴스 호재악재 분류

BERT: fine-tuning 긍정,부정 분류모델

```python
max_seq_len = 64
def convert_examples_to_features(examples, labels, max_seq_len, tokenizer):

    input_ids, attention_masks, token_type_ids, data_labels = [], [], [], []

    for example, label in tqdm(zip(examples, labels), total=len(examples)):
        # input_id는 워드 임베딩을 위한 문장의 정수 인코딩
        input_id = tokenizer.encode(example, max_length=max_seq_len, pad_to_max_length=True)

        # attention_mask는 실제 단어가 위치하면 1, 패딩의 위치에는 0인 시퀀스.
        padding_count = input_id.count(tokenizer.pad_token_id)
        attention_mask = [1] * (max_seq_len - padding_count) + [0] * padding_count

        # token_type_id은 segment 인코딩(한 문장이므로 모두 0)
        token_type_id = [0] * max_seq_len

        input_ids.append(input_id)
        attention_masks.append(attention_mask)
        token_type_ids.append(token_type_id)
        data_labels.append(label)

    input_ids = np.array(input_ids, dtype=int)
    attention_masks = np.array(attention_masks, dtype=int)
    token_type_ids = np.array(token_type_ids, dtype=int)

    data_labels = np.asarray(data_labels, dtype=np.int32)

    return (input_ids, attention_masks, token_type_ids), data_labels
```

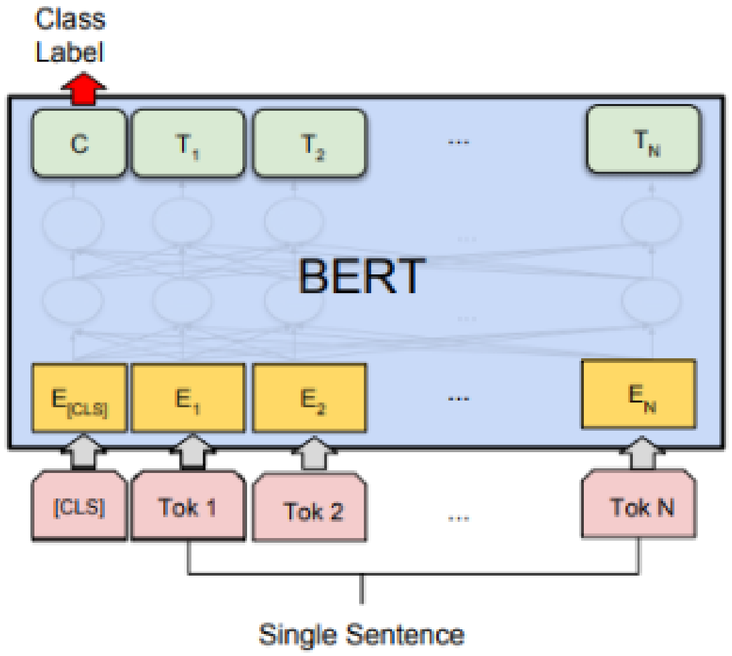
```python
model.evaluate(test_X, test_y, batch_size=1024)
```

executed in 2.16s, finished 11:29:04 2022-06-25

```
1/1 [==============================] - 2s 2s/step - loss: 0.4109 - accuracy: 0.8416

[0.41085386276245117, 0.8416149020195007]
```

Accuracy: 84% 성능을 가진 fine-tuning 모델 생성

한문장에 대한 긍정,부정,중립을 분류하는 fine-tuning



(b) Single Sentence Classification Tasks: SST-2, CoLA

04.

BERT로 뉴스 호재악재 분류

분류결과

2.2.1 긍정

```
df_n.iloc[pos_idx].sample(10)
```

executed in 21ms, finished 17:06:05 2022-06-24

	날짜	뉴스제목
10365	2021-03-30	네이버보다 못 오른 카카오, 5대1 액면분할로 주가 부양 가능할까?
1602	2022-05-27	삼성 금융앱 '모니모' 4월 사용자 수 74만 증가...카카오페이 66만·토스 63...
1941	2022-05-04	카카오, 1분기 영업익 1587억원...광고 줄고 콘텐츠 성장세(상보)
12454	2022-04-27	"폰타나로 시작하는 든든한 하루!" 폰타나, 카카오 쇼핑라이브 첫 방송 진행
8408	2022-05-03	카카오게임즈, 1분기 매출 2663억원...전년比 105%↑
10171	2021-05-07	최고 실적 또 갈아치운 카카오... 모빌리티·페이 '일등 공신'
3389	2022-05-10	서울대병원-카카오헬스케어, 정밀의료 지식은행 구축 위한 업무협약 체결
1967	2022-05-18	더 똑똑해진 카카오 '톡비서 죠르디'...카톡 메시지도 예약 해준다
16975	2022-05-04	[리포트 브리핑]카카오게임즈, '신작 없이도 선방한 1분기' 목표가 110,000원...
17801	2020-08-06	[잠정실적]카카오 2Q 실적, 영업이익 978억원... 전년동기 대비 142% 증가 (연...

2.2.2 부정

```
df_n.iloc[neg_idx].sample(10)
```

executed in 20ms, finished 17:07:07 2022-06-24

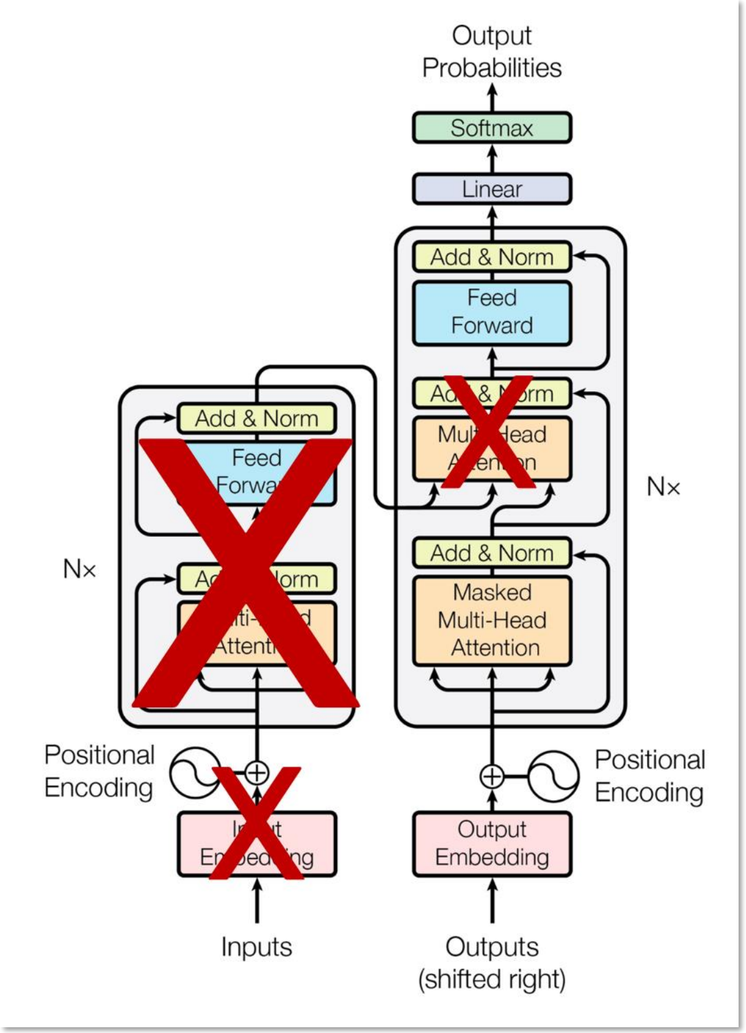
	날짜	뉴스제목
8283	2022-05-19	"바닥 안보이네"...네이버·카카오 신저가
6968	2017-08-01	'돌풍' 카카오뱅크, 실질이용률은 아직 낮아
6832	2017-08-31	카카오뱅크 한 달째 상담·대출 '안돼'...소비자 '답답'
5435	2018-11-13	카카오 카풀・타다 인기몰이...결단 못내리는 정부
5955	2016-02-29	카카오가 헤어숍까지..."배달앱 사태 재현" 우려
2085	2022-05-19	[특징주] 카카오, 투자심리 위축에 또 '신저가'... 8만원도 위태
9469	2022-04-13	주가 힘 못 쓰는 네이버·카카오...실적 부진·연봉 인상이 발목
3032	2022-05-06	고점 대비 반토막 카카오…증권사 목표가 줄줄이 하향
8392	2022-05-03	오버행 발목 잡힌 카카오페이...장중 한때 신저가
17765	2021-11-10	10일, 외국인 코스닥에서 카카오게임즈(-7.21%), 심텍(0.00%) 등 순매수

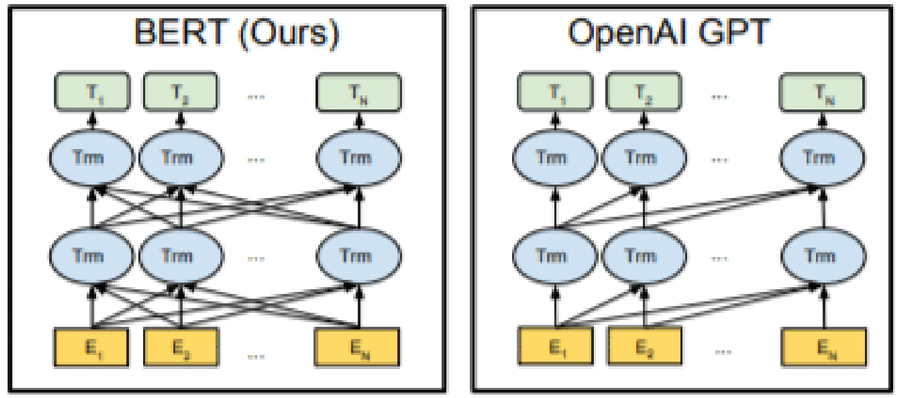
04.

KoGPT을 활용한 챗봇 구현

➢ GPT(Generative Pre-trained Transformer)

- ✓ Tramsformer 구조에서 Decoder 부분만 사용한 일방향모델
- ✓ 문장생성에 강점을 가짐





GPT

어제 카페 갔었어 거기 사람 많더라

BERT

어제 카페 갔었어 ☐ 사람 많더라

04.

KoGPT을 활용한 챗봇 구현

➢ KoGPT2

✓ 일반대화 + 금융대화를 추가하여 fine-tuning

In [26]:

```
Q_TKN = "<usr>"
A_TKN = "<sys>"
BOS = '</s>'
EOS = '</s>'
MASK = '<unused0>'
SENT = '<unused1>'
PAD = '<pad>'
```

executed in 17ms, finished 19:36:37 2022

In [27]:

```
koGPT2_TOKENIZER = PreTrainedTokenizerFast.from_pretrained("skt/kogpt2-base-v2",
            bos_token=BOS, eos_token=EOS, unk_token='<unk>',
            pad_token=PAD, mask_token=MASK)
model = GPT2LMHeadModel.from_pretrained('skt/kogpt2-base-v2')
```

executed in 7.74s, finished 19:36:46 2022-06-15

	Q	A
53	Q	A
54	주식투자 위험한가요?	어떻게 하느냐에 따라 다를 것 같아요.
55	주식보다 저축이 나을까요?	어떻게 하느냐에 따라 다를 것 같아요
56	주식투자 해도 될까?	어떻게 하느냐에 따라 다를 것 같아
57	주식투자 해야하나요?	어떻게 하느냐에 따라 다를 것 같아
58	믿어도 될까요?	완벽하진 않지만 도움이 되실거에
59	예측이 정확한가요?	완벽하진 않지만 도움이 되실거에
60	예측이 믿을만해?	완벽하진 않지만 도움이 되실거에
61	믿어도 돼?	완벽하진 않지만 도움이 되실거에
62	괜찮은 주식 있을까요?	고슴도치에게 물어보세요
63	괜찮은 업종 있을까요?	
64	언제쯤 수익이 날까?	
65	주식 물어봐도 돼?	
66	괜찮은 주식 있을까요?	
67	괜찮은 업종 있을까요?	
68	언제쯤 수익이 날까?	
69	주식 물어봐도 돼?	
70		

Q	A
내 주식 망했어	주식으로
내 주식 떨어졌어	주식으로
내 주식 어떡해?	주식으로
내 주식 망했어	마음이 아
내 주식 떨어졌어	마음이 아
내 주식 어떡해?	마음이 아
내 주식 올랐어	축하해요!
덕분에 수익봤어	축하해요! 다음에도 고슴도치가 도와드릴게요.
주식 대박남!	축하해요! 다음에도 고슴도치가 도와드릴게요.
내 주식 올랐어	축하드려요~ 오늘 파티각?
덕분에 수익봤어	축하드려요~ 오늘 파티각?
주식 대박남!	축하드려요~ 오늘 파티각?

Fine-tuning 결과

```
user > 주식 투자 해볼까?
Chatbot > 어떻게 하느냐에 따라 다를 것 같아요
user > 주식 추천해줘
Chatbot > 고슴도치에 검색해 보세요
user > 예측 결과 어때?
Chatbot > 완벽하진 않지만 도움이 되실거에요
user > 결과 보내줄 수 있어?
Chatbot > 네 전송해 드릴게요
user > 주식 망했어
Chatbot > 주식으로 돈버는사람은 몇명 없대요
user > 주식 대박났어!
Chatbot > 축하해요  다음에도 고슴도치가 도와드릴게요
```

04.

Telegram UI 구현

```python
######## 텔레그램 관련 코드 ########
token = "5403110188:AAEbcgi6cDNmgdRHERhGhprFQgMUHzi-rtI"
id = 5322933876

bot = telegram.Bot(token)
bot1 = TelegramBot(token,id)
info_message = '''안녕하세요 금융챗봇 고슴도치입니다~'''
bot.sendMessage(chat_id=id, text=info_message)

updater = Updater(token=token, use_context=True)
dispatcher = updater.dispatcher
updater.start_polling()

### 챗봇 답장

def handler(update, context): ...



echo_handler = MessageHandler(Filters.text, handler)
dispatcher.add_handler(echo_handler)
##################################
```

```python
# 실시간 주가 정보
def stockinfo_now(corp_name): ...
```

```python
# BERT 긍부정 판별
def bert_clsfy_news(corp,senti_model,
```

```python
# DB에서 정보추출
def DB_info(name,db_type,date):
```

```python
# 재무데이터 AI 추천
def company_recomend(date): ...
```

```python
# 퀀트 기반 종목 추천
def quant_invest_result(date): ...
```

```python
# 텍스트 모델기반 종목 추천
def text_invest_result(month,period_rate,date): ...
```

```python
# 코스피 업종 추천
def kospi_kind_recomend(): ...
```

프로젝트 결과

04.
결과

단위 (%)

수익모델		기간	수익률 평균	코스피수익률	BM대비
AI모델 (주가+재무)		1개월	-6.50	-10.89	4.39
		3개월	29.87	-7.73	37.60
		6개월	-2.01	-22.42	20.41
AI모델 (뉴스)		3개월	15.95	-7.86	23.81
		6개월	36.67	-20.93	57.60
퀀트 전략	볼린저 밴드	3개월	5.51	-8.32	13.83
	마법의 공식		9.51	-8.32	17.83
	Fammal LSV		7.66	-8.32	13.54
	절대 모멘텀		1.31	-8.32	9.63

04.

시연영상

Thank You

Fintech KDT4기 | TEAM C