# 프로젝트 제목

- `yfinance` : 주가 데이터를 가져 오는 라이브러리입니다.
- `pandas` : 데이터를 표 형식으로 쉽게 다루게 해주는 파이썬 도구입니다.
- `numpy` : 숫자 데이터를 배열 형태로 빠르게 계산할 수 있게 도와주는 라이브러리입니다.
- `matplotlib` : 데이터를 선 그래프나 막대 그래프 등 다양한 형태로 시각화할 수 있게 해주는 도구입니다.
- `lightgbm` : **lightgbm**이라는 머신러닝 모델을 사용하기 위한 라이브러리입니다.
- `sklearn` :머신러닝을 위한 다양한 기능들을 모아놓은 종합 도구입니다.
    - `LinearRegression` : 주어진 데이터를 이용해서 직선을 그려서 예측하는 방식입니다.
    - `train_test_split` : 이건 데이터를 학습용과 테스트용으로 나누는 도구입니다.
    - `mean_squared_error` :예측한 값이 실제 값과 얼마나 차이가 나는지를 알려주는 도구입니다.
- `tensorflow` : 인공지능을 쉽게 만들 수 있도록 도와주는 도구입니다.
    - `Sequential` : 층을 차례대로 쌓는 딥러닝 모델
    - `Dense, LSTM` : **LSTM**은 일단 모델 쓰기 위한 도구
    - `MinMaxScaler` :데이터를 0과 1사이로 바꿔주는 도구

```
# 📌 STEP 0: 설치 및 라이브러리 불러오기
!pip install yfinance lightgbm --quiet

import yfinance as yf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

import lightgbm as lgb
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, LSTM
from sklearn.preprocessing import MinMaxScaler

from datetime import datetime, timedelta
```

# yfinance 라이브러리란?

```
# 📌 STEP 1: 삼성전자 주가 데이터 로드
ticker = '005930.KS' #삼전주가 가져오기
end_date = datetime.today().strftime('%Y-%m-%d') #끝 날자 정하기
start_date = (datetime.today() - timedelta(days=3*365)).strftime('%Y-%m-%d') # 시작날짜 정하기

df = yf.download(ticker, start=start_date, end=end_date) #시작, 끝, 삼성전자 로 가져옴
df = df[['Close']].dropna() #종가를 가져오기,na, null값을 삭제, 결측치
df.reset_index(inplace=True) # 주식데이터를 가져오면 인덱스로 변환

target_date = '2025-08-26'#예측하고 싶은 날짜
df.tail() #최근 5일치의 데이터 보기
```

```
/tmp/ipython-input-4046124521.py:6: FutureWarning: YF.download() has changed argument auto_ad
  df = yf.download(ticker, start=start_date, end=end_date) #시작, 끝, 삼성전자 로 가져옴
[*********************100%***********************]  1 of 1 completed
Price   Date         Close
Ticker               005930.KS
  728   2025-08-18     70000.0
  729   2025-08-19     70000.0
  730   2025-08-20     70500.0
  731   2025-08-21     70600.0
  732   2025-08-22     71400.0
```

## LSTM 모델이란?

```
# 📌 STEP 2: LSTM 모델
scaler = MinMaxScaler()  #데이터 정규화
scaled_data = scaler.fit_transform(df[['Close']])  #종가를 0~1사이의 수로 변환

def create_sequences(data, seq_length):
  X, y = [], []
  for i in range(len(data) - seq_length):
      X.append(data[i:i+seq_length])
      y.append(data[i+seq_length])
  return np.array(X), np.array(y)

seq_len = 30
X_lstm, y_lstm = create_sequences(scaled_data, seq_len)

X_train_lstm = X_lstm[:-1]
y_train_lstm = y_lstm[:-1]

model_lstm = Sequential([
    LSTM(50, return_sequences=False, input_shape=(seq_len, 1)),
    Dense(1)
])
model_lstm.compile(optimizer='adam', loss='mse')
model_lstm.fit(X_train_lstm, y_train_lstm, epochs=10, batch_size=16, verbose=1)

last_seq = scaled_data[-seq_len:] #마지막 30일 데이
predicted_scaled = model_lstm.predict(np.expand_dims(last_seq, axis=0))  #예측된 데이터
predicted_price_lstm = scaler.inverse_transform(predicted_scaled)[0][0]  #원래 가격으로 복원

print(f"[LSTM] 예측 종가 (2025-08-22): {predicted_price_lstm:.2f} 원")
```

```
Epoch 1/10
/usr/local/lib/python3.12/dist-packages/keras/src/layers/rnn/rnn.py:199: UserWarning: Do not
  super().__init__(**kwargs)
44/44 ━━━━━━━━━━━━━━━━━━━━ 3s 18ms/step - loss: 0.0697
Epoch 2/10
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 15ms/step - loss: 0.0049
Epoch 3/10
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 11ms/step - loss: 0.0039
Epoch 4/10
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 11ms/step - loss: 0.0031
Epoch 5/10
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 11ms/step - loss: 0.0032
Epoch 6/10
44/44 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - loss: 0.0031
Epoch 7/10
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 11ms/step - loss: 0.0026
Epoch 8/10
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 11ms/step - loss: 0.0026
Epoch 9/10
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 11ms/step - loss: 0.0024
Epoch 10/10
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 11ms/step - loss: 0.0025
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 175ms/step
[LSTM] 예측 종가 (2025-08-22): 71064.80 원
```

## Linear Regression (선형 회귀) 모델이란?

```
# 📌 STEP 3: 선형 회귀 모델
# 날짜 정보로 특성 만들기
df['year'] = df['Date'].dt.year
df['month'] = df['Date'].dt.month
df['day'] = df['Date'].dt.day
df['weekday'] = df['Date'].dt.weekday

X = df[['year', 'month', 'day', 'weekday']]
y = df['Close']

# 예측할 날짜 설정
target = datetime.strptime(target_date, '%Y-%m-%d')
target_input = pd.DataFrame([{
    'year': target.year,
    'month': target.month,
    'day': target.day,
    'weekday': target.weekday()
}])
```

```
# 선형 회귀 학습 및 예측
model_lr = LinearRegression()
model_lr.fit(X.values, y.values)

predicted_price_lr = model_lr.predict(target_input.values)[0]
predicted_price_lr = float(predicted_price_lr)

print(f"[Linear Regression] 예측 종가 ({target_date}): {predicted_price_lr:.2f} 원")
```

```
[Linear Regression] 예측 종가 (2025-08-26): 64661.79 원
/tmp/ipython-input-2156884906.py:25: DeprecationWarning: Conversion of an array with ndim > 0
  predicted_price_lr = float(predicted_price_lr)
```

## LightGBM 모델이란?

```
# 📌 STEP 4: LightGBM 모델
model_lgb = lgb.LGBMRegressor()
model_lgb.fit(X.values, y.values.ravel())  # ravel()로 warning 제거

predicted_price_lgb = float(model_lgb.predict(target_input.values)[0])

print(f"[LightGBM] 예측 종가 (2025-08-22): {predicted_price_lgb:.2f} 원")
```

```
[LightGBM] [Info] Auto-choosing col-wise multi-threading, the overhead of testing was 0.000
You can set `force_col_wise=true` to remove the overhead.
[LightGBM] [Info] Total Bins 55
[LightGBM] [Info] Number of data points in the train set: 733, number of used features: 4
[LightGBM] [Info] Start training from score 63798.618051
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
```

```
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
```

```
# 📌 STEP 5: 최종 비교
print("₩n 예측 요약:")
print(f"LSTM 예측 종가:          {predicted_price_lstm:.2f} 원")
print(f" Linear Regression 예측:   {predicted_price_lr:.2f} 원")
print(f"LightGBM 예측 종가:       {predicted_price_lgb:.2f} 원")
```

```
 예측 요약:
LSTM 예측 종가:          71064.80 원
 Linear Regression 예측:   64661.79 원
LightGBM 예측 종가:       70749.25 원
```

# 결론

## 모델별 예측 결과 비교

| 모델 | 날짜 | 예측값 | 실제값 | 오차 |
|---|---|---|---|---|
| **LSTM** | 250725 | 64603.30 원 | 65,900 원 | 1296.7 원 |
| | 250801 | 66981.09 원 | 68,900 원 | 1918.91 원 |
| | 250808 | 70025.97 원 | 71,800 원 | 1774.03 원 |
| | 250815 | 70550.30 원 | 71,600 원 | 1049.70 원 |

| 모델 | 날짜 | 예측값 | 실제값 | 오차 |
|---|---|---|---|---|
| **선형 회귀** | 250725 | 64628.49 원 | 65,900 원 | 1271.51 원 |
| | 250801 | 64380.93 원 | 68,900 원 | 4519.07 원 |
| | 250808 | 64136.99 원 | 71,800 원 | 7663.01 원 |
| | 250815 | 64325.33 원 | 71,600 원 | 7274.67 원 |
| **LightGBM** | 250725 | 61004.81 원 | 65,900 원 | 4895.19 원 |
| | 250801 | 62798.45 원 | 68,900 원 | 6101.55 원 |
| | 250808 | 62178.25 원 | 71,800 원 | 9621.75 원 |
| | 250815 | 66932.25 원 | 71,600 원 | 4667.75 원 |