

Nama Peserta:

- Azriel Naufal Aulia (NIM : 1301190374)
- Dicky Irianto (NIM : 180155201004)
- Syahrir Aliffudin Muharram (NIM : 24060117130063)
- Fanuel Phalosa Handiono (NIM : A11.2019.11717)

Universitas Host: Universitas Dian Nuswantoro

Kelas: UDINUS-01

Kelompok: 6

Tema Project Kelompok:

Sistem Prediksi Harga Saham berbasis website

Latar belakang masalah :

Salah satu pelajaran kepada para pelaku pasar saham seperti investor dan trader, yang cukup terkenal adalah *“don't try to time the market”*. Pesan itu berasal dari Peter Lynch, seorang manajer investasi terkenal di *wall street* akan kesuksesannya dalam menggandakan uang kliennya dari senilai US\$ 18 Juta menjadi US\$ 14 Miliar selama 23 tahun dia berkarir. Selama beliau berkarir, Peter Lynch berpendapat bahwa merupakan hal yang mustahil untuk memprediksi harga saham dari suatu perusahaan. Karena sifat alami dari pasar saham yang dikenal dengan pergerakan harga yang dinamis, volatile dan non-linier. Prediksi harga saham yang akurat sangat menantang karena banyak faktor (makro dan mikro), seperti politik, kondisi ekonomi global, kejadian tak terduga, kinerja keuangan perusahaan, dan sebagainya.

Tapi, semua ini juga berarti ada banyak data untuk menemukan polanya. Jadi, analis keuangan, peneliti, dan ilmuwan data terus mengeksplorasi teknik analitik untuk mendeteksi tren pasar saham. Yang dimana, terlepas dari volatilitas, harga saham bukan hanya angka yang dihasilkan secara acak. Harga saham dapat dianalisis sebagai urutan data *time series*. Prediksi data *time series* atau *time series forecasting* (memprediksi nilai masa depan berdasarkan nilai historis) merupakan salah satu eksperimen yang masih memiliki ruang eksplorasi yang cukup banyak sehingga mengundang para ilmuwan data untuk mencoba membuat algoritma yang bisa

memprediksi dengan optimal data *time series* kedepannya, seperti prediksi harga saham suatu perusahaan.

Deskripsi Projek :

- Projek merupakan sistem aplikasi web untuk memprediksi harga saham sesuai pilihan dari user.
- Untuk dataset menggunakan API dari alpha vantage dengan mengambil dataset harga saham dari tanggal 1 januari 2015 (bisa diubah dalam code-nya)
- Pada aplikasi web projek kami, terdapat 3 page, homepage yang berisi latar belakang dan deskripsi projek, regression page yang berisi sistem prediksi harga saham dengan berdasarkan input harga opening dari user. Dan deep learning page yang berisi prediksi harga saham di hari esok berdasarkan data historis (misal : 5 harga opening di hari sebelumnya).
- untuk model regresi, menggunakan algoritma linear regression dan support vector machine. Untuk model ini, model akan memprediksi variabel (y) berdasarkan data fitur masukan dari user
- untuk model forecast, menggunakan algoritma Long Short Term Memory (LSTM). Model akan

Kebutuhan masalah sesuai dengan tema/judul project kelompok

- Dataset yang berisi data historis suatu saham, seperti harga pembukaan, harga penutupan dan volume perdagangan berdasarkan tanggal, dan lain-lainnya, dalam jangka waktu tertentu. Pemakaian dataset dibantu dengan menggunakan API dari platform Alpha Vantage.

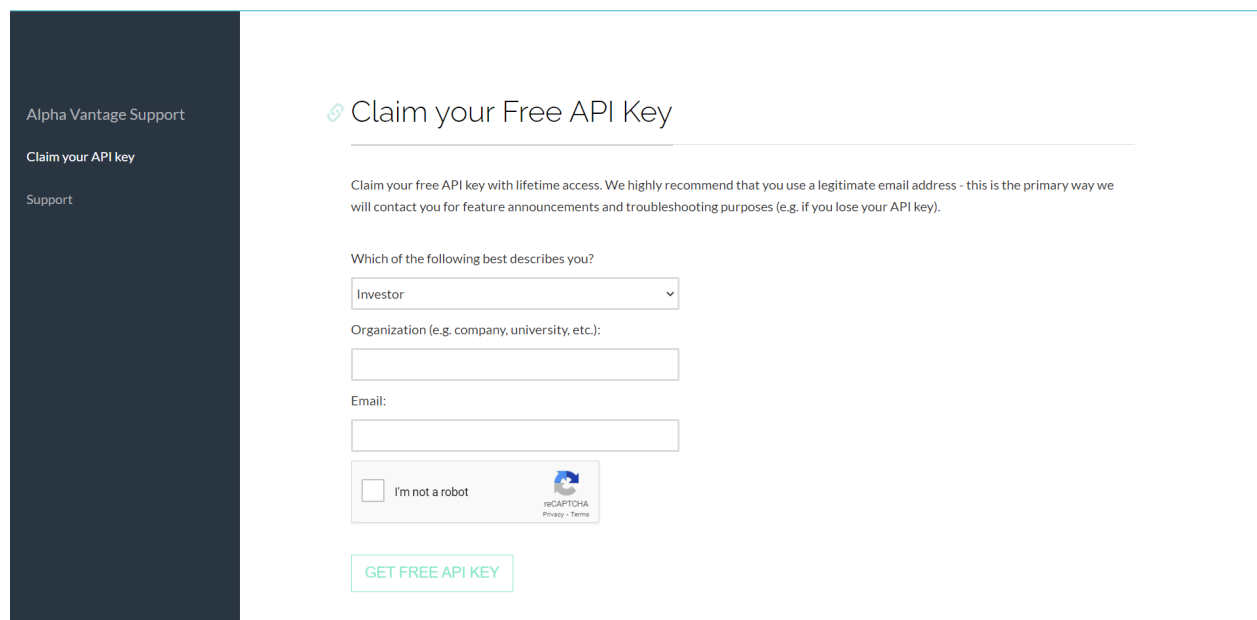
Tahap eksplorasi dan persiapan data untuk membangun model

Pada tahap eksplorasi dan persiapan data yang nanti akan dibutuhkan dalam membangun model pembelajaran mesin, pertama kami akan memanggil library yang dibutuhkan

```
In [99]: import pandas as pd
import requests
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import json
%matplotlib inline
```

Kemudian untuk memanggil data dengan menggunakan API, kami harus membuat API key terlebih dahulu yang nanti digunakan untuk proses request untuk dataset. Dengan menggunakan platform Alpha Vantage, kita bisa membuat API key pada link dibawah :

<https://www.alphavantage.co/support/#api-key>



Kemudian API-key disimpan di dalam variable

```
api_key = "API key di simpan didalam variabel"
```

Membuat fungsi untuk mengambil data historis dari saham yang di input di parameter.

```
# HISTORICAL DATA ~ TIME_SERIES_DAILY
def get_historical_data(symbol, start_date = None):
    api_url = f'https://www.alphavantage.co/query?function=TIME_SERIES_DAILY_ADJUSTED&symbol={symbol}&apikey={api_key}&outputsize=full'
    raw_df = requests.get(api_url).json()
    df = pd.DataFrame(raw_df[f'Time Series (Daily)']).T
    df = df.rename(columns = {'1. open': 'open', '2. high': 'high', '3. low': 'low', '4. close': 'close',
                             '5. adjusted close': 'adj close', '6. volume': 'volume'})
    for i in df.columns:
        df[i] = df[i].astype(float)
    df.index = pd.to_datetime(df.index)
    # menghapus kolom 'dividend amount' dan kolom 'split coefficient'
    df = df.iloc[:, -1].drop(['7. dividend amount', '8. split coefficient'], axis = 1)
    if start_date:
        df = df[df.index >= start_date]
    return df
```

Python

Penjelasan coding diatas lebih lanjut :

Hal pertama yang kami lakukan adalah mendefinisikan fungsi bernama 'get_historic_data' yang mengambil simbol saham / ticker saham ('symbol') sebagai parameter yang diperlukan dan tanggal mulai data historis ('start_date') sebagai parameter optional. Apabila pada parameter 'start_date' tidak di isi maka, data historis akan diambil dari waktu tanggal berdirinya suatu perusahaan dari ticker saham yang dimasukkan. Selanjutnya, kami mengekstrak data historis dalam format JSON menggunakan fungsi 'get' dan menyimpannya ke dalam variabel 'raw_df'. Setelah melakukan beberapa proses untuk membersihkan dan memformat data mentah JSON, mulai dari,

- Mengganti nama kolom
- Mengganti tipe data pada setiap kolom menjadi tipe data float
- Membuat tanggal sebagai index pada dataframe
- Membuang kolom 'dividen amount' dan kolom 'split coefficient'

Setelah itu, kami mengembalikannya dalam bentuk pandas dataframe.

Untuk contoh percobaan kami mencoba mengambil data historis dari saham microsoft, dengan tanggal 2020-01-01 sebagai tanggal mulainya data historis diambil. Kode saham dan Tanggal mulainya data historis bisa diedit di dalam code.

```
msft_hist = get_historical_data('MSFT', '2020-01-01')
msft_hist
```

	open	high	low	close	adj close	volume
2020-01-02	158.78	160.73	158.330	160.62	157.615386	22634546.0
2020-01-03	158.32	159.95	158.060	158.62	155.652798	21121681.0
2020-01-06	157.08	159.10	156.510	159.03	156.055129	20826702.0
2020-01-07	159.32	159.67	157.320	157.58	154.632253	21881740.0
2020-01-08	158.93	160.80	157.950	160.09	157.095300	27762026.0
...
2021-11-15	337.54	337.88	334.034	336.07	335.456697	16723009.0
2021-11-16	335.68	340.67	335.510	339.51	338.890420	20886832.0
2021-11-17	338.94	342.19	338.000	339.12	339.120000	19053380.0
2021-11-18	338.18	342.45	337.120	341.27	341.270000	22463533.0
2021-11-19	342.64	345.10	342.200	343.11	343.110000	21963403.0

477 rows × 6 columns

Mengelola data apabila ada nilai yang hilang (dari hasil dataset yang diperoleh diatas)

```
msft_hist.info()
```

✓ 0.5s

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 477 entries, 2020-01-02 to 2021-11-19
Data columns (total 6 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   open        477 non-null    float64
1   high        477 non-null    float64
2   low         477 non-null    float64
3   close       477 non-null    float64
4   adj close   477 non-null    float64
5   volume      477 non-null    float64
dtypes: float64(6)
memory usage: 26.1 KB
```

Dari hasil diatas, dapat dikatakan dataset yang diperoleh tidak ada nilai yang kosong.

Pemilihan Fitur Untuk pembuatan model

Setelah analisis masing-masing pengertian dari setiap fitur, fitur 'adj close' dibuang karena data tersebut dikatakan cukup sama dengan kolom 'close'. keduanya merepresentasikan harga penutupan pada suatu saham.

```
In [106]: def drop_feature(df):  
          return df.drop(['adj close'], axis = 1)
```

```
In [107]: msft_hist = drop_feature(msft_hist)
```

```
In [108]: msft_hist
```

```
Out[108]:
```

	open	high	low	close	volume
2010-01-04	30.62	31.10	30.590	30.950	38409100.0
2010-01-05	30.85	31.10	30.640	30.960	49749600.0
2010-01-06	30.88	31.08	30.520	30.770	58182400.0
2010-01-07	30.63	30.70	30.190	30.452	50559700.0
2010-01-08	30.28	30.88	30.240	30.660	51197400.0
...
2021-11-15	337.54	337.88	334.034	336.070	16723009.0
2021-11-16	335.68	340.67	335.510	339.510	20886832.0
2021-11-17	338.94	342.19	338.000	339.120	19053380.0
2021-11-18	338.18	342.45	337.120	341.270	22463533.0
2021-11-19	342.64	345.10	342.200	343.110	21963403.0

2993 rows x 5 columns

Untuk fitur yang akan dipakai dalam pembuatan model pada tahap 4, kami memutuskan untuk menggunakan kolom 'open', 'high', 'low', 'close', dan 'volume'. karena ke empat fitur tersebut cukup merepresentasikan harga dari suatu saham.

Normalisasi

Untuk normalisasi kami menggunakan formula normalisasi min-max supaya setiap fitur, nilainya di antara 0 sampai 1. Sehingga pembuatan model akan lebih optimal.

normalisasi

```
In [109]: # normalisasi menggunakan formula normalisasi min - max
def normalise_min_max(df):
    return (df - df.min()) / (df.max() - df.min())
```

```
In [117]: msft_hist = normalise_min_max(msft_hist)
```

```
In [118]: msft_hist
```

```
Out[118]:
```

	open	high	low	close	volume
2010-01-04	0.023564	0.024178	0.024603	0.024805	0.099341
2010-01-05	0.024284	0.024178	0.024760	0.024836	0.135701
2010-01-06	0.024378	0.024116	0.024384	0.024242	0.162738
2010-01-07	0.023596	0.022935	0.023351	0.023249	0.138298
2010-01-08	0.022500	0.023494	0.023508	0.023899	0.140343
...
2021-11-15	0.984040	0.977562	0.974439	0.978007	0.029810
2021-11-16	0.978219	0.986233	0.979059	0.988754	0.043160
2021-11-17	0.988421	0.990957	0.986853	0.987535	0.037282
2021-11-18	0.986043	0.991765	0.984099	0.994252	0.048215
2021-11-19	1.000000	1.000000	1.000000	1.000000	0.046612

2993 rows x 5 columns

Pengelompokan data testing dengan data training

berikut adalah fungsi untuk memisahkan dataset menjadi data training dan data testing menggunakan bantuan library sklearn.

berikut adalah fungsi untuk memisahkan dataset menjadi data training dan data testing dengan bantuan library sklearn.

```
[ ] from sklearn.model_selection import train_test_split
```

```
• y1 = pd.DataFrame(msft_hist['close'])
  y1
```

```
• X_train, X_test, y_train, y_test = train_test_split(msft_hist, y1, train_size = 0.8, shuffle=False)
```

Untuk contoh diatas, data testing sebesar 20% dari dataset (dapat diubah di dalam parameter fungsinya). Dari pembagian data testing dan data training terbagi 4 variabel data:

- X_train : data training berisi fitur data seperti ('open', 'high', 'low', 'volume') sebesar 80% dataset
- X_test : data testing sebesar 20 % dataset
- y_train : label (harga penutupan / 'close') dari X_Train
- y_test : label dari X_test

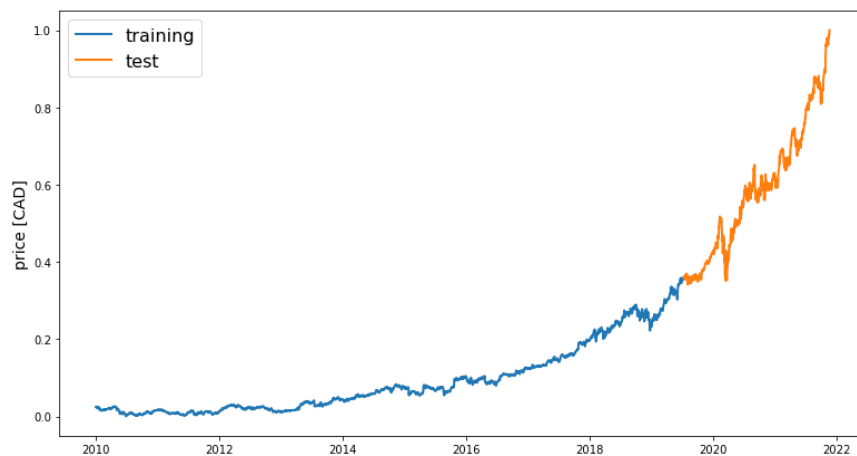
Fungsi untuk membuat visualisasi dataset berdasarkan data testing dan data training

```
In [115]: def line_plot(line1, line2, label1=None, label2=None, title='', lw=2):  
fig, ax = plt.subplots(1, figsize=(13, 7))  
ax.plot(line1, label=label1, linewidth=lw)  
ax.plot(line2, label=label2, linewidth=lw)  
ax.set_ylabel('price [CAD]', fontsize=14)  
ax.set_title(title, fontsize=16)  
ax.legend(loc='best', fontsize=16);
```

Hasil visualisasi nya

visualisasi data testing dengan data training

```
In [116]: target_col = 'close'  
line_plot(train[target_col], test[target_col], 'training', 'test', title='')
```



Type Markdown and LaTeX: α^2

hasil eksplorasi dan persiapan data dari tahap 3 :

data training dan data testing dengan ukuran perbandingan 80% dan 20% dari dataset. untuk masing-masing data, terdapat fitur 'open', 'high', 'low', 'close', dan 'volume'. Dari kelima fitur tersebut cukup merepresentasikan harga dari suatu saham.

Tahap Membuat Model

Projek kami merupakan sistem prediksi harga saham, dimana saham dipilih oleh user. Pada projek prediksi harga saham, kita bisa memprediksi harga saham dengan melihat input data

fitur ke dalam model, disebut dengan model regresi. Namun karena harga saham bukan merupakan angka yang dihasilkan secara acak, melainkan terpengaruh dari banyaknya macam variabel seperti politik, kondisi ekonomi global, kejadian tak terduga, kinerja keuangan perusahaan, dan variabel lainnya.

Sehingga semua ini juga berarti ada banyak data untuk menemukan polanya, bisa dikatakan proyek prediksi harga saham dapat dikategorikan sebagai Prediksi data *time series* atau *time series forecast*. Data *time series* merupakan jenis data yang memiliki titik data yang diindeks atau diurutkan dalam urutan berbasis waktu. Prediksi data *time series* adalah salah satu metode untuk membuat model untuk memprediksi nilai masa depan berdasarkan data deret waktu saat ini dan pola data historis. Untuk pembuatan model prediksi tersebut, kami menggunakan algoritma *Long Short Term Memory* (LSTM).

Untuk prediksi harga saham, dengan melihat input data fitur ke dalam model, kami menggunakan model regresi linear, dan *support vector regression*. Data input kami menggunakan harga pembukaan saham untuk memprediksi harga penutupan di hari yang sama. Alasan dipilihnya harga pembukaan sebagai data masukan ke model karena nilai korelasi yang tinggi antara harga pembukaan dan harga penutupan pada suatu saham.

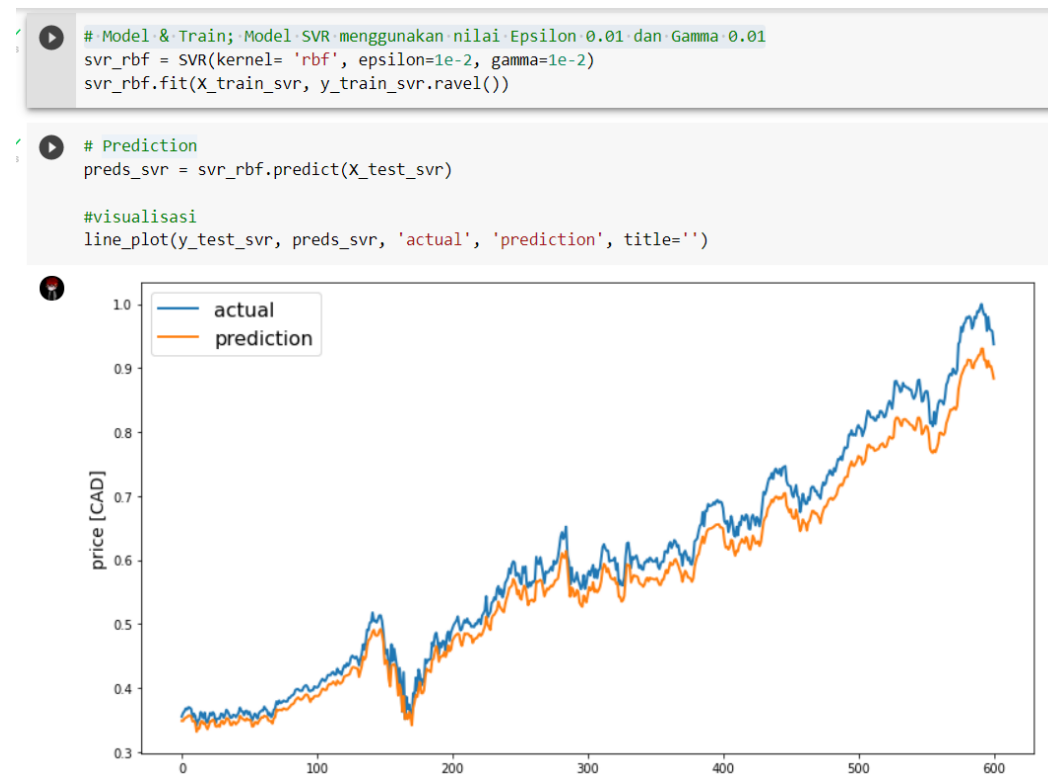
Dalam eksperimen, hasil data prediksi akan kami lakukan evaluasi menggunakan Mean Squared Error (MAE) untuk melihat seberapa optimal model tersebut.

Eksperimen Menggunakan Model Support Vector Regression

Analisis Support Vector Machines (SVM) adalah salah satu dari tool machine learning yang populer untuk klasifikasi dan regresi, mendukung regresi linier dan nonlinier yang dapat kita sebut sebagai SVR. SVR juga dapat digunakan dalam memprediksi data yang berbentuk time series, seperti harga saham dan harga emas. Keuntungan menggunakan SVM adalah SVM bekerja dengan baik ketika ada pemisahan margin yang jelas antar kelas, SVM lebih efektif dalam ruang dimensi tinggi (dalam kasus di mana jumlah dimensi lebih besar dari jumlah sampel), dan SVM menggunakan subset poin *train* dalam fungsi keputusan (disebut *support vector*), sehingga relatif hemat memori.

Parameter yang digunakan dalam membangun model SVR adalah C , ϵ (Epsilon) dan γ (Gamma). Parameter C menyeimbangkan trade-off antara kompleksitas model dan kesalahan empiris. Jika nilai C terlalu besar, dapat menyebabkan *Overfitting*, sedangkan jika nilai C terlalu kecil, dapat menyebabkan *Underfitting*. Parameter ϵ merupakan nilai seberapa besar batas ambang kesalahan yang dapat ditoleransi dan dapat mempengaruhi seberapa halus respon yang dihasilkan SVR. Sedangkan parameter γ menentukan seberapa jauh pengaruh dari model latihan. Apabila nilai γ terlalu besar, maka dapat menyebabkan *Overfitting*, begitu pula sebaliknya, dapat mengakibatkan *Underfitting* jika nilai gamma terlalu kecil. Pada eksperimen ini digunakan nilai default 1 untuk parameter C dan 0.01 untuk parameter ϵ dan γ .

Screenshot Source code model dan visualisasi hasilnya



Screenshot hasil evaluasi model menggunakan MAE (mean squared error)

```
from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test_svr, preds_svr))
print('Mean Squared Error:', metrics.mean_squared_error(y_test_svr, preds_svr))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test_svr, preds_svr)))
```

```
Mean Absolute Error: 0.030066334321603633
Mean Squared Error: 0.0011476297423317807
Root Mean Squared Error: 0.033876684346786076
```

Eksperimen Menggunakan Model Linear Regression

Salah satu model regresi yang paling sederhana dalam pembelajaran mesin. *Simple linear Regression* hanya mempunyai 1 independent variable (x). Walaupun sederhana, algoritma ini merupakan salah satu algoritma yang sangat populer karena *simple* tapi *powerful*. Secara matematis, persamaan dari *Simple Linear Regression* adalah sebagai berikut

$$y = mx + b + e$$

y = dependent variable

m = slope dari garis (persamaan diatas merupakan sebuah garis)

x = independent variable

b = intercept

e = error

Jadi, secara sederhana tujuan dari *Simple Linear Regression* adalah untuk memprediksi nilai dari y dengan mengetahui nilai x dan menemukan nilai **m** dan **b** yang errornya paling minimal. Karena ini merupakan sebuah prediksi, maka persamaan diatas harus ditambahkan nilai error.

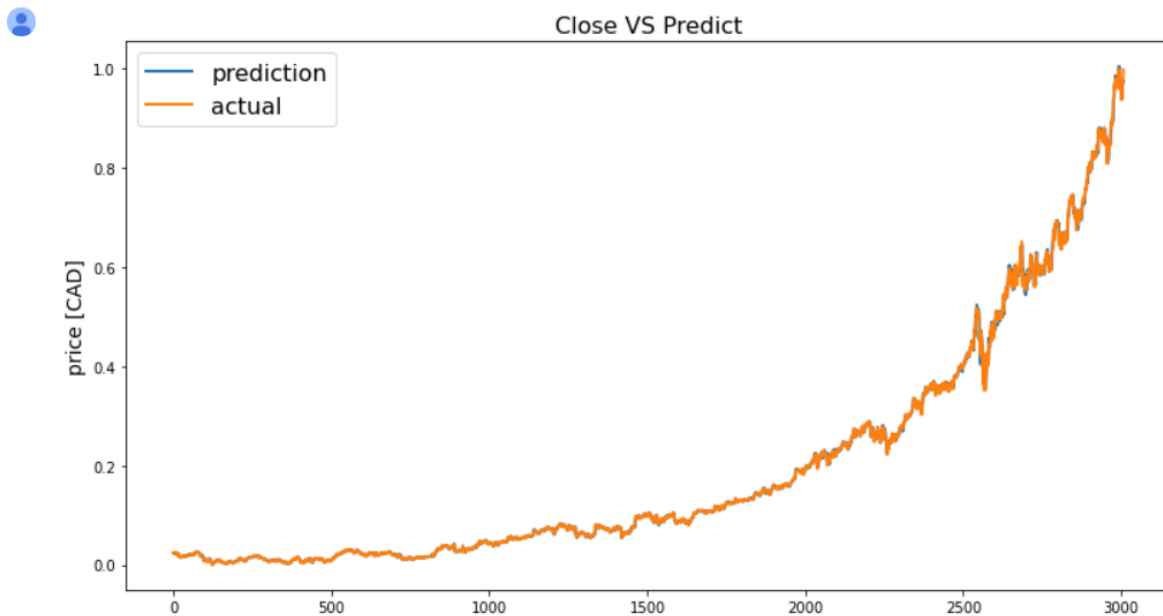
Screenshot source code model

```
model_temp = LinearRegression()
model_temp.fit(x_temp,y_temp)
result = []
i = 0
while i < 3007:
    temp_result = model_temp.predict(x_temp[i].reshape(-1, 1))
    result.append(temp_result[0].tolist())
    i += 1
```

```
[ ] close_arr = y_temp.tolist()
    open_arr = x_temp.tolist()
```

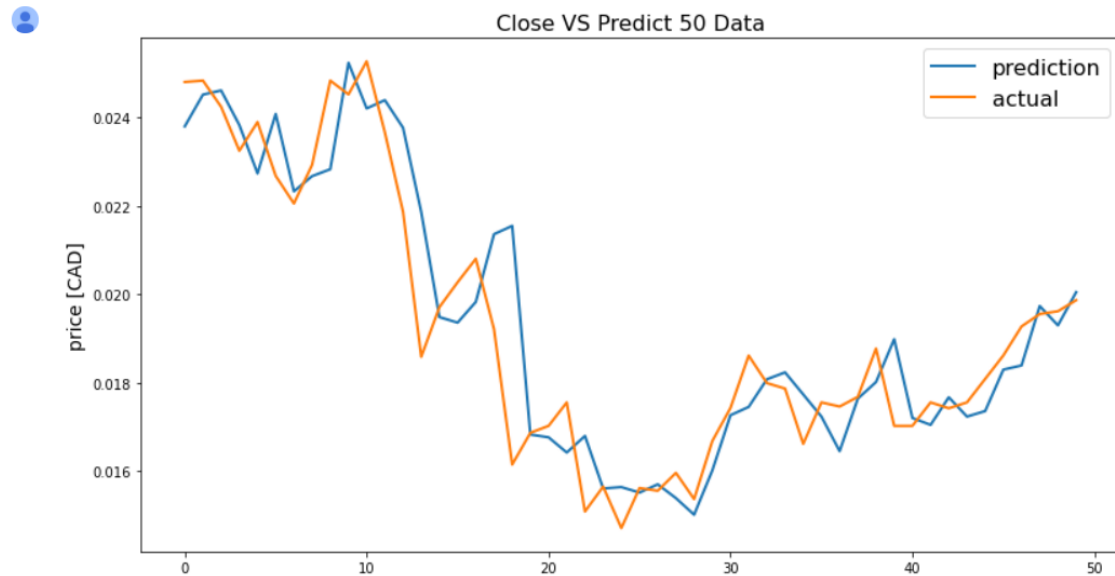
Screenshot visualisasi hasil model dengan nilai sebenarnya

```
line_plot(result, close_arr, 'prediction', 'actual', title='Close VS Predict')
```



50 Data

```
line_plot(result[:50], close_arr[:50], 'prediction', 'actual', title='Close VS Predict 50 Data')
```



Screenshot hasil evaluasi model menggunakan MAE (mean squared error)

```
[301] from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test_lr, y_predict))
print('Mean Squared Error:', metrics.mean_squared_error(y_test_lr, y_predict))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test_lr, y_predict)))
```

```
Mean Absolute Error: 0.006572475858807917
Mean Squared Error: 7.684914347237855e-05
Root Mean Squared Error: 0.008766364324643287
```

Eksperimen Menggunakan Model LSTM (Long Short term Memory)

Berbeda dengan model yang lainnya, model *support vector regression* dan model regresi linear sederhana yang digunakan untuk memprediksi harga saham (harga penutupan) di hari yang sama, Disini LSTM digunakan untuk memprediksi harga saham di masa depan berdasarkan harga saham di hari sebelumnya. Model LSTM (Long Short Term Memory) adalah model yang cukup optimal dalam melakukan model deret waktu atau data yang bersifat *time-series*. Model LSTM memiliki 5 komponen penting yang memungkinkannya untuk memodelkan data jangka panjang dan jangka pendek. Antara lain :

- Cell state

Ini mewakili memori internal sel yang menyimpan memori jangka pendek dan memori jangka panjang.

- Hidden state

Mengeluarkan status informasi yang dihitung dari *current input*, *hidden state* sebelumnya dan *current cell input* yang akhirnya digunakan untuk memprediksi harga pasar saham di masa depan. Selain itu, *hidden state* dapat memutuskan untuk hanya mengambil memori jangka pendek atau jangka panjang atau kedua jenis memori yang disimpan dalam *cell state* untuk membuat prediksi berikutnya.

- Input gate

Memutuskan berapa banyak informasi dari *current input* mengalir ke *cell state*.

- Forget gate

Memutuskan berapa banyak informasi dari *current input* dan *cell state* sebelumnya yang akan mengalir ke *cell state* yang baru.

- Output gate

Memutuskan berapa banyak informasi dari *cell state* saat ini mengalir ke *hidden state*, sehingga jika diperlukan, LSTM hanya dapat memilih memakai ingatan jangka panjang atau ingatan jangka pendek

Screenshot source code

```
#membangun model lstm
model = Sequential()

model.add(LSTM(units=40, return_sequences=True))
model.add(Dropout(0.15))

model.add(LSTM(units=40, return_sequences=True))
model.add(Dropout(0.15))

model.add(LSTM(units=40))
model.add(Dropout(0.15))

model.add(Dense(1))
#prediksi dengan model lstm

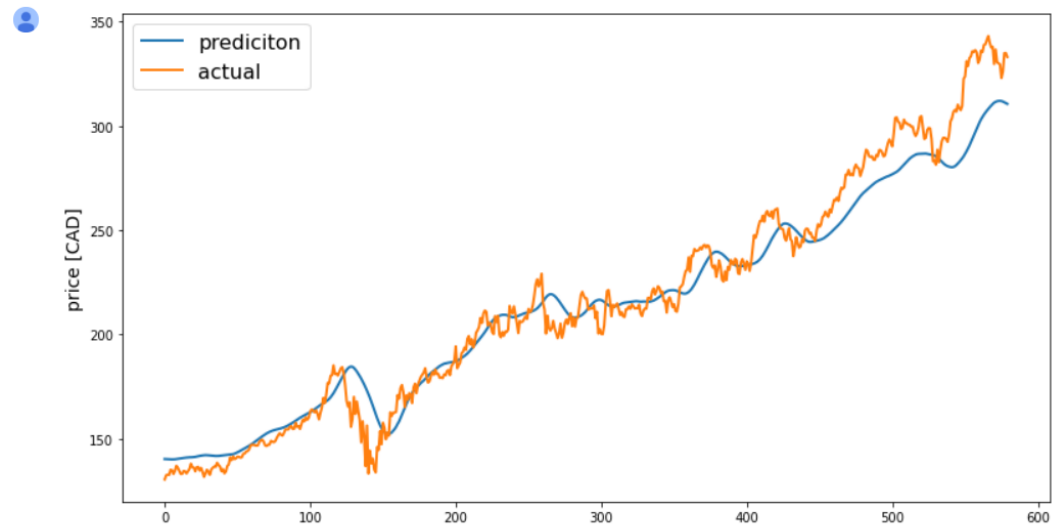
model.compile(optimizer='adam', loss='mean_squared_error')
model.fit(X_train_lstm, y_train_lstm, epochs=20, batch_size=100)
```

Epoch 1/20
25/25 [=====] - 7s 44ms/step - loss: 0.0042
Epoch 2/20
25/25 [=====] - 1s 41ms/step - loss: 3.6676e-04
Epoch 3/20
25/25 [=====] - 1s 41ms/step - loss: 2.6354e-04
Epoch 4/20
25/25 [=====] - 1s 40ms/step - loss: 2.0090e-04
Epoch 5/20
25/25 [=====] - 1s 41ms/step - loss: 2.0090e-04

Screenshot visualisasi hasil model dengan nilai sebenarnya

```
# menggabungkan hasil prediction ke dataframe
lstm_predictions = model.predict(X_test_lstm)

lstm_predictions = scaler.inverse_transform(lstm_predictions)
y_test_lstm = scaler.inverse_transform(y_test_lstm.reshape(-1,1))
# visualisasi prediksi dan actual values
line_plot(lstm_predictions, y_test_lstm, 'prediciton', 'actual', title='')
```



Screenshot hasil evaluasi model menggunakan MAE (mean Vector regression)

```
[231] #hitung nilai MAE
      preds = model.predict(X_test_lstm)
      mean_absolute_error(preds, y_test_lstm)
```

0.02727040686496983

Kesimpulan

Berikut adalah hasil evaluasi masing-masing model menggunakan evaluasi *Mean Squared Error* (MAE):

eksperimen	Nilai MAE
Eksperimen Menggunakan Model Support Vector Regression	0.030066334321603633
Eksperimen Menggunakan Model LSTM (Long Short term Memory)	0.02727040686496983
Eksperimen Menggunakan Model Linear Regression	0.006572475858807917

Dengan menggunakan algoritma mean squared error, didapat model yang paling optimal adalah model regresi linear, namun beberapa faktor yang harus diperhatikan adalah semakin kecil perbedaan antara nilai dengan prediksi maka kemungkinan model tersebut mengalami overfit. Oleh karena itu, dalam aplikasi web nantinya tetap akan menggunakan kedua model diatas sebagai pembanding.

Untuk hasil prediksi menggunakan model LSTM, walaupun hasil evaluasi model tidak sekecil menggunakan model regresi linear. Model LSTM tetap menjadi model yang paling optimal dalam memprediksi data *time-series* (dalam kasus kami, harga saham). Dikarenakan harga saham yang dipengaruhi dari banyaknya variabel sehingga sulit untuk menemukan satu variabel yang paling optimal dalam memprediksi harga saham

Tahap membuat model menjadi aplikasi web

Dalam mengaplikasikan model yang telah dibuat, kami menggunakan web framework streamlit, karena penulisan codenya yang cukup mudah. Berikut Screenshot UI dari web aplikasi kami :

×

Select page

Homepage

Homepage

Regression page

forecast page

☐ show prediction

Projek prediksi harga saham menggunakan machine learning

Latar Belakang :

Salah satu pelajaran kepada para pelaku pasar saham seperti investor dan trader, yang cukup terkenal adalah "don't try to time the market". Pesan itu berasal dari Peter Lynch, seorang manajer investasi terkenal di wall street akan kesuksesannya dalam menggandakan uang kliennya dari senilai US\$ 18 Juta menjadi US\$ 14 Miliar selama 23 tahun dia berkarir. Selama beliau berkarir, Peter Lynch berpendapat bahwa merupakan hal yang mustahil untuk memprediksi harga saham dari suatu perusahaan. Karena sifat alami dari pasar saham yang dikenal dengan pergerakan harga yang dinamis, volatile dan non-linier. Prediksi harga saham yang akurat sangat menantang karena banyak faktor (makro dan mikro), seperti politik, kondisi ekonomi global, kejadian tak terduga, kinerja keuangan perusahaan, dan sebagainya.

Tapi, semua ini juga berarti ada banyak data untuk menemukan polanya. Jadi, analisis keuangan, peneliti, dan ilmuwan data terus mengeksplorasi teknik analitik untuk mendeteksi tren pasar saham. Yang dimana, terlepas dari volatilitas, harga saham bukan hanya angka yang dihasilkan secara acak. Harga saham dapat dianalisis sebagai urutan data time series. Prediksi data time series atau time series forecasting (memprediksi nilai masa depan berdasarkan nilai historis) merupakan salah satu eksperimen yang masih memiliki ruang eksplorasi yang cukup banyak sehingga mengundang para ilmuwan data untuk mencoba membuat algoritma yang bisa memprediksi dengan optimal data time series kedepannya, seperti prediksi harga saham suatu perusahaan.

×

Select page

Regression page


Select model

support vector regression

☐ show prediction

Hasil prediksi model

prediction Data Visualization



input your stock open price

predict



Link presentasi projek :

<https://youtu.be/gdHe8O7fcBM>

Link github source code :

<https://github.com/azrielnaufal/tugas-akhir-kelas-microcredential-data-science>