Stock Price Report

Overall

Saham dapat diartikan sebagai tanda penyertaan modal seseorang atau pihak (badan usaha) pada suatu perusahaan atau perseroan [1]. Indonesia sendiri memiliki pasar sahamnya sendiri yang dinamakan Bursa Efek Jakarta yang digabungkan dengan Bursa Efek Surabaya. Pada projek ini, mencoba memprediksi harga pasar saham menggunakan model machine learning serta mengetahui proses pengambilan data dari database. Dataset yang akan digunakan berasal dari saham apple selama 10 tahun yang didapatkan dari website Kaggle [2]. Software yang digunakan antara lain Pentaho, PostgreSQL (Pg Admin 4), JupyterNotebook, Microsoft Excel dengan model machine learning yang akan digunakan antara lain Decision Tree Regressor, Support Vector Regressor dan Convolutional Neural Network.

Business Understanding

Pada pernyataan yang telah dijelaskan, sehingga masalah yang diangkat adalah

1. Bagaimana proses pengambilan data dari database?
2. Bagaimana performa model machine learning yang digunakan?

Tujuan dari masalah yang diangkat adalah

1. Mengetahui proses pengambilan data dari database
2. Mengetahui performa model machine learning yang digunakan

Solusi yang dapat dilakukan

1. Menggunakan korelasi antar fitur pada dataset menggunakan library sklearn dan heatmap seaborn
2. Mengevaluasi hasil model menggunakan MSE

Data Understanding

Dataset yang digunakan memiliki 2518 baris dan 6 fitur, antara lain.

- Date: Tanggal harga saham dibuka
- Close: Harga terakhir/penutup saham pada hari tersebut
- Volume: Jumlah saham yang diperdagangkan pada hari tersebut.
- Open: Pembukaan harga saham pada hari tersebut
- High: Harga tertinggi sebuah saham pada hari tersebut

- Low: Harga terendah sebuah saham pada hari tersebut.

Dari 6 fitur, fitur High akan digunakan sebagai **label** dikarenakan untuk mengetahui harga tertinggi sebuah saham pada hari tersebut. Pada tahapan ini akan dilakukan proses ETL dari database PostgreSQL dan Pentaho. Banyangkan dataset yang dimiliki berada dalam sebuah database perusahaan dan ingin dilakukan pengambilan data untuk dilakukan proses lebih lanjut, disitulah ETL digunakan. ETL adalah proses pengumpulan data dari berbagai macam sumber dengan tujuan untuk memperoleh data yang berkualitas [3]. Berikut proses ETL yang dilakukan.

1. **Pembuatan Database**

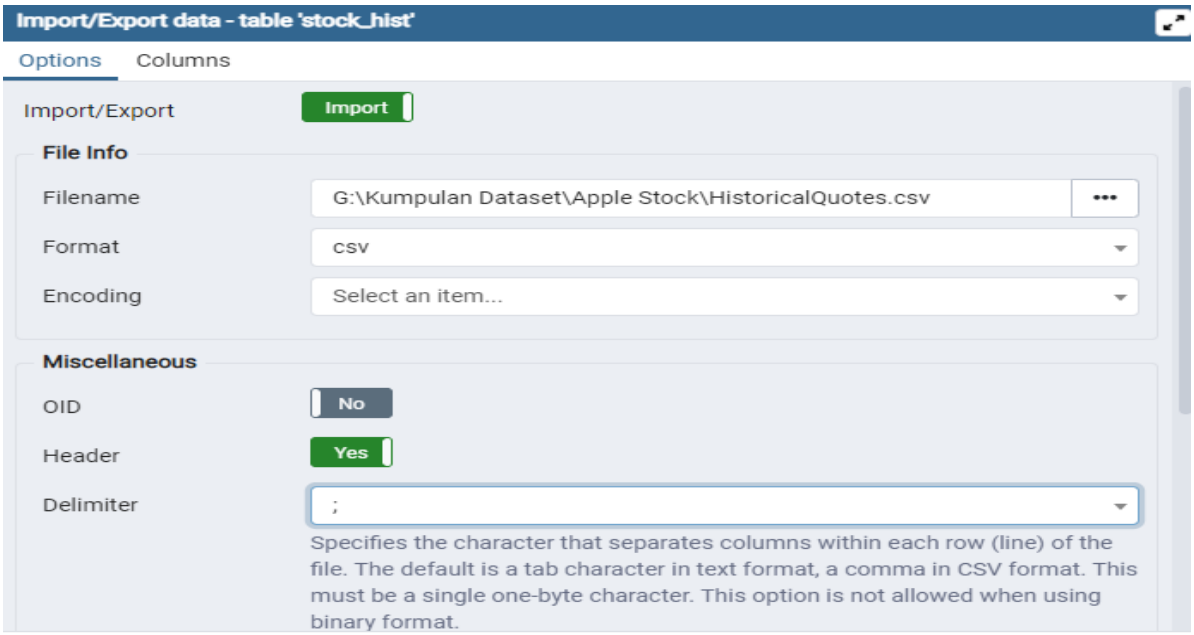
```
CREATE DATABASE "Stock_market"
    WITH
    OWNER = postgres
    ENCODING = 'UTF8'
    CONNECTION LIMIT = -1;
```

2. **Pembuatan Tabel dan Fitur**

```
CREATE TABLE IF NOT EXISTS public.stock_hist
(
    date date,
    close character(10),
    volume integer,
    open character(10),
    high character(10),
    low character(10)
);

ALTER TABLE public.stock_hist
    OWNER to postgres;
```

3. **Import Dataset dan Hasil Import**



Gambar 1 Import data ke PostgreSQL

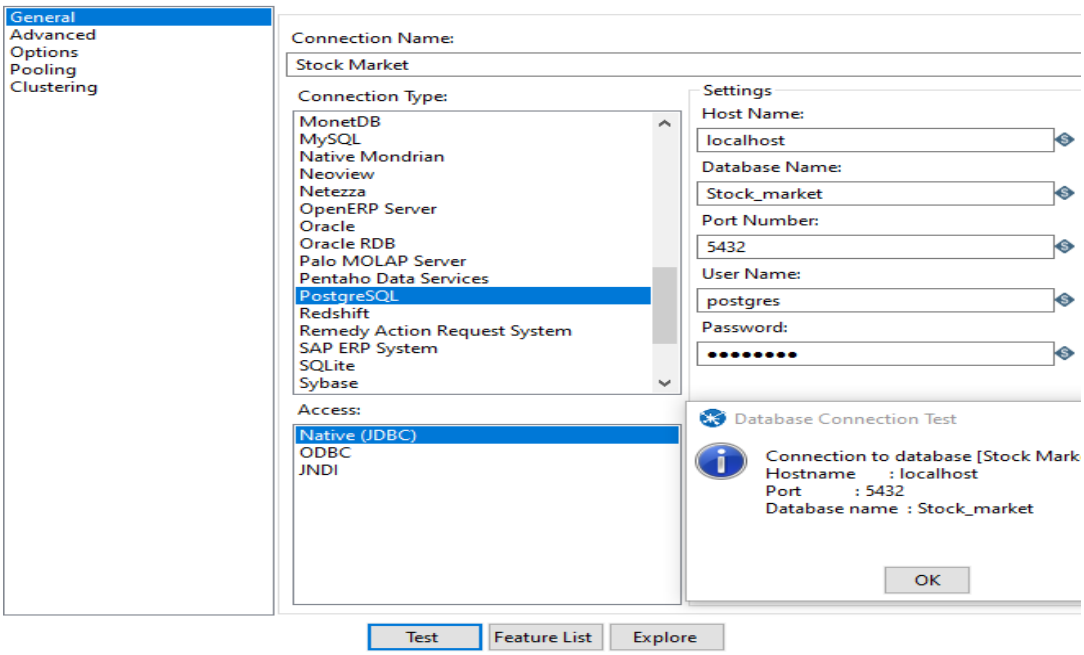
SELECT * FROM stock_hist LIMIT 5

Data Output

	date date	close character (10)	volume integer	open character (10)	high character (10)	low character (10)
1	2020-02-28	[...] $273.36	106721200	[...] $257.26	[...] $278.41	[...] $256.37
2	2020-02-27	[...] $273.52	80151380	[...] $281.1	[...] $286	[...] $272.96
3	2020-02-26	[...] $292.65	49678430	[...] $286.53	[...] $297.88	[...] $286.5
4	2020-02-25	[...] $288.08	57668360	[...] $300.95	[...] $302.53	[...] $286.13
5	2020-02-24	[...] $298.18	55548830	[...] $297.26	[...] $304.18	[...] $289.23

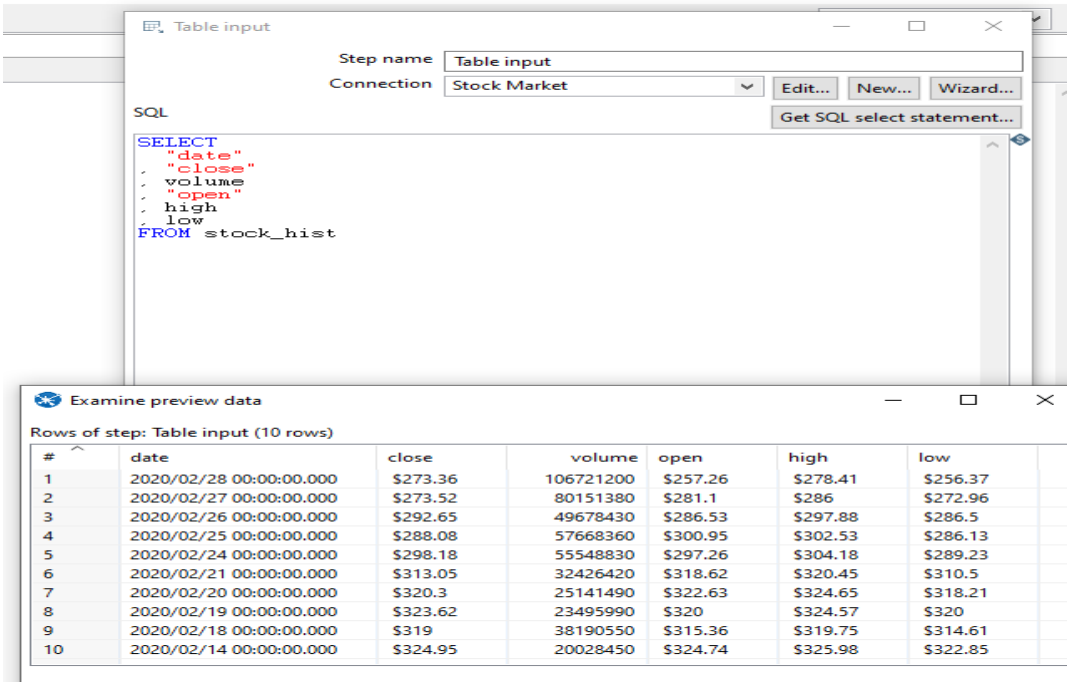
4. Pengambilan Data dari Database Menggunakan Pentaho

Pertama yang perlu dilakukan adalah melakukan connection ke PostgreSQL. Untuk melakukan connection ke PostgreSQL dari Pentaho diperlukan sebuah file bernama **postgresql-42.5.0.jar**. File tersebut tergantung dari versi Java yang diinstal dan pada projek ini versi Java yang terinstall adalah JDK 8. File yang telah di download akan diletakkan pada folder lib pada Pentaho. Setelah diletakkan, maka akan dilakukan connection ke PostgreSQL.



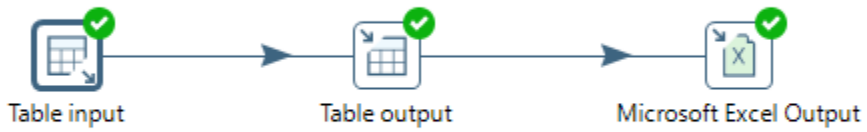
Gambar 2 Connection Pentaho ke PostgreSQL

Setelah melakukan koneksi, tahapan selanjutnya adalah pengambilan data. Pengambilan data dilakukan pada fitur design dengan menggunakan fungsi **Table Input**. Pada Table Input, pilih koneksi database yang ingin diperoleh datanya dan klik **Get SQL select statement.**



#	date	close	volume	open	high	low
1	2020/02/28 00:00:00.000	$273.36	106721200	$257.26	$278.41	$256.37
2	2020/02/27 00:00:00.000	$273.52	80151380	$281.1	$286	$272.96
3	2020/02/26 00:00:00.000	$292.65	49678430	$286.53	$297.88	$286.5
4	2020/02/25 00:00:00.000	$288.08	57668360	$300.95	$302.53	$286.13
5	2020/02/24 00:00:00.000	$298.18	55548830	$297.26	$304.18	$289.23
6	2020/02/21 00:00:00.000	$313.05	32426420	$318.62	$320.45	$310.5
7	2020/02/20 00:00:00.000	$320.3	25141490	$322.63	$324.65	$318.21
8	2020/02/19 00:00:00.000	$323.62	23495990	$320	$324.57	$320
9	2020/02/18 00:00:00.000	$319	38190550	$315.36	$319.75	$314.61
10	2020/02/14 00:00:00.000	$324.95	20028450	$324.74	$325.98	$322.85

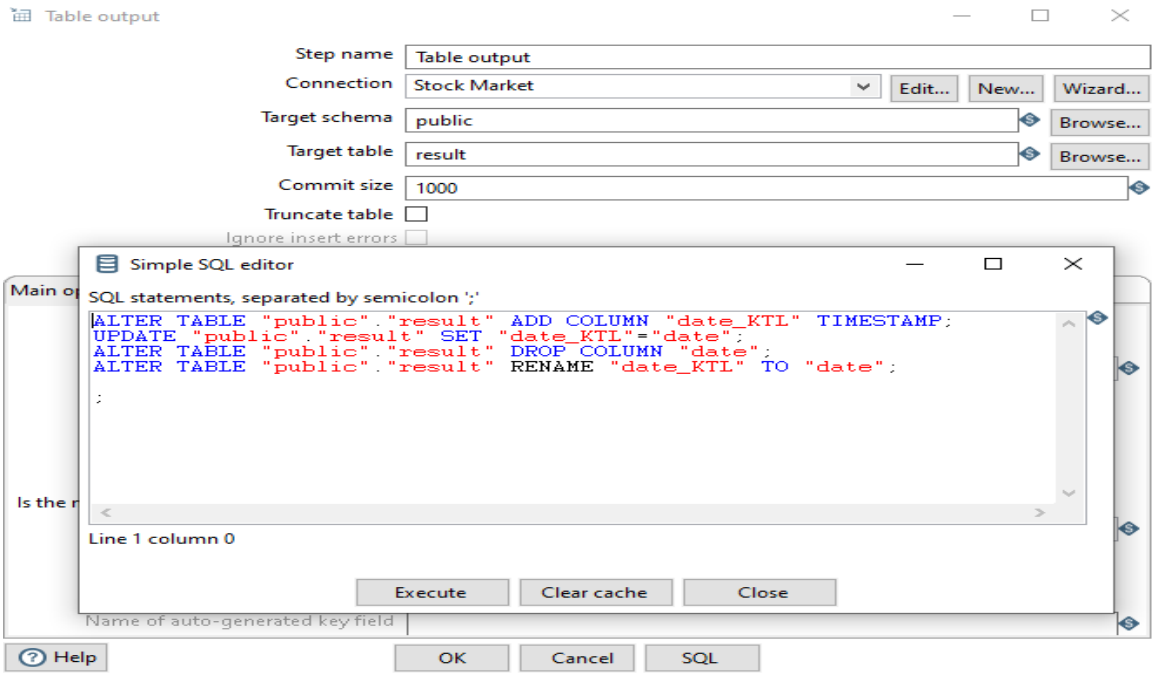
Gambar 3. Tabel Input



Gambar 4. Rangkaian ETL

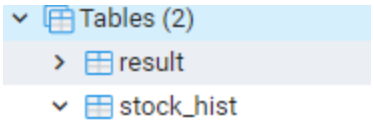
Rangkaian diatas merupakan rangkaian ETL sederhana yang hanya sekedar pengambilan data dari sebuah database. Hal ini dikarenakan proses processing data akan dilakukan pada bahasa program Python. Namun akan dijelaskan singkat Table Output dan Microsoft Excel Output.

Table Output berfungsi sebagai pembentukan tabel baru pada database terkait. Dengan mengatur parameter seperti pada gambar dibawah ini.



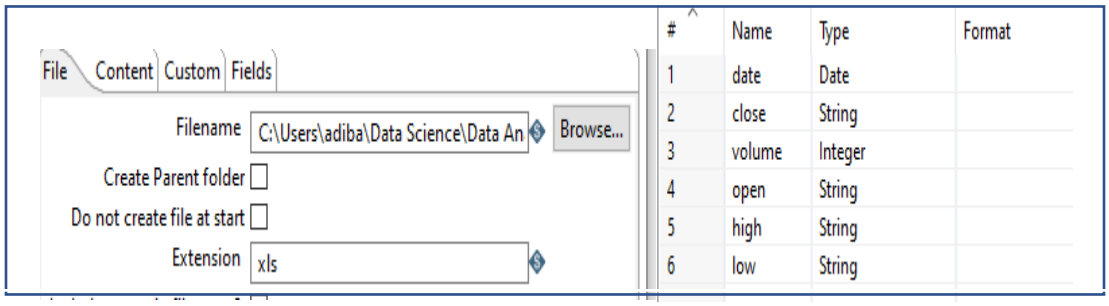
Gambar 5. Parameter Table Output

Step name berguna untuk menamai step pada pentaho, connection sebagai database terkait, Target schema sebagai schema table dimana schema table yang digunakan pada projek ini adalah public, dan Target table sebagai nama table yang akan dibuat pada database Stock Market.



Gambar 6 Hasil Table Output

Microsoft Excel Output digunakan sebagai file output dalam format xls. Dengan mensetting lokasi hasil file dan kolom yang digunakan.



Gambar 7 Parameter Microsoft Excel Output

Gambar 8 Hasil Microsoft Excel Output

Data Preparation

Tahapan yang dilakukan pada data preparation antara lain Exploratory Data Analisis dan Preprocessing

```
df.head()
```

	date	close	volume	open	high	low
0	2020-02-28	$273.36	106721200	$257.26	$278.41	$256.37
1	2020-02-27	$273.52	80151380	$281.1	$286	$272.96
2	2020-02-26	$292.65	49678430	$286.53	$297.88	$286.5
3	2020-02-25	$288.08	57668360	$300.95	$302.53	$286.13
4	2020-02-24	$298.18	55548830	$297.26	$304.18	$289.23

Gambar 9 Tampilan 5 dataset pertama

Exploratory Data Analisis (EDA)

Dataset tidak memiliki missing value pada tiap fitur

```
df.isna().sum()
```

```
date      0
close     0
volume    0
open      0
high      0
low       0
dtype: int64
```

Gambar 10 Missing value

Dataset memiliki 1 tipe datetime, 4 tipe object, dan 1 tipe integer. Jika dilihat 4 tipe object seharusnya float, untuk merubah tipe object ke tipe float akan dilakukan penghapusan symbol '$' pada fitur-fitur tersebut.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2518 entries, 0 to 2517
Data columns (total 6 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   date    2518 non-null   datetime64[ns]
 1   close   2518 non-null   object
 2   volume  2518 non-null   int64
 3   open    2518 non-null   object
 4   high    2518 non-null   object
 5   low     2518 non-null   object
dtypes: datetime64[ns](1), int64(1), object(4)
memory usage: 118.2+ KB
```

Gambar 11 Tipe fitur

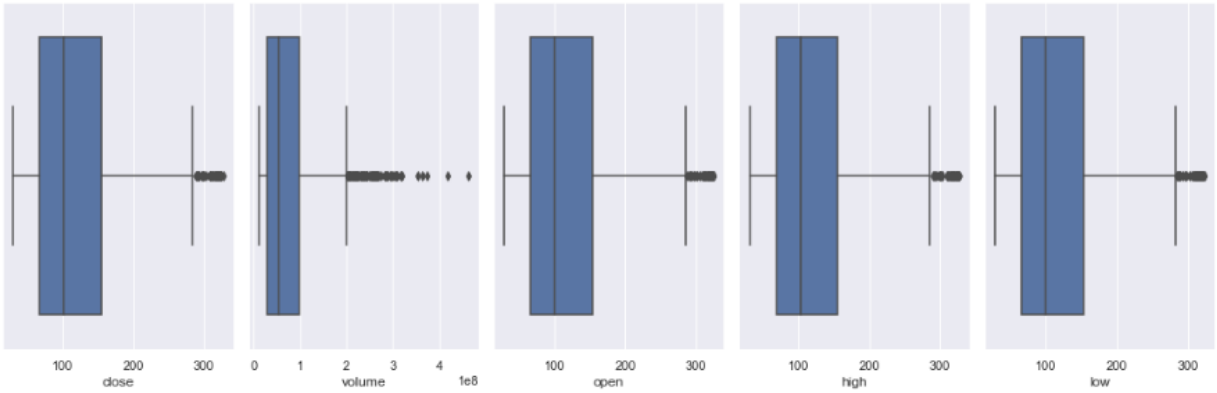
	date	close	volume	open	high	low
0	2020-02-28	273.36	106721200	257.26	278.41	256.37
1	2020-02-27	273.52	80151380	281.1	286	272.96
2	2020-02-26	292.65	49678430	286.53	297.88	286.5
3	2020-02-25	288.08	57668360	300.95	302.53	286.13
4	2020-02-24	298.18	55548830	297.26	304.18	289.23

Gambar 12 Hasil removes symbol

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2518 entries, 0 to 2517
Data columns (total 6 columns):
 #   Column  Non-Null Count  Dtype         
---  ------  --------------  -----         
 0   date    2518 non-null   datetime64[ns]
 1   close   2518 non-null   float64       
 2   volume  2518 non-null   int64         
 3   open    2518 non-null   float64       
 4   high    2518 non-null   float64       
 5   low     2518 non-null   float64       
dtypes: datetime64[ns](1), float64(4), int64(1)
memory usage: 118.2 KB
```

Gambar 13 Tipe data baru pada fitur

Tipe data yang telah dirubah akan dilakukan pengecekan outlier pada dataset



Dapat dilihat bahwa tiap-tiap fitur memiliki outlier masing dan jika dihitung didapatkan sebagai berikut

```
close     42
volume    96
open      41
high      42
low       42
dtype: int64
```

Namun pada projek ini, tidak akan dilukan pembersihan outlier dan dilakukan split dataset menjadi train dan test

```
training_data = df[:2226]
testing_data = df[2226:]
```

Setelah dilakukan pemisahan, diperlukan normalisasi. Hal ini dilakukan karena untuk menyamakan distribusi pada tiap data train dan testing dengan testing hanya dilakukan transform tanpa fit.

```
scaler = MinMaxScaler()
scaler_training = scaler.fit_transform(training_data)
scaler_testing = scaler.transform(testing_data)
```

```
X_train = []
y_train = []

for i in range(len(scaler_training) - 6):
    X_train.append(scaler_training[i:i+6])
    y_train.append(scaler_training[i,0])
```

```
X_train, y_train = np.array(X_train), np.array(y_train)
```

```
X_train.shape, y_train.shape
```

```
((2220, 6, 1), (2220,))
```

```
X_testing = np.array(X_testing)
X_testing = np.reshape(X_testing,(X_testing.shape[0], X_testing.shape[1], 1))
X_testing.shape
```

```
(286, 6, 1)
```

```
y_testing.shape
```

```
(286, 1)
```

Jika dilakukan normalisasi terlebih dahulu, setelah itu split dataset akan menyebabkan persamaan semua distribusi dan hal tersebut dapat menyebabkan bias pada distribusi data. Berikut visualisasi split dataset

Setelah data prepation selesai dilakukan, tahapan selanjutnya adalah modeling.

Modeling & Evaluation

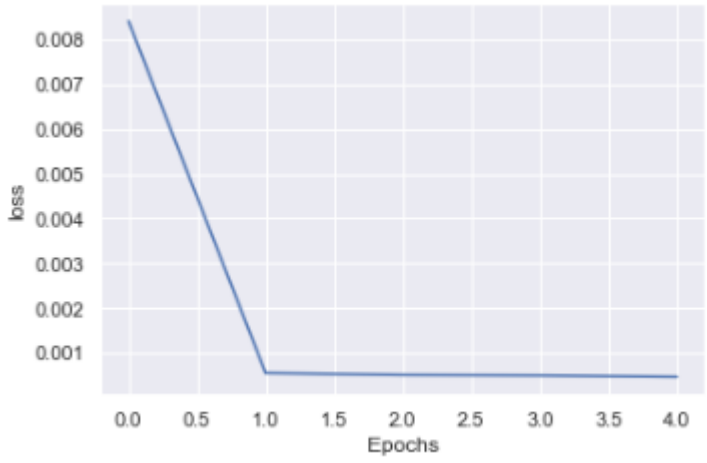
Model yang digunakan pada projek ini adalah LSTM dengan optimasi Adam, loss MAE dan metrics RMSE

```
model_lstm = tf.keras.Sequential([
    tf.keras.layers.LSTM(64, return_sequences = True, input_shape = (X_train.shape[1],1)),
    tf.keras.layers.LSTM(32, return_sequences = False),
    tf.keras.layers.Dense(30, activation = 'relu'),
    tf.keras.layers.Dense(1)
])
model_lstm.summary()
```

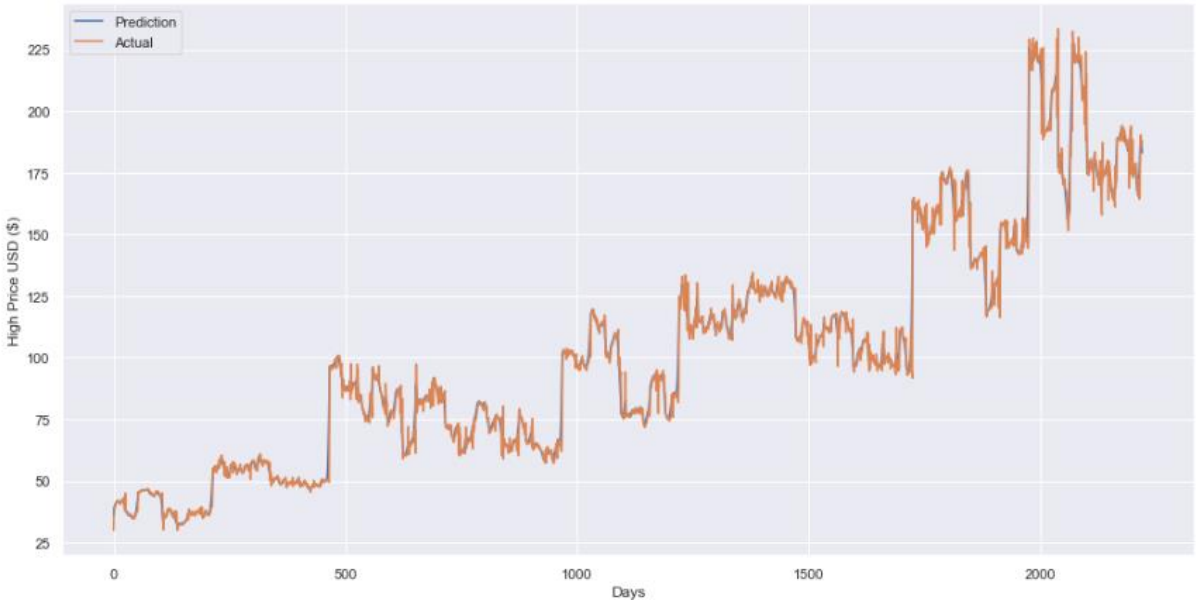
```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 6, 64)             16896
_________________________________________________________________
lstm_1 (LSTM)                (None, 32)                12416
_________________________________________________________________
dense (Dense)                (None, 30)                990
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 31
=================================================================
Total params: 30,333
Trainable params: 30,333
Non-trainable params: 0
```

```
model_lstm.compile(loss=MeanSquaredError(), optimizer=Adam(learning_rate=0.0001), metrics=[RootMeanSquaredError()])
```

Menggunakan epochs = 5, didapatkan loss sebagai berikut



Berikut hasil predict dari sisi Train dan sisi Testing.



Jika dilihat pada Train, prediksi yang didapatkan mendekati dengan nilai aslinya. Namun pada data testing menjauhi nilai aslinya, untuk mengetahui penyebabnya mari melihat nilai RMSE data testing

```
21.335990410499765
```

Ternyata nilai RMSE yang didapatkan cukup tingggi, hal ini dapat ditingkatkan dengan menambah layer pada model dan mengubah nilai parameter optimizer dan jumlah epochs.

[1] “Saham .:: SIKAPI ::.” https://sikapiuangmu.ojk.go.id/FrontEnd/CMS/Category/64 (accessed Aug. 25, 2022).

[2] Nasdaq, “Apple (AAPL) Historical Stock Data | Kaggle.” https://www.kaggle.com/datasets/tarunpaparaju/apple-aapl-historical-stock-data (accessed Aug. 25, 2022).

[3] R. Dharayani, K. A. Laksitowening, and A. P. Yanuarfiani, “Implementasi ETL (Extract , Transform , Load) Pangkalan Data Perguruan Tinggi dengan Menggunakan State-Space Problem,” *e-Proceeding Eng.*, vol. Vol.2, No., no. 2355–9365, pp. 1159–1165, 2015.