Snapshot of the algorithm and data collected for the RNN-LSTM with attention

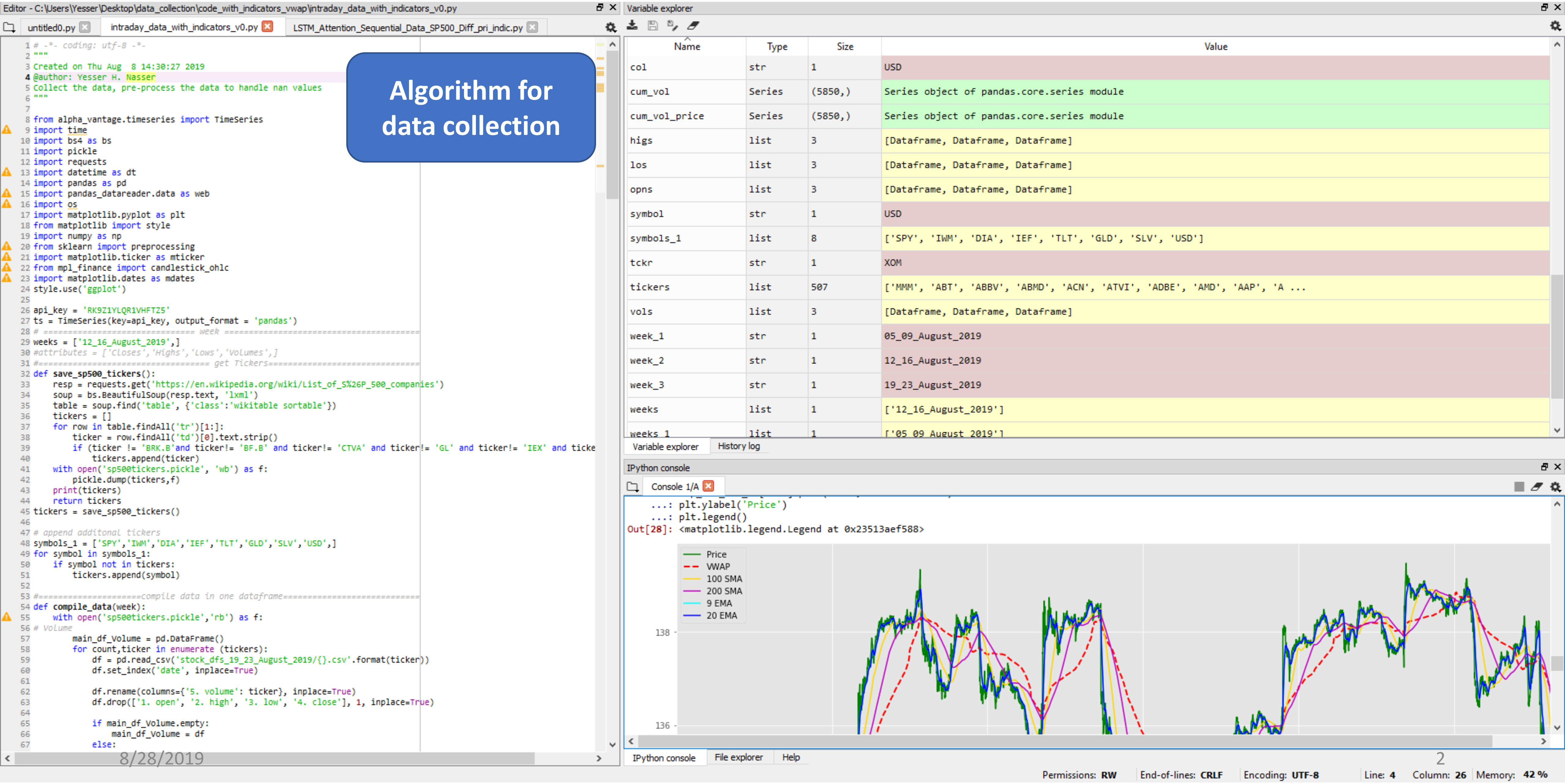
Ongoing Application: High Frequency Quant Trading strategies using machine learning.

Author: Yesser Nasser, PhD

Quantitative Data Scientist

Yesser.Nasser@icloud.com

Algorithm Part 1: data collection



Editor - C:\Users\Yesser\Desktop\data_collection\code_with_indicators_vwap\intraday_data_with_indicators_v0.py
untitled0.py
intraday_data_with_indicators_v0.py
LSTM_Attention_Sequential_Data_SP500_Diff_pri_indic.py
Variable explorer
History log
IPython console
Console 1/A
File explorer
Help

Algorithm for data collection

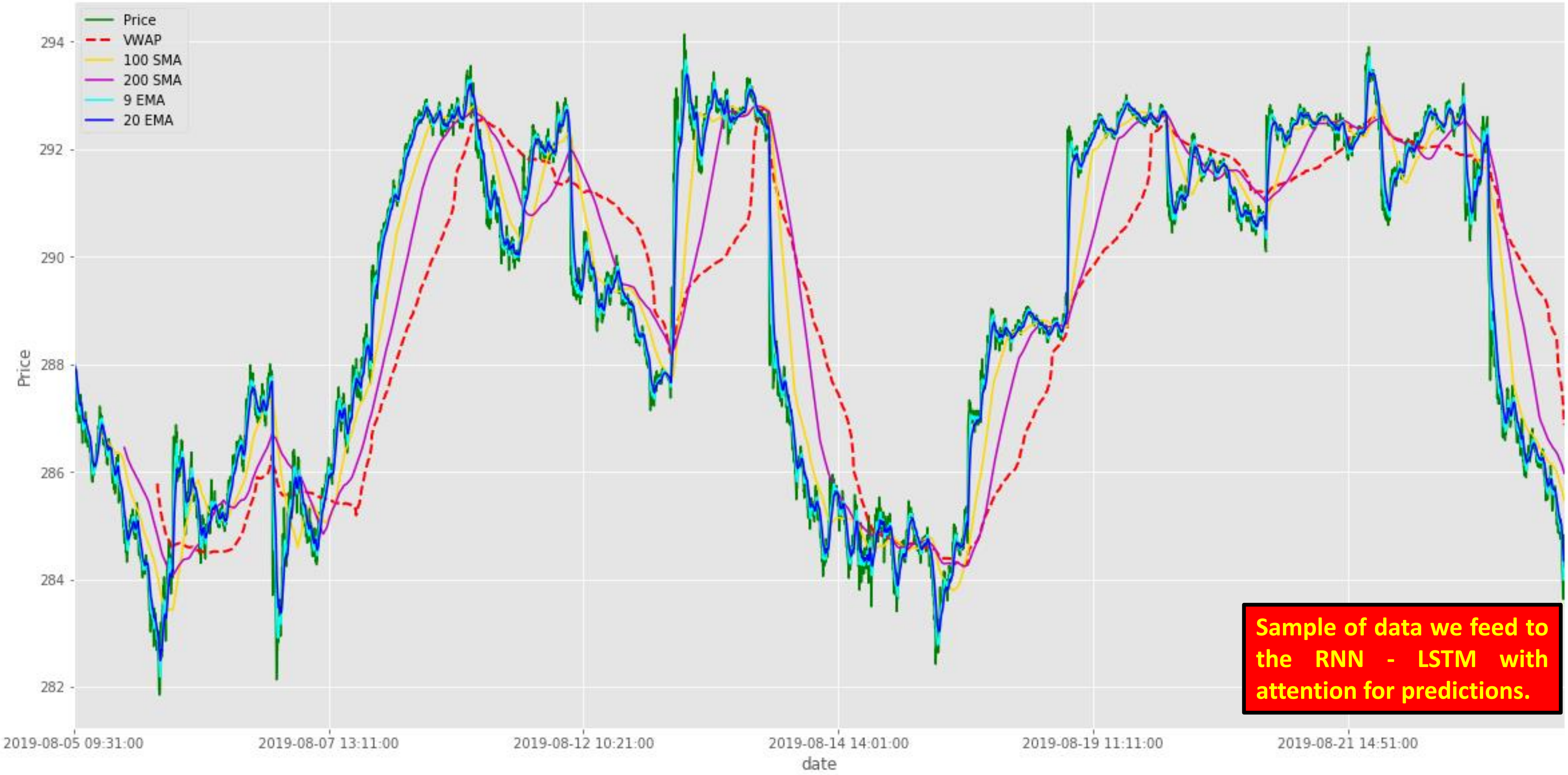
```python
# -*- coding: utf-8 -*-
"""
Created on Thu Aug  8 14:30:27 2019
@author: Yesser H. Nasser
Collect the data, pre-process the data to handle nan values
"""

from alpha_vantage.timeseries import TimeSeries
import time
import bs4 as bs
import pickle
import requests
import datetime as dt
import pandas as pd
import pandas_datareader.data as web
import os
import matplotlib.pyplot as plt
from matplotlib import style
import numpy as np
from sklearn import preprocessing
import matplotlib.ticker as mticker
from mpl_finance import candlestick_ohlc
import matplotlib.dates as mdates
style.use('ggplot')

api_key = 'RK9Z1YLQR1VHFTZ5'
ts = TimeSeries(key=api_key, output_format = 'pandas')
# =============================== week =======================================
weeks = ['12_16_August_2019',]
#attributes = ['Closes','Highs','Lows','Volumes',]
#================================== get Tickers================================
def save_sp500_tickers():
    resp = requests.get('https://en.wikipedia.org/wiki/List_of_S%26P_500_companies')
    soup = bs.BeautifulSoup(resp.text, 'lxml')
    table = soup.find('table', {'class':'wikitable sortable'})
    tickers = []
    for row in table.findAll('tr')[1:]:
        ticker = row.findAll('td')[0].text.strip()
        if (ticker != 'BRK.B'and ticker!= 'BF.B' and ticker!= 'CTVA' and ticker!= 'GL' and ticker!= 'IEX' and ticke
            tickers.append(ticker)
    with open('sp500tickers.pickle', 'wb') as f:
        pickle.dump(tickers,f)
    print(tickers)
    return tickers
tickers = save_sp500_tickers()

# append additonal tickers
symbols_1 = ['SPY','IWM','DIA','IEF','TLT','GLD','SLV','USD',]
for symbol in symbols_1:
    if symbol not in tickers:
        tickers.append(symbol)

#====================compile data in one dataframe===========================
def compile_data(week):
    with open('sp500tickers.pickle','rb') as f:
# Volume
        main_df_Volume = pd.DataFrame()
        for count,ticker in enumerate (tickers):
            df = pd.read_csv('stock_dfs_19_23_August_2019/{}.csv'.format(ticker))
            df.set_index('date', inplace=True)

            df.rename(columns={'5. volume': ticker}, inplace=True)
            df.drop(['1. open', '2. high', '3. low', '4. close'], 1, inplace=True)

            if main_df_Volume.empty:
                main_df_Volume = df
            else:
```

Name	Type	Size	Value
col	str	1	USD
cum_vol	Series	(5850,)	Series object of pandas.core.series module
cum_vol_price	Series	(5850,)	Series object of pandas.core.series module
higs	list	3	[Dataframe, Dataframe, Dataframe]
los	list	3	[Dataframe, Dataframe, Dataframe]
opns	list	3	[Dataframe, Dataframe, Dataframe]
symbol	str	1	USD
symbols_1	list	8	['SPY', 'IWM', 'DIA', 'IEF', 'TLT', 'GLD', 'SLV', 'USD']
tckr	str	1	XOM
tickers	list	507	['MMM', 'ABT', 'ABBV', 'ABMD', 'ACN', 'ATVI', 'ADBE', 'AMD', 'AAP', 'A ...
vols	list	3	[Dataframe, Dataframe, Dataframe]
week_1	str	1	05_09_August_2019
week_2	str	1	12_16_August_2019
week_3	str	1	19_23_August_2019
weeks	list	1	['12_16_August_2019']
weeks_1	list	1	['05_09_August_2019']

```
   ...: plt.ylabel('Price')
   ...: plt.legend()
Out[28]: <matplotlib.legend.Legend at 0x23513aef588>
```

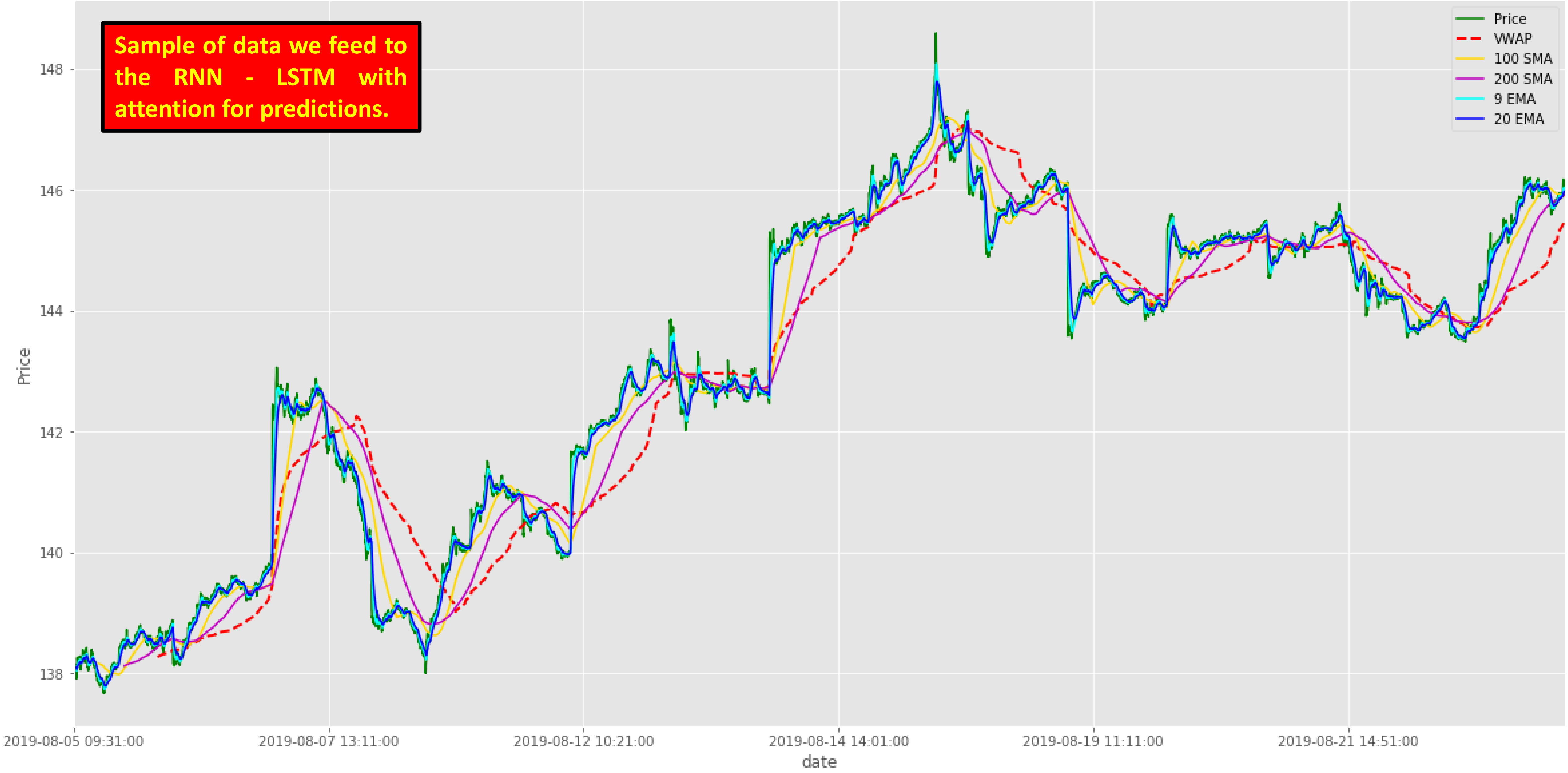
Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 4 Column: 26 Memory: 42 %

SPY



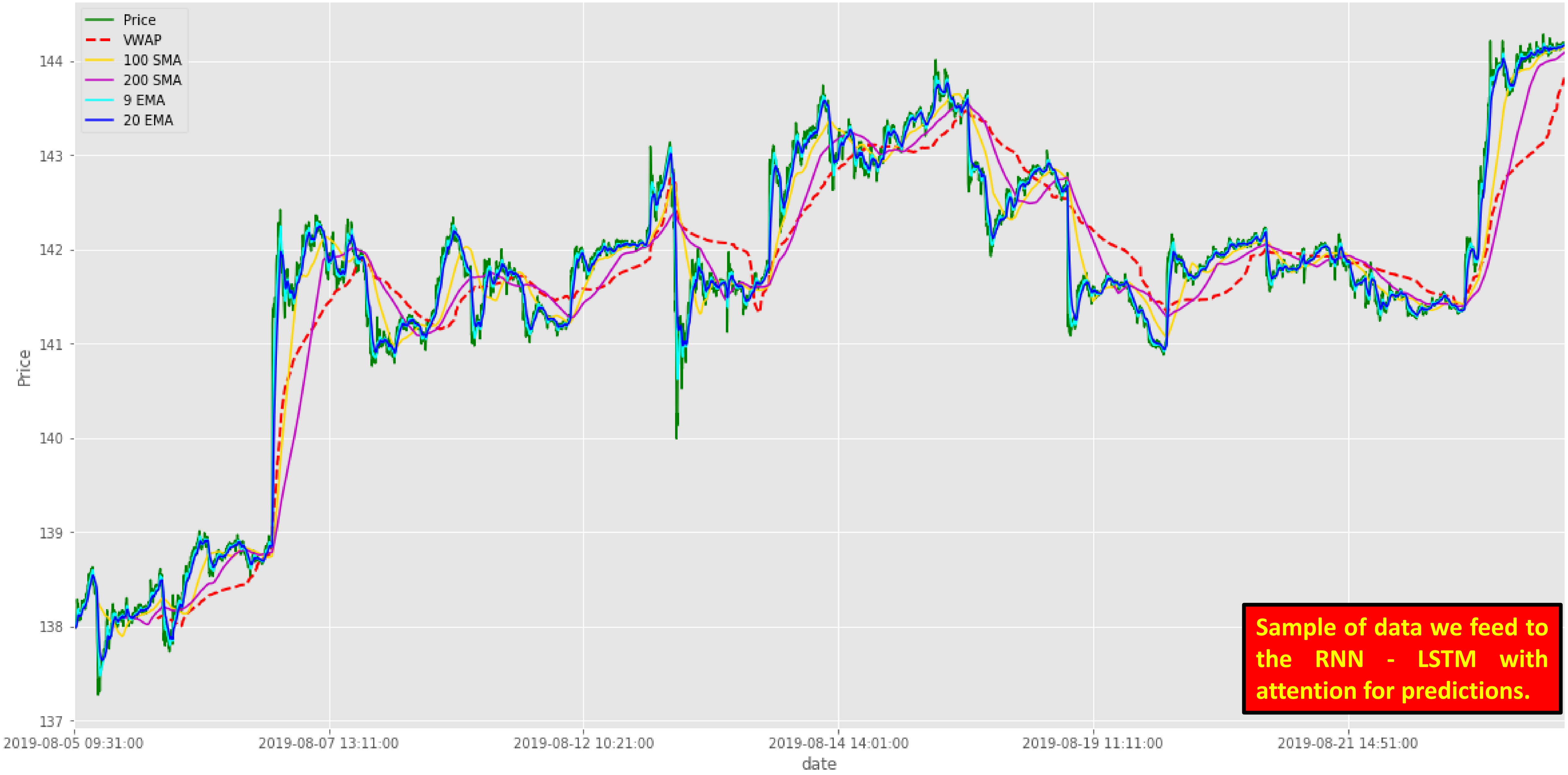
Sample of data we feed to the RNN - LSTM with attention for predictions.

TLT



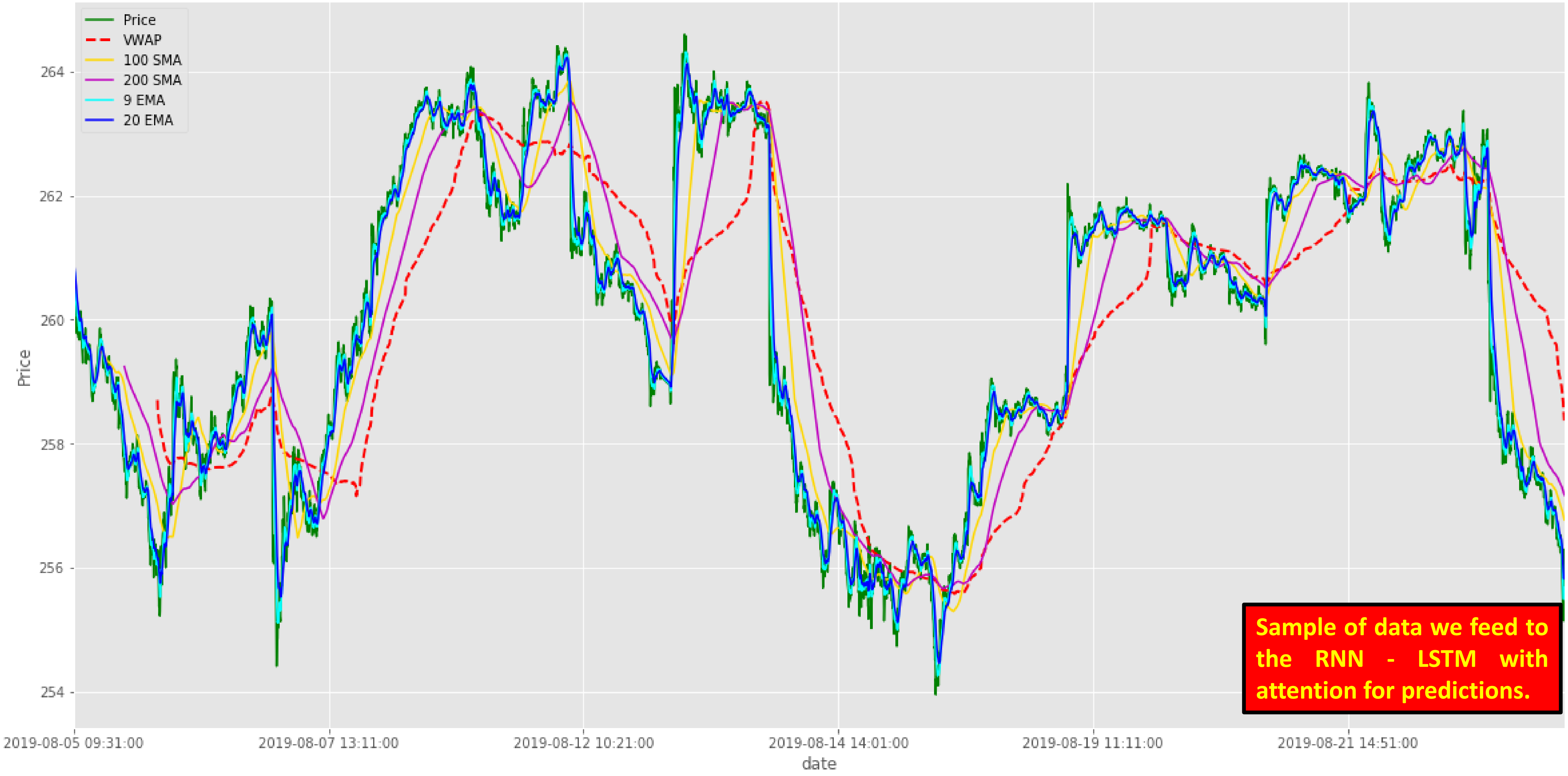
Sample of data we feed to the RNN - LSTM with attention for predictions.

GLD



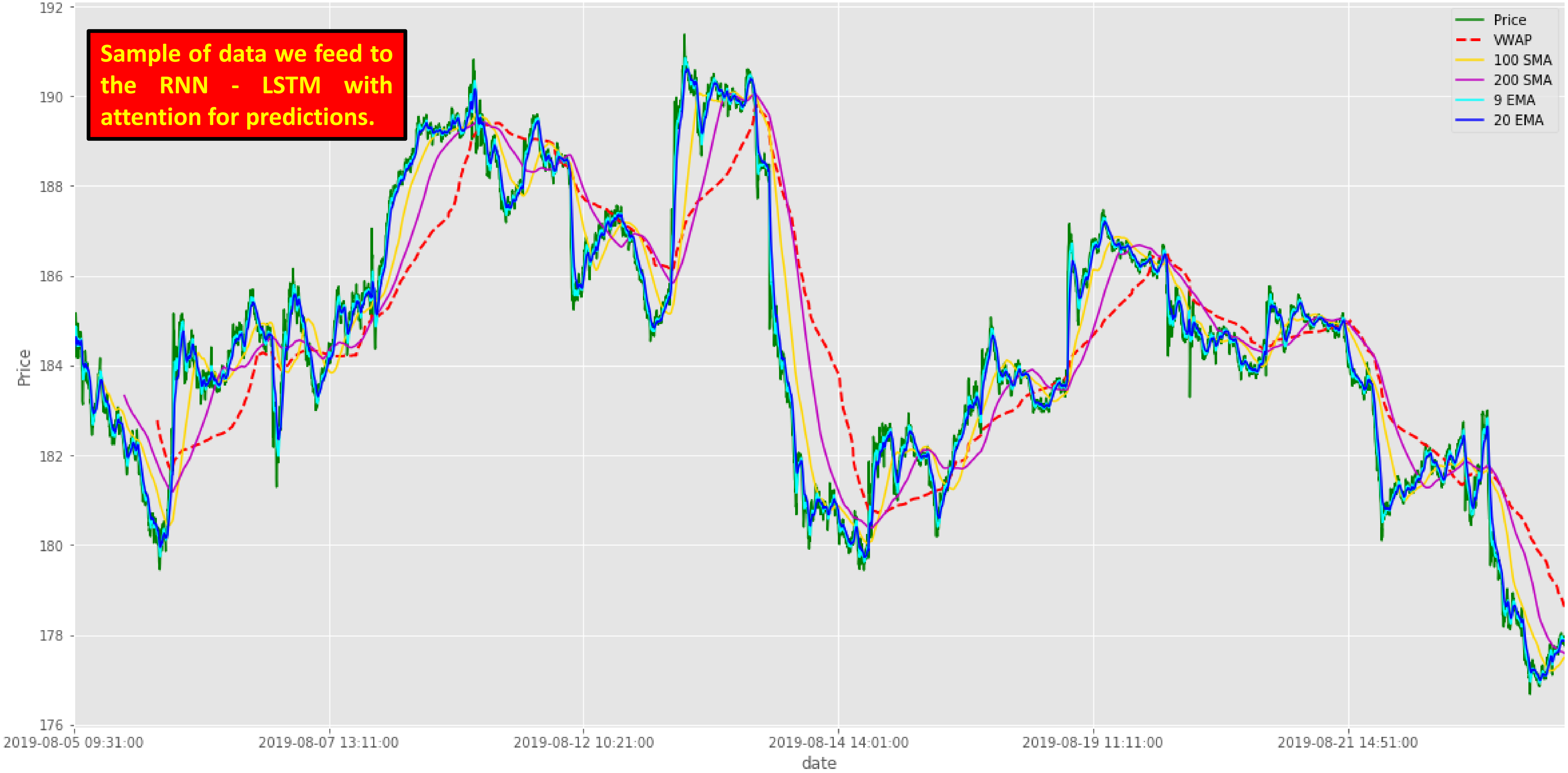
Sample of data we feed to the RNN - LSTM with attention for predictions.

DIA



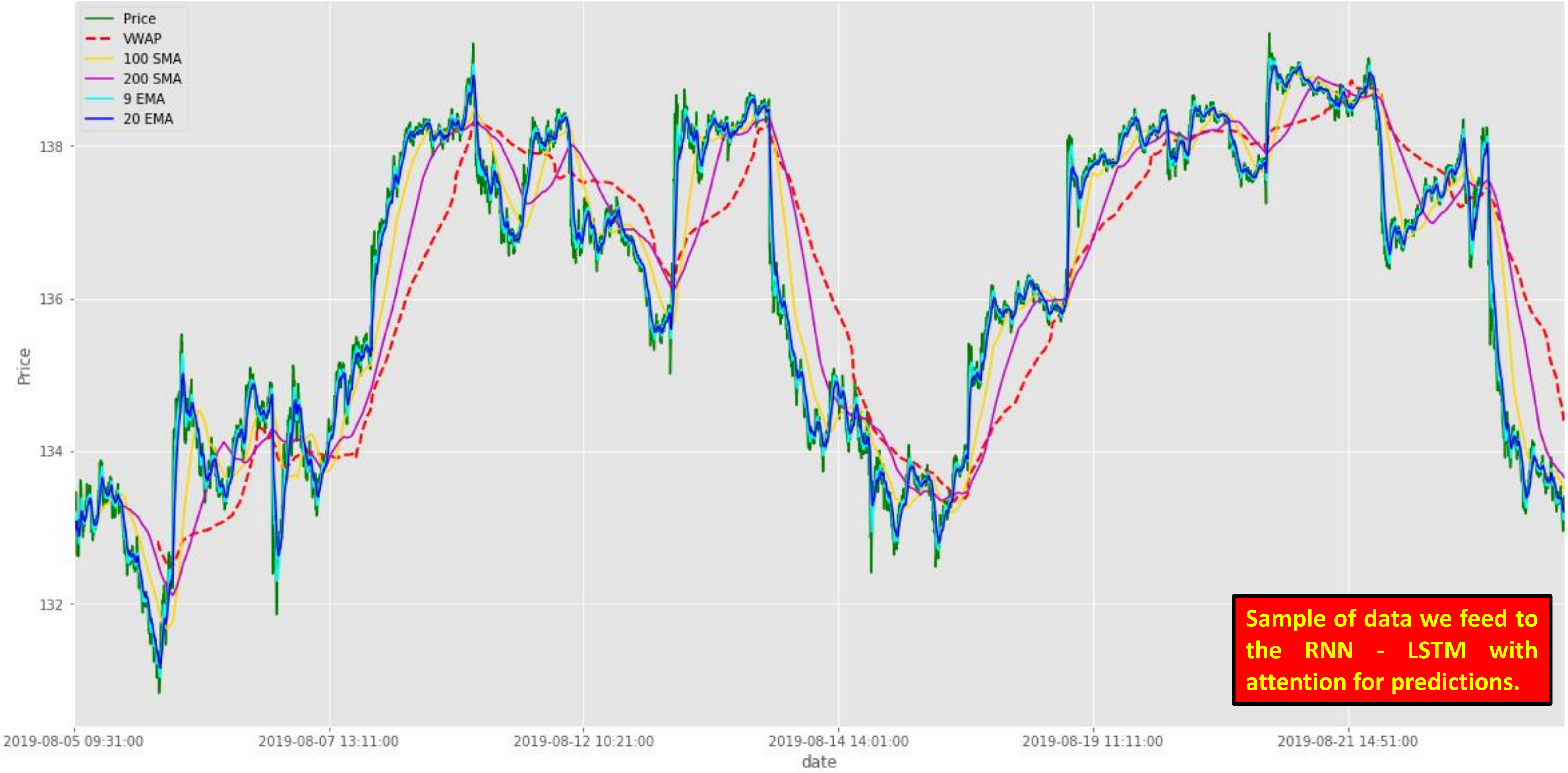
Sample of data we feed to the RNN - LSTM with attention for predictions.

Stock Price - FB



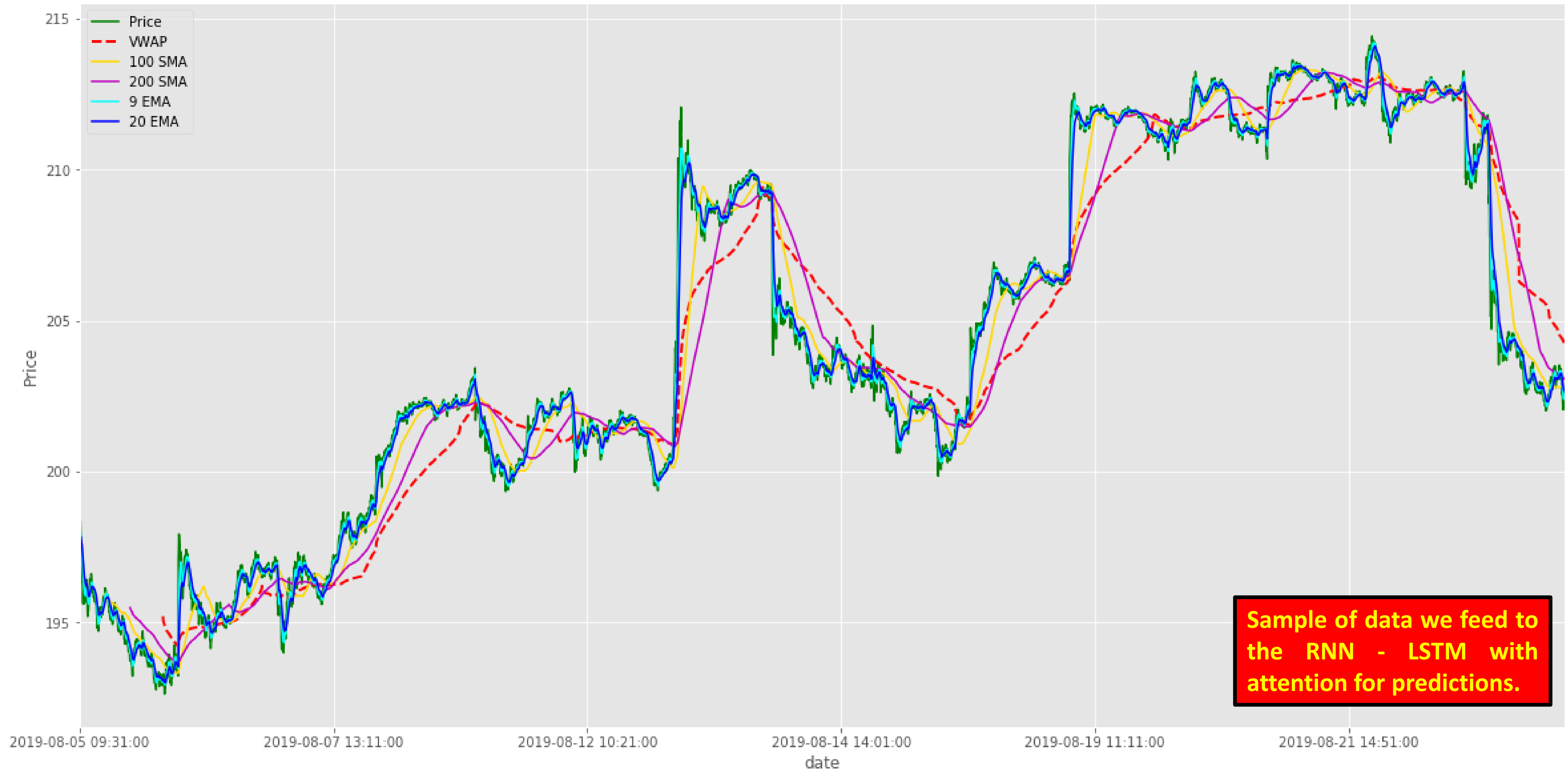
Sample of data we feed to the RNN - LSTM with attention for predictions.

Stock Price - MSFT



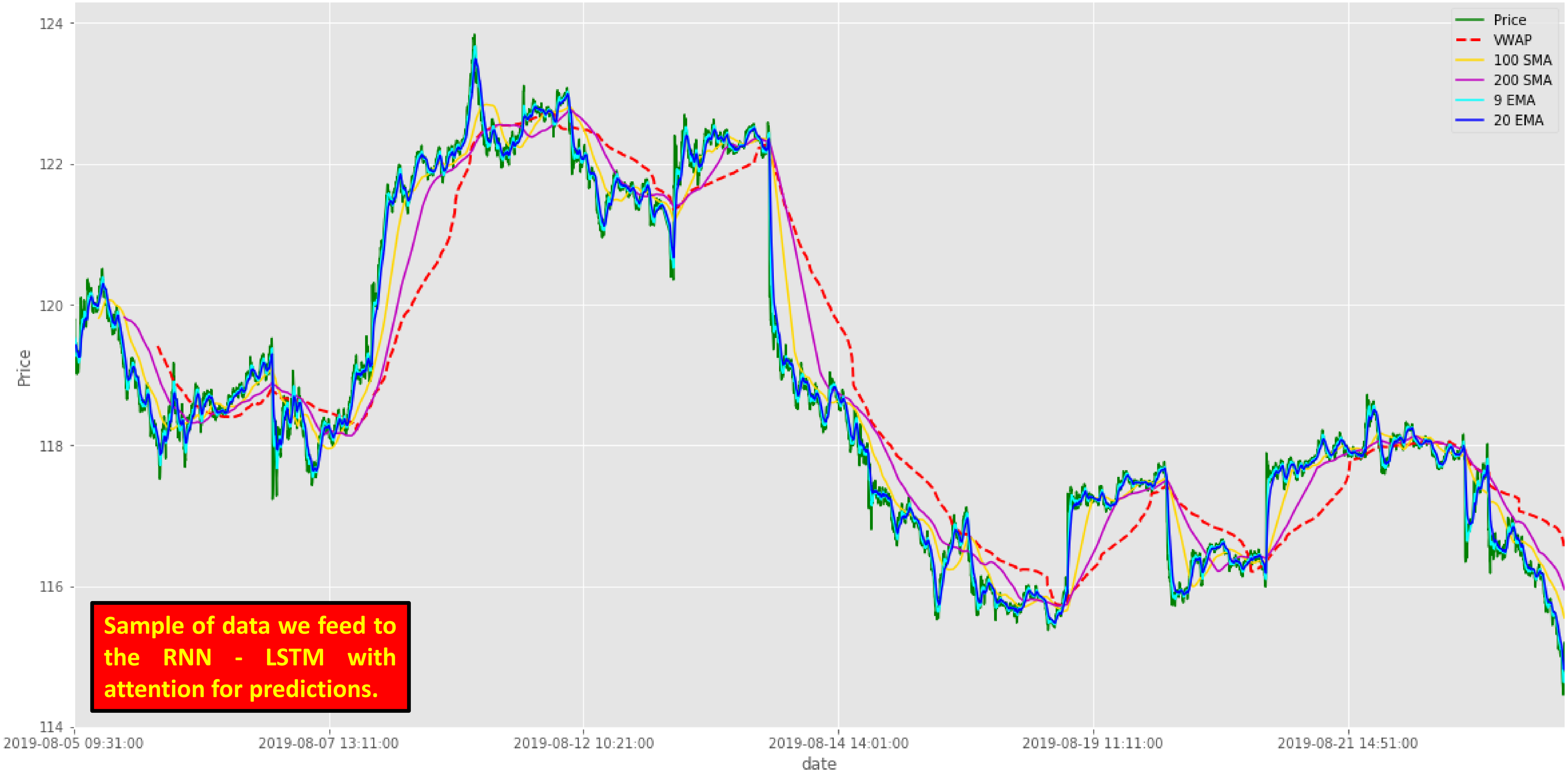
Sample of data we feed to the RNN - LSTM with attention for predictions.

Stock Price - AAPL



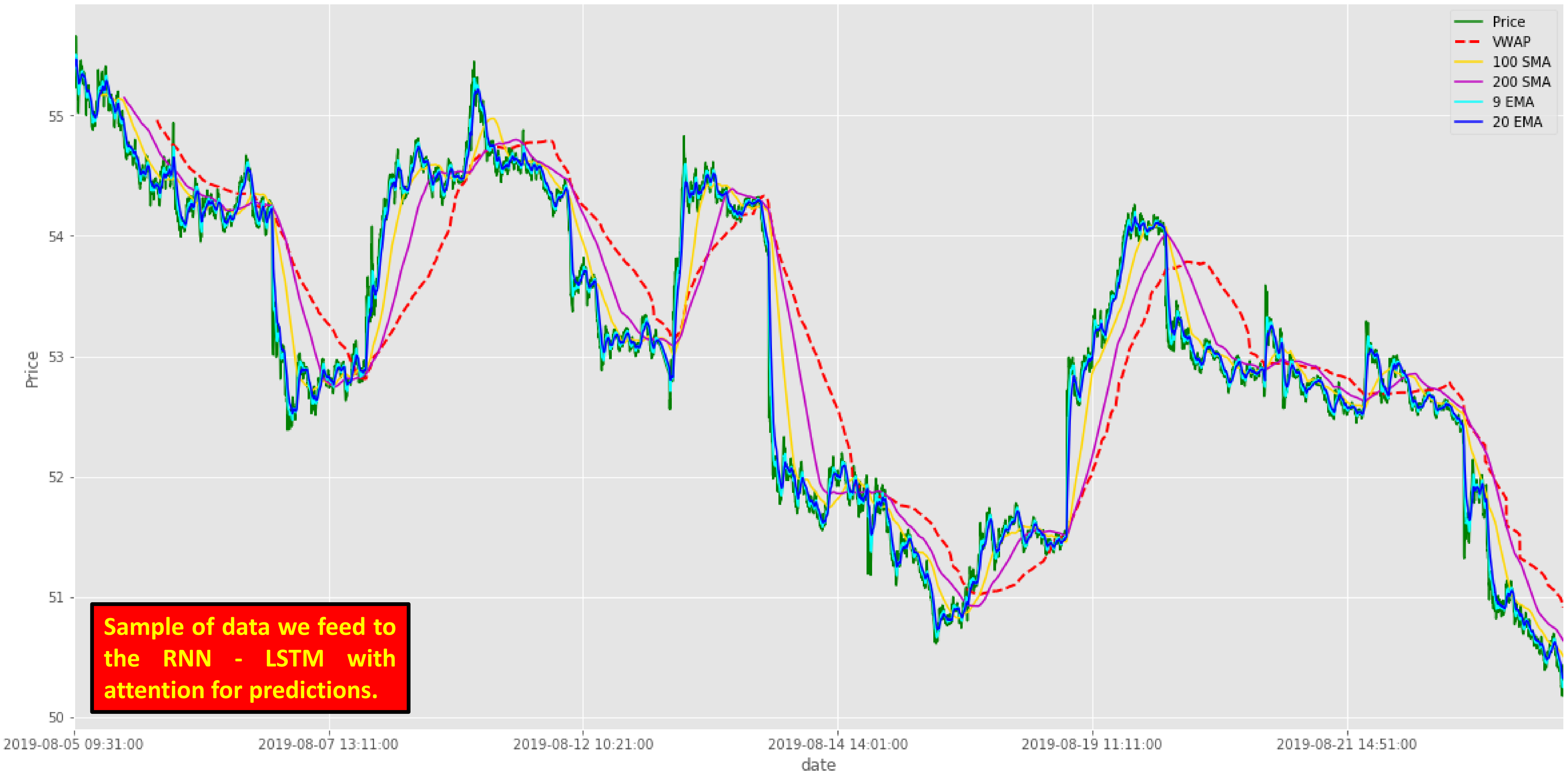
Sample of data we feed to the RNN - LSTM with attention for predictions.

Stock Price - CVX



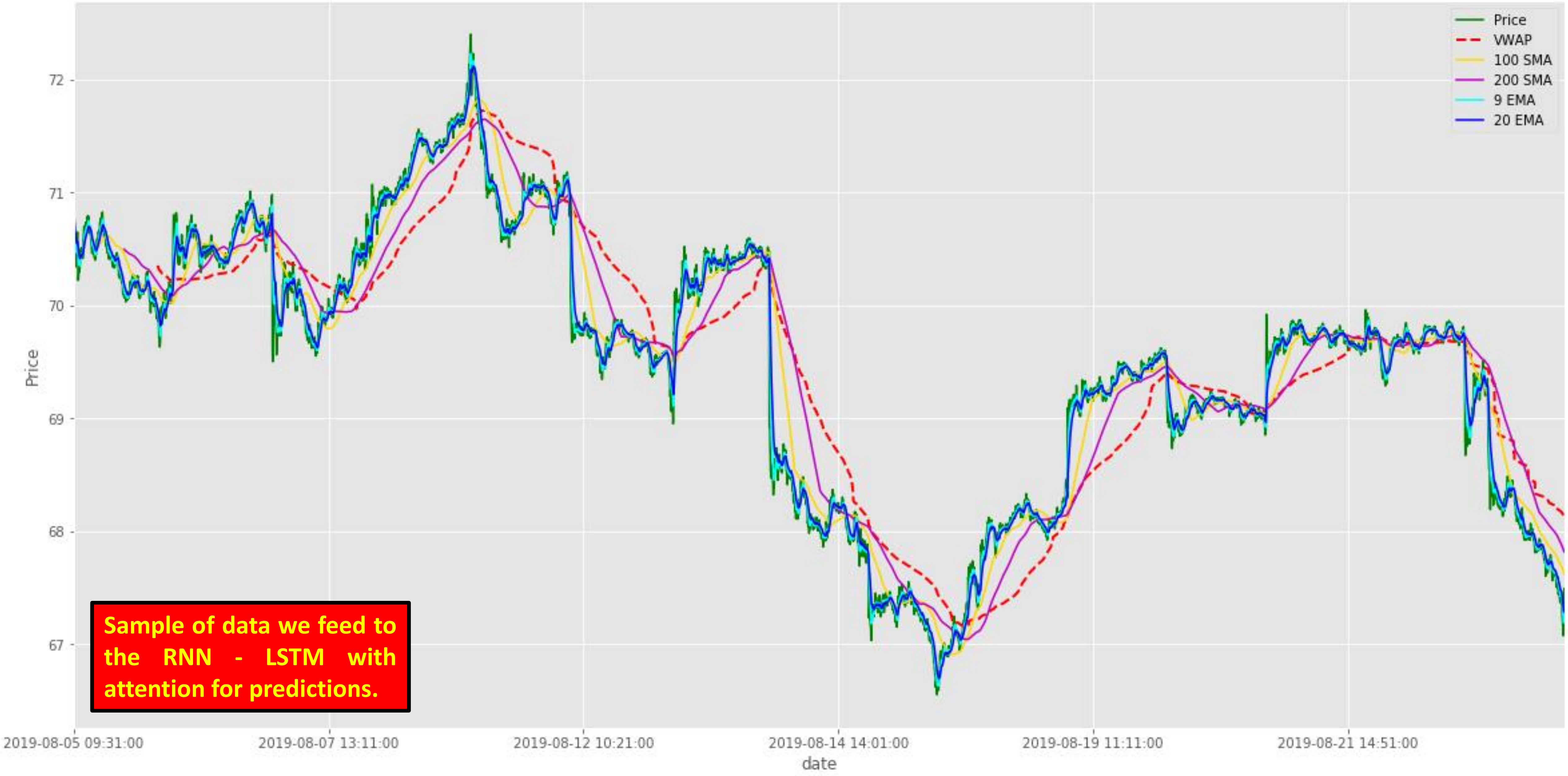
Sample of data we feed to the RNN - LSTM with attention for predictions.

Stock Price - COP



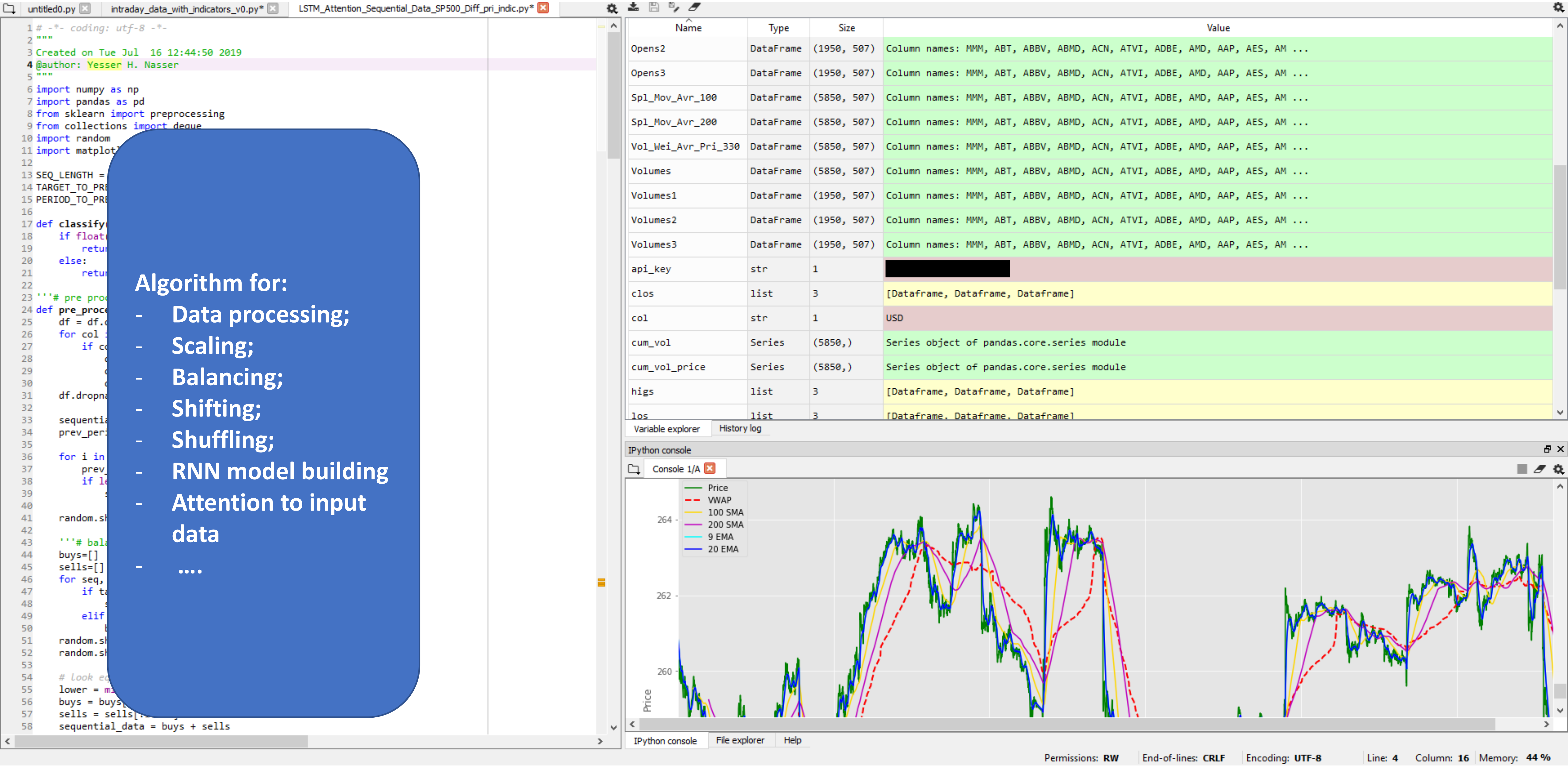
Sample of data we feed to the RNN - LSTM with attention for predictions.

Stock Price - XOM



Sample of data we feed to the RNN - LSTM with attention for predictions.

Algorithm Part 2: Model building



Algorithm for:
- Data processing;
- Scaling;
- Balancing;
- Shifting;
- Shuffling;
- RNN model building
- Attention to input data
- ....

untitled0.py
intraday_data_with_indicators_v0.py*
LSTM_Attention_Sequential_Data_SP500_Diff_pri_indic.py*

```
# -*- coding: utf-8 -*-
"""
Created on Tue Jul  16 12:44:50 2019
@author: Yesser H. Nasser
"""
import numpy as np
import pandas as pd
from sklearn import preprocessing
from collections import deque
import random
import matplotl

SEQ_LENGTH =
TARGET_TO_PRE
PERIOD_TO_PRE

def classify(
    if float(
        retur
    else:
        retur

'''# pre proc
def pre_proc
    df = df.c
    for col i
        if co
            c
            c
            c
    df.dropna

    sequentia
    prev_peri

    for i in
        prev_
        if le
            s

    random.sh

    '''# bala
    buys=[]
    sells=[]
    for seq,
        if ta
            s
        elif
            b
    random.sh
    random.sh

    # look eq
    lower = mi
    buys = buys
    sells = sells[
    sequential_data = buys + sells
```

Name	Type	Size	Value
Opens2	DataFrame	(1950, 507)	Column names: MMM, ABT, ABBV, ABMD, ACN, ATVI, ADBE, AMD, AAP, AES, AM ...
Opens3	DataFrame	(1950, 507)	Column names: MMM, ABT, ABBV, ABMD, ACN, ATVI, ADBE, AMD, AAP, AES, AM ...
Spl_Mov_Avr_100	DataFrame	(5850, 507)	Column names: MMM, ABT, ABBV, ABMD, ACN, ATVI, ADBE, AMD, AAP, AES, AM ...
Spl_Mov_Avr_200	DataFrame	(5850, 507)	Column names: MMM, ABT, ABBV, ABMD, ACN, ATVI, ADBE, AMD, AAP, AES, AM ...
Vol_Wei_Avr_Pri_330	DataFrame	(5850, 507)	Column names: MMM, ABT, ABBV, ABMD, ACN, ATVI, ADBE, AMD, AAP, AES, AM ...
Volumes	DataFrame	(5850, 507)	Column names: MMM, ABT, ABBV, ABMD, ACN, ATVI, ADBE, AMD, AAP, AES, AM ...
Volumes1	DataFrame	(1950, 507)	Column names: MMM, ABT, ABBV, ABMD, ACN, ATVI, ADBE, AMD, AAP, AES, AM ...
Volumes2	DataFrame	(1950, 507)	Column names: MMM, ABT, ABBV, ABMD, ACN, ATVI, ADBE, AMD, AAP, AES, AM ...
Volumes3	DataFrame	(1950, 507)	Column names: MMM, ABT, ABBV, ABMD, ACN, ATVI, ADBE, AMD, AAP, AES, AM ...
api_key	str	1	[redacted]
clos	list	3	[Dataframe, Dataframe, Dataframe]
col	str	1	USD
cum_vol	Series	(5850,)	Series object of pandas.core.series module
cum_vol_price	Series	(5850,)	Series object of pandas.core.series module
higs	list	3	[Dataframe, Dataframe, Dataframe]
los	list	3	[Dataframe, Dataframe, Dataframe]

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 4 Column: 16 Memory: 44 %

To learn more about the progress on this project and looking to apply this workflow please get in touch at: Yesser.Nasser@icloud.com