

# 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](mailto:Yesser.Nasser@icloud.com)

## Algorithm Part 1: data collection

Algorithm for  
data collection

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Aug 8 14:30:27 2019
4 @author: Yesser H. Nassen
5 Collect the data, pre-process the data to handle nan values
6 """
7
8 from alpha_vantage.timeseries import TimeSeries
9 import time
10 import bs4 as bs
11 import pickle
12 import requests
13 import datetime as dt
14 import pandas as pd
15 import pandas_datareader.data as web
16 import os
17 import matplotlib.pyplot as plt
18 from matplotlib import style
19 import numpy as np
20 from sklearn import preprocessing
21 import matplotlib.ticker as mticker
22 from mpl_finance import candlestick_ohlc
23 import matplotlib.dates as mdates
24 style.use('ggplot')
25
26 api_key = 'RK921YLQR1VWF7Z5'
27 ts = TimeSeries(key=api_key, output_format='pandas')
28 # ===== WEEK =====
29 weeks = ['12_16_August_2019',]
30 #attributes = ['Closes', 'Highs', 'Lows', 'Volumes',]
31 # ===== get Tickers =====
32 def save_sp500_tickers():
33     resp = requests.get('https://en.wikipedia.org/wiki/List_of_S%26P_500_companies')
34     soup = bs.BeautifulSoup(resp.text, 'lxml')
35     table = soup.find('table', {'class': 'wikitable sortable'})
36     tickers = []
37     for row in table.findAll('tr')[1:]:
38         ticker = row.findAll('td')[0].text.strip()
39         if (ticker != 'BRK.B' and ticker != 'BF.B' and ticker != 'CTVA' and ticker != 'GL' and ticker != 'IEK' and ticker != 'JPM') and (ticker != 'K' and ticker != 'KO' and ticker != 'LLY' and ticker != 'MDLZ' and ticker != 'MRK' and ticker != 'NKE' and ticker != 'PFE' and ticker != 'PG' and ticker != 'PYPL' and ticker != 'TSLA' and ticker != 'V' and ticker != 'VZ'):
40             tickers.append(ticker)
41     with open('sp500tickers.pickle', 'wb') as f:
42         pickle.dump(tickers, f)
43     print(tickers)
44     return tickers
45 tickers = save_sp500_tickers()
46
47 # append additional tickers
48 symbols_1 = ['SPY', 'IWM', 'DIA', 'IEF', 'TLT', 'GLD', 'SLV', 'USD',]
49 for symbol in symbols_1:
50     if symbol not in tickers:
51         tickers.append(symbol)
52
53 # ===== compile data in one dataframe =====
54 def compile_data(week):
55     with open('sp500tickers.pickle', 'rb') as f:
56         # Volume
57         main_df_Volume = pd.DataFrame()
58         for count, ticker in enumerate(tickers):
59             df = pd.read_csv('stock_dfs_19_23_August_2019/{}.csv'.format(ticker))
60             df.set_index('date', inplace=True)
61
62             df.rename(columns={'5. volume': ticker, 'inplace': True})
63             df.drop(['1. open', '2. high', '3. low', '4. close'], 1, inplace=True)
64
65             if main_df_Volume.empty:
66                 main_df_Volume = df
67             else:
68                 main_df_Volume = main_df_Volume.append(df, ignore_index=True)
```

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']

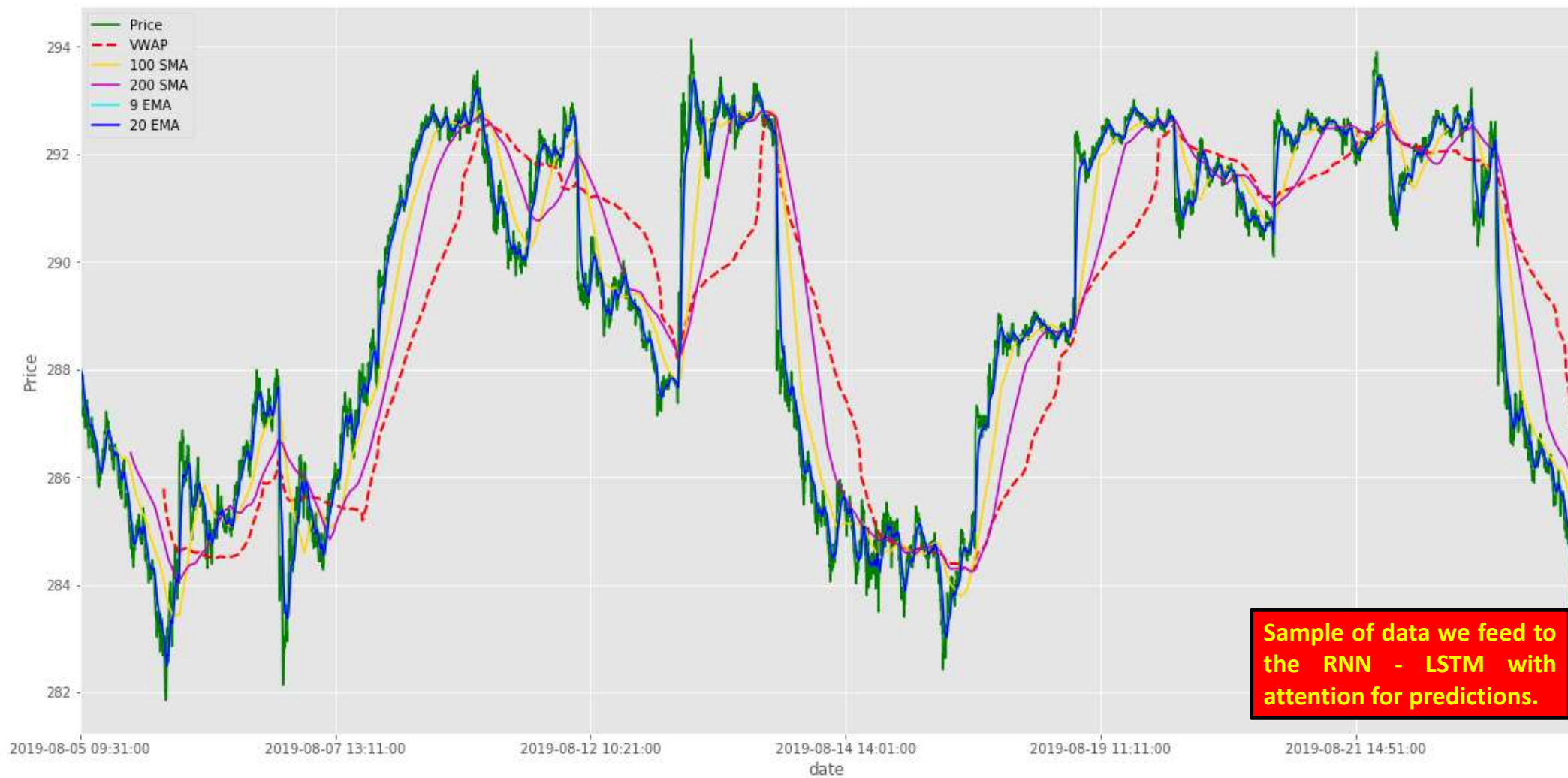
IPython console

Console 1/A

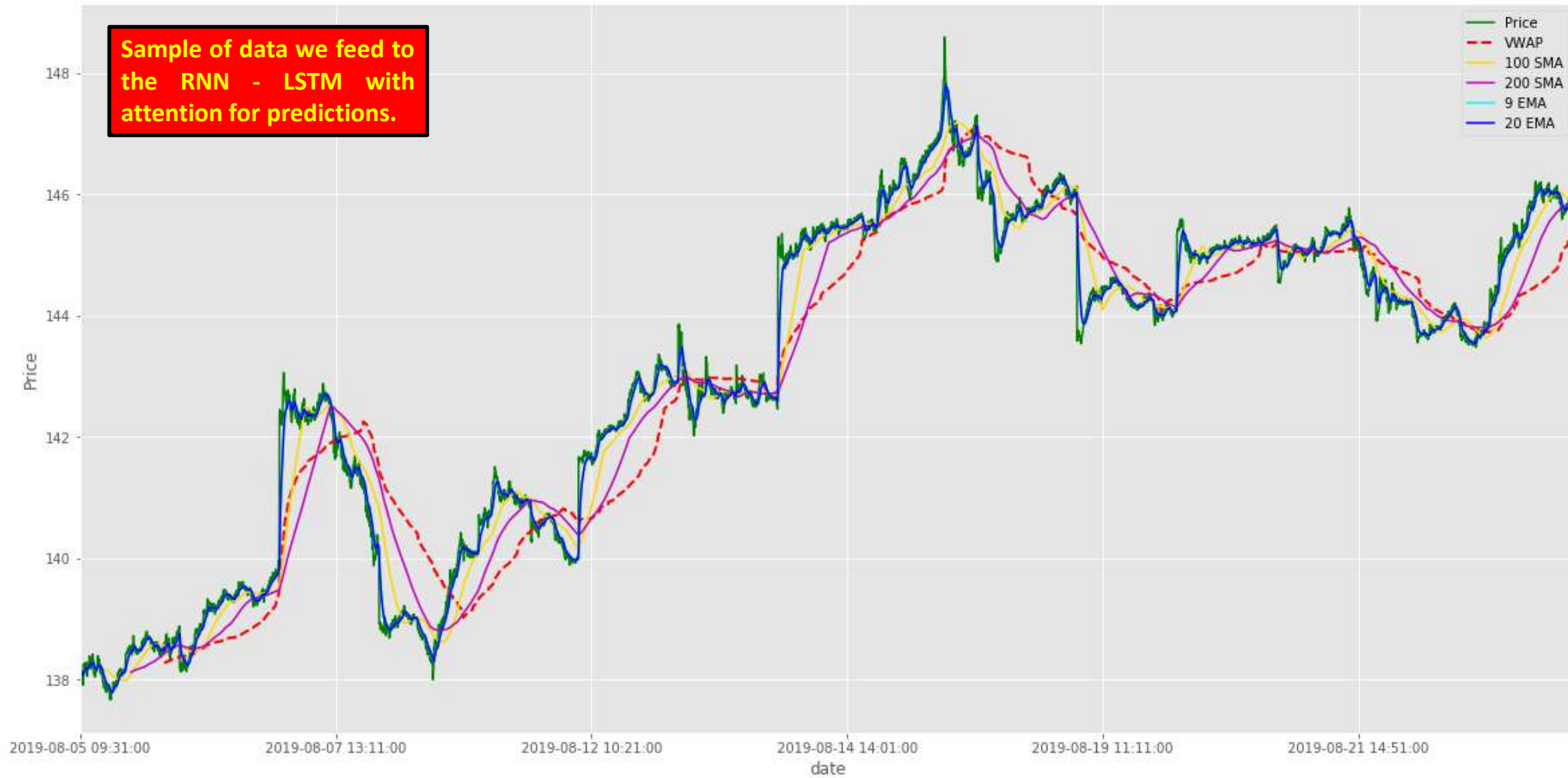
...: plt.ylabel('Price')  
...: plt.legend()  
Out[28]: <matplotlib.legend.Legend at 0x23513aef588>

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

# SPY



## TLT



8/26/2019

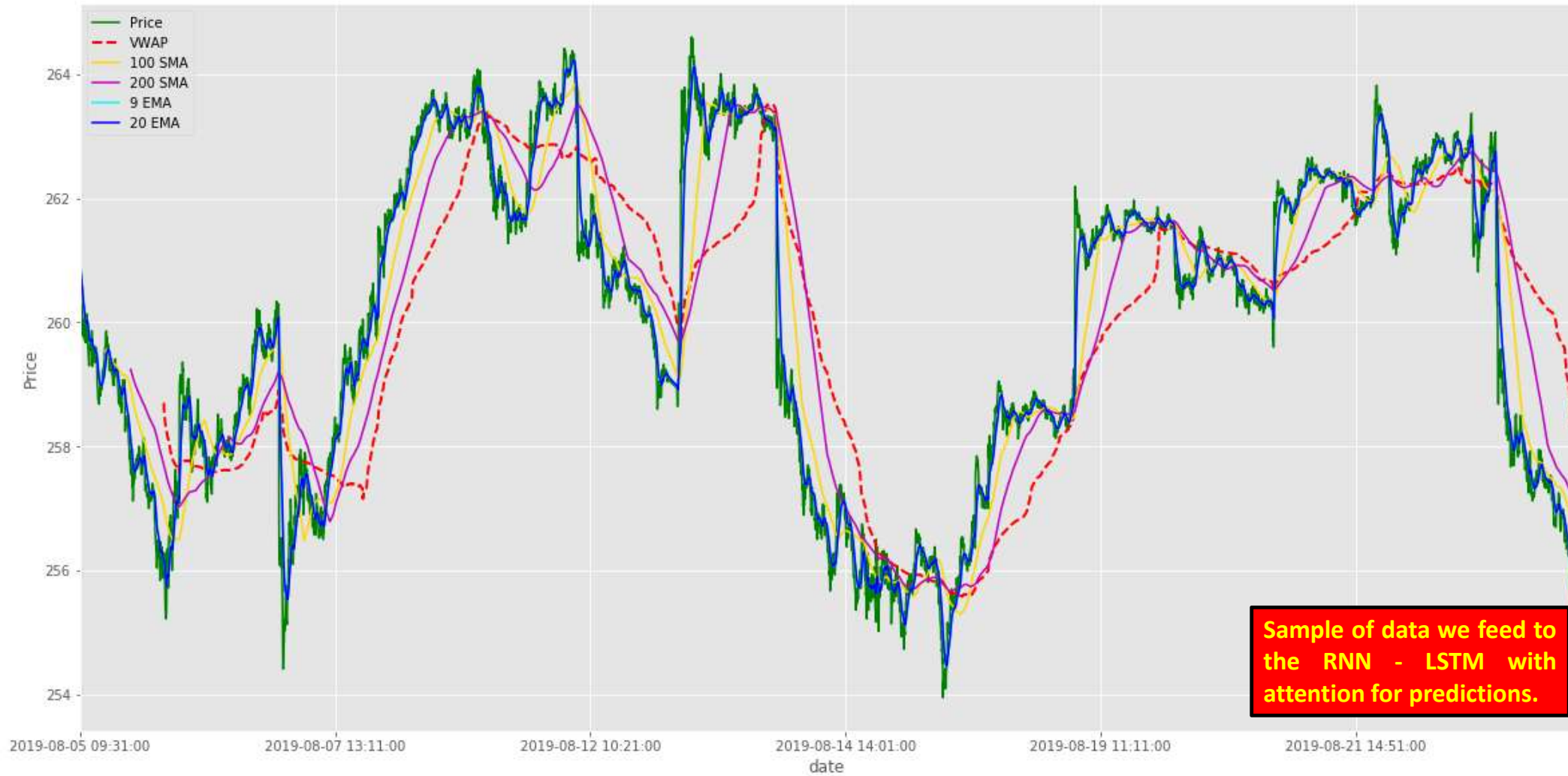
# GLD



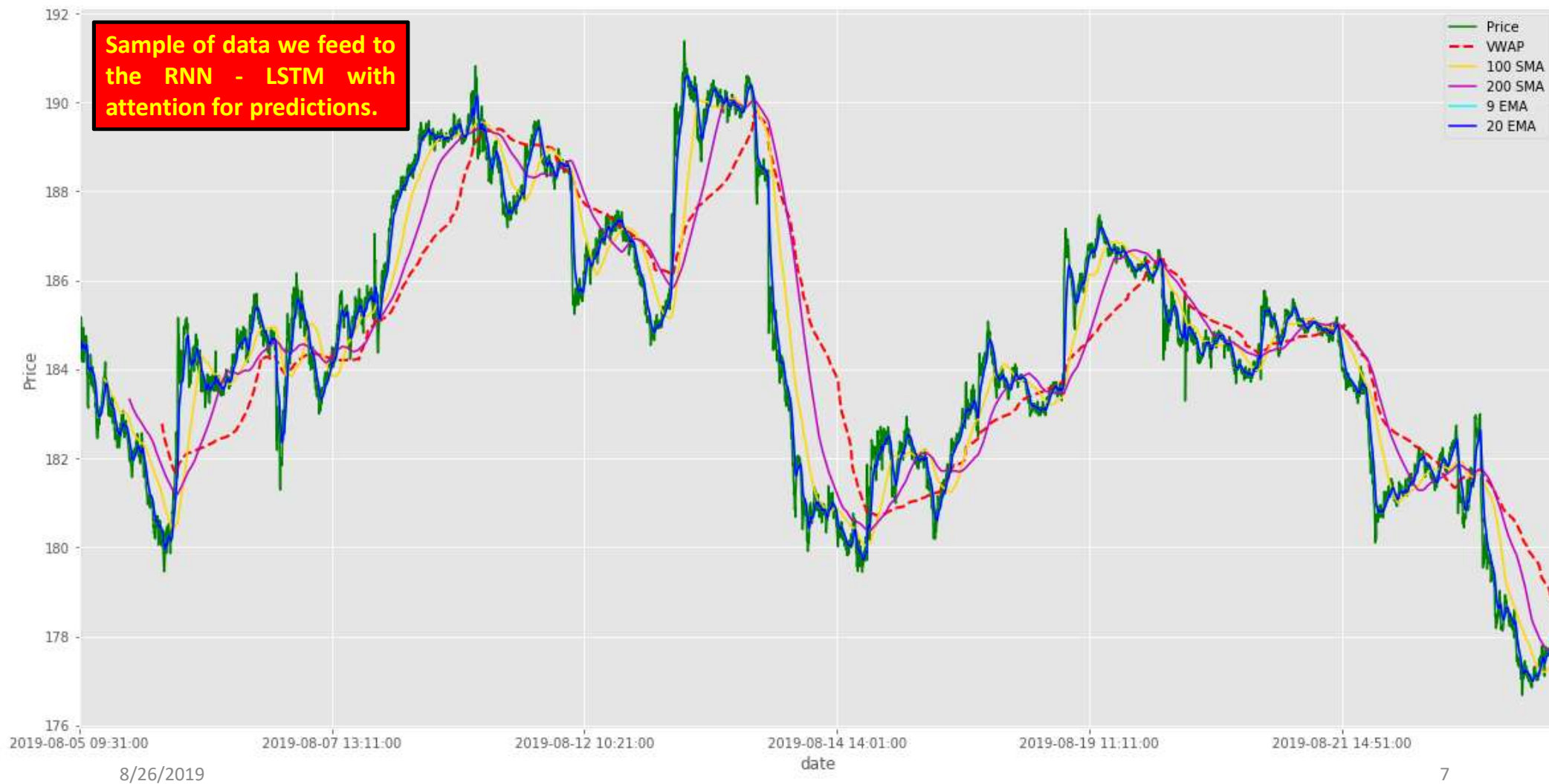
8/26/2019



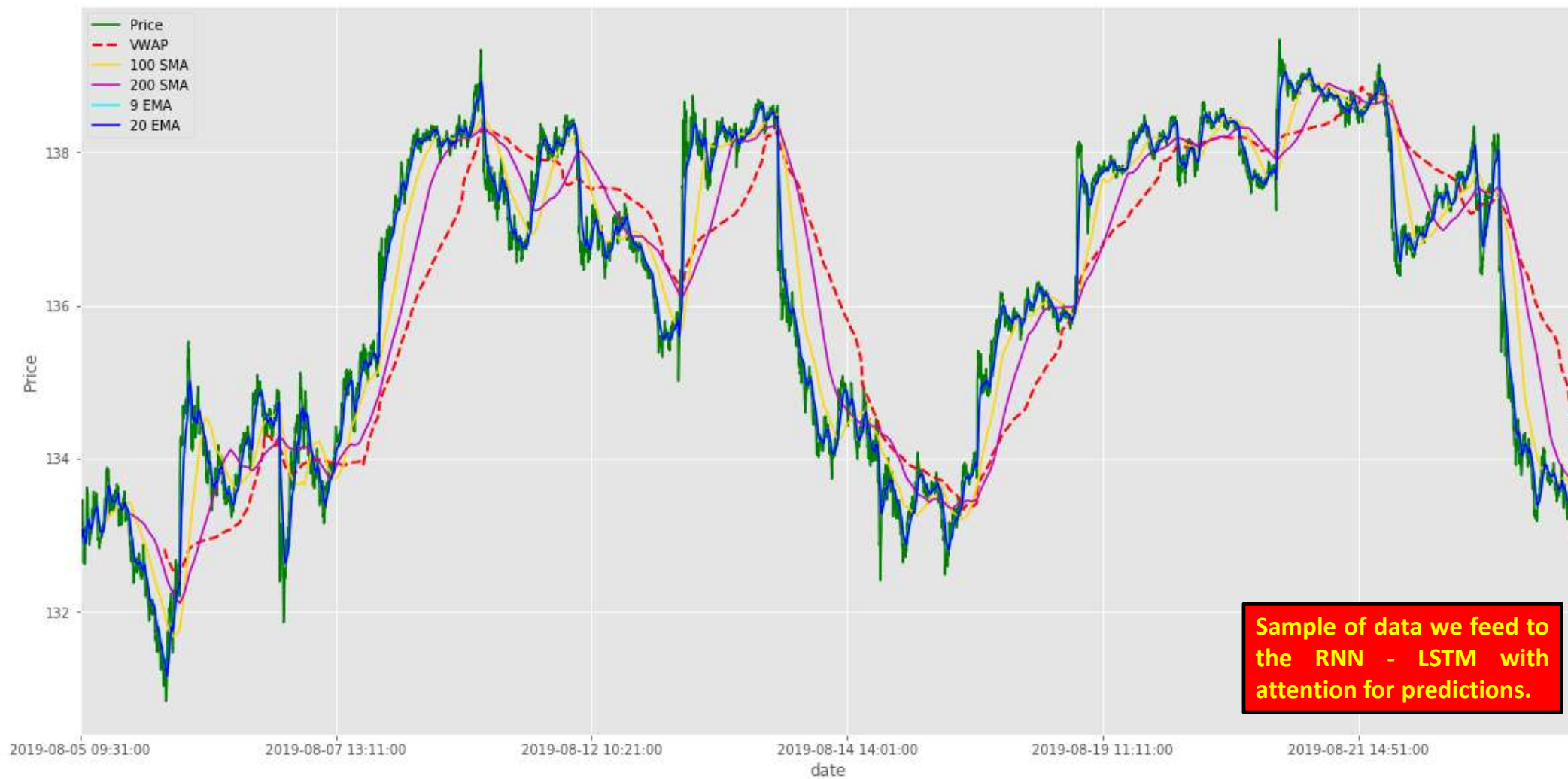
# DIA



## Stock Price - FB



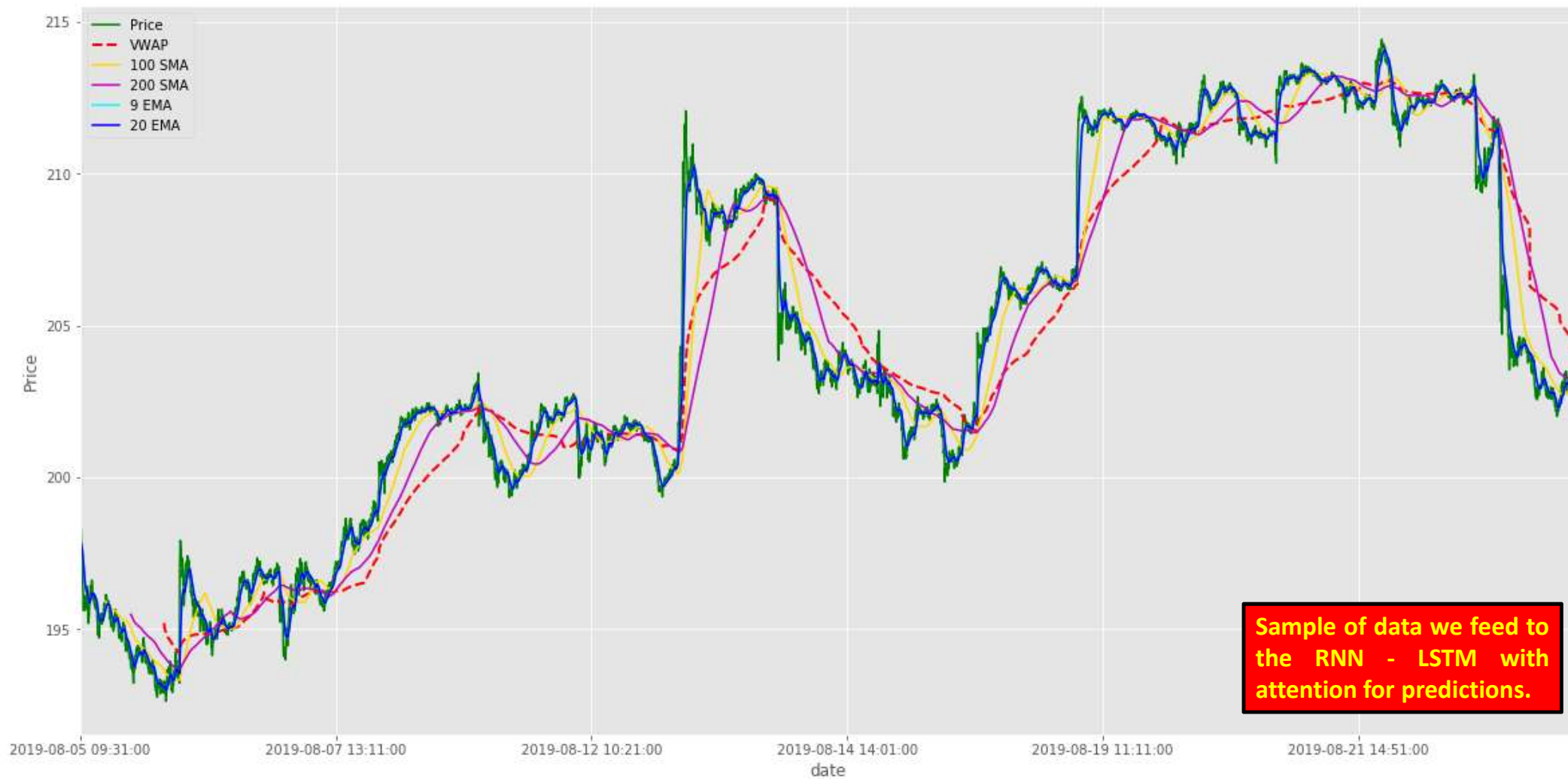
## Stock Price - MSFT



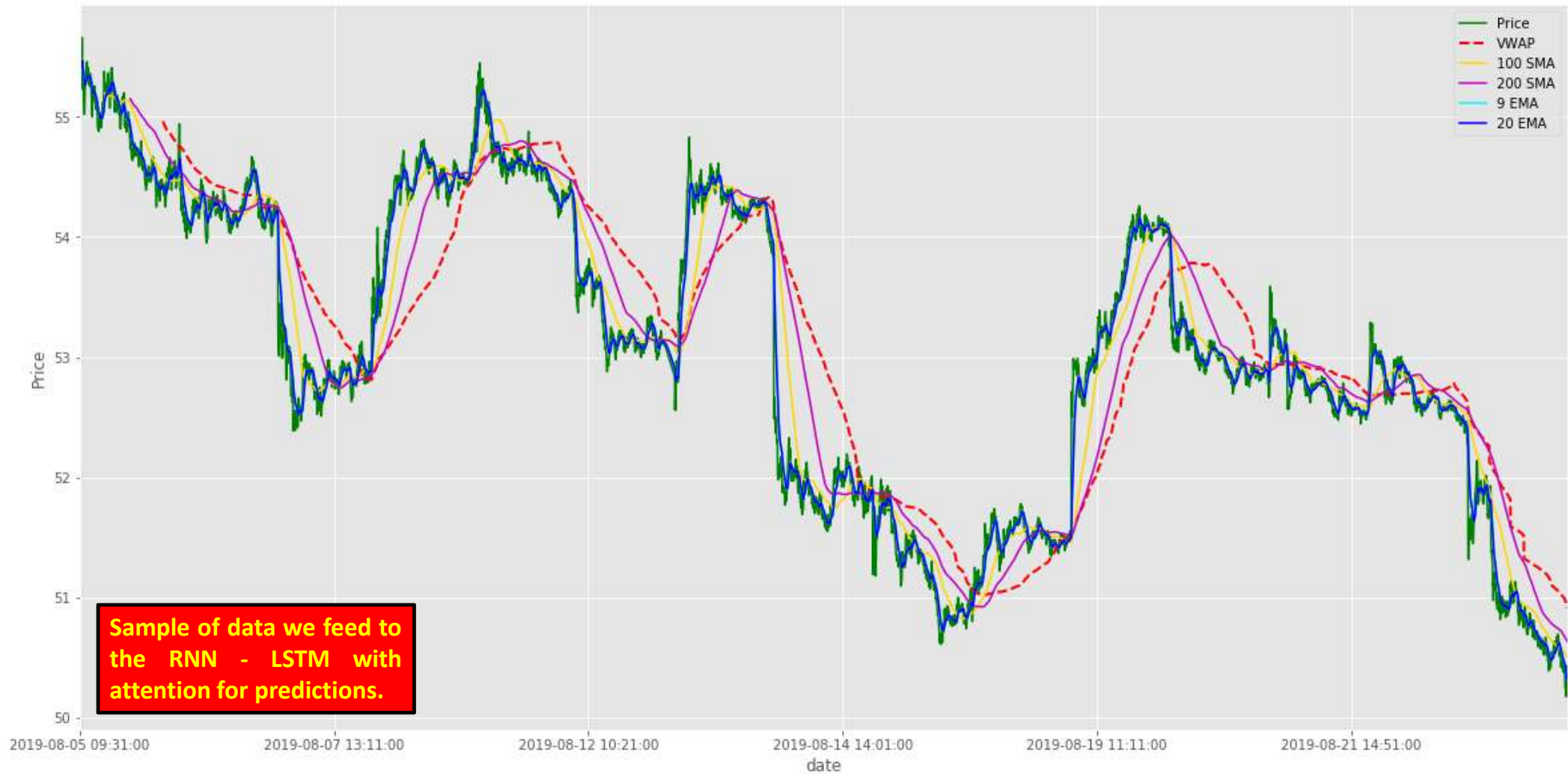
8/26/2019



## Stock Price - AAPL

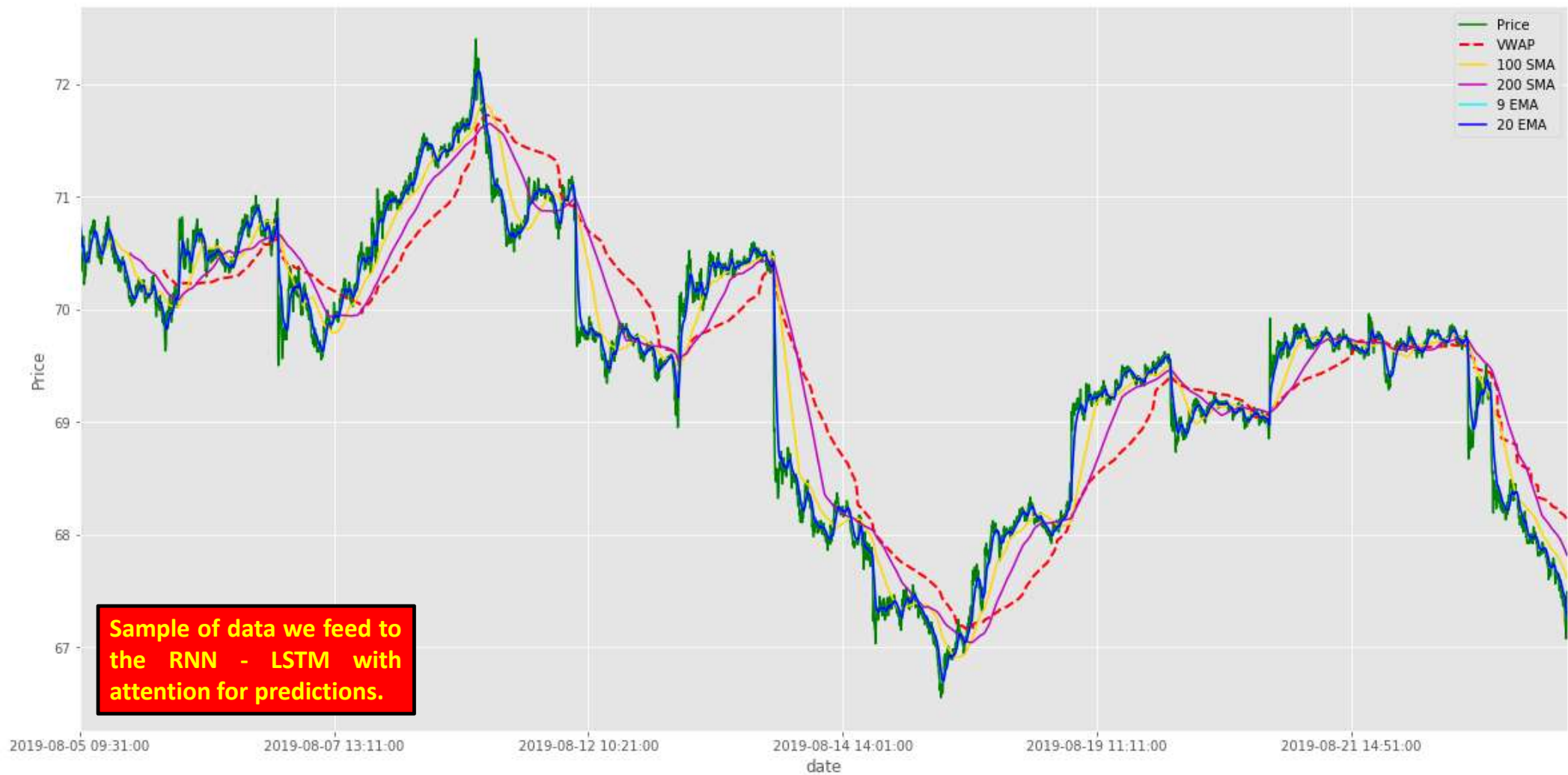


## Stock Price - COP



8/26/2019

## Stock Price - XOM



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

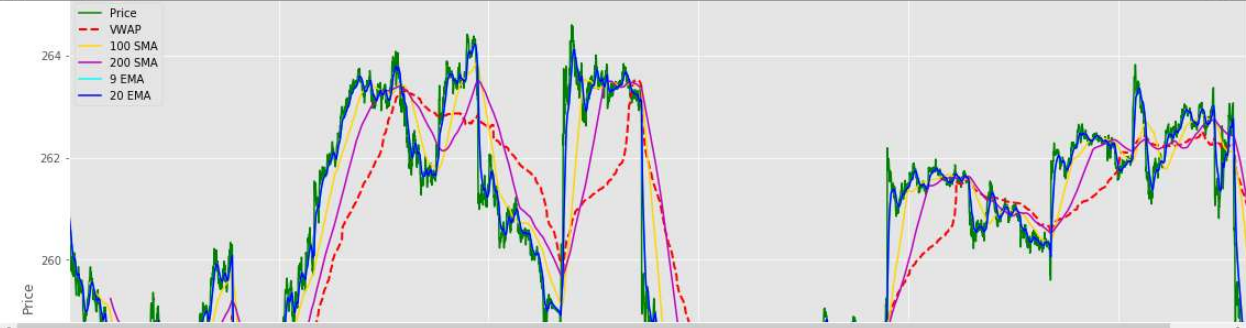
## Algorithm Part 2: Model building

**Algorithm for**

- Data processing;
- Scaling;
- Balancing;
- Shuffling;
- RNN model building
- Attention to input data

```
1 #- coding: utf-8 -*-
2 """
3 Created on Tue Jul 16 12:44:50 2019
4 @author: Yesser H. Nasser
5 """
6 import numpy as np
7 import pandas as pd
8 from sklearn import preprocessing
9 from collections import deque
10 import random
11 import matplotlib.pyplot as plt
12
13 SEQ_LENGTH = 15
14 TARGET_TO_PREV = 1
15 PERIOD_TO_PREV = 1
16
17 def classify(seq, target):
18     if float(target) == 0:
19         return 0
20     else:
21         return 1
22
23 '''# pre processing'''
24 def pre_processing(df):
25     df = df.dropna()
26     for col in df.columns:
27         if col == 'Date':
28             continue
29         df[col] = df[col].astype(float)
30     df.dropna(inplace=True)
31
32     sequential_data = []
33     prev_period = []
34
35     for i in df.iterrows():
36         prev_period.append([n for n in i[1][:-1]])
37         if len(prev_period) == SEQ_LENGTH:
38             sequential_data.append([np.array(prev_period), i[1][-1]])
39             prev_period = []
40
41     random.shuffle(sequential_data)
42
43     '''# balance the data make there are buys as many sells'''
44     buys = []
45     sells = []
46     for seq, target in sequential_data:
47         if target == 0:
48             sells.append([seq, target])
49         elif target == 1:
50             buys.append([seq, target])
51     random.shuffle(buys)
52     random.shuffle(sells)
53
54     # Look equal number of sells and buys
55     lower = min(len(buys), len(sells))
56     buys = buys[:lower]
57     sells = sells[:lower]
58     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]



IPython console

Console 1/A

Price

VWAP

100 SMA

200 SMA

9 EMA

20 EMA

IPython console | File explorer | Help

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](mailto:Yesser.Nasser@icloud.com)