Time Series Analysis of Day 2

Simple Moving Average Using different window size

Comparison of Different plots of Simple Moving Average Using different window size

Disadvantages of Simple Moving Average

Cumulative Moving Average (Expanding)

Formula of Exponential Weighted Moving Average & their different parameters

Exponential Weighted Moving Average Using different alpha size

Exponential Weighted Moving Average Using span Parameter

Comparison of Different plots of Exponential Weighted moving Average Using different alpha size & Span for specific period using xlim

Comparison of Different plots of Exponential Weighted moving Average with Simple Moving Average

How Exponential Weighted moving Average is better than other Moving averages

How Exponential Weighted moving Average is smoothern the Cyclic Data (Their is lot of Upword & Downword directional Data) than the other Moving averages

In [2]:
```
import pandas_datareader as pdr
import pandas as pd
from datetime import datetime
```

In [3]:
```
df_tesla = pdr.get_data_yahoo('TSLA')
df_tesla
```

Out[3]:

Date	High	Low	Open	Close	Volume	Adj Close
2017-11-03	20.416668	19.675333	19.966667	20.406000	133410000.0	20.406000
2017-11-06	20.500000	19.934000	20.466667	20.185333	97290000.0	20.185333
2017-11-07	20.433332	20.002001	20.068001	20.403334	79414500.0	20.403334
2017-11-08	20.459333	20.086666	20.366667	20.292667	70879500.0	20.292667
2017-11-09	20.297333	19.753332	20.166668	20.199333	81706500.0	20.199333
...	...	...	...	...	...	...
2022-10-27	233.809998	222.850006	229.770004	225.089996	61638800.0	225.089996
2022-10-28	228.860001	216.350006	225.399994	228.520004	69152400.0	228.520004
2022-10-31	229.850006	221.940002	226.190002	227.539993	61554300.0	227.539993
2022-11-01	237.399994	227.279999	234.050003	227.820007	62688800.0	227.820007
2022-11-02	227.869995	214.820007	226.039993	214.979996	62887200.0	214.979996

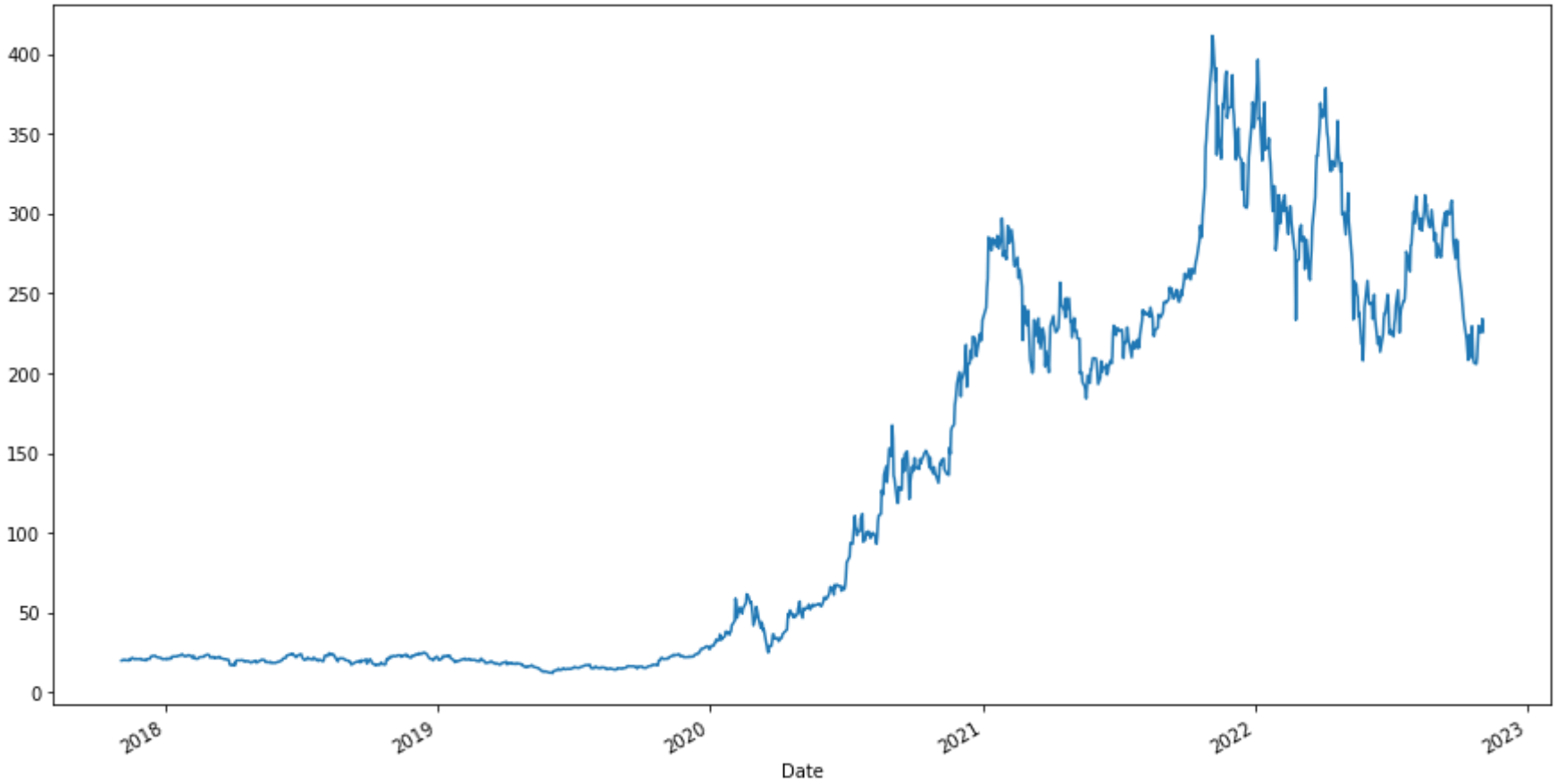
1258 rows × 6 columns

Simple Moving Average

A simple moving average tells us the unweighted mean of the previous K data points.

In [4]:
```
df_tesla['Open'].plot(figsize=(15,8))
```

Out[4]:
```
<AxesSubplot:xlabel='Date'>
```



Rolling Parameter Information

In [5]:
```
# https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.rolling.html
```

Open : 10 days rolling

In [6]:
```
df_tesla['Open : 10 days rolling'] = df_tesla['Open'].rolling(window=10, min_periods=1).mean()
```

In [7]:
```
df_tesla.head()
```

Out[7]:

Date	High	Low	Open	Close	Volume	Adj Close	Open : 10 days rolling
2017-11-03	20.416668	19.675333	19.966667	20.406000	133410000.0	20.406000	19.966667
2017-11-06	20.500000	19.934000	20.466667	20.185333	97290000.0	20.185333	20.216667
2017-11-07	20.433332	20.002001	20.068001	20.403334	79414500.0	20.403334	20.167112
2017-11-08	20.459333	20.086666	20.366667	20.292667	70879500.0	20.292667	20.217000
2017-11-09	20.297333	19.753332	20.166668	20.199333	81706500.0	20.199333	20.206934

In [8]:
```
df_tesla[['Open','Open : 10 days rolling']]
```

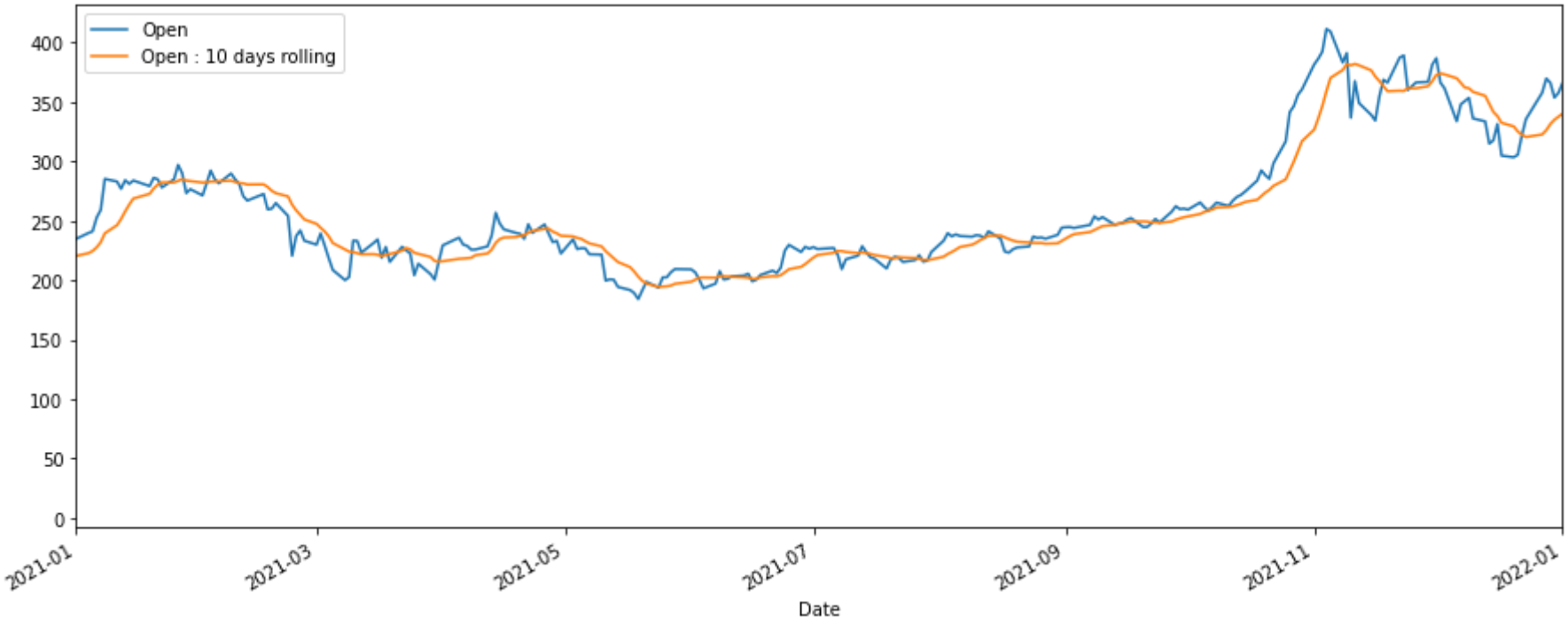
Out[8]:

Date	Open	Open : 10 days rolling
2017-11-03	19.966667	19.966667
2017-11-06	20.466667	20.216667
2017-11-07	20.068001	20.167112
2017-11-08	20.366667	20.217000
2017-11-09	20.166668	20.206934
...	...	...
2022-10-27	229.770004	216.314000
2022-10-28	225.399994	216.453000
2022-10-31	226.190002	218.068001
2022-11-01	234.050003	218.523001
2022-11-02	226.039993	219.147000

1258 rows × 2 columns

In [9]:
```
df_tesla[['Open','Open : 10 days rolling']].plot(xlim=['2021-01-01', '2022-01-01'], figsize=(15, 6))
```

Out[9]: <AxesSubplot:xlabel='Date'>



Open : 30 days rolling

In [10]:
```
df_tesla['Open : 30 days rolling'] = df_tesla['Open'].rolling(window=30, min_periods=1).mean()
```

Open : 50 days rolling

In [11]:
```
df_tesla['Open : 50 days rolling'] = df_tesla['Open'].rolling(window=50, min_periods=1).mean()
```

Plotting of Open, Open 10, Open 30 and Open 50 Days Rolling

In [12]:
```
df_tesla[['Open', 'Open : 10 days rolling', 'Open : 30 days rolling', 'Open : 50 days rolling']]
```

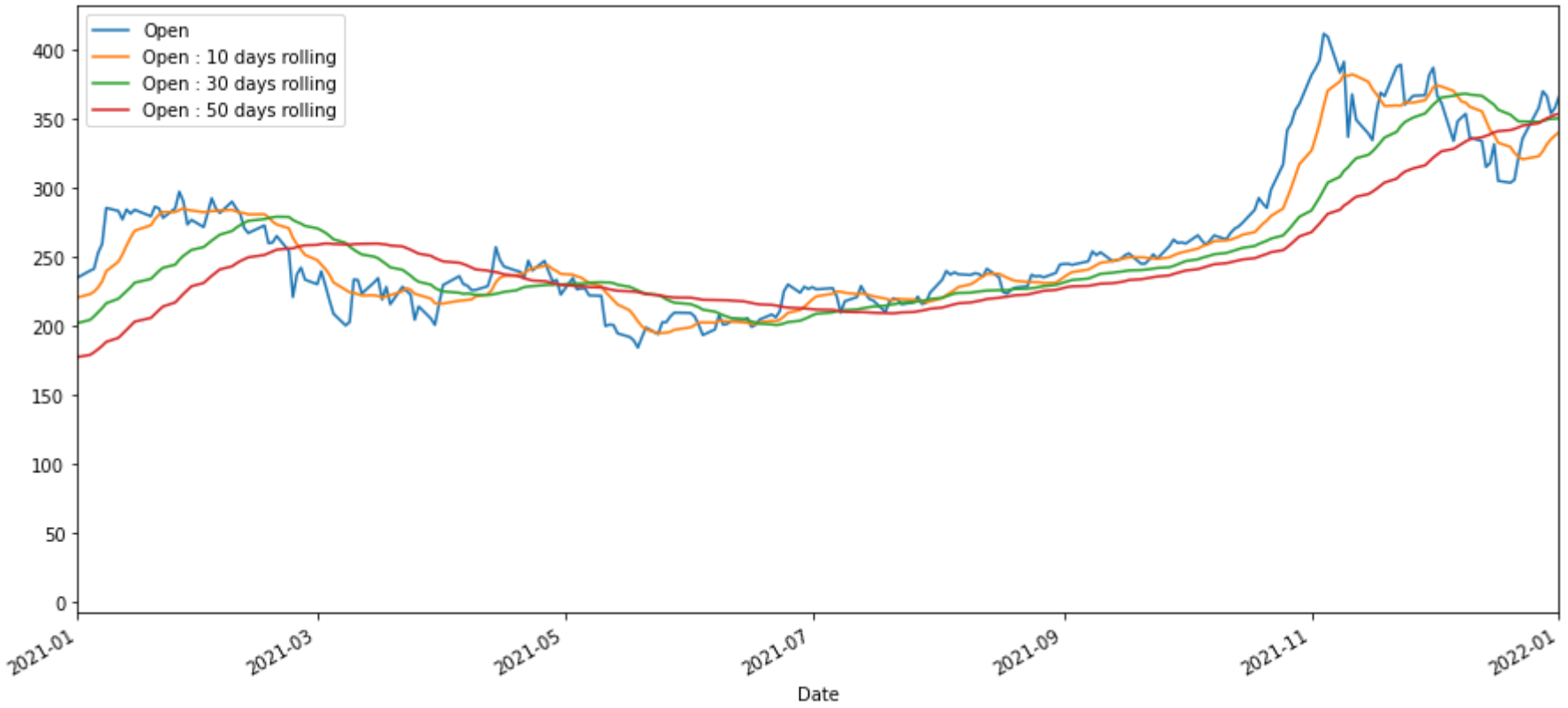
Out[12]:

Date	Open	Open : 10 days rolling	Open : 30 days rolling	Open : 50 days rolling
2017-11-03	19.966667	19.966667	19.966667	19.966667
2017-11-06	20.466667	20.216667	20.216667	20.216667
2017-11-07	20.068001	20.167112	20.167112	20.167112
2017-11-08	20.366667	20.217000	20.217000	20.217000
2017-11-09	20.166668	20.206934	20.206934	20.206934
...	...	...	...	...
2022-10-27	229.770004	216.314000	248.018998	264.753997
2022-10-28	225.399994	216.453000	245.545332	263.141997
2022-10-31	226.190002	218.068001	243.081999	261.685797
2022-11-01	234.050003	218.523001	240.653332	260.528531
2022-11-02	226.039993	219.147000	237.911665	259.220264

1258 rows × 4 columns

In [13]:
```
df_tesla[['Open','Open : 10 days rolling', 'Open : 30 days rolling', 'Open : 50 days rolling']].plot(xlim=['2021-01-01',
```

Out[13]: <AxesSubplot:xlabel='Date'>



Disadvantages of Simple Moving Average

1) Similar importance to all the Data [(x1 + x2 + x3...xn)/N]

```
Time Series => Recent Data => More Weight to Recent data, that weight provided by the some Parameters (We will See in EWMA)
```

Cumulative Moving average (Expanding)

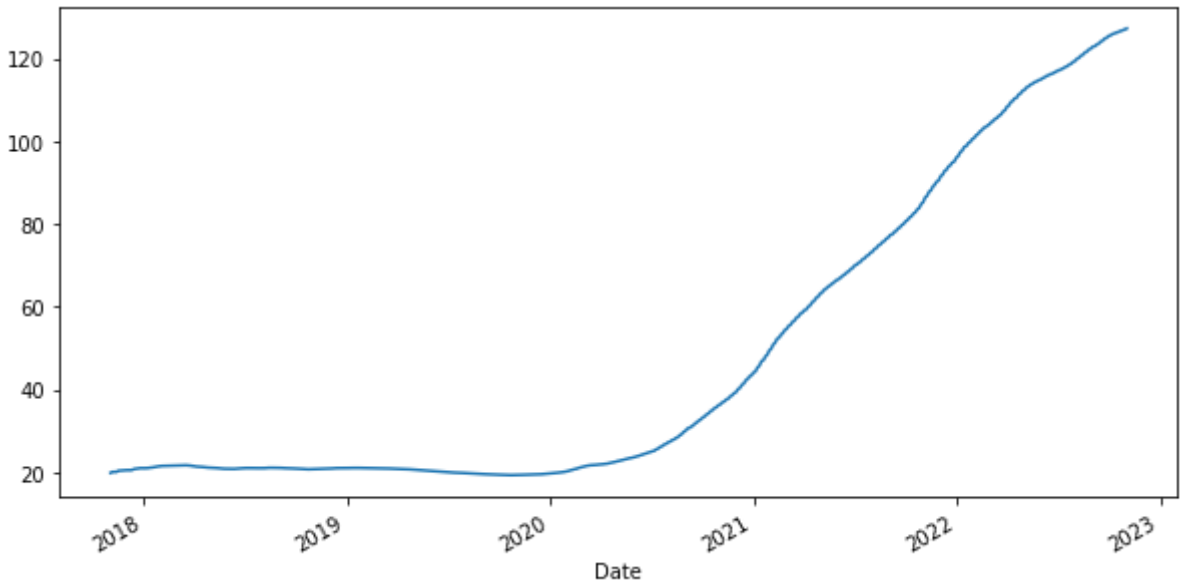
The Cumulative Moving Average is the mean of all the previous values up to the current value.

In [14]:
```
df_tesla['Open'].expanding().mean().head()
```

Out[14]:
```
Date
2017-11-03    19.966667
2017-11-06    20.216667
2017-11-07    20.167112
2017-11-08    20.217000
2017-11-09    20.206934
Name: Open, dtype: float64
```

In [15]:
```
df_tesla['Open'].expanding().mean().plot(figsize=(10, 5))
```

Out[15]: <AxesSubplot:xlabel='Date'>



Exponential Moving Average

```
EMA  =  [(Column - Prev EMA) * Multiplier]  + Prev EMA


        Where Column = To be decide by ourself on which column we have to calculate EMA

        Where Multiplier = [2 / (Window Size)+1 ]
```

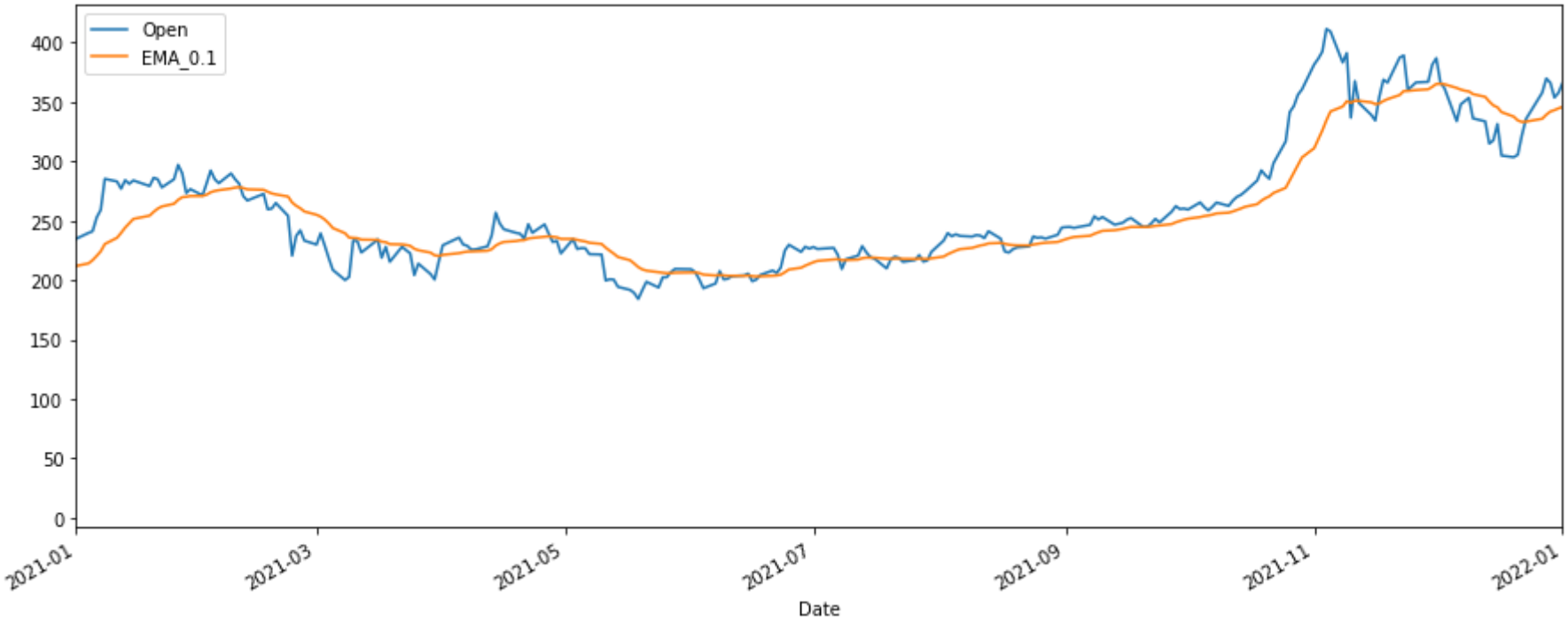
```
EWMA(t) = a * x(t) + ( 1 - a ) EWMA(t-1)
```

EWM for Open alpha 0.1

In [16]:
```
df_tesla['EMA_0.1'] = df_tesla['Open'].ewm(alpha=0.1, adjust=False).mean()
```

In [17]:
```
df_tesla[['Open' , 'EMA_0.1']].plot(xlim=['2021-01-01', '2022-01-01'], figsize=(15, 6))
```

Out[17]: <AxesSubplot:xlabel='Date'>

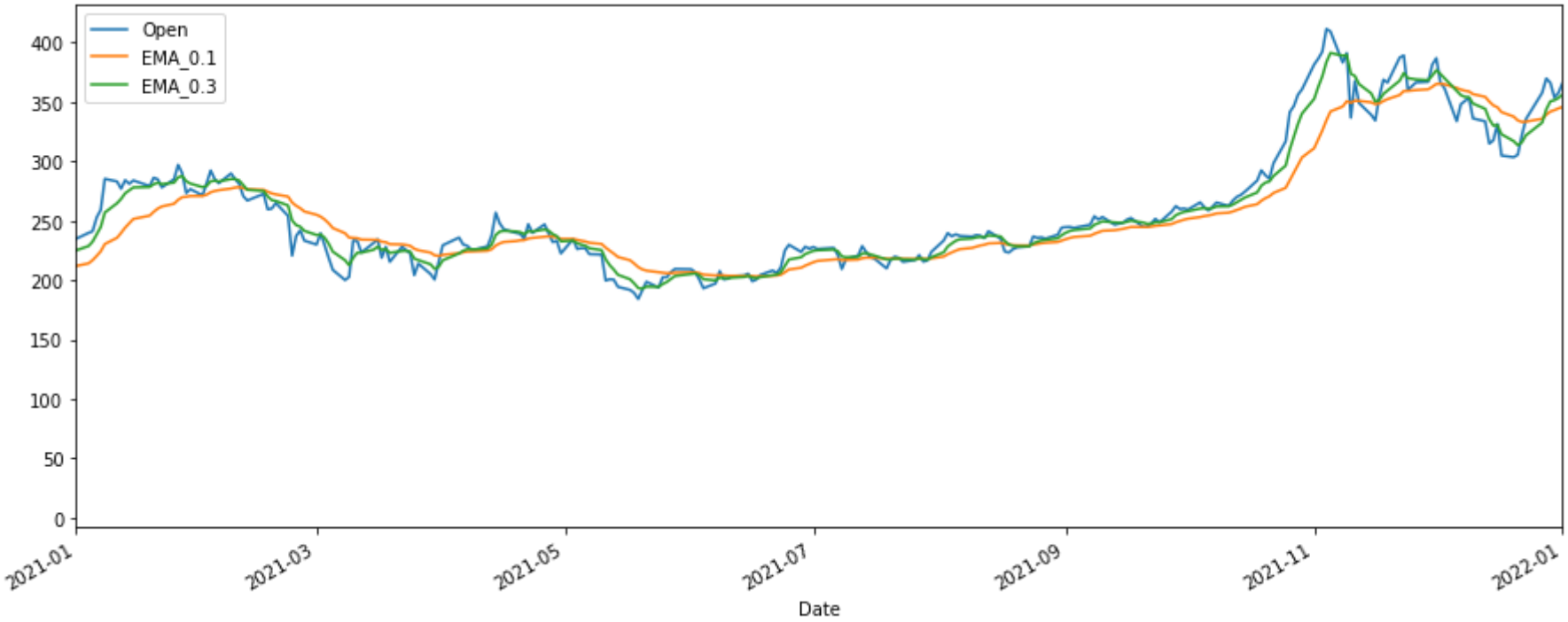


EWM for Open alpha 0.3

In [18]:
```
df_tesla['EMA_0.3'] = df_tesla['Open'].ewm(alpha=0.3, adjust=False).mean()
```

In [19]:
```
df_tesla[['Open' ,'EMA_0.1' , 'EMA_0.3']].plot(xlim=['2021-01-01', '2022-01-01'], figsize=(15, 6))
```

Out[19]: <AxesSubplot:xlabel='Date'>



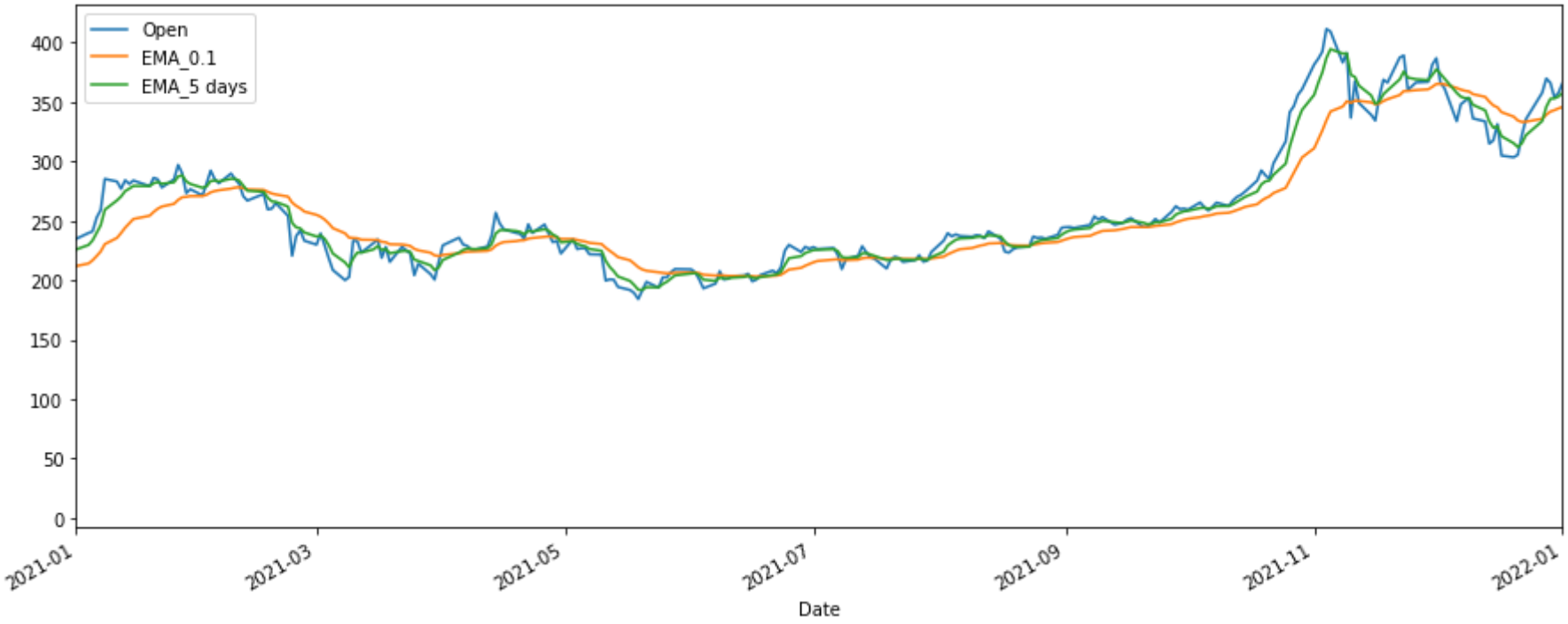
EWM parameter Span

In [20]:
```
# df_tesla['EMA_5 days'] = df_tesla['Open'].ewm(span =5, alpha=0.3, adjust=False).mean()
```

In [21]:
```
df_tesla['EMA_5 days'] = df_tesla['Open'].ewm(span=5).mean()
```

In [22]:
```
df_tesla[['Open' ,'EMA_0.1' , 'EMA_5 days']].plot(xlim=['2021-01-01', '2022-01-01'], figsize=(15, 6))
```

Out[22]: <AxesSubplot:xlabel='Date'>



ARIMA

```
AR (Autoregression) +    I   +   MA ( Moving Average)


ARIMA is used for Forcasting of Sales and Monthly Production (Dont used for Stocks)
```

Interview Questions on Time Series

```
1) For moving Average do u used PACF plot OR ACF Plot ??
```

```
2) Where PACF is Used

3) Where is ACF Plot Used...???

3) Where PACF used for Moving Average ???

4) Where ACF used for Autoregression ????
```

Moving Average Models

```
Ma = q + $ e(t-1)
```