

# 03-Rolling-and-Expanding

October 19, 2022

---

Copyright Pierian Data

For more information, visit us at [www.pieriandata.com](http://www.pieriandata.com)

## 1 Rolling and Expanding

A common process with time series is to create data based off of a rolling mean. The idea is to divide the data into "windows" of time, and then calculate an aggregate function for each window. In this way we obtain a simple moving average. Let's show how to do this easily with pandas!

```
[1]: import pandas as pd
      %matplotlib inline
```

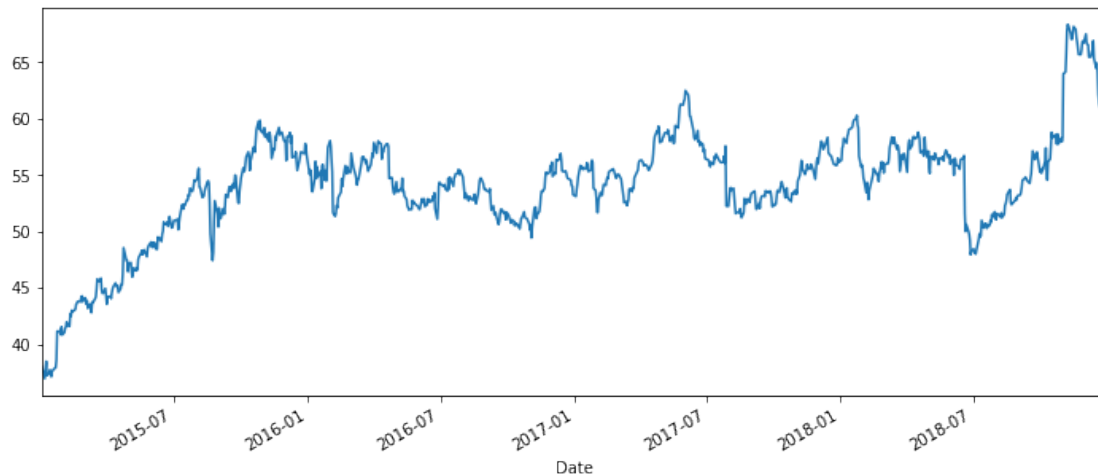
```
[2]: # Import the data:
      df = pd.read_csv('../Data/starbucks.csv', index_col='Date', parse_dates=True)
```

```
[3]: df.head()
```

```
[3]:
```

	Close	Volume
Date		
2015-01-02	38.0061	6906098
2015-01-05	37.2781	11623796
2015-01-06	36.9748	7664340
2015-01-07	37.8848	9732554
2015-01-08	38.4961	13170548

```
[4]: df['Close'].plot(figsize=(12,5)).autoscale(axis='x',tight=True);
```



Now let's add in a rolling mean! This rolling method provides row entries, where every entry is then representative of the window.

```
[5]: # 7 day rolling mean
df.rolling(window=7).mean().head(15)
```

```
[5]:
```

	Close	Volume
Date		
2015-01-02	NaN	NaN
2015-01-05	NaN	NaN
2015-01-06	NaN	NaN
2015-01-07	NaN	NaN
2015-01-08	NaN	NaN
2015-01-09	NaN	NaN
2015-01-12	37.616786	1.238222e+07
2015-01-13	37.578786	1.297288e+07
2015-01-14	37.614786	1.264020e+07
2015-01-15	37.638114	1.270624e+07
2015-01-16	37.600114	1.260380e+07
2015-01-20	37.515786	1.225634e+07
2015-01-21	37.615786	9.868837e+06
2015-01-22	37.783114	1.185335e+07
2015-01-23	38.273129	1.571999e+07

```
[6]: df['Close'].plot(figsize=(12,5)).autoscale(axis='x',tight=True)
df.rolling(window=30).mean()['Close'].plot();
```



The easiest way to add a legend is to make the rolling value a new column, then pandas does it automatically!

```
[7]: df['Close: 30 Day Mean'] = df['Close'].rolling(window=30).mean()
df[['Close', 'Close: 30 Day Mean']].plot(figsize=(12,5)).
    ↪autoscale(axis='x',tight=True);
```

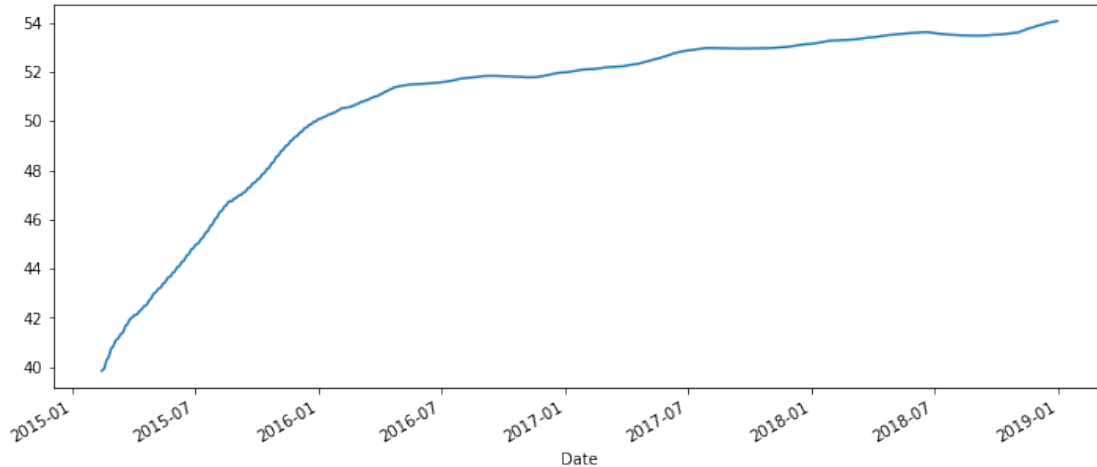


## 1.1 Expanding

Instead of calculating values for a rolling window of dates, what if you wanted to take into account everything from the start of the time series up to each point in time? For example, instead of considering the average over the last 7 days, we would consider all prior data in our expanding set of averages.

```
[10]: # df['Close'].plot(figsize=(12,5)).autoscale(axis='x',tight=True)

# Optional: specify a minimum number of periods to start from
df['Close'].expanding(min_periods=30).mean().plot(figsize=(12,5));
```



That's it! It doesn't help much to visualize an expanding operation against the daily data, since all it really gives us is a picture of the "stability" or "volatility" of a stock. However, if you do want to see it, simply uncomment the first plot line above and rerun the cell.

Next up, we'll take a deep dive into visualizing time series data!