Rolling window features: part 3

Window features

pandas.DataFrame.rolling

DataFrame. rolling(window, min_periods=None, center=False, win_type=None, on=None, axis=0, closed=None)

[source]

Provide rolling window calculations.

Parameters: window: int, offset, or BaseIndexer subclass

Size of the moving window. This is the number of observations used for calculating the statistic. Each window will be a fixed size.

If its an offset then this will be the time period of each window. Each window will be a variable sized based on the observations included in the time-period. This is only valid for datetimelike indexes.

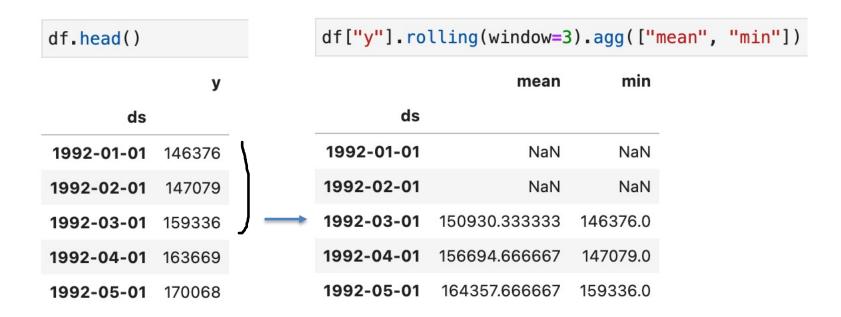
If a BaseIndexer subclass is passed, calculates the window boundaries based on the defined get_window_bounds method. Additional rolling keyword arguments, namely min_periods, center, and closed will be passed to get_window_bounds.

min_periods: int, default None

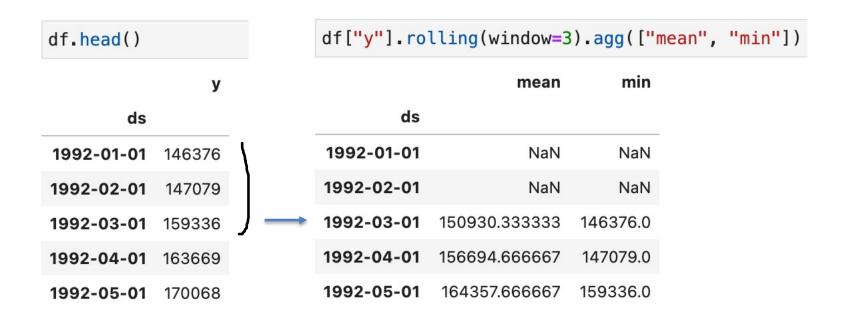
Minimum number of observations in window required to have a value (otherwise result is NA). For a window that is specified by an offset, *min_periods* will default to 1. Otherwise, *min_periods* will default to the size of the window.

center: bool, default False

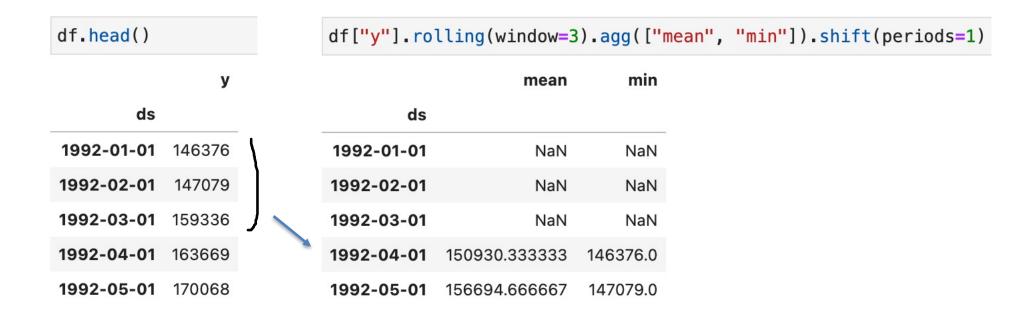
Set the labels at the center of the window.



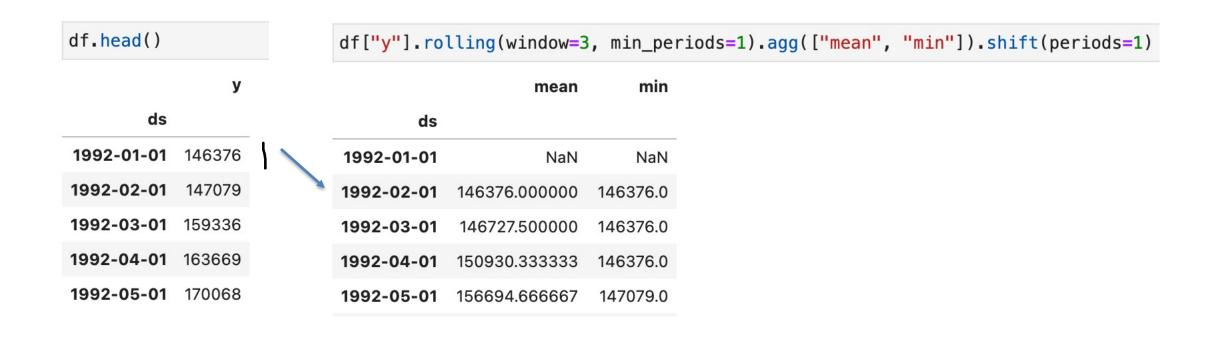
The `rolling` method by default assigns the rolling statistics to the edge of the window.

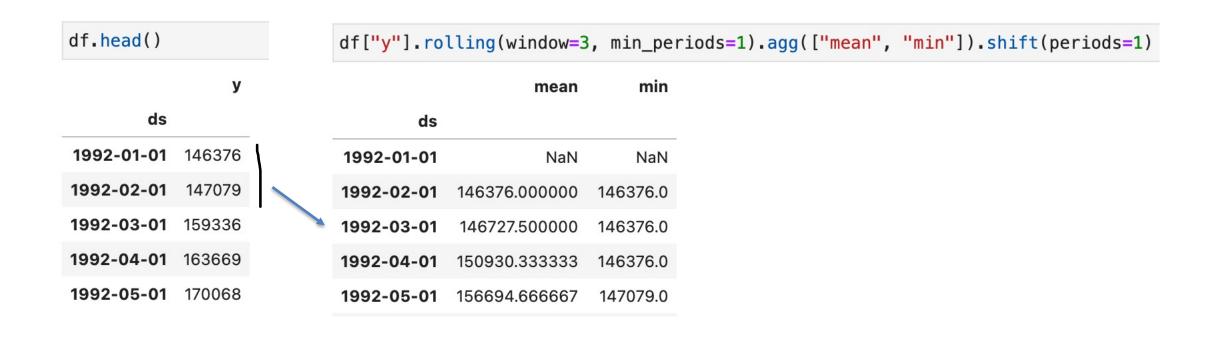


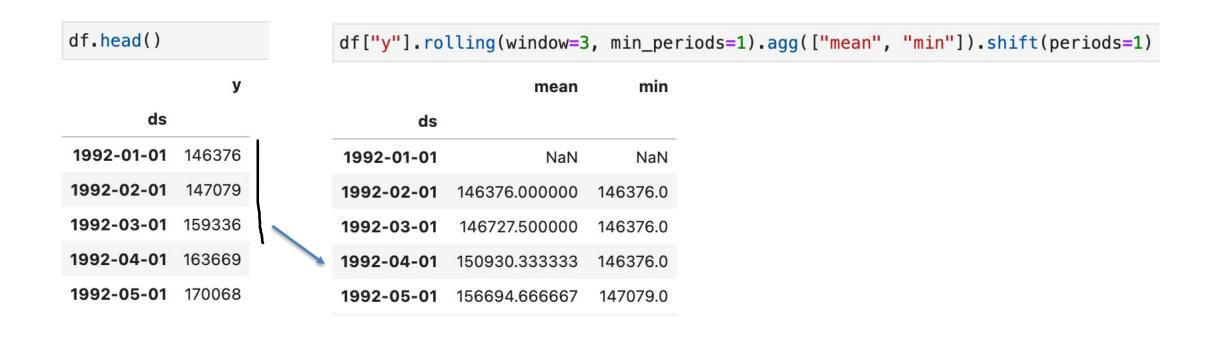
For forecasting we want features which only use information that we will have at predict time (i.e, the past). So we want to shift the output of the row down by one to avoid data leakage.

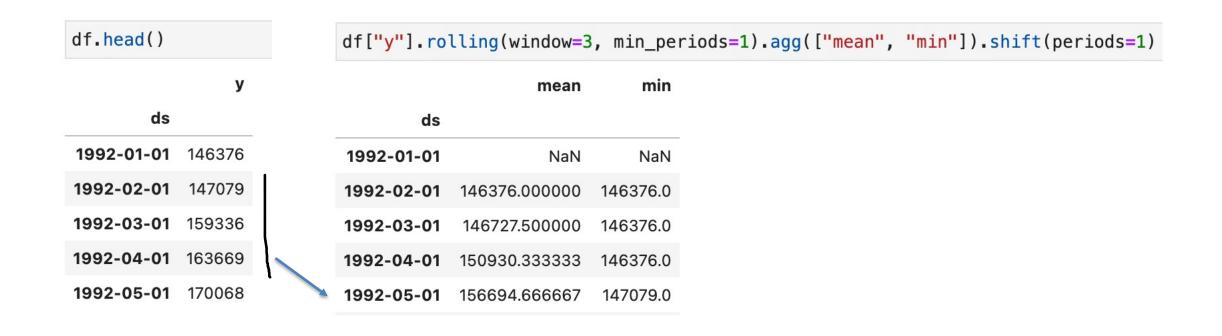


For forecasting we want features which only use information that we will have at predict time (i.e, the past). So we want to shift the output of the row down by one to avoid data leakage.









Rolling window features: Feature-engine

	У	y_window_1_mean	y_window_1_std	y_window_3_mean	y_window_3_std	y_window_6_mean	y_window_6_std
ds							
2016-01-01	400928	518253.00	NaN	469239.67	42447.39	458781.00	30709.03
2016-02-01	413554	400928.00	NaN	454562.67	59305.36	449317.33	38790.92
2016-03-01	460093	413554.00	NaN	444245.00	64402.97	442186.33	41105.40
2016-04-01	450935	460093.00	NaN	424858.33	31160.32	447049.00	41231.17
2016-05-01	471421	450935.00	NaN	441527.33	24654.57	448045.00	41242.76

Rolling window features: Feature-engine

	у	y_window_1_mean	y_window_1_std	y_window_3_mean	y_window_3_std	y_window_6_mean	y_window_6_std
ds							
2016-01-01	400928	518253.00	NaN	469239.67	42447.39	458781.00	30709.03
2016-02-01	413554	400928.00	NaN	454562.67	59305.36	449317.33	38790.92
2016-03-01	460093	413554.00	NaN	444245.00	64402.97	442186.33	41105.40
2016-04-01	450935	460093.00	NaN	424858.33	31160.32	447049.00	41231.17
2016-05-01	471421	450935.00	NaN	441527.33	24654.57	448045.00	41242.76

Rolling window features: Feature-engine

	У	y_window_1_mean	y_window_1_std	y_window_3_mean	y_window_3_std	y_window_6_mean	y_window_6_std
ds							
2016-01-01	400928	518253.00	NaN	469239.67	42447.39	458781.00	30709.03
2016-02-01	413554	400928.00	NaN	454562.67	59305.36	449317.33	38790.92
2016-03-01	460093	413554.00	NaN	444245.00	64402.97	442186.33	41105.40
2016-04-01	450935	460093.00	NaN	424858.33	31160.32	447049.00	41231.17
2016-05-01	471421	450935.00	NaN	441527.33	24654.57	448045.00	41242.76

```
from sktime.transformations.series.summarize import WindowSummarizer
transformer = WindowSummarizer(
    lag_feature={
        "lag": [1, 2] # Create lag features
        "mean": [[1, 3], [3, 6]], # [[lag, window size], ...]
        "std": [[1, 3]],
    target_cols=["y"],
transformer.fit_transform(df)
```

	у			y_lag_1	y_lag_2	y_mean_1_3	y_mean_3_6	y_std_1_4
ds			ds					
1992-01-01	146376		1992-01-01	NaN	NaN	NaN	NaN	NaN
1992-02-01	147079		1992-02-01	146376.00	NaN	NaN	NaN	NaN
1992-03-01	159336		1992-03-01	147079.00	146376.00	NaN	NaN	NaN
1992-04-01	163669		1992-04-01	159336.00	147079.00	150930.33	NaN	NaN
1992-05-01	170068		1992-05-01	163669.00	159336.00	156694.67	NaN	8716.56
•••			•••					
2016-01-01	400928		2016-01-01	518253.00	444507.00	469239.67	450128.33	39602.33
2016-02-01	413554		2016-02-01	400928.00	518253.00	454562.67	447110.33	48660.13
2016-03-01	460093		2016-03-01	413554.00	400928.00	444245.00	458781.00	52584.96
2016-04-01	450935		2016-04-01	460093.00	413554.00	424858.33	449317.33	53178.48
2016-05-01	471421		2016-05-01	450935.00	460093.00	441527.33	442186.33	28588.60

	у			y_lag_1	y_lag_2	y_mean_1_3	y_mean_3_6	y_std_1_4
ds			ds					
1992-01-01	146376		1992-01-01	NaN	NaN	NaN	NaN	NaN
1992-02-01	147079		1992-02-01	146376.00	NaN	NaN	NaN	NaN
1992-03-01	159336		1992-03-01	147079.00	146376.00	NaN	NaN	NaN
1992-04-01	163669		1992-04-01	159336.00	147079.00	150930.33	NaN	NaN
1992-05-01	170068		1992-05-01	163669.00	159336.00	156694.67	NaN	8716.56
•••			•••					
2016-01-01	400928		2016-01-01	518253.00	444507.00	469239.67	450128.33	39602.33
2016-02-01	413554		2016-02-01	400928.00	518253.00	454562.67	447110.33	48660.13
2016-03-01	460093		2016-03-01	413554.00	400928.00	444245.00	458781.00	52584.96
2016-04-01	450935		2016-04 01	460093.00	413554.00	424858.33	449317.33	53178.48
2016-05-01	471421		2016-05-01	450935.00	460093.00	441527.33	442186.33	28588.60

	у			y_lag_1	y_lag_2	y_mean_1_3	y_mean_3_6	y_std_1_4
ds			ds					
1992-01-01	146376		1992-01-01	NaN	NaN	NaN	NaN	NaN
1992-02-01	147079		1992-02-01	146376.00	NaN	NaN	NaN	NaN
1992-03-01	159336		1992-03-01	147079.00	146376.00	NaN	NaN	NaN
1992-04-01	163669		1992-04-01	159336.00	147079.00	150930.33	NaN	NaN
1992-05-01	170068		1992-05-01	163669.00	159336.00	156694.67	NaN	8716.56
			•••					
2016-01-01	400928		2016-01-01	518253.00	444507.00	469239.67	450128.33	39602.33
2016-02-01	413554	1	2016-02-01	400928.00	518253.00	454562.67	447110.33	48660.13
2016-03-01	460093		2016-03-01	413554.00	400928.00	444245.00	458781.00	52584.96
2016-04-01	450935		2016-04-01	460093.00	413554.00	424858.33	449317.33	53178.48
2016-05-01	471421		2016-05-01	450935.00	460093.00	441527.33	442186.33	28588.60

	у			y_lag_1	y_lag_2	y_mean_1_3	y_mean_3_6	y_std_1_4
ds			ds					
1992-01-01	146376		1992-01-01	NaN	NaN	NaN	NaN	NaN
1992-02-01	147079		1992-02-01	146376.00	NaN	NaN	NaN	NaN
1992-03-01	159336		1992-03-01	147079.00	146376.00	NaN	NaN	NaN
1992-04-01	1-01 163669		1992-04-01	159336.00	147079.00	150930.33	NaN	NaN
1992-05-01	170068		1992-05-01	163669.00	159336.00	156694.67	NaN	8716.56
•••		1	•••					•••
2016-01-01	400928		2016-01-01	518253.00	444507.00	469239.67	450128.33	39602.33
2016-02-01	413554		2016-02-01	400928.00	518253.00	454562.67	447110.33	48660.13
2016-03-01	460093		2016-03-01	413554.00	400928.00	444245.00	458781.00	52584.96
2016-04-01	450935		2016-04-01	460093.00	413554.00	424858.33	449317.33	53178.48
2016-05-01	471421		2016-05-01	450935.00	460093.00	441527.33	442186.33	28588.60

Summary

New features can be created by applying a rolling window to the target or other features.

The window features still need to be lagged to ensure there is no data leakage.

Multiple different window sizes could be helpful. The seasonal period and different time scales can be a good starting point.

Mean and standard deviation are common. Use feature selection methods for more statistics.