Overview

This document covers common patterns in time series analysis, and implementation in certain libraries.

Here we separate time series functionalities into two categories:

(1) Data manipulation. These are relational-like functionalities on time series data, such as join, window, aggregation and etc.
(2) Time series models and techniques. These are specific stats and ML techniques used on time series data, such as AR, ARMA, ARIMA and etc.

This is similar to Spark SQL vs MLlib and *pandas* vs *statsmodels*, the former is often used for data cleaning and prepare inputs for the latter.

This documentation focus on functionalities in category (1).

This doc currently covers the following libraries:

- Spark SQL
- Pandas
- Flint (Time series library on Spark): https://github.com/twosigma/flint
- Kdb (A widely used time series database in finance): http://code.kx.com/q/

Gourav Sengupta: Asof Join

Asof Join is a left Join and inexact matching on join key. Rather than matching equal keys, asof Join joins matches keys that are nearest. In the context of time series analysis, asof join is often used for matching nearest points in time.

Existing implementations

Spark
None

Pandas
https://pandas.pydata.org/pandas-docs/stable/generated/pandas.merge_asof.html

Flint
http://ts-flint.readthedocs.io/en/latest/reference.html?highlight=summarizeIntervals#ts.flint.TimeSeriesDataFrame.leftJoin

Kdb
https://code.kx.com/wiki/Reference/aj

Window Function

Window functions operate on a set of rows and return a single value for each row. The term window describes the set of rows on which the function operates. A window function uses values from the rows in a window to calculate the returned values.

In time series analysis, window functions are used to compute certain value for each row using the rows around it (in time). Window functions can be used for moving average, exponential smoothing and etc.

Existing implementations

Spark
Window functions (with the limitation that a partition key must be provided)
https://databricks.com/blog/2015/07/15/introducing-window-functions-in-spark-sql.html

Pandas
https://spark.apache.org/docs/latest/api/python/pyspark.sql.html#pyspark.sql.functions.window
https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.rolling.html#pandas-dataframe-rolling

Flint
http://ts-flint.readthedocs.io/en/latest/reference.html?highlight=summarizeIntervals#ts.flint.TimeSeriesDataFrame.summarizeWindows

Rolling Window Function

Rolling window functions are a special case of groupby, where groups are defined by a rolling window. A rolling window can be a rolling time window or a rolling count window. Rolling window can be used for, e.g., rolling window regression.

Rolling window is similar to window functions. The difference is:

- With window functions, each input is associated with a window. The output size is the same as the input.
- With rolling window functions, the output size is the same as number of windows defined by the window function, not the input size.

Spark doc explains rolling time window quite nicely:
http://spark.apache.org/docs/2.1.0/api/python/pyspark.sql.html#pyspark.sql.functions.window

Existing implementations

Spark
http://spark.apache.org/docs/2.1.0/api/python/pyspark.sql.html#pyspark.sql.functions.window

Pandas
None

Flint
None

Resample

In time series analysis, resample is used for changing the frequency of the data. There are two types of resample, upsample and downsample.

Upsample means resample the data to a higher frequency, e.g. monthly to daily. Upsample uses interpolation to fill the missing data, e.g., linear interpolation.

Downsample means resample the data to a lower frequency, e.g., daily to monthly. Downsample uses aggregation to aggregate multiple samples to one sample, e.g., mean.

This article explains upsampling and downsampling quite nicely:
https://machinelearningmastery.com/resample-interpolate-time-series-data-python

Existing implementations

Spark
groupby for downsample

Pandas
resample function for both upsample and downsample.
https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.resample.html

groupby can also be used for downsample

Flint
`summarizeIntervals` for downsample:
http://ts-flint.readthedocs.io/en/latest/reference.html?highlight=summarizeIntervals#ts.flint.TimeSeriesDataFrame.summarizeIntervals

Question:

- Do we need a new resample function in Spark or is groupby sufficient?
- Time series resample maintains time order, i.e., after resample, data are still ordered by the new time column. Can we have groupby maintain the same property when used for resampling?

Time Shift

Time shift is a operator that shifts values in a time series back and forth in time. Different from "changing timestamp", time shift has special property:

- An ordered series is still ordered after time shift
- If the time bound of a series is known before time shift, the time bound is still know after the shift, and can be computed by shifting the time bounds.

Time shift is useful for autoregression, among many other things.

Existing implementations

Spark
withColumn. But doesn't preserve the property of a time shift.

Pandas
https://pandas.pydata.org/pandas-docs/stable/timeseries.html#shifting-lagging

Flint
http://ts-flint.readthedocs.io/en/latest/reference.html?highlight=summarizeIntervals#ts.flint.TimeSeriesDataFrame.shiftTime

Accumulation

Accumulation is a sequence of partial aggregation of a given sequence. A simple example of accumulation is cumulative sum.

Existing implementations

Spark
Window functions (with the limitation that a partition key must be provided)

Pandas
expanding
https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.expanding.html

Flint
http://ts-flint.readthedocs.io/en/latest/reference.html?highlight=summarizeIntervals#ts.flint.TimeSeriesDataFrame.addSummaryColumns