resample_data

June 14, 2020

1 Resample Data

1.1 Pandas Resample

You've learned about bucketing to different periods of time like Months. Let's see how it's done. We'll start with an example series of days.

```
In [19]: import numpy as np
         import pandas as pd

         dates = pd.date_range('10/06/2020', periods=11, freq='D')
         close_prices = np.arange(len(dates))

         close = pd.Series(close_prices, dates)
         close
```

```
Out[19]: 2020-10-06     0
         2020-10-07     1
         2020-10-08     2
         2020-10-09     3
         2020-10-10     4
         2020-10-11     5
         2020-10-12     6
         2020-10-13     7
         2020-10-14     8
         2020-10-15     9
         2020-10-16    10
         Freq: D, dtype: int64
```

Let's say we want to bucket these days into 3 day periods. To do that, we'll use the DataFrame.resample function. The first parameter in this function is a string called `rule`, which is a representation of how to resample the data. This string representation is made using an offset alias. You can find a list of them here. To create 3 day periods, we'll set `rule` to "3D".

```
In [11]: close.resample('3D')
```

```
Out[11]: DatetimeIndexResampler [freq=<3 * Days>, axis=0, closed=left, label=left, convention=st
```

This returns a `DatetimeIndexResampler` object. It's an intermediate object similar to the `GroupBy` object. Just like group by, it breaks the original data into groups. That means, we'll have to apply an operation to these groups. Let's make it simple and get the first element from each group.

```
In [12]: close.resample('3D').first()
```

```
Out[12]: 2020-10-06    0
         2020-10-09    3
         2020-10-12    6
         2020-10-15    9
         dtype: int64
```

You might notice that this is the same as `.iloc[::3]`

```
In [13]: close.iloc[::3]
```

```
Out[13]: 2020-10-06    0
         2020-10-09    3
         2020-10-12    6
         2020-10-15    9
         Freq: 3D, dtype: int64
```

So, why use the `resample` function instead of `.iloc[::3]` or the `groupby` function?

The `resample` function shines when handling time and/or date specific tasks. In fact, you can't use this function if the index isn't a time-related class.

```
In [14]: try:
             # Attempt resample on a series without a time index
             pd.Series(close_prices).resample('W')
         except TypeError:
             print('It threw a TypeError.')
         else:
             print('It worked.')
```

```
It threw a TypeError.
```

One of the resampling tasks it can help with is resampling on periods, like weeks. Let's resample `close` from it's days frequency to weeks. We'll use the "W" offset allies, which stands for Weeks.

```
In [15]: pd.DataFrame({
             'days': close,
             'weeks': close.resample('W').first()})
```

```
Out[15]:             days  weeks
         2020-10-06   0.0    NaN
         2020-10-07   1.0    NaN
         2020-10-08   2.0    NaN
```

```
2020-10-09   3.0    NaN
2020-10-10   4.0    NaN
2020-10-11   5.0    0.0
2020-10-12   6.0    NaN
2020-10-13   7.0    NaN
2020-10-14   8.0    NaN
2020-10-15   9.0    NaN
2020-10-16  10.0    NaN
2020-10-18   NaN    6.0
```

The weeks offset considers the start of a week on a Monday. Since 2018-10-10 is a Wednesday, the first group only looks at the first 5 items. There are offsets that handle more complicated problems like filtering for Holidays. For now, we'll only worry about resampling for days, weeks, months, quarters, and years. The frequency you want the data to be in, will depend on how often you'll be trading. If you're making trade decisions based on reports that come out at the end of the year, we might only care about a frequency of years or months. ## OHLC Now that you've seen how Pandas resamples time series data, we can apply this to Open, High, Low, and Close (OHLC). Pandas provides the `Resampler.ohlc` function will convert any resampling frequency to OHLC data. Let's get the Weekly OHLC.

```
In [20]: close.resample('W').ohlc()
```

```
Out[20]:             open  high  low  close
         2020-10-11     0     5    0      5
         2020-10-18     6    10    6     10
```

Can you spot a potential problem with that? It has to do with resampling data that has already been resampled.

We're getting the OHLC from close data. If we want OHLC data from already resampled data, we should resample the first price from the open data, resample the highest price from the high data, etc..

To get the weekly closing prices from `close`, you can use the `Resampler.last` function.

```
In [21]: close.resample('W').last()
```

```
Out[21]: 2020-10-11     5
         2020-10-18    10
         Freq: W-SUN, dtype: int64
```

1.2 Quiz

Implement `days_to_weeks` function to resample OHLC price data to weekly OHLC price data. You find find more Resampler functions here for calculating high and low prices.

```
In [23]: import quiz_tests
```

```
def days_to_weeks(open_prices, high_prices, low_prices, close_prices):
    """Converts daily OHLC prices to weekly OHLC prices.
```

```
        Parameters
        ----------
        open_prices : DataFrame
            Daily open prices for each ticker and date
        high_prices : DataFrame
            Daily high prices for each ticker and date
        low_prices : DataFrame
            Daily low prices for each ticker and date
        close_prices : DataFrame
            Daily close prices for each ticker and date

        Returns
        -------
        open_prices_weekly : DataFrame
            Weekly open prices for each ticker and date
        high_prices_weekly : DataFrame
            Weekly high prices for each ticker and date
        low_prices_weekly : DataFrame
            Weekly low prices for each ticker and date
        close_prices_weekly : DataFrame
            Weekly close prices for each ticker and date
        """

        open_prices_weekly = open_prices.resample('W').first()
        high_prices_weekly = high_prices.resample('W').max()
        low_prices_weekly = low_prices.resample('W').min()
        close_prices_weekly = close_prices.resample('W').last()

        return open_prices_weekly, high_prices_weekly, low_prices_weekly, close_prices_week


quiz_tests.test_days_to_weeks(days_to_weeks)
```

```
Tests Passed
```

1.3 Quiz Solution

If you're having trouble, you can check out the quiz solution here.