Fitbit-sleep-analysis

March 23, 2017

1 Fitbit Sleep Analysis

Studying springtime sleep habits though analysis of my Fitbit sleep data In this notebook, I will study a student's sleep habits with data extracted from his/her Fitbit Charge HR tracker. Each day's sleep data is stored in the /logs/ directory.

1.1 Setup

Let's begin with importing what we need and configuring plots for later.

```
In [1]: %matplotlib inline

        import matplotlib
        from matplotlib import pyplot as plt
        import numpy as np
        import pandas as pd
        import json
        import datetime
        import scipy.stats

        matplotlib.style.use('ggplot')
        plt.rcParams['figure.figsize'] = [12.0, 8.0]
```

Before we dive in, we must first understand the shape of the data. Refer to the Fitbit documentation for detailed information on the format of the log files. As a quick overview, each day contains two top-level attributes, `sleep` and `summary`: - `sleep` is a list of one or more entries for different sleep events in a day (e.g., a nap and then one's main sleep). Each entry has several attributes, like `isMainSleep`, `minutesAsleep`, and `minuteData`. `minuteData` contains minute-by-minute sleep data, where each minute has an identifier, 1, 2, or 3, to denote asleep, restless, or awake. - `summary` contains summary statistics. It includes the number of sleep events (`totalSleepRecords`) and the total time in bed or asleep (`totalTimeInBed` and `totalMinutesAsleep`).

We can run a couple of simple commands to illustrate this.

```
In [27]: with open('logs/A07_20160401_sleep.json') as f:
             sample_data = json.loads(f.read())
         list(sample_data.keys())
```

```
Out[27]: [u'sleep', u'summary']
```

```
In [83]: #sample_data

In [29]: list(sample_data['summary'].keys())

Out[29]: [u'totalTimeInBed', u'totalMinutesAsleep', u'totalSleepRecords']

In [30]: list(sample_data['sleep'][0].keys())

Out[30]: [u'logId',
          u'dateOfSleep',
          u'minutesToFallAsleep',
          u'awakeningsCount',
          u'minutesAwake',
          u'timeInBed',
          u'minutesAsleep',
          u'awakeDuration',
          u'efficiency',
          u'isMainSleep',
          u'startTime',
          u'restlessCount',
          u'duration',
          u'restlessDuration',
          u'minuteData',
          u'awakeCount',
          u'minutesAfterWakeup']
```

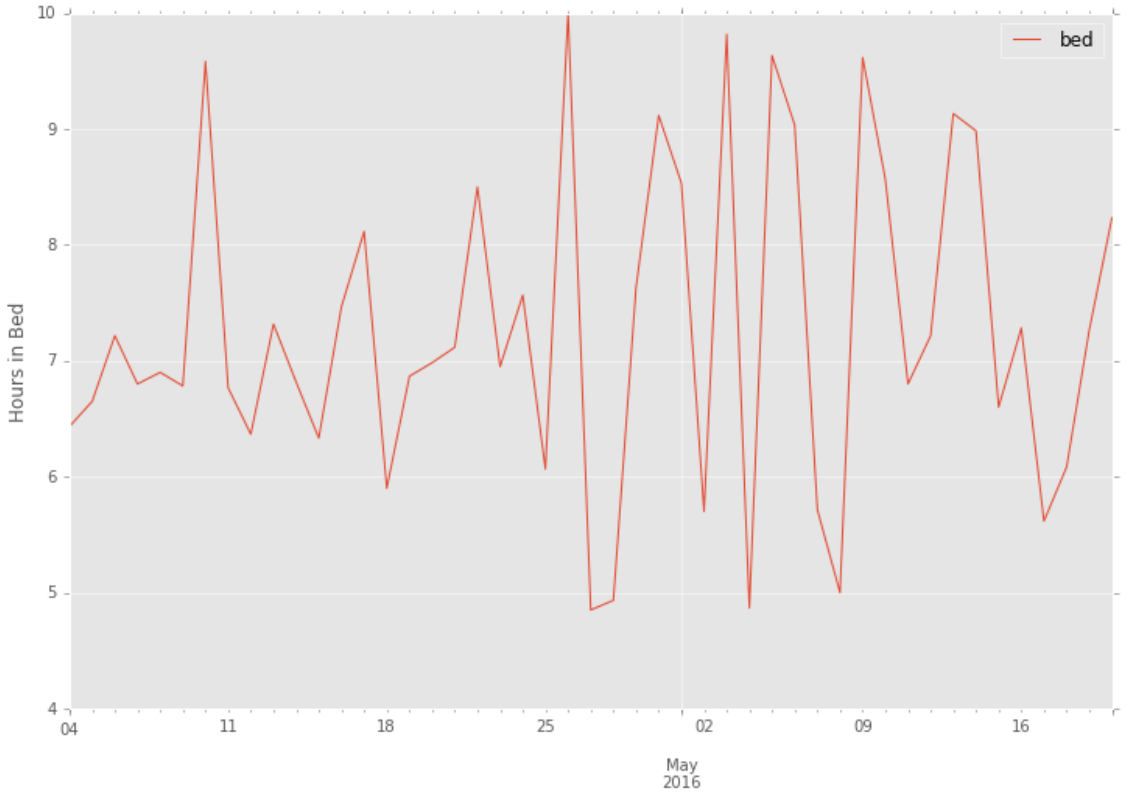
1.2 Time Spent in Bed

How long was he/she in bed each night?

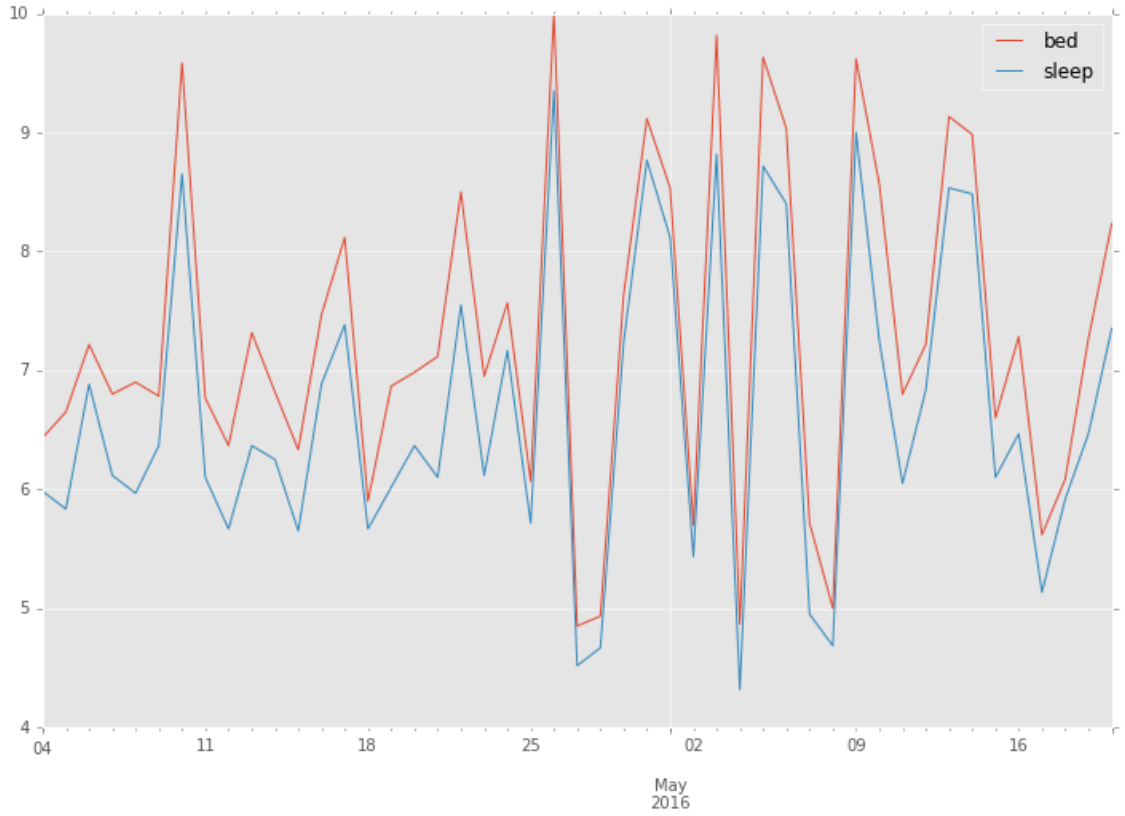
(I am choosing to study `totalTimeInBed`, rather than `totalMinutesAsleep`, for a couple of reasons. People generally fall asleep quickly and sleep well; People rarely get up in the middle of the night; and a quick look at the distributions shows them as being very similar.)

```
In [65]: dates = pd.date_range('20160404', '20160520')
         time_in_bed = []
         sleep_time = []

         for date in dates:
             fname = 'logs/A07_' + date.strftime('%Y%m%d') + '_sleep.json'
             try: f = open(fname)
             except IOError as e:
                 print(str(e))
             else:
                 #print "file read " + fname
                 date_data = json.loads(f.read())
                 #print date_data['summary']['totalTimeInBed']
                 time_in_bed.append(date_data['summary']['totalTimeInBed'] / 60.0)
                 sleep_time.append(date_data['summary']['totalMinutesAsleep'] / 60.0)
                 #print time_in_bed
```

```
df = pd.DataFrame(time_in_bed, index = dates)
df.columns = ['bed']

#df.plot()
#plt.ylabel('Hours in Bed');
```




```
In [75]: df2 = pd.DataFrame(sleep_time, index=dates)
         df2.columns = ['sleep']

         #df2.plot()
         #plt.ylabel('Hours')

         fig, axes = plt.subplots()
         df.plot(ax=axes)
         df2.plot(ax=axes)

Out[75]: <matplotlib.axes._subplots.AxesSubplot at 0x116f5cc90>
```

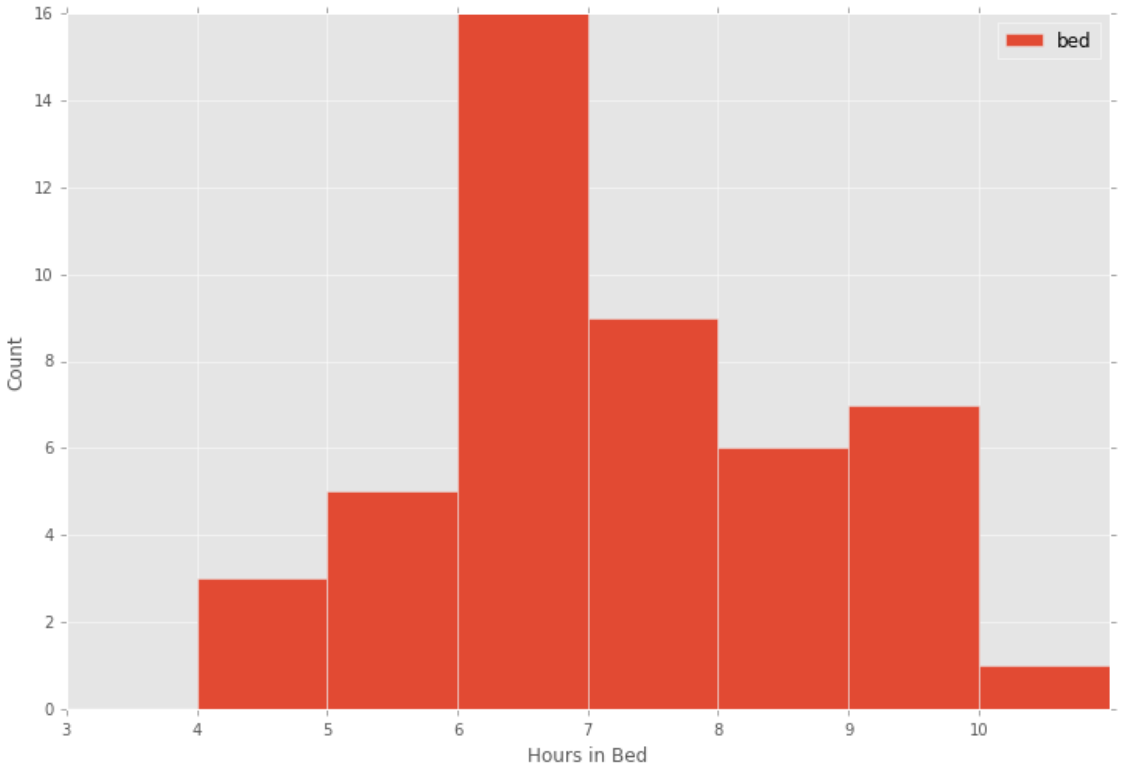


As interesting as this looks, there isn't a whole lot to take from it on the surface. The vague periodicity of the graph sugggests that we look at different days of the week, so we'll do that soon. Let's first look at the distribution of how long a student was in bed.

```
In [76]: df.describe().transpose()
```

```
Out[76]:       count      mean       std   min  25%       50%       75%   max
         bed    47.0  7.270922  1.394652  4.85  6.4  6.983333  8.366667  10.0
```

```
In [77]: df.plot.hist(bins = 8, range = (3, 11))

         plt.xlim(3, 11)
         plt.xticks(range(3, 11))
         plt.xlabel('Hours in Bed')
         plt.ylabel('Count');
```



As far as behavioral data goes, this is reasonably well-behaved. Notice that this distribution varies quite a bit, with a standard deviation of nearly 1.3 hours. (Anyone familiar with the sleep habits of a college student will not be surprised.)

1.3 Sleep Paterns by Day of Week

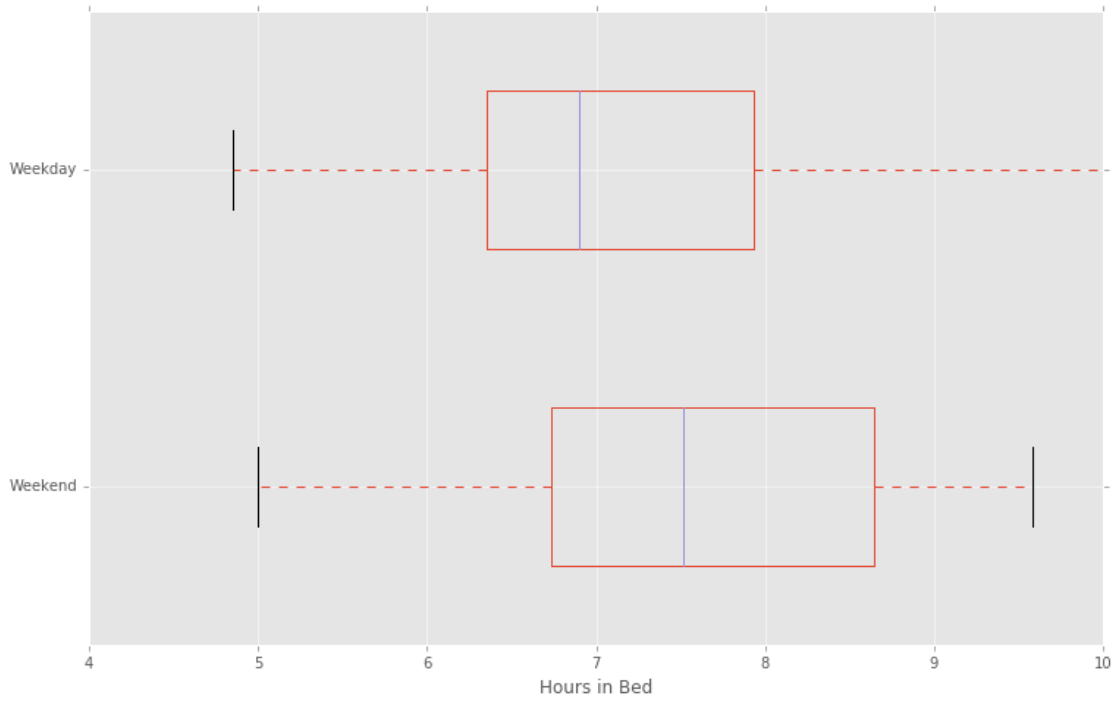
Let's now look at the different days of the week. Did a student sleep more on weekends? (I certainly hope so.) What nights were the worst? We can add another identifier denoting the day of the week, and then group the days based on whether they were weekdays or weekends. We'll then look at the distriutions and compare them with a t-test.

Note that a sleep record for a certain day (e.g., Friday) contains data from the previous night (in that case, Thursday night). Also, `pandas` numbers days as 0 = Monday, ... , 6 = Sunday. Altogether, this means weekdays are labeled 0 through 4, and weekends are labeled 5 and 6.

```
In [79]: df['day_of_week'] = df.index.weekday
         df['day_type'] = df['day_of_week'].apply(lambda x: 'Weekend' if x >= 5 else 'Weekday')
         df.head()
```

```
Out[79]:                 bed  day_of_week day_type
         2016-04-04  6.433333            0  Weekday
         2016-04-05  6.650000            1  Weekday
         2016-04-06  7.216667            2  Weekday
         2016-04-07  6.800000            3  Weekday
         2016-04-08  6.900000            4  Weekday
```

```
In [80]: df.boxplot(column = 'bed', by = 'day_type', positions = [2, 1],
                    vert = False, widths = 0.5)
         plt.xlabel('Hours in Bed')
         plt.suptitle('')
         plt.title('');
```




```
In [43]: # Group dataframe by weekday vs. weekend
         df_weekdays = df[df.day_of_week < 5]
         df_weekend = df[df.day_of_week >= 5]

         scipy.stats.ttest_ind(df_weekdays['bed'], df_weekend['bed'])
```

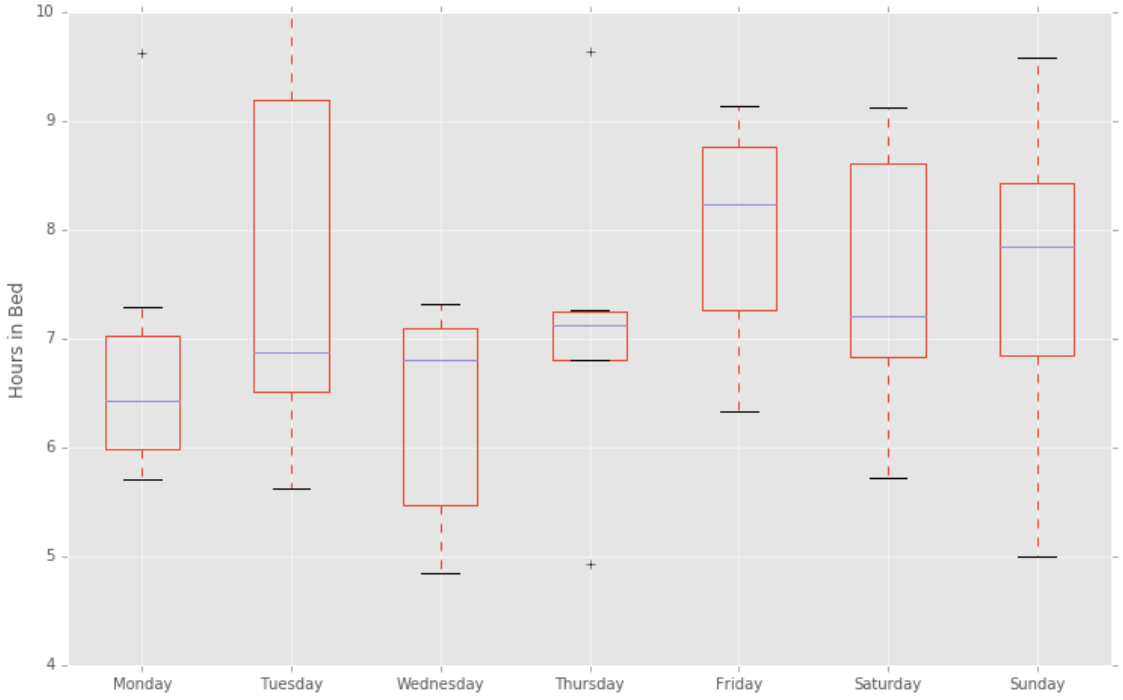
```
Out[43]: Ttest_indResult(statistic=-0.75575501148685065, pvalue=0.45373413586527755)
```

A slightly low p-value (0.45) would does not suggest a student sleep statistically significantly more on weekends than on weekdays. We can notice that the two boxplots illustrate the difference between weekdays and weekends.

What about individual days? We can consider each night of the week separately, and look at the distributions there.

```
In [81]: # Add a label for day name, to make the boxplot more readable
         days = {0: 'Monday', 1: 'Tuesday', 2: 'Wednesday', 3: 'Thursday', 4: 'Friday',
                 5: 'Saturday', 6: 'Sunday'}
         df['day_name'] = df['day_of_week'].apply(lambda x: days[x])
```

```
df.boxplot(column = 'bed', by = 'day_name', positions = [5, 1, 6, 7, 4, 2, 3])

# Configure title and axes
plt.suptitle('')
plt.title('')
plt.ylabel('Hours in Bed')
plt.xlabel('');
```



This is very illustrative. Mondas, Tuesdays, Wednesday, and Thursdays were the days on which a student got the least amount of sleep the previous night. Looking back on his/her Spring semeter, I can find explanations for all of these -- From Friday a student schedule is free.

1.4 Bedtimes

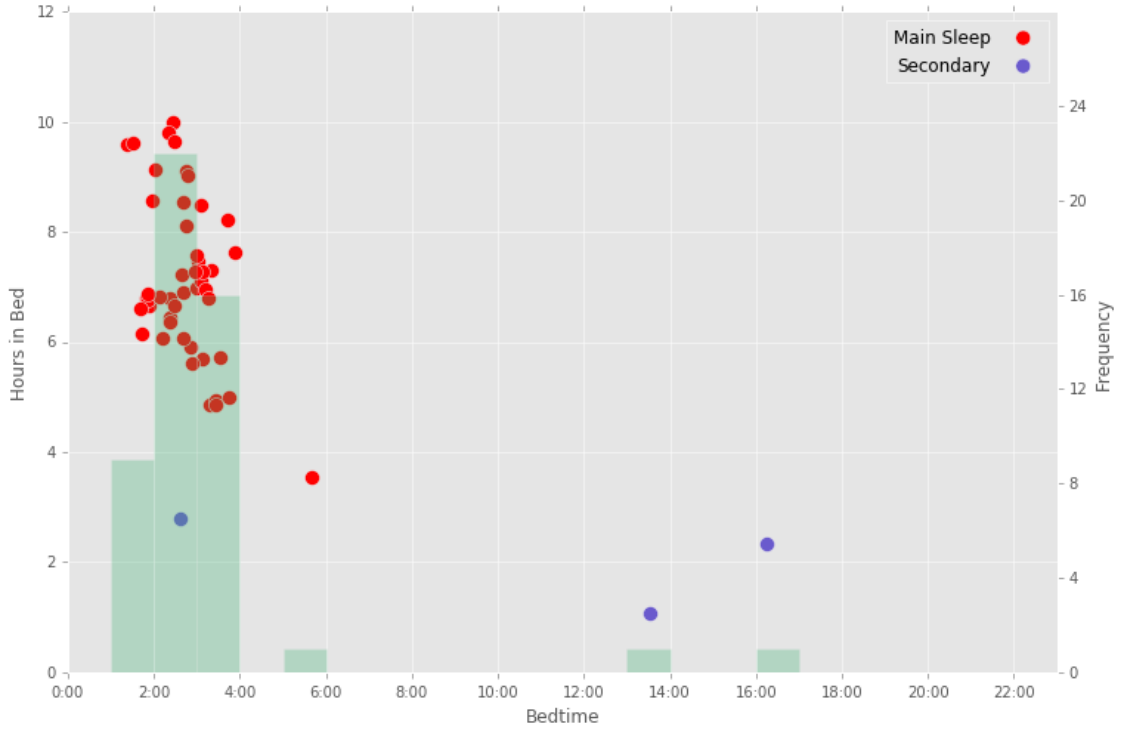
We've learned quite a bit about how much a student sleeps. What about *when* he/she sleep? Let's create a new dataset; this one will have columns for bedtime (formally defined as the start of any sleep event, including naps), for the time he/she was in bed (as before), and for whether or not it was a 'main sleep' (the longest sleep event of a 24-hour period, distinguishing regular sleep from naps).

Let's look at his/her sleep duration versus bedtime. While the next two code blocks are pretty large, their purposes are straightforward: we want to create the dataset described above, then plot it.

```
In [82]: bedtimes = []

         # Read data into list
```

```
        for date in dates:
            fname = 'logs/A07_' + date.strftime('%Y%m%d') + '_sleep.json'
            with open(fname) as f:
                date_data = json.loads(f.read())

                # Note that sleep_event['startTime'][11:16] gets the hh:mm characters
                # from the start of a sleep event; it is then converted to a datetime
                for sleep_event in date_data['sleep']:
                    bedtimes.append((pd.to_datetime(sleep_event['startTime'][11:16]),
                                     sleep_event['timeInBed'] / 60.0,
                                     sleep_event['isMainSleep']))

        # Convert to dataframe, and make 'bedtime' a float (e.g., 5:30 -> 5.5)
        df = pd.DataFrame(bedtimes, columns = ['bedtime', 'duration', 'main'])
        df['bedtime'] = df['bedtime'].dt.hour + df['bedtime'].dt.minute / 60.0
```

In [46]:
```
        # Make first plot: scatterplot of bedtime vs. duration, colored by main sleep
        ax = df[df.main == True].plot.scatter(x = 'bedtime', y = 'duration',
                                              color = 'Red', s = 100,
                                              label = 'Main Sleep')

        df[df.main == False].plot.scatter(x = 'bedtime', y = 'duration',
                                          color = 'SlateBlue', ax = ax, s = 100,
                                          label = 'Secondary')

        # List of times to use for labels
        times = [str(2 * h) + ':00' for h in range(12)]

        # Configure legend, x-axis, labels
        plt.legend(scatterpoints = 1, markerfirst = False)
        plt.xticks(range(0, 24, 2), times)
        plt.xlim(0, 24)
        plt.xlabel('Bedtime')
        plt.ylabel('Hours in Bed');

        # Overlay a histogram of bedtimes on the same plot, using a secondary y-axis
        ax2 = ax.twinx()
        df['bedtime'].map(lambda x: int(x)).plot.hist(bins = range(24),
                                                      color = 'MediumSeaGreen',
                                                      alpha = 0.3, grid = False)

        # Configure secondary y-axis
        plt.yticks(range(0, 28, 4))
        plt.ylim(0, 28);
```



You're a late owl !!! There is a clear negative correlation between bedtime and time spent in bed; it's no surprise that the later he/she goes to bed, the less sleep he/she gets.

But some other things can be read from this plot -- he/she most frequently went to bed between 2:00 and 3:00 am, for instance, followed by between 3:00 and 4:00, then 1:00 to 2:00. He/she rarely slept before 1:00, and even less so before midnight. He/she didn't take very many naps (note also that not every secondary sleep event is a nap), but when he/she did, they were around two or three hours in length.

1.5 Final Remarks

Data is great, but its purpose is ultimately to give insight into a larger picture. What can we learn from this dive into my sleep habits?

Clearly, **sleep durations vary a lot.** We noticed a standard deviation of nearly two hours on the distribution of sleep durations. A student sleep wildly different amounts each night; it would likely be healthier for him/her to work towards sleeping more consistently.

From the section "Sleep Patterns by Day of Week," we notice that **student classes/homework workload is reflected in sleep habits**. A student slept noticably less on nights where he/she had something due the following day. The solution, which is much easier said than done, is to start work earlier -- every student's goal. Similarly, **A studnet sleeps more on weekends.** The immediate explanation for this observation is that you're trying to make up for getting less sleep during the week. But this idea of "catching up on sleep" has been often debated by scientists, and it is unclear whether this is actually possible.

Future directions for this project include analyzing and comparing sleep data from fall, winter, and spring terms. Each quarter was very different, posing its own set of challenges; can you see

this reflected in your sleep patterns? Similarly, I could consider how these patterns change during summer. Finally, I might also consider making use of the nightly, minute-by-minute data.