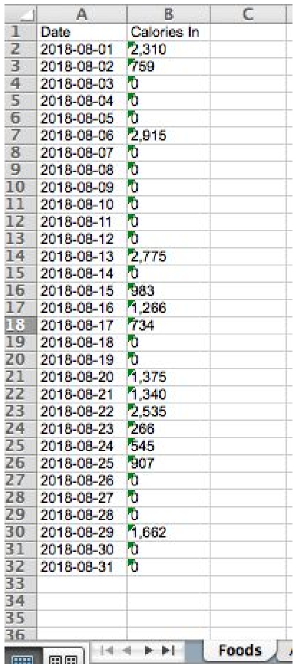
Capstone 1: Data Wrangling

My capstone project focuses on analysis of an individual's Fitbit data. Data was exported via Fitbit's website, and only up to 31 days of data can be collected at a time. Instinctively, I exported the data by month, all of which are Excel files. Activity, Sleep, and Food data were collected for all months, and are saved as separate sheets within an Excel file. In addition, daily food logs are saved as its own sheets, meaning each Excel file has at least 30 sheets.

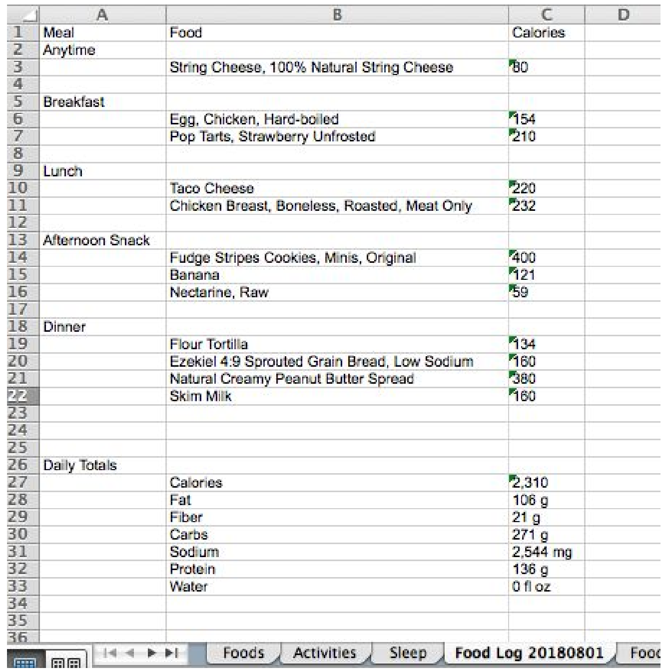
Fitbit's activity and sleep data are saved in a traditional database format, where each column is a variable and each row is an instance of the data. To consolidate the monthly data into one dataframe, the activity and sleep data were first parsed separately and stored into two lists of dataframes. Second, we concatenate the lists of dataframes to get two big dataframes. Fitbit displays their data in a user-friendly manner, so commas are used for numbers when needed. In Python, unfortunately, numeric data types are strictly just numbers, so the presence of a comma automatically renders the value to be treated as a string. To convert columns of strings into numeric values, commas were removed using regular expressions then converted using Python's handy "pd.to_numeric" function.

Upon analysis of the datasets, it was discovered that on days when Tracy does not use her Fitbit, data will still be displayed for that day with default values (e.g., minimum calories burned value based on Tracy's BMI). A check was performed to determine how many missing values (days Tracy doesn't use her Fitbit) were present in the dataset. It turns out only a small portion of the data contained missing values, so instead of filling in missing values, they were removed instead. The default value for Steps is 0, so we remove instances of data where the Steps column contains 0. Outliers were determined to be any values 2 standard deviations away from the mean. Outliers were easily handled by being replaced with the mean value.

The trickiest dataset to work with was the Food data.



	A	B	C
1	Date	Calories In	
2	2018-08-01	2,310	
3	2018-08-02	759	
4	2018-08-03	0	
5	2018-08-04	0	
6	2018-08-05	0	
7	2018-08-06	2,915	
8	2018-08-07	0	
9	2018-08-08	0	
10	2018-08-09	0	
11	2018-08-10	0	
12	2018-08-11	0	
13	2018-08-12	0	
14	2018-08-13	2,775	
15	2018-08-14	0	
16	2018-08-15	983	
17	2018-08-16	1,266	
18	2018-08-17	734	
19	2018-08-18	0	
20	2018-08-19	0	
21	2018-08-20	1,375	
22	2018-08-21	1,340	
23	2018-08-22	2,535	
24	2018-08-23	266	
25	2018-08-24	545	
26	2018-08-25	907	
27	2018-08-26	0	
28	2018-08-27	0	
29	2018-08-28	0	
30	2018-08-29	1,662	
31	2018-08-30	0	
32	2018-08-31	0	
33			
34			
35			
36			



	A	B	C	D
1	Meal	Food	Calories	
2	Anytime			
3		String Cheese, 100% Natural String Cheese	80	
4				
5	Breakfast			
6		Egg, Chicken, Hard-boiled	154	
7		Pop Tarts, Strawberry Unfrosted	210	
8				
9	Lunch			
10		Taco Cheese	220	
11		Chicken Breast, Boneless, Roasted, Meat Only	232	
12				
13	Afternoon Snack			
14		Fudge Stripes Cookies, Minis, Original	400	
15		Banana	121	
16		Nectarine, Raw	59	
17				
18	Dinner			
19		Flour Tortilla	134	
20		Ezekiel 4:9 Sprouted Grain Bread, Low Sodium	160	
21		Natural Creamy Peanut Butter Spread	380	
22		Skim Milk	160	
23				
24				
25				
26	Daily Totals			
27		Calories	2,310	
28		Fat	106 g	
29		Fiber	21 g	
30		Carbs	271 g	
31		Sodium	2,544 mg	
32		Protein	136 g	
33		Water	0 fl oz	
34				
35				
36				

This kind of data layout is not in a traditional dataset format. In order to transform the data to achieve the desired structure, extensive manipulation of data was involved.

In the “Foods” sheet, it should be intuitive that no food data is entered for a particular day if the value in “Calories In” is 0. Since there is a sheet for each day of the month (for daily food log), we are only interested in the sheets where food data is entered--when “Calories In” is not 0. We store the dates of interest in a list and use it to get the corresponding daily food log sheets. Each sheet is converted into a dataframe, where further manipulation is done. We associate each food item with the type of meal it was eaten as (e.g., Breakfast, Lunch, etc.) by taking the existing “Meal” column and forward filling each value. Any rows with missing values are removed.

Fitbit includes daily food composition totals within each sheet, which has sufficient information to be a separate dataframe itself. These data were extracted out of the food dataframe and put into its own dataframe. As a result of manipulating the food data, two dataframes were created.

	Meal	Food	Calories	Date	Weekday
0	Breakfast	American Cheese	61	2015-11-09	Monday
1	Breakfast	Bagel thins, Everything	110	2015-11-09	Monday
2	Breakfast	Egg, Chicken, Fried	184	2015-11-09	Monday
3	Breakfast	Ham Steak, Traditional	30	2015-11-09	Monday
4	Morning Snack	Dark Chocolate Dreams	170	2015-11-09	Monday
5	Morning Snack	Banana	90	2015-11-09	Monday
6	Morning Snack	Rice Cakes, Salt Free	70	2015-11-09	Monday
7	Breakfast	English Muffin, Original	129	2015-11-11	Wednesday
8	Breakfast	Egg, Chicken, Fried	184	2015-11-11	Wednesday
9	Breakfast	Bacon Pre-Cooked (S)	75	2015-11-11	Wednesday
10	Breakfast	American Cheese	79	2015-11-11	Wednesday

Food	Calories	Carbs	Fat	Fiber	Protein	Sodium	Water	Weekday
Date								
2015-11-09	715	72	34	8	35	943	0	Monday
2015-11-11	797	74	39	4	37	1064	0	Wednesday
2015-11-12	1049	108	45	11	53	1216	0	Thursday
2015-11-30	90	20	0	1	1	2	0	Monday
2015-12-02	240	29	6	3	17	152	0	Wednesday
2015-12-09	860	101	35	8	37	1105	0	Wednesday
2015-12-10	1054	135	40	26	58	1210	0	Thursday
2015-12-11	1157	155	35	23	68	679	0	Friday
2015-12-15	1162	142	44	13	57	1402	0	Tuesday