|  |  |  |
| --- | --- | --- |
| Action Description | Findings | Script Name / Location |
| I started by downloading the main bike share data from the Capital Bikeshare website and putting the files into folders on my desktop. I split the files into their respective years via folders. The entire folder is 1.69GB. |  | Path: Desktop/Springboard/Bike\_Sharing/Capital Bikeshare Data |
| In Anaconda, I used the data filepath to extract all files into a main dictionary where each key is part of the name of the original file and each respective value is a Pandas dataframe object. |  | Path: Desktop/Springboard/Bike\_Sharing/  Py file: Data\_Load.py |
| I wrote each file name (each key for the entire data dictionary), each column name for each value, and 5 header rows for each data frame to a text file for further analysis. Since not all feature names across each data frame are the same, the data may be inconsistent as well and I will need to address that. If the data is consistent, I can just change the feature names and then start to combine each data frame. I then found the number of unique column names and associated counts across all 25 data frames. | See below under preliminary full bike share data analysis and associated original text file.  Conclusion:  I found inconsistencies across the 25 dataframes including: Capitalization of column names, dropping of zipcodes between start and end stations (will address later), different arrangements of the same column names, and extra features which I consolidated into new features that matched the majority of features. I decided on 7 unique feature names and successfully standardized all feature names across the 25 dataframes as shown in the associated edit text file.  Issues to fix:   * Decide how to treat the start and end station variables: Only zip, only address, or keep both? * Make sure the Duration variable is consistent. Newer files show the measure in milliseconds and older files in hours, minutes, and seconds. | Path: Desktop/Springboard/Bike\_Sharing/  Text file:   * full\_data\_dict info - ORIGINAL.txt * full\_data\_dict info - EDIT.txt   Py file: Data\_Load.py |
| I created a new script that holds all relevant defined functions for ease of reading and calling same functions later. |  | Path: Desktop/Springboard/Bike\_Sharing/  Py file: Bike\_Functions.py |
| I used a regex function and mapped it to every Start Station and End Station feature in each data set, and added in new features for Start Zip and End Zip. I then wrote the results to a new file. Each data frame now has 9 features. | The issue above related to how to treat zipcodes has become a rather big issue to solve. The quarters between 2012 Q1 and 2015 Q2 all exclude respective zipcodes. Now, I could try to map all addresses and back out the implied zipcodes, but since there are already so many inconsistencies I feel that there is a low chance that this would actually work well. Further, I feel that zipcode is a much less confusing feature from a machine learning standpoint (although implied order may become an issue – TBD – maybe make it a nominal/string variable?) as compared to address, and I would like to move ahead with zipcode only. Finally, since there is a full year of consistent available data (2015 Q3 – 2016 Q3) and it is the most recent that I can continue using only this subset of the data. Using this subset of the data has the advantages of: Reduced computation time (which has already been a slowing factor to me), the Duration feature is consistently recorded in milliseconds, zipcodes were already included in the original data, and any changes to major routes or even station addition/removal is less likely to be a factor since the time horizon is shorter.  Conclusion:   * Move on and use only the 2015 Q3 – 2016 Q3 subset for the remainder of the project * Reinstate original zipcode features | Path: Desktop/Springboard/Bike\_Sharing/  Text file: Create Zipcodes.txt  Py file:   * Data\_Load.py * Select\_Subset.py |
| Split the data and now using only 2015 Q3 – 2016 Q3 subset. I reloaded only the relevant data from the originally downloaded csv files, kept original zipcode data, and renamed all feature names so that everything matched up. I then wrote to a new file so I could confirm that it worked well, and it did. I then combined all dataframes into one main dataframe, which I then wrote to a csv file. | New features: Duration, Start Date, End Date, Start Zip, End Zip, Bike Number, Member Type.  I confirmed that there are no missing values in any dataframe.  New subset data file contains about 4.3 million instances across 7 features. File is about 330MB. | Path: Desktop/Springboard/Bike\_Sharing/  Text file:   * subset\_data\_dict info.txt * full\_subset\_data.csv   Py file: Select\_Subset.py |
| While searching for more information on the original dataset, I realized that what I thought was the station zipcode is actually the station number. Considering the station number plus the address may become useful. I want to add in the station address once more and rename the zipcode features to reflect their true nature. I still want to continue with only the most recent data due to the advantages of doing so. I will make changes to the Select\_Subset.py script. | New features: Duration, Start Date, End Date, Start Station Number, Start Station, End Station Number, End Station, Bike Number, and Member Type  I confirmed that there are no missing values in any dataframe.  New subset data file contains about 4.3 million instances across 9 features. File is about 535MB. | Path: Desktop/Springboard/Bike\_Sharing/  Text file:   * subset\_data\_dict info.txt * full\_subset\_data.csv   Py file: Select\_Subset.py |
| Using an outside script (<https://kwkelly.com/blog/analyzing-capital-bikeshare-data-with-python-and-pandas/>) I was able to pull data regarding each station’s latitude and longitude by station number. I then merged the previous subset dataframe with the new data using a left join. I found that, when using Start Station as a key, there are 87 instances of a missing lat/long (all related to station 31709), and, when using End Station as a key, there are 76 instances of a missing lat/long (all related to station 31709). I used each End and Start station as keys to perform two left joins, but did not dropp the instances with missing information even though station 31709 is not a station as of today (since it will still affect bike counts at other stations). The dataframe now has 11 features. I also did a little sanity check to see if the start and end locations matched up correctly, and they do match. | New subset data file contains about 4.3 million instances across 11 features: Duration, Start Date, End Date, Start Station Number, Start Station, End Station Number, End Station, Bike Number, Member Type, End Location, and Start Location. File is about 900MB. | Path: Desktop/Springboard/Bike\_Sharing/  Text file:   * full\_subset\_data - V2.csv   Py file: Station\_Data.py |
| Also using the same outside script ((<https://kwkelly.com/blog/analyzing-capital-bikeshare-data-with-python-and-pandas/>) I created a Distance feature using both Start and End Location features. Distance serves as a lower bound since the function used to create it calculates the "distance between two points on the surface of a spheroid" and acts much like Euclidean distance rather than what would be a more accurate Manhattan distance. | New subset data file contains about 4.3 million instances across 12 features: Duration, Start Date, End Date, Start Station Number, Start Station, End Station Number, End Station, Bike Number, Member Type, End Location, Start Location, and Distance. File is about 980MB. | Path: Desktop/Springboard/Bike\_Sharing/  Text file:   * full\_subset\_data – V3.csv   Py file: Distance\_Feature.py |
| Using data provided at <http://cabitracker.com/station_outage.php?id=149> I am trying to map empty and full stations so that I can begin to create counts for how many bikes are at each station at any moment. The idea is that since I know when a station is empty or full, and all comings and goings between stations, I should be able to figure out station capacity and real numbers for station inventory numbers. One problem is that dates/times don’t match up exactly, so I can’t just join tables based on date/time and fill in values that way – Is there a way to do a good fuzzy match/join? <http://stackoverflow.com/questions/13636848/is-it-possible-to-do-fuzzy-match-merge-with-python-pandas> but having some trouble downloading difflib  **ALTERNATIVE**: See if I can get real station capacities, assume all stations start at half capacity or something similar, and then create station inventory counts and build a model on those assumptions.  I’m getting ideas from here: <https://github.com/joepaolicelli/cabi-prediction-api> |  | Path: Desktop/Springboard/Bike\_Sharing/  Text file:   * full\_subset\_data – V4.csv   Py file: Full\_Empty\_2.py |
| Exploring the subset data in full\_subset\_data – V3.csv, there appear to be stations that are named differently, but refer to the same station. I removed Start Station and End Station name features, and wrote to a new CSV file. | There are 415 unique start and end station names, but there are only 407 unique start and end station numbers. Further, there are 407 unique starting and ending locations measured by latitude and longitude.  Below are station numbers, respective station names, and number of times someone used the station as a starting location:   * 31000:   + 20th & Bell St; 2669   + Eads St & 15th St S; 2669 * 31030:   + N Adams St & Lee Hwy; 6269   + Lee Hwy & N Adams St; 6269 * 31039:   + N Quincy St & Wilson Blvd; 6318   + Wilson Blvd & N Quincy St; 6318 * 31048:   + King St Metro; 6495   + King St Metro South; 6495 * 31092:   + N Nelson St & Lee Hwy; 1516   + Lee Hwy & N Nelson St; 1516   + Lee Hwy & N Monroe St; 1516 * 31232:   + 8th & F St NW; 30106   + 7th & F St NW / National Portrait Gallery; 30106 * 31314:   + 34th & Water St NW; 15229   + 33rd & Water St NW; 15229   Same for ending location:   * 31232:   + 8th & F St NW; 33206   + 7th & F St NW / National Portrait Gallery; 33206 * 31314:   + 34th & Water St NW; 21433   + 33rd & Water St NW; 21433 * 31039:   + N Quincy St & Wilson Blvd; 5678   + Wilson Blvd & N Quincy St; 5678 * 31048:   + King St Metro; 4932   + King St Metro South; 4932 * 31030:   + N Adams St & Lee Hwy; 3956   + Lee Hwy & N Adams St; 3956 * 31000:   + 20th & Bell St; 2189   + Eads St & 15th St S; 2189 * 31092:   + N Nelson St & Lee Hwy; 1338   + Lee Hwy & N Nelson St; 1338   + Lee Hwy & N Monroe St; 1338   Since station numbers are constant and station names are not, and station names are basically unique identifiers, I can discard the station name. I know that this will not affect results since all station counts for each different name have the same counts and the unique numbers of start/end station numbers match those of start/end lat/long numbers. This means that the stations seem to all be recognized under the station number, which correspond to static latitudes and longitudes.  I then deleted the station names feature and wrote the data to a new file. New data file contains about 4.3 million instances across 10 features: Duration, Start Date, End Date, Start Station Number, End Station Number, Bike Number, Member Type, End Location, Start Location, and Distance. File is about 765MB. | Path: Desktop/Springboard/Bike\_Sharing/  Text file:   * full\_subset\_data – V3.csv   Py file: Clean\_Stations\_Numbers.py |
| I made two visualizations using Tableau Public:   1. Capital Bikeshare Starting Station Popularity Bubble Map [2015 Q3 - 2016 Q3] 2. Capital Bikeshare Total Trip Popularity Spider Map [2015 Q3 - 2016 Q3] |  | <https://public.tableau.com/profile/dpalbrecht>  Py file: Lat\_Long\_Split.py |
| After not being able to successfully join together the historical Capital Bikeshare dataset and the CaBi Tracker dataset, I’m going back to a GitHub account (<https://github.com/joepaolicelli/cabi-prediction-api>) where I found a good way to grab the live XML feeds from Capital Bikeshare. I used this code and changed it so I could use sqlite since I’m more familiar with it. I’ve scheduled my script to run every 10 minutes, and it grabs the last activity for each station (446 rows every run). The database has 7 features: station ID, station name, latitude, longitude, timestamp, #bikes, and #spaces. | I will need to check it after a day and determine:   * It is generating enough data? * How many days should I collect for? * Should I schedule it to run on an even smaller time interval? 5 minutes, 1 minute?   I’ll be able to find all capacities and make a new feature using the #bikes and #spaces features. | Path: Desktop/Springboard/Bike\_Sharing/  Py file: Get\_XML.py  Database: bike\_count.sqlite |
| Need to find more data: public transport data patterns, weather data  Need to think of how this can be framed to answer the how many bikes are at a station at a given time? |  |  |
| I could group the latitudes and longitudes to be close or far from each other or the most popular stations? Make a new feature for “far” or “close” |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Questions:

* (PROBLEM: How do I get starting numbers and come up with a running total for the current hour? The data does not seem to capture current bike rack numbers)
* Is it possible some bike share locations no longer exist? I would have to delete these entries from the training data since they are no longer relevant to rebalancing [SOLVED]

**Preliminary full bike share data analysis:**

*(Copied from text file)* Unique column names and counts grouped by color:

* End Station: 12
* Bike number: 6
* Start station: 13
* Bike #: 2
* End station: 13
* Subscription Type: 8
* Total duration (ms): 1
* Bike Key: 1
* Subscriber Type: 2
* Bike#: 17
* Member type: 2
* Start time: 1
* Account type: 1
* End station number: 6
* Duration: 17
* Member Type: 9
* Start date: 24
* Start Station: 12
* Duration (ms): 7
* Type: 1
* End date: 25
* Subscription type: 1
* Start station number: 6

Given the number of different colors, the combined dataframe should end up with 7 total features. The names will be: Duration, Start Date, End Date, Start Station, End Station, Bike Number, and Member Type.

*(Copied from text file)* Column names per separate dataframe in ORIGINAL FILE:

2010-Q4: 7

Duration, Start date, End date, Start station, End station, Bike#, Member Type,

2011-Q1: 7

Duration, Start date, End date, Start station, End station, Bike#, Member Type,

2011-Q2: 7

Duration, Start date, End date, Start station, End station, Bike#, Member Type,

2011-Q3: 7

Up to here, all features and values are alike per headers in the text file.

Duration, Start date, End date, Start station, End station, Bike#, Member Type,

2011-Q4: 7

Duration, Start date, End date, Start station, End station, Bike#, Member Type,

2012-Q1: 7

* Start and End Station change capitalization. Keep case.
* Start and End Station drop the zipcode component. I could either strip all zipcode components entirely and use just them or just addresses, or try to map zipcodes and addresses and use them jointly.
* Member Type is changed to Type, but data looks to be the same. Change feature name to Member Type.

Duration, Start date, End date, Start Station, End Station, Bike#, Type,

2012-Q2: 7

Duration, Start date, End date, Start Station, End Station, Bike#, Bike Key,

2012-Q3: 7

Duration, Start date, End date, Start Station, End Station, Bike#, Subscriber Type,

2012-Q4: 7

Duration, Start date, End date, Start Station, End Station, Bike#, Subscription Type,

2013-Q1: 7

Duration, Start date, End date, Start Station, End Station, Bike#, Subscription Type,

2013-Q2: 7

Duration, Start time, End date, Start Station, End Station, Bike#, Subscription Type,

2013-Q3: 7

Duration, Start date, End date, Start Station, End Station, Bike#, Subscription Type,

2013-Q4: 7

Duration, Start date, Start Station, End date, End Station, Bike#, Subscription Type,

2014-Q1: 7

Duration, Start date, Start Station, End date, End Station, Bike#, Member Type,

2014-Q2: 7

Duration, Start date, Start Station, End date, End Station, Bike#, Subscriber Type,

DANGER: This yellow grouping of files changes the order of the feature names. Will have to pass in feature names in a different order to keep integrity.

2014-Q3: 7

Duration, Start date, Start Station, End date, End Station, Bike#, Subscription Type,

2014-Q4: 7

Duration, Start date, Start Station, End date, End Station, Bike#, Subscription Type,

2015-Q1: 7

Total duration (ms), Start date, Start station, End date, End station, Bike number, Subscription Type,

2015-Q2: 7

Duration (ms), Start date, Start station, End date, End station, Bike number, Subscription type,

2015-Q3: 9

Duration (ms), Start date, End date, Start station number, Start station, End station number, End station, Bike #, Member type,

2015-Q4: 9

Duration (ms), Start date, End date, Start station number, Start station, End station number, End station, Bike #, Member type,

2016-Q1: 9

Duration (ms), Start date, End date, Start station number, Start station, End station number, End station, Bike number, Member Type,

2016-Q2: 9

Duration (ms), Start date, End date, Start station number, Start station, End station number, End station, Bike number, Account type,

2016-Q3-1: 9

Duration (ms), Start date, End date, Start station number, Start station, End station number, End station, Bike number, Member Type,

2016-Q3-2: 9

Duration (ms), Start date, End date, Start station number, Start station, End station number, End station, Bike number, Member Type,

DANGER: This green grouping of files now adds back in zipcodes as a new field name. I will create a new feature that uses the addition of address and zipcode so that it matches the other features for now.