Data Wrangling Steps

1. Imported pandas and the two csv files that are now named complaints and subways.
    a. Complaints is a large file, 1.36 GB. To speed up the import process only columns of interest will be brought in. Using the column descriptions from a separate file, the following column indexes will be imported: 0, 1, 2, 7, 9, 10, 13, 19, 20, 21, 22. The corresponding column titles are as follows: CMPLNT_NUM, CMPLNT_FR_DT, CMPLNT_FR_TM, PD_DESC, CRM_ATPT_CPTD_CD, BORO_NM, X_COORD_CD, Y_COORD_CD, Latitude, Longitude.
2. Once imported, looking at the complaints.info() and subways.info() will a nice overview of both dataframes. I started with looking at and cleaning complaints.

    ### Complaints

    a. The info for complaints does not show the number of non-null entries per column. It also has imported the date column, time column, x-coordinate, and y-coordinate columns as objects.
    b. In my attempts to convert the x and y coordinate columns to float after import, none of them worked. astype(float) and apply.(pd.to_numeric). To fix this issue, I rewrote the read_csv to include the argument thousands=','. Now the columns are imported as float.
    c. To convert the date and time columns to datetime, I first printed complaints.head(1) to see how the date and times are formatted. Date is MM//DD/YYYY and time is in 24-hour format. I used the to_datetime and to_timedelta functions put these columns in the proper format with the errors='coerce' argument. Checked the info and head on complaints at the end just to make sure everything is in the correct format.
    d. Since the info command is not returning count of non-null entries, I wrote a for loop to count the NaN values in each column.
    e. Since the foundation of this analysis is on proximity to subway station entrances, having records with no lat/long data does not make sense. I removed all records with NaN x and y coordinates and/or lat and long coordinates.
    f. Ran the for loop again to make sure those rows were dropped. Lastly, I checked the info to make sure everything is how I want it to be.
    g. Having over 5 million crimes is a lot for one machine and one processor to deal with. To make the dataset more manageable I selected only crimes that occurred in 2016.
    h. This reduced the number of crimes to about 459,000 entries. I took a look at occurrence of each type of crime.

    i. It makes sense to focus on the crimes with highest counts to be more accurate. I then isolated the top 9 offenses (10 was a miscellaneous entry which does not give any insight to the type of actual crime).
    ii. Now the total of crimes for analysis are approximately 365,000.

i. This is still a high number which would have taken my machine too long to calculate. I further reduced the crimes by only selecting the ones that occurred in the borough of Manhattan. Now there are only 90,483 crimes.

Subways

j. I printed the info and head for subways to see its key characteristics. The name is missing for some stops but everything else has a value. I only need to work with the 'the_geom' so I extracted it into a new dataframe, subways_loc.
k. The column geom is a text column that has 'POINT (longitude latitude)'. I split geom into three new columns, point, Longitude, and Latitude.
l. I removed the parentheses from Latitude and Longitude with a lamba function to select on the part of the str that I wanted.
m. Then I removed point from subways_loc dataframe and converted Latitude and Longitude to float with astype(float).
n. Calculating distances is much easier with State Plane Coordinates (in ft) than with Universal Transverse Mercator (UTM). Luckily, the complaints dataframe has x and y coordinates in New York State Plane Coordinate System Long Island Zone NAD 83.
    i. To do this in python and in jupyter notebook I had to take a few extra steps. I had to create a new virtual environment in conda, activate py35, install Microsoft Visual Studio 14 C++, and install pyproj.
    ii. To figure out the exact projection of the subway coordinates, I had to do some investigation. I sent an email request to NYC open data to tell me the coordinate system and they replied WGS84 Zone 18N, EPSG:32618 http://spatialreference.org/ref/epsg/wgs-84-utm-zone-18n/
    iii. Using the code written by Elise Huard as a guide, I transformed the coordinates from WGS84 to New York State Plane Coordinate System Long Island Zone NAD 83 using a for loop. https://gist.github.com/elisehuard/10259449

3. Now I calculated distances between the crime locations and subway locations.
    a. Even though the crime counts were paired down quite a bit from the original number, memory errors happen when trying to use all 90,483 instances at once.

    i. I split the crimes dataframe into 5 smaller dataframes so there would be no memory error when distances were totaled.

b. I created a function that calculated the absolute distance between all pairs of crime and subway coordinates.
    i. The results were put into a new, empty dataframe.

c. Then I found the minimum distance between each crime and a subway location and put into a new column in the dataframe.

d. I then transposed the minimum distance values into the original spreadsheet. The values were matched on the crime index.