Process Log
Ilica, Dingzhang, Michelle

Github Repo:
https://github.com/ilica/fairfares

1. We have two goals for the analysis
    a. Goal 1: Make a chart that has the median household income for the neighborhood around a given subway station on the y-axis and the drop in ridership at that station as a percent on the x-axis before and after the corona virus lockdowns to try to understand how income of a neighborhood affects whether or not people in that neighborhood are still commuting
    b. Goal 2: Repeat the analysis of what percent of one's income is spent on a 30 day monthly pass if they are at the federal poverty line that is in this report: https://issuu.com/cssnyorg/docs/the_transit_affordability_crisis_fi
        i. Redo it for 2016 to get back and check their numbers
        ii. Do it again for updated numbers for 2019 according to the updated cost of a metrocard and the updated federal poverty line
2. Goal 1: Make a scatter plot, one dot will be a subway station, the y axis will be the median household income surrounding that subway station and the x axis will indicate the drop in ridership between two dates on either side of the lockdown order for corona virus shelter in place
    a. Reason: Fair fares is even more important now if poorer people (less of a drop in ridership) are still moving around, but their ability to access it is being hampered by closures like in the offices
    b. Open a jupyter notebook to do the analysis in, named "produce_scatter_plot"
        i. Import libraries I expect to use like pandas, requests, json, and Socrata, which is the library for helping pull government data
        ii. Go to nyc open data and look for recent turnstile data, searched: 2020 Turnstile and found https://data.ny.gov/Transportation/Turnstile-Usage-Data-2020/py8k-a8wg
        iii. Click Export, csv and download the turnstile data. Once downloaded, move to folder where the ipython notebook is, and rename is Turnstile_2020.csv
        iv. In notebook, write to read in 2020 Turnstile data
            1. turnstile = pd.read_csv("Turnstile_2020.csv")
            2. The data seems to have duplicate rows, so drop duplicates like this:
                a. subway_to_latlong = subway_to_latlong.drop_duplicates(subset="Station")
        v. I'm going to need the information for the longitude and latitude for the subway stations in order to be able to understand what census tract they

are in to try to look for the median household income around the subway stations.

1. Talk to some map-making friends about tips searching for such a file, because the one provided by MTA is hard to join on and contains a lot of problems - they suggest looking for "geocoded" files
2. Google turnstile geocode subway stations
3. Click on 1st entry: https://chriswhong.com/open-data/visualizing-the-mtas-turnstile-data/
4. This person to make his visualization must have needed a geocoded list of the subway stations, so i typed in Chris Whong github
5. Found his github, https://github.com/chriswhong, search repositories for turnstile, found the project page for the above data viz on his website, find https://github.com/chriswhong/nycturnstiles/blob/master/geocoded.csv
6. Download the file:
    a. Mv the file into the fairfares directory where I am working
7. Read file into pandas:
    a. pd.read_csv("geocoded.csv") in my notebook

vi. I need the census data for median household income
1. Google american community survey because census data is now 10 years old and ACS is more recent
    a. Click data in the google search results under the american community survey result which was the first result
    b. Click Data > data.census.gov > 2014-2018 ACS 5-year Data Profiles > Customize table
        i. Geographies: Census tract -> within state: New York
        ii. Year: 2018
        iii. https://data.census.gov/cedsci/table?d=ACS%205-Year%20Estimates%20Data%20Profiles&table=DP03&tid=ACSDP5Y2018.DP03&g=0400000US36.140000&hidePreview=true&y=2018
        iv. Click Download

            v. Unzip and put files in the same folder as fairfares, rename the data file to census_data and the metadata file to headers

    c. In my notebook, write: census_to_median_income = pd.read_csv('census_data/census_data.csv') to read in the data

2. Write a method to convert the lat longs to see what census tract they fall under:
    a. Google lat long to census data: click on first stack overflow link, which points to the FCC API: https://geo.fcc.gov/api/census/
    b. Great, we can now use python's requests library to convert any lat long into census tract info:
        i. Format the string with a given lat long, format_str = "https://geo.fcc.gov/api/census/area?lat={}&lon={}&format=json".format(lat, long)
        ii. Have the requests library make the request: response = requests.get(format_str)
        iii. Parse the json and return the block_fips which codes for the census tract which i determined by opening the census data from the ACS in excel and looking at what id it uses for each census tract and compared that against a few test hits on the lat and long for a few subway station, and determined that the geo id in the census data is the first 11 digits of the block_fips in the FCC data API
3. Write a method to convert the census tract to the median income:
    a. From the headers file for the census data find the column needed for the median household income and code that into a variable
    b. name_of_median_income_col = 'DP03_0062E'
    c. Convert the census tract info from the lat long to census tract to the GEO ID needed by the ACS dataset, and filter the dataframe for the row that matches the GEO_ID, and then pull out the household median income using the above column name, if its sensible, return it and if it doesn't make any sense, return a None
    d. Code anything that is >250,000 as the median household income as 250,001 to denote the fact that it was a string

that means that its larger than 250,000 but we needed it as an int

vii. For the turnstile data:

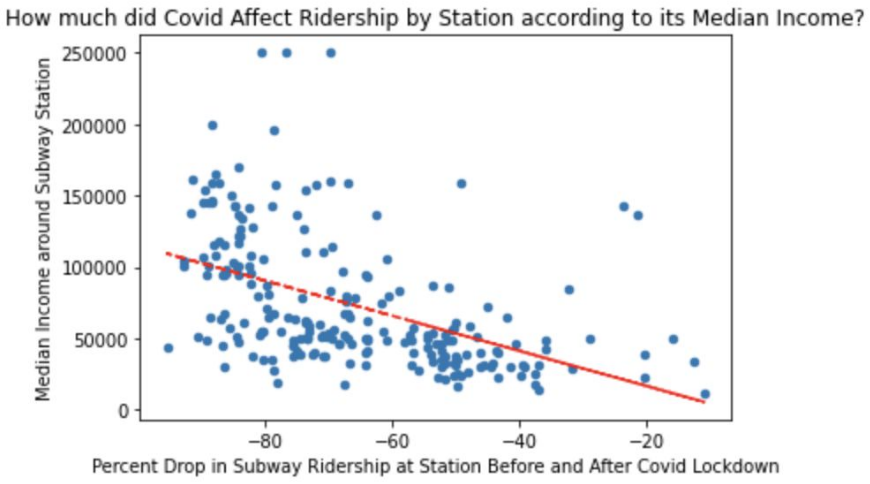
1. Drop the unnecessary columns
2. Choose a date much before corona on a weekday and compare against a week day after corona and the lockdown were introduced
    a. Crop the pandas dataframe down to only those two dates by
        i. cropped = cleaned[cleaned.Date.isin(["03/26/2020", "02/15/2020"])]
3. Group by the station, unit, SCP and date and then compute the max and min for that date by turnstile by station so that we know what the starting and ending "entry" count was for the day
    a. Using the grouping features of pandas do grouped = cropped.groupby(["Station", "Unit", "SCP", "Date"]).agg({"Entries":[np.max, np.min]})
4. Add a row to subtract the difference between the max and and min so that we know the number of entries at each station, turnstile on each date
    a. grouped["num_entries"]=grouped.apply(lambda row: row.Entries.amax-row.Entries.amin, axis=1)
5. Reset the index to bring it back to a pandas dataframe object and not a grouped object by using the reset_index() method in pandas
6. Group again by station and date and sum all the number of entries during the grouping so we wind up with two rows for each station, one for each date, in preparation to calculate the percent difference by using the grouping features of pandas again and doing a .groupby(['Station','Date'])["num_entries"].sum().reset_index()
7. Reset the index straight away to bring back a dataframe
8. Create a method to calculate the percent difference and use that function to go through all the rows and calculate the percent difference between entries on feb 15th and march 26th.
9. Join with the lat long geocoded subway station file and by using the pandas .join() method, indexing on subway stations
10. Drop the columns of the data that is no longer relevant like the lines, code, division
11. Apply a function to every row of the data frame that takes the latitude and longitude of each of the subway stations and creates a new column that is the census tract that that subway station belongs to using the function described earlier:

a. lat_long["census_tract"] = lat_long.apply(lambda row: lat_long_to_census_tract(row["Lat"], row["Long"]), axis=1)

12. Using the census tract to median income function described above, apply a function to every row of the data frame that take the census tract id number and adds a new column with the median income in the census tract around each subway station
13. Multiply each of the percentage difference column by 100 to make the column a real percent to read the graph easier

viii. With the dataframe that has the percentage drop in ridership at each station and the median income around each station, generate a scatter plot with an overlay trend line to understand the correlation of median income around a subway station to the ridership drop

1. plot = lat_long.plot.scatter(y="income", x="pct"):



c. Goal 2: Repeat the analysis of what percent of one's income is spent on a 30 day monthly pass if they are at the federal poverty line that is in this report: https://issuu.com/cssnyorg/docs/the_transit_affordability_crisis_fi

i. Open new jupyter notebook named: produce_bar_chart_pct_income

ii. Google the cost of a 30 day unlimited pass in 2016:

1. Click on the news tab, find nytimes article with 2015 price hike of MTA 30 day pass:

a. https://www.nytimes.com/2015/01/23/nyregion/mta-raises-fares-subways-and-buses.html

iii. Current cost of a 30 day unlimited pass is 127, googled from 30 day unlimited pass, click second link https://new.mta.info/fares-and-tolls/subway-bus-and-staten-island-railway/fares-at-a-glance/unlimited-ride-metrocard

iv. The Cost of 30 day unlimited pass in 2016 and 2019, hardcode the numbers
    1. cost_30_day_unlimited_2019 = 127
    2. cost_30_day_unlimited_2016 = 116.50

v. Google federal poverty 2016, look for filetype pdf, click on second link, https://www.govinfo.gov/content/pkg/FR-2016-01-25/pdf/2016-01450.pdf , get poverty line for person in family 1-household, 11,880

vi. Google federal poverty line in 2019, click on first link, For 2019 https://aspe.hhs.gov/2019-poverty-guidelines 12,490

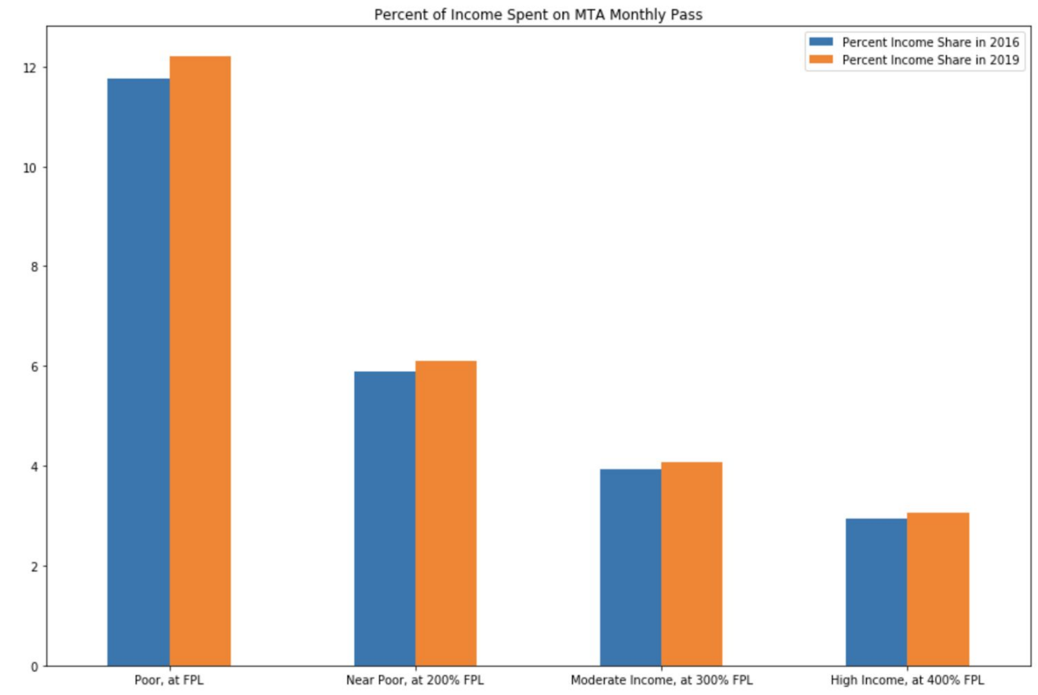
vii. The Federal Poverty Lines in 2016 and 2019, hardcode the numbers
    1. fpl_2016 = 11880
    2. fpl_2019 = 12490

viii. The percent of income spend on transportation for a 30 day unlimited pass if you are at the federal poverty line in 2016 is pct_of_income_transportation_2016 = cost_30_day_unlimited_2016*12*100/fpl_2016 and same for 2019 but with the 2019 numbers

ix. Write the column names out in preparation to build a dataframe at the various income level multipliers of the federal poverty income line
    1. column_names = ['Poor, at FPL', 'Near Poor, at 200% FPL', 'Moderate Income, at 300% FPL', 'High Income, at 400% FPL']

x. Build the two arrays that have that information in them as to match the column names:
    1. array with percent of 2016 FPL at different levels of income pct_2016_FPL = [pct_of_income_transportation_2016, pct_of_income_transportation_2016/2, pct_of_income_transportation_2016/3, pct_of_income_transportation_2016/4]
    2. array with percent of 2019 FPL at different levels of income pct_2019_FPL = [pct_of_income_transportation_2019, pct_of_income_transportation_2019/2, pct_of_income_transportation_2019/3, pct_of_income_transportation_2019/4]

xi. Build the data frame and show the chart
    1. df = pd.DataFrame({'Percent Income Share in 2016': pct_2016_FPL, 'Percent Income Share in 2019': pct_2019_FPL}, index=column_names)
    2. ax = df.plot.bar(rot=0, figsize=(15,10), title="Percent of Income Spent on MTA Monthly Pass")

xii. 

Percent of Income Spent on MTA Monthly Pass
Percent Income Share in 2016
Percent Income Share in 2019
0
2
4
6
8
10
12
Poor, at FPL
Near Poor, at 200% FPL
Moderate Income, at 300% FPL
High Income, at 400% FPL