Final Project Write-up: The Correlation of City Walkability/Bikeability with Obesity Rates

Names: Abigail Finn, Jake Hays

Team Name: Adventure Time w/ Finn & Jake

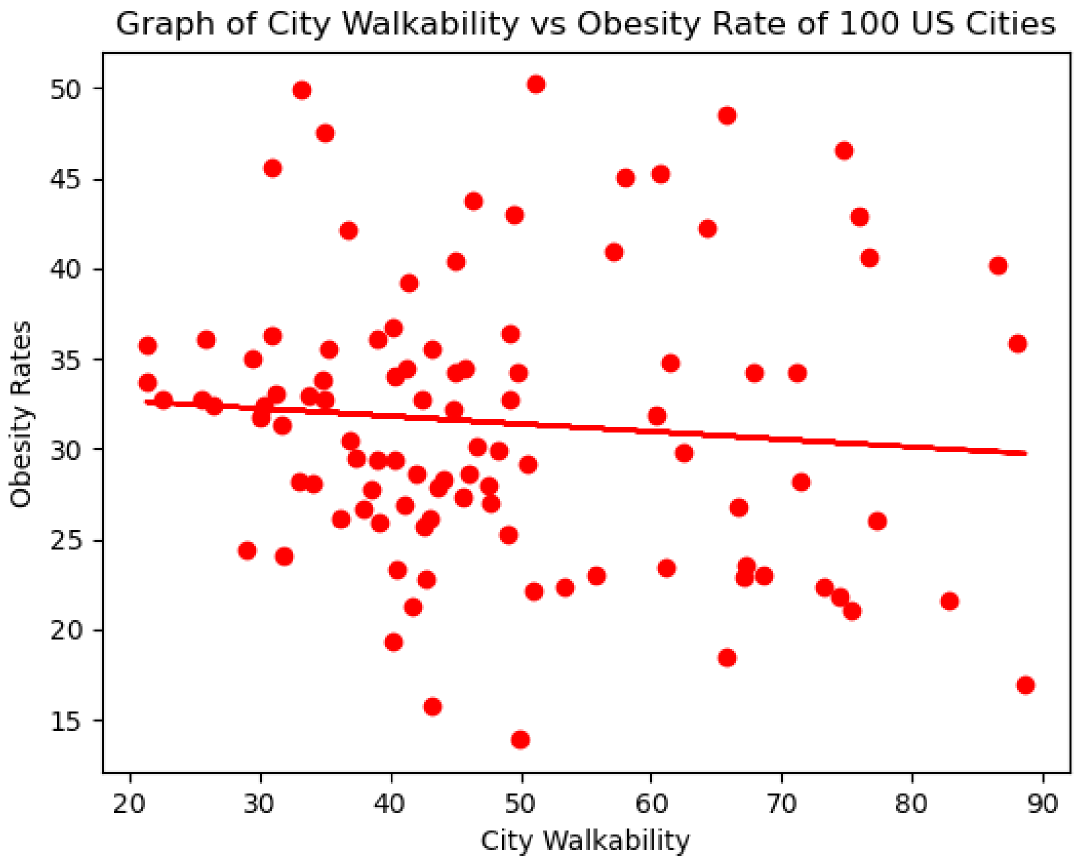
Github Link: https://github.com/afinn02/MOB206

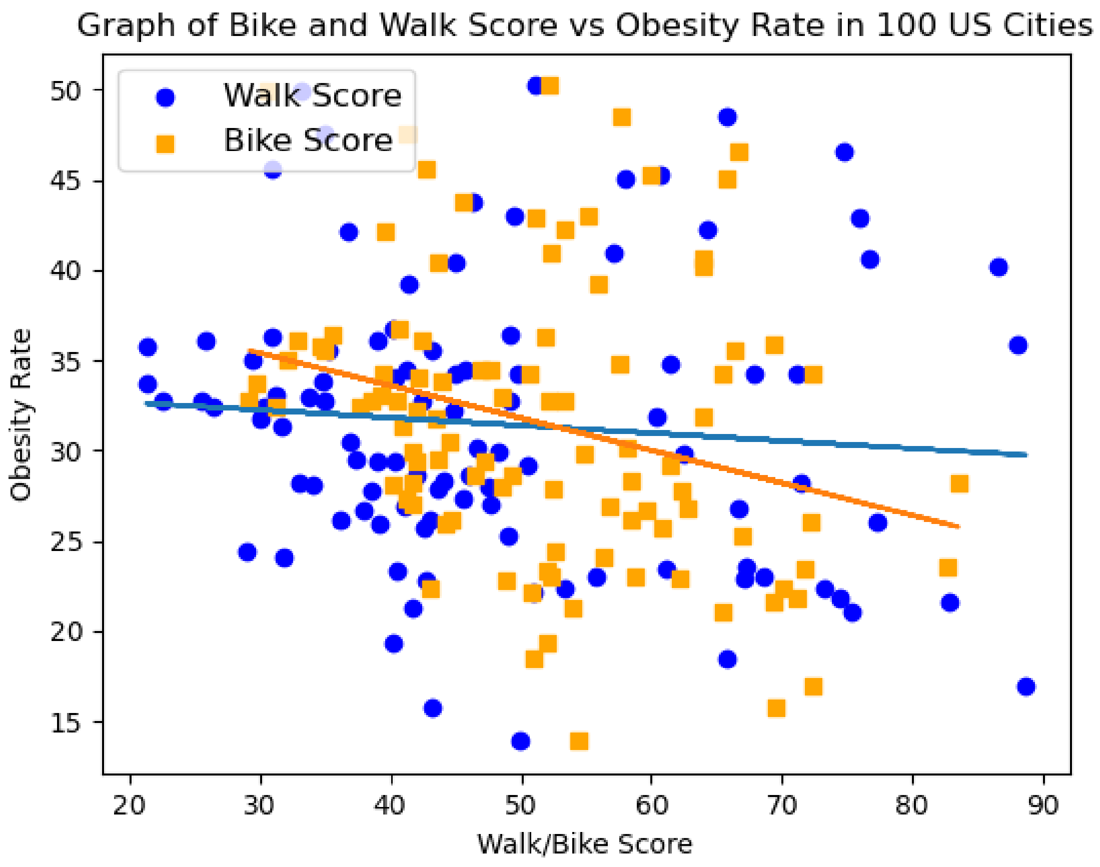
The major goals for this project were to determine if there is a correlation between how walkable or bikeable United States cities are with their obesity rates, and create visuals to represent this correlation. We accomplished this by calculating a correlation coefficient and a line of best fit for the respective data sets. The walkability index is a national measurement that quantifies a given city's accessibility by foot. Bikeability index measures how desirable it is to bike instead of drive. To achieve these goals, we pulled data from a website that ranked major cities in the U.S. by these two metrics. We then pulled corresponding data on obesity rates from the CDC's API. We stored this information in the "sqliteMove" database which we used to make calculations.

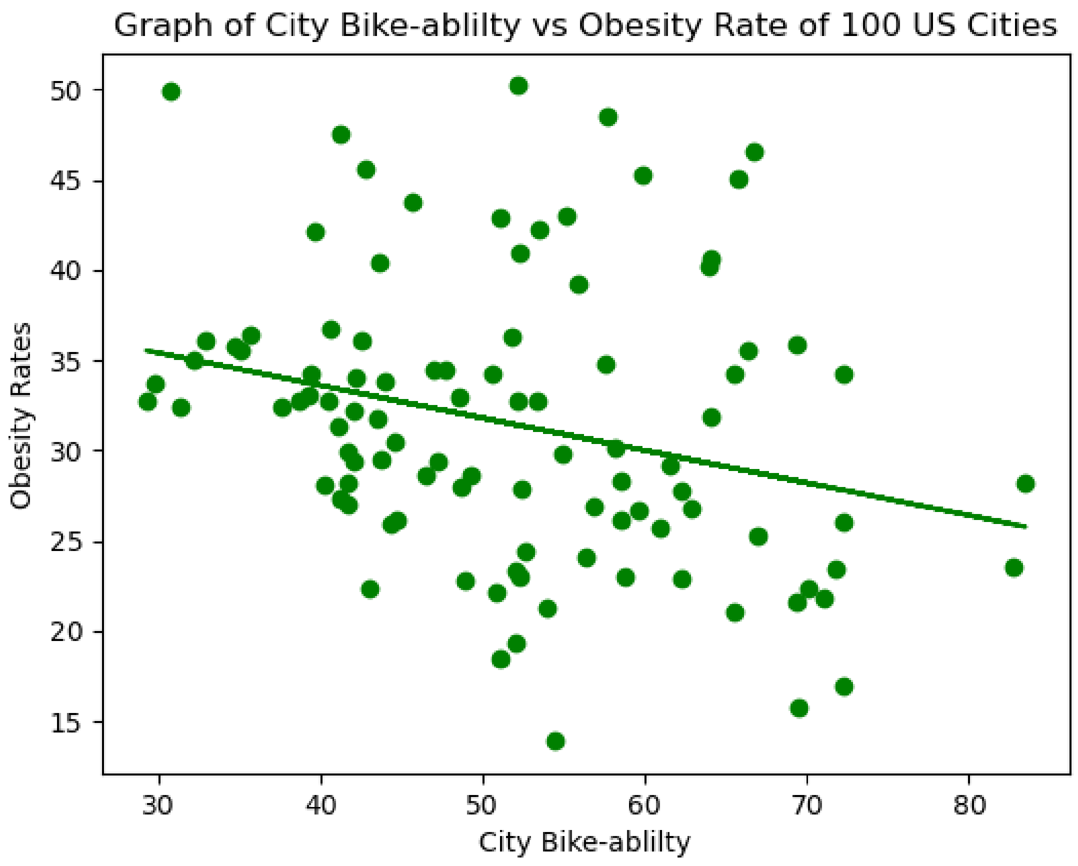
We achieved our goals of creating strong visualizations to represent the relationship of these variables as well as calculate the correlation coefficients; however, we did not find a strong relationship between either walkability and obesity and bikeability and obesity with r values of -0.08 and -0.275 respectively. If we were to redo this study, we would gather a larger dataset of cities across the U.S instead of 100 major ones. With more data, it would be easier to identify a potential correlation that is statistically significant.

The main problem we faced was accessing specific cities when working with the obesity rates API. The movement website only had around 100 major cities. The obesity API limited the amount of cities returned with none of them matching the ones in the "Movement" table. To combat this, we called the API based on the cities we already collected; this allowed us to gather more specific information. Although this was effective, calling individual cities from the "Movement" table was slower than the other method and needed far more requests. We also initially struggled with duplicate string data representing city names, but fixed it by identifying cities with a unique integer and keeping a separate table to connect the city name and city id.

Graphs:




Graph of Bike and Walk Score vs Obesity Rate in 100 US Cities
Walk Score
Bike Score
Obesity Rate
Walk/Bike Score
50
45
40
35
30
25
20
15
20
30
40
50
60
70
80
90



Link to calculations output file:

https://github.com/afinn02/MOB206/blob/main/AdventureTime.txt

Function Documentation:

Walking.py:

setupBase(sq, file): (Also used in Obesity.py)

- Purpose: To establish a base within a given file
- Input: SQL command to execute in form of a string, database file name in form of a string
- Output: no output

getSiteData():

- Purpose: Access data from the walkscore website and store it in a temporary list of tuples containing city, walkability index, bikeability index
- Input: nothing
- Output: list of tuples containing information we plan on sending to the database

countInsertions(curr, conn, basename): (Also used in Obesity.py)

- Purpose: Count the current number of items in the database in order to help us insert 25 at a time.
- Input: cursor to the database, connection to the database, name of the table in string form
- Output: integer value of the current number of items in the table

insertData(curr, conn, dataLst):

- Purpose: Insert the data contained in a list of tuples 25 at a time. This function iterates through the data list while calling the countInsertions function to accomplish this. It also creates a separate table to hold the ID integer and corresponding city name.
- Input: Cursor for the data set, connection object to the data set, table name in string form
- Output: No output

Obesity.py:

getCities():

- Purpose: Get the list of cities we gathered data for from the "Cities" table
- Input: No input
- Output: returns a list of strings representing different U.S cities

fetchAPI(cityList):

- Purpose: Get obesity rate from API for the cities passed in through the city list.
- Input: list of tuples containing city string names

- Output: returns a list of tuples with a city name (string) and obesity rate value (float)

insertObData(curr, conn, dataLst):

- Purpose: Insert the obesity rate with a corresponding city ID integer into the database 25 at a time. It utilizes the countInsertions function from the “Walking.py” file
- Input: cursor for the database, connection to the database, list of tuples representing the city name (string) and obesity rate (float)
- Output: No output

Calculations file:

createConnection(filename):

- Purpose: to create a connection between the filename input and a SQL cursor which will be used later to make tables
- Input: filename is the name of a database that is taken as a string
- Output: Returns a cur and conn SQL object so that the user can access the data from the database input later

joinTables(cur, conn)

- Function: This function serves to access data from the three tables within the database and use a select from join statement then uses fetchall to create a datalist with all of the data needed then iterate through the length of the data list and save the different columns as their own data list
- Input: this function takes the cur and conn objects that connect the database to the sql tables

- Output: walkingIndex, obesityRate, bikingIndex, cityNames are all data lists that have the values from the data that we got from the database that can be used later for calculations

getCorrelation(var1, var2):

- Purpose: This function takes two variables and finds the correlation coefficient between the two of them ®
- Input: The function takes two lists of values one being the explanatory variable and the other being the dependent variable
- Output: The function returns the correlation coefficient between the independent variable (var 1) and the dependent variable (var 2)

makeGraph(var1, var2, labelx, labely, color):

- Purpose: This function takes two lists of variable data two axis labels and a string of the desired graph color and creates a scatter plot that plots the points and line of best fit of the two variables
- Inputs: var1 is the independent variable that is passed in as a list of data, var2 is the dependent variable that is passed in as a list of data, labelx is the label of the x axis that is passed in as a string, labely is the label of the y axis that is passed in as a string, the color is input as a string
- Output: The function shows a graph of the two variables with var 1 on the x axis and var2 on the y axis and shows the line of best fit

def makeDoubleGraph(x1,x2,y):

- Purpose: This function takes two explanatory variables and one dependent variable and returns a graph of the two scatter plots layered on top of each other

- Input: x1 is a list of data for the first explanatory variable, x2 is a list of data for the second explanatory variable, y is a list of data for the dependent variable
- Output: The function shows a scatterplot of the two sets of data superimposed on each other and also returns the equation of best fit for the two correlations between x1 and y and x2 and y respectively.

def calculationOutput(filename, correlation1, correlation2, line1, line2):

- Purpose: Output calculations to a given file
- Input: filename = name of a file taken as a string, correlation 1 = correlation coefficient value as a float, correlation 2 = correlation coefficient value as a float, line1 = line of best fit (string), line2 = line of best fit (string)
- Output: None. Writes these calculations to a separate file

Instructions for Running Code:

1. Run the "Walking.py" file 5 times to insert city data into the base 25 at a time. You MUST run this file before the other two.
    a. This file creates two tables in "sqliteMove.db." "Movement" contains an integer representing the city name as well as the corresponding bike and walk index. "Cities" contains the unique integer with its corresponding city name as a string value.
2. Run the "Obesity.py" file 5 times to insert the obesity rates and corresponding cities into the base. This takes about 45 seconds per run.
    a. This file creates the table "Obesity" that contains the unique city ID integer with its corresponding obesity rate.

3. Run the "calculations.py" file once.
    a. This file calculates correlation coefficients for walkability and obesity as well as bikeability and obesity. It also creates 3 scatter plot graphs to visualize the relationship between the variables. The calculations are written out to the "AdventureTime.txt" file.

Date	Issue Description	Location of Resource	Did it Resolve the Issue?
12/1/2022	Needed specific city obesity data	Socrate CDC API https://dev.socrata.com/foundry/chronicdata.cdc.gov/bjvu-3y7d	Yes, we utilized their documentation to call data of specific cities in the walking data
12/9/2022	Was unsure how to add x and y labels as well as title to plt	Geeks for Geeks Matplotlib https://www.geeksforgeeks.org/matplotlib-pyplot-title-in-python/	Yes, this resource provided me with examples to help me add these features to our figures
12/9/2022	Didn't know how to add trend line to pyplot	Statology Matplot Trendline https://www.statology.org/matplotlib-trendline/	Yes, this resource provided me with examples to help me add these features to our figures. This resource had the added benefit of teaching me how to get the equation of the line of best fit for the data as well.
12/9/2022	I had to learn how to plot two scatter plots superimposed on the same graph	Stack Overflow https://stackoverflow.com/questions/4270301/matplotlib-multiple-datasets-on-the-same-scatter-plot	Yes, this discussion post provided me with an example of how to make the graph that we wanted and add a legend and change the shape to represent each

			variable
12/6/2022	I was having trouble inserting data into the table 25 at a time.	https://www.w3schools.com/sql/sql_select.asp	Yes. After reading more on select statements, I realized that I could use it to count how many items are in the database. From here I did a check for initial values and compared it to the current amount in order to go 25 at a time.
12/7/2022	I wanted to use functions that I had already created in a different python file, but I wasn't sure how to connect the two.	https://stackoverflow.com/questions/60318699/how-to-import-my-python-file-in-another-file-in-visual-studio-code	Yes this resolved the issue. I did not have previous experience connecting two files in vscode.
12/10/22	I was having issues understanding unique integer keys to avoid duplicate string data.	https://stackoverflow.com/questions/32661308/create-table-of-unique-foreign-keys	This did help slightly. I still needed clarification on how I should modify my tables, but it did increase my understanding.