

SI 206 Final Project Report

Jordan Shefman

Mark Qian

Project Goals

For our project, it was important to us that we were able to demonstrate all of the skills that we have learned over the course of the semester. While doing so, we aimed to learn about an interesting topic that we were both passionate about. The goal for this project was to check all of the boxes on the rubric. Additionally, it was a good experience to build a full project from scratch, rather than being presented with some starter code.

Goals Achieved

We believe that we have achieved all of these goals. We were able to build a report about different E-Bike companies and which countries in the world that they are located in. Throughout the process, we were able to take advantage of the CityBike API as well as using BeautifulSoup to obtain data from *The Most Bicycle-Friendly Cities of 2019* likes from Copenhagenize. After obtaining the raw data, we were able to present it in a few different ways, including CSVs and graphical visualization. Most importantly, we have become comfortable with receiving an assignment and planning out every step. This is a very important skill to have in the industry, when you are not given starter code.

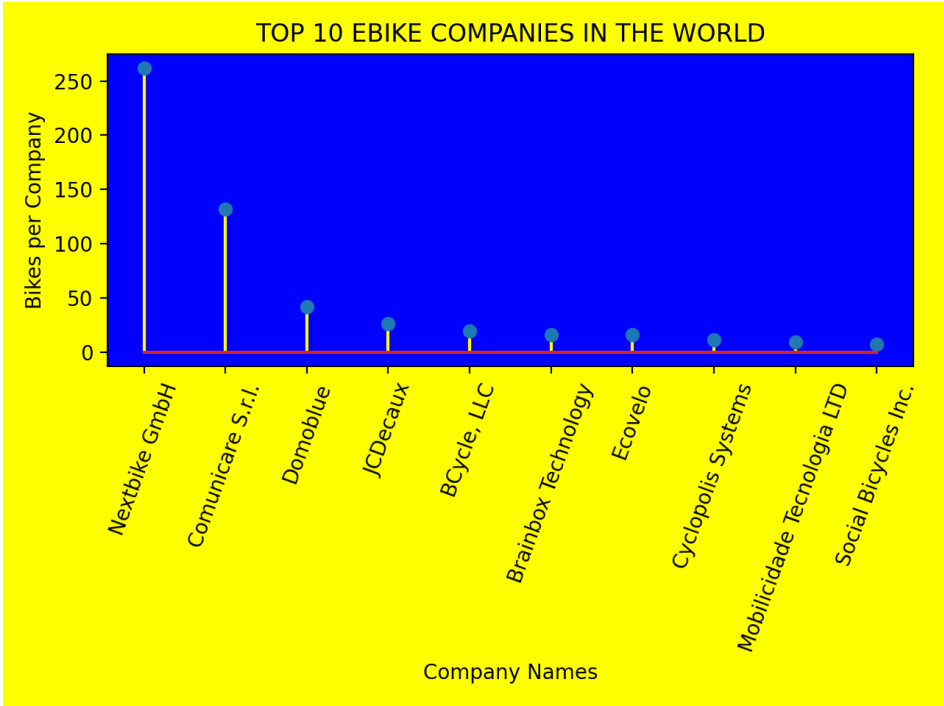
Problems Faced

The biggest problem that we faced was our conflicting schedules. It was difficult for us to find time to work together, so we had to collaborate remotely. We created miniature deadlines in order to stay on track and finish on time. The other problem we faced was with Matplotlib. We found it difficult to come up with plots that would represent our data, not copy each other, and still remain different from the plots that were presented to us in class. However, we were able to overcome both of these problems.

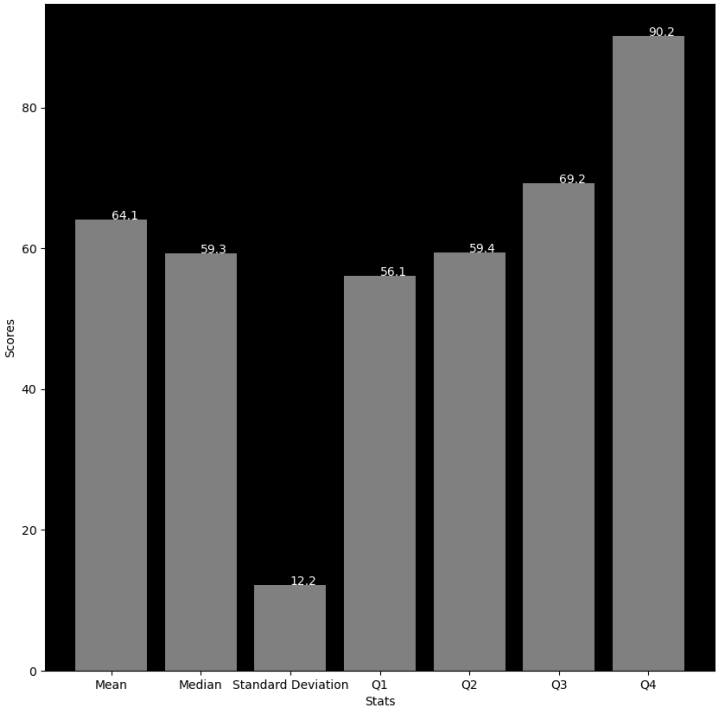
Output File Sample

1 CITY,COUNTRY,NUMBER OF BIKES	24 São Paulo,BR,2	47 Baeza,ES,1	70 Fort Worth, TX,US,1
2 Moscow,RU,1	25 Aranjuez,ES,1	48 Boise, ID,US,1	71 Udine,IT,1
3 Antequera,ES,1	26 A Coruña,ES,1	49 Boulder, CO,US,1	72 Fargo, ND,US,1
4 Igoumenitsa,GR,1	27 Ferrol - Narón,ES,1	50 Santa Monica, CA,US,1	73 Salt Lake City,US,1
5 Lecce,IT,1	28 Curtin University, Perth, WA,	51 Rimini,IT,1	74 Phoenix, AZ,US,1
6 Ioannina,GR,1	29 Melbourne, AU,AU,1	52 Fort Lauderdale, FL,US,1	75 Omaha, NE,US,1
7 Karditsa,GR,1	30 Ferrara,IT,1	53 Milwaukee, WI,US,1	76 Houston, TX,US,1
8 Manerba del Garda,IT,1	31 Montréal, QC,CA,1	54 Buffalo, NY,US,1	77 Boston, MA,US,1
9 Kavala,GR,1	32 Parma,IT,1	55 Washington, DC,US,1	78 Ustica,IT,1
10 Kinouria,GR,1	33 Toronto, ON,CA,1	56 Charlotte, NC,US,1	79 Philadelphia, PA,US,1
11 Komotini,GR,1	34 Hamilton, ON,CA,1	57 Saluzzo,IT,1	80 Cieza,ES,1
12 Patra,GR,1	35 Ottawa, ON,CA,1	58 Benidorm,ES,2	81 Indianapolis, IN,US,1
13 Marciana Marina,IT,1	36 Badajoz,ES,1	59 Cincinnati, OH,US,1	82 Orlando, FL,US,1
14 Aranda de Duero,ES,1	37 Ann Arbor, MI,US,1	60 Savigliano,IT,1	83 Como,IT,1
15 Rethymno,GR,1	38 Pavia,IT,1	61 New York, NY,US,1	84 Kansas City, MO,US,1
16 Samos,GR,1	39 Perugia,IT,1	62 Tampa, FL,US,1	85 Dayton, OH,US,1
17 Novara,IT,1	40 Austin, TX,US,1	63 Terni,IT,1	86 Madison, WI,US,1
18 Nicosia,CY,2	41 Chattanooga, TN,US,1	64 Columbus, OH,US,1	87 Los Angeles, CA,US,1
19 Mani,GR,1	42 Piacenza,IT,1	65 Denver, CO,US,1	88 Ketchum / Sun Valley, ID,US,1
20 Irakleio,GR,1	43 Pinerolo,IT,1	66 Chicago, IL,US,1	89 Minneapolis, MN,US,1
21 Thessaloniki,GR,1	44 Portland, OR,US,1	67 Tirano,IT,1	90 La Spezia,IT,1
22 Athens,GR,1	45 Reggio Emilia,IT,1	68 El Paso, TX,US,1	91 Don Benito - Villanueva,ES,1
23 Padova,IT,1	46 San Ramon, CA,US,1	69 Blanca,ES,1	92 San Antonio, TX,US,1

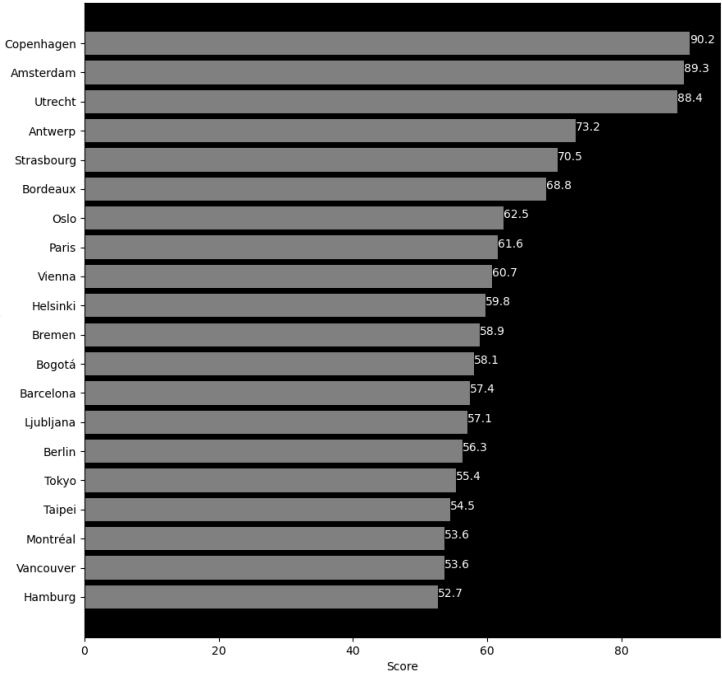
Visualizations

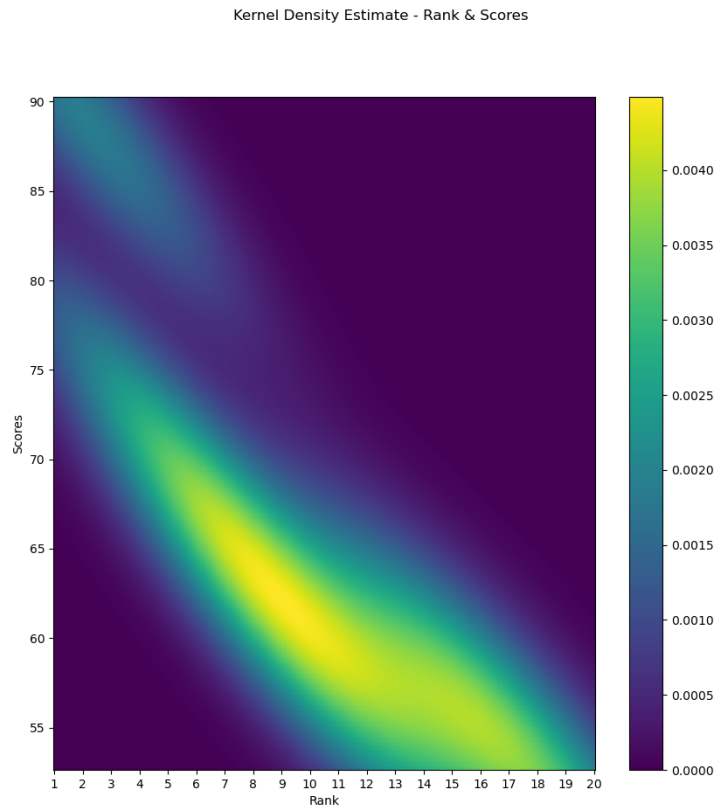


Statistics of the Copenhagenize Index



The Most Bicycle-Friendly Cities in 2019





Instructions for Running the Code

To run the code, simply press run. The first several times the code is run, the data will not be accurate, because only 25 lines of data can be added to the database at a time. There were over 600 total lines of data, so only after running the program enough times to get it all in the database will the output be accurate. The only other thing to do while the code is being run is to exit the visualization to allow the rest of the code to execute.

Function Documentation

getData()

Input: none

Output: sorted list of best bike friendly cities according to copenhagenize index

Action: uses beautiful soup to scrape data and prints the sorted list

setUpDatabase(db_name)

Input: db_name: string

Output: cur, conn

Action: creates a database with the given name

createBikesTable(cur, conn)

Input: cur, conn

Output: none

Action: creates a bikes table if one does not already exist

createCitiesTable(cur, conn)

Input: cur, conn

Output: none

Action: creates a cities table if one does not already exist

createCountriesTable(cur, conn)

Input: cur, conn

Output: none

Action: creates a countries table if one does not already exist

createCompaniesTable(cur, conn)

Input: cur, conn

Output: none

Action: creates a companies table if one does not already exist

createNamesTable(cur, conn)

Input: cur, conn

Output: none

Action: creates a names table if one does not already exist

readAPI()

Input: none

Output: results: dict

Action: makes a request to the city bikes API. if request is OK, returns results as a dict. If not OK, prints error message and returns none

addBikes(cur, conn, bikes_dict)

Input: cur, conn, bikes_dict

Output: none

Action: adds bikes to the bike table 25 at a time. Looks up city, country, etc in other tables to get their ids, so bike table does not store repeating strings

addCities(cur, conn, bikes_dict)

Input: cur, conn, bikes_dict

Output: none

Action: adds cities and ids to their table

addCountries(cur, conn, bikes_dict)

Input: cur, conn, bikes_dict

Output: none

Action: adds countries and ids to their table

addCompanies(cur, conn, bikes_dict)

Input: cur, conn, bikes_dict

Output: none

Action: adds companies and ids to their table

addNames(cur, conn, bikes_dict)

Input: cur, conn, bikes_dict

Output: none

Action: adds names and ids to their table

getCounts(cur, conn)

Input: cur, conn

Output: res: list of tuples (city, country, count)

Action: computes the counts of of bikes per city

getMaxCount(count_list)

Input: count_list

Output: (max, max_city)

Action: finds the city with the most bikes and outputs as a tuple of the count and the city

bikesByCompany(cur, conn)

Input: cur, conn

Output: matplotlib visualization

Action: computes the number of bikes that each company has in operation and creates a lollipop chart of the top ten companies with the most bikes

get_data()

Input: nothing

Output: city_list

Action: retrieves the data we want from the website and puts them in a tuple list

city_lst(city_list)

Input: city_list

Output: cities

Action: splits the cities from the tuple with their rank and score and puts them in their own list

score_lst(city_list)

Input: city_list

Output: scores

Action: splits the scores from the tuple with their rank and city and puts them in their own list

rank_lst(city_list)

Input: city_list

Output: ranks

Action: splits the ranks from the tuple with their city and score and puts them in their own list

stats(scores)

Input: scores

Output: stats

Action: reads in the scores and computes mean, median, standard deviation, Q1, Q2, Q3, and 100th quantile

city_score_visual(cities, scores)

Input: cities: list, scores: list

Output: matplotlib visualization

Action: creates a horizontal bar chart that demonstrates the scores that each city has received on their bike friendliness

stats_visual(scores_stats)

Input: scores_stats: list

Output: matplotlib visualization

Action: creates a bar chart that demonstrates the stats that were computed in stats()

kernal_estimate_visual(ranks, scores)

Input: ranks: list, scores: list

Output: matplotlib visualization

Action: creates a kernel density estimate plot with a color bar that demonstrates the relationship between the rank and score

Resources Used

Date	Issue Description	Location of Resource	Result
3/28/22	Needed website to scrape data from	https://copenhagenizeindex.eu	Solved
2/28/22	Needed an API	http://api.citybik.es/v2/	Solved
4/4/22	Needed to find visualization technique not explicitly used in class	https://seaborn.pydata.org/introduction.html	Not solved
4/4/22	Needed to find visualization technique not explicitly used in class	https://matplotlib.org/stable/gallery/index	Solved