Data Mechanics
Project Report
J. David Brawn, Sarah Larbi, Jennifer Liang, Taylor Potye

Boston is known across the US as one of the biggest college towns in the country, but what effect does this have on the city itself? With over 150,000 students in the city of Boston alone, schools and their young populations are bound to impact the surrounding area, just as the surrounding area impacts each school. We chose to use datasets pertaining to three main factors, social, safety, and accessibility, to create a weighted ranking for each university based on its surroundings. To do this we have used the following datasets: Boston College and Universities from Analyze Boston[1] , Boston Crime Incident Reports from Analyze Boston[2], MassDOT Car Crash Data from MassDOT[3], MBTA Bus Station Location Data from MBTA[4], Boston Food Establishment Licenses from City of Boston[5], Boston Entertainment Licenses from City of Boston[6], and Boston Police Station from City of Boston[7]. With these data sets we performed transformations that make this data useful for analysis. We then aimed to first see what weight would establish the highest ranking for any individual school and then to use k-means in order to establish the optimal placement for new police stations in order to have the highest positive impact on the safety scores. To run our project the following libraries are required: gpxpy, dml, prov, tqdm, scikit-learn, and scipy.

[1] https://data.boston.gov/dataset/colleges-and-universities
[2] https://data.boston.gov/dataset/crime-incident-reports-august-2015-to-date-source-new-system
[3] http://www.massdot.state.ma.us/highway/Departments/TrafficandSafetyEngineering/CrashData.aspx
[4] http://realtime.mbta.com/portal
[5] https://data.cityofboston.gov/Permitting/Active-Food-Establishment-Licenses/gb6y-34cq
[6] https://data.cityofboston.gov/Permitting/Entertainment-Licenses/qq8y-k3gp
[7] https://data.cityofboston.gov/resource/pyxn-r3i2.json

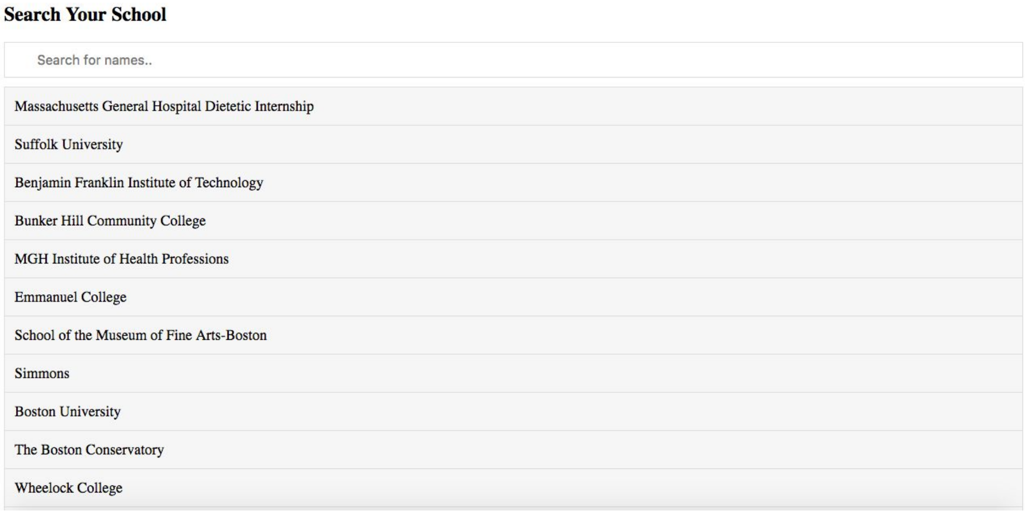
The first transformation we did on the data creates a union of the zipcode and business names from the entertainment and food license datasets. It then takes the aggregate and returns a count of the number of 'social businesses' with the zip code corresponding to those businesses. Ultimately this could be assigned some sort of weight in regards to how social a college is given the number of entertainment and food vicinities in its area. The code for this transformation can be seen in the file transformation1.py. The next transformation is found under the file safetyTransformation.py which ultimately makes a data set that includes each college and how many crimes and crashes are located within a mile of each school, in the form {'Name': , 'Number of Crimes': , 'Number of Crashes': } using the colleges dataset, crash data, and the crime incident reports. To do this we take the crash and crime data, and for each set, loop through every crash or crime for every college, and determine using the gpxpy library if each incident is within a mile of the school. If so we append to a list the name of the school, and the number 1, representing the incident. Then we aggregate these with the sum function to find the total number of crimes or crashes within a mile of each school. Now we have two of these sets, one with the totals for crime, and one with the totals for crashes, and we take the product of the two, select if the schools are the same, then project to just include the school and the two totals. We believe this new data set can be used to perform an analysis on the safety of the surrounding areas of the different universities in Boston. Another transformation we computed dealt with the MBTA data we used and is found under the file mbtaTransformation.py. This transformation makes a data set that includes each college, the number of MBTA bus stops located within a mile, and the number of students for that college, in the form {'Name': , 'Number of MBTA stops': , 'Number of Students': } using the colleges and MBTA stops datasets. To do this we first make a list aggregating the sum of the number of stops within a mile of each school in the same way we did above for crimes and accidents. We then take a list of the schools and their student populations which we have from the college data set, and similarly perform a product, selection, and projection on the two lists to combine the two statistics with the school name in one data set. We believe this data set could be used to perform an analysis on the accessibility of transit for each school, factoring in student population for how many stops might be needed, or the density of availability. Our final transformation uses the Boston Police station location data and the college dataset. This transformation determines the number of police stations within a mile of each school through distance calculations and sum aggregation. This transformation can be found under policeAnalysis.py.

Now that we have transformed our data, we can use this to develop our college ranking optimizer. We started off by finding the individual scores per category per university in our safetyScore.py, transitScore.py, and socialScore.py scripts by normalizing the number of relevant datapoints within a mile of each school (e.g. number of bus stops for the transit score). From there, we used these scores to find an overall ranking for each university assuming that each category were to be weighted equally in our overallScore.py script. In rankingOptimizer.py, we aim to find the optimal weight for each category that would maximize an individual school's own ranking. The use case of this idea is from the perspective of an admissions office wanting to present the maximum possible ranking for their own university. We set a lower bound on the category weight to be 20% and the upper bound at 50% to eliminate the possibility of any school setting their highest category score as 100% or lowest as 0%. In order to play around with this, just change the SCHOOL_NAME variable at the top of the file to the school whose ranking you would like to maximize. We found that new weights can actually have profound impacts on a particular school's ranking. For example Boston University School of Public Health's rating when optimized increased from 16/57 to 9/57, just by changing category weights from 33/33/33 to 44/36/20. On the other hand we found that Boston College is so bad, there is no possible weighting to improve their rank of dead last.

Given our ranking system, some schools may find that they wish to improve their score in one of our categories to improve their ranking. We came up with a way a school could improve their safety score by adding new police stations. In safetyCorrelation.py we found a significant and very high positive correlation between the proximity of police stations to a university and it's safety score (with a correlation coefficient of 0.77 and a p-value of 0). Because of this high correlation, we decided to find optimal locations for new police stations in order to improve the safety scores of the lowest scoring universities in that category. In other words, if the city was going to invest in a couple police stations, we want to find the best places to improve university safety across the city where it is needed most. In newStations.py, using our police stations data and the locations of universities with bottom half safety scores (score < 0.5), we use the k-means algorithm to find optimal placement for new police stations in order to improve safety scores for the lowest scoring universities. You can change the number of means run in our algorithm (i.e. number of new police station locations) by changing the NUM_CLUSTERS variable at the top of the file. We found 3 to work well.

For our ranking system and new police station generator, we have created interactive web services for our users. Our ranking system web service can be used by running the file app.py in the RankingApp subdirectory, and it will bring you to a screen where you can type in the name of the school you wish to optimize the ranking of. You must type in the name exactly as it appears in the option bar and press enter, if it's not a valid input the page will refresh. For example if you type in, "Northeastern University" it will give you the original ranking with the scores for each category and it give you the improved ranking with the new scores for each category. This web service takes a few moments to run since it uses our ranking optimizer algorithm that analyzes multiple combinations of ways to improve rankings, and progress can be tracked in the terminal with a status bar. This web service can be illustrated in Figures 1 and 2.

Figure 1.

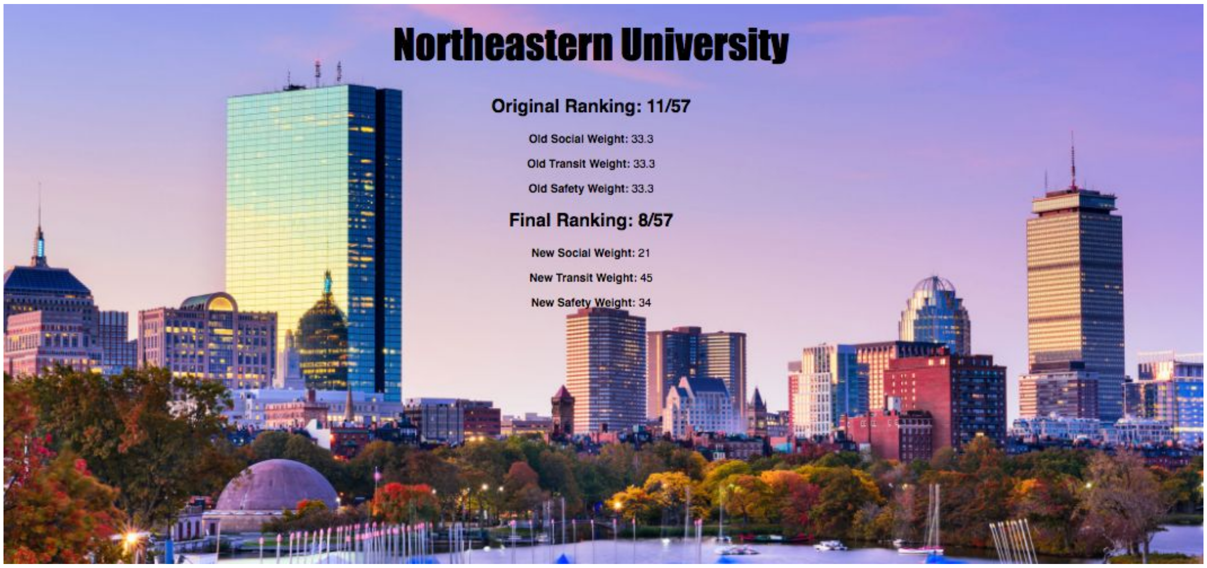


Search Your School

Search for names..

Massachusetts General Hospital Dietetic Internship
Suffolk University
Benjamin Franklin Institute of Technology
Bunker Hill Community College
MGH Institute of Health Professions
Emmanuel College
School of the Museum of Fine Arts-Boston
Simmons
Boston University
The Boston Conservatory
Wheelock College

Figure 2.



Northeastern University

Original Ranking: 11/57

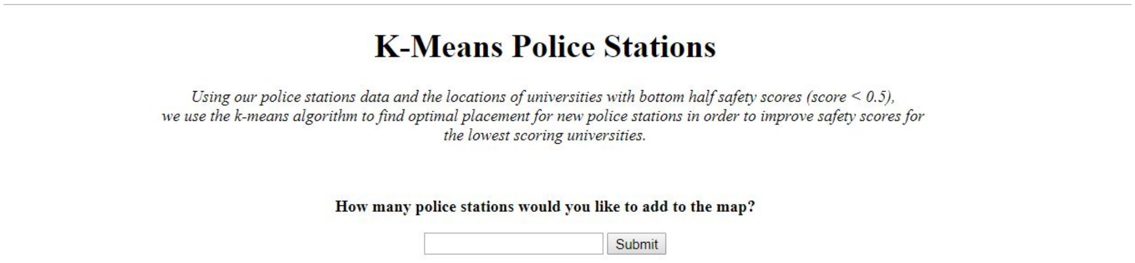
Old Social Weight: 33.3
Old Transit Weight: 33.3
Old Safety Weight: 33.3

Final Ranking: 8/57

New Social Weight: 21
New Transit Weight: 45
New Safety Weight: 34

Our other web service can be used by running kmeansapp.py inside the mapvisualizations subdirectory. This web service takes in the number of police stations you would like to generate and returns the geographical locations on a map, based on the results of our k-means algorithm, of where those police stations should be placed in order to improve safety in low safety score areas. An illustration of this web service can shown in Figures 3 and 4.

Figure 3.

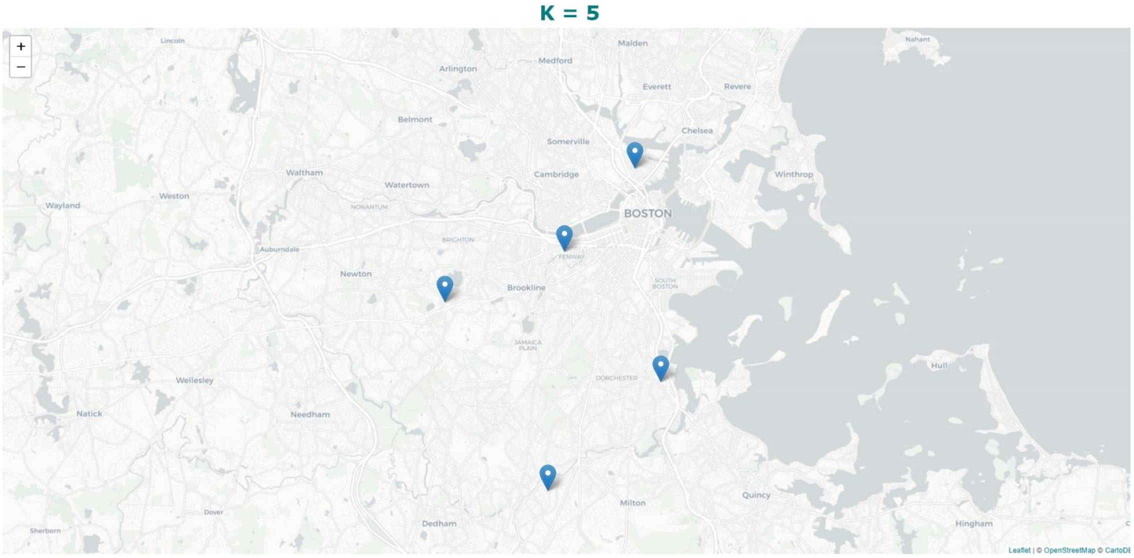


K-Means Police Stations

Using our police stations data and the locations of universities with bottom half safety scores (score < 0.5), we use the k-means algorithm to find optimal placement for new police stations in order to improve safety scores for the lowest scoring universities.

How many police stations would you like to add to the map?

Figure 4.



K = 5

To continue this work, we have thought about whether the introduction of new police stations to low-scoring areas would improve the surrounding school's safety scores over time. If so, would this be the best approach to helping school's improve safety on campus? We have also thought about whether the rankings we have determined are significantly impacted by Boston's anomaly of having a dense population of schools spread over a small area. Could the algorithms we have designed be applied to any area with a large number of universities? We also believe that our rankings would be influenced by adding datasets of surrounding areas such as Brookline and Cambridge, so this would be a good next step to take if these datasets are available and comparable.