COMP20008 PHASE 4 REPORT

1. Title:
City of Melbourne Survival Guide for the Homeless

2. Domain:
communities, public amenities

3. Question Statement:
My project will be trying to seek the best spots in the City of Melbourne for the homeless to hang around. There are three assumptions to the project. 1) Money and food would be the top priorities for the homeless. 2) Larger pedestrian counts will more likely yield more income for a homeless individual. 3) It is in their interest to get as close to a cheap/free food place as possible. The project will try to find the spots in the City of Melbourne with large pedestrian flows that are also close to places that provide free/cheap meal. The time was also taken into consideration, since not all community food places provide all three meals. The result will hopefully benefit the homeless in Melbourne and help them get financial income and food more easily than before.

4. Datasets:

1) Community Food Map:
URL: https://data.melbourne.vic.gov.au/People-Events/Community-Food-Map/uwyu-5y9e

Attribute: Category
Expected Type: text
Description: This attribute categorizes the type of food service community centers provide.

Attribute: Who
Expected Type: text
Description: Specifies to whom the food service is open to

Attribute: Monday - Sunday (multiple fields)
Expected Type: datetime
Description: Indicates the open time of the food service. Very poorly formatted.

Attribute: Latitude, Longitude
Expected Type: float
Description: The geographical coordinates of community centers.

Attribute: Name
Expected Type: text
Description: Name of community centers.

Other attributes: Attributes irrelevant to the project, including address, contact details, websites etc.

2) Pedestrian Counter Locations:
URL: https://data.melbourne.vic.gov.au/Transport-Movement/Pedestrian-Sensor-Locations/ygaw-6rzq

Attribute: Sensor ID
Expected Type: int
Description: ID's of pedestrian counters

Attribute: Sensor Description
Expected Type: text
Description: The general locations of pedestrian counters.

Attribute: Status
Expected Type: text
Description: "Installed" or "Test"

Attribute: Latitude, Longitude
Expected Type: float
Description: The geographical coordinates of pedestrian counters

Other attributes: Sensor Name, Year Installed, Sensor Name. Irrelevant to the project.

3) Pedestrian Counts:
URL: https://data.melbourne.vic.gov.au/Transport-Movement/Pedestrian-Counts/b2ak-trbp

Attribute: Date_Time
Expected Type: datetime
Description: At which hour of the day the count is registered.

Attribute: Sensor_ID
Expected Type: int
Description: ID of the sensor where the count is registered.

Attribute: Sensor_Name
Expected Type: text
Description: The general locations of pedestrian counters.

Attribute: Hourly_Counts
Expected Type: int
Description: The hourly pedestrian count.

5. Pre-processing:

1) Community Food Map:

- Filter on “Who” attribute: Since I want the result of the project to be useful to homeless people in general, despite their gender and age, my program picked records containing any keyword in [“everyone”, “homeless”, “over 18”].
- Filter on “Category” attribute: My program picked records of community centers that provide cheap/free food service. A simple string matching for “Free and cheap meals” will do.
- Normalizing open time of community centers: Three formats of data exists for these attributes (Monday - Sunday): “Closed” indicating the community center will be closed on that day, with indication of what meal the food service is open for eg. “Breakfast: 8.00am - 9.30am”, and no indication at all eg. “12.30pm - 3.00pm”. The ideal format would be a tuple, storing 0 if the community center is open for breakfast, 1 for lunch, 2 for dinner, and empty for closed. eg. (1, 2) would indicate open for lunch and dinner. Keywords “Closed” and “Breakfast” (or “Lunch”, “Dinner”) can be easily picked up by string matching. The problem lies in the third format (“12.30pm - 3.00pm”) where no keyword is given. My approach was to first convert them into 24hr clock format and store the data in a list of two floats (eg. [12.30, 15.00]). Then I matched it against the meal time specified by myself (breakfast: 7-10, lunch: 11-14, dinner: 18-21), looking for overlap, which indicates the food service will be open for that meal.
- Cleaning up: Finally the irrelevant fields need getting rid of. ID’s for community centers are also added to the dataset, since there are no ID’s for community centers in the original dataset.

2) Pedestrian Counters Locations:

- Filter on “Status” attribute: Most sensors’ status are “Installed”. But there still exist sensors, of which the status is “Test”. Testing sensors will be ditched for the purpose of this project.

3) Pedestrian Counts:

- Data aggregation: The assumption is that the average hourly pedestrian count of one sensor would a fairly good representation of the pedestrian count at that location at any hour of the day. My approach is to calculate the total pedestrian count of one sensor divided by the

number of records of that sensor in the dataset, which will give the average hourly pedestrian count of that sensor.

6. Integration:

1) Load in data:

- Community Food Map: Loading data into a dictionary made data manipulation fairly easy. The load_food_data function in file explore.py will load in community food map datasets and return a dictionary in the format of {community_center_id: ((latitude, longitude), (meals for mon), (meals for tue), …, (meals for sun)), …}. So the index 0 will store geographical coordinates as a tuple, and index 1 - 7 will store meal information of Monday - Sunday.
- Pedestrian Counter Locations: The function load_sensors_data in explore.py will load in pedestrian counters locations dataset and return a dictionary in the format of {sensor_id: (latitude, longitude), …}
- Pedestrian Counts: The function load_counts_data will load in pedestrian counts dataset and return a dictionary in the format of {sensor_id: avg_count, …}

2) A simple scoring system:

- Score pedestrian counter locations by average hourly pedestrian counts: Each pedestrian counter location was given a normalized score_count = (val - min)/(max - min) between 0 which indicates the worst pedestrian count in the dataset and 1 which indicates the best pedestrian count in the dataset, where val is the average hourly pedestrian count of the location being assessed, max and min are the maximum and minimum of the average hourly pedestrian counts in the set respectively.
- Score pedestrian counter locations by the distance to the nearest community food service: The logic is very similar to the calculation of score by pedestrian count. Only this time, the smaller the value of minimum distance to a food service is, the higher the score is supposed to be. Thus a score was given by score_dist = (max - val)/(max - min), where val is the minimum distance from the location being assessed to a community food service, and max and min are the maximum and minimum of the distance to a nearest community food service in the set respectively. And same with score by pedestrian count, score_dist is also between 0 and 1, with a higher score indicating closer distance to a community food service.
- Calculating final score: The final score was calculated by giving score_count and score_dist from the previous two steps different weight. The formula used in the current state of the project is score_final = score_count * 0.6 + score_dist * 0.4, where the average pedestrian count was given slightly more weight than the distance to the nearest community food service, because firstly a homeless individual would spend most of his/her time on the street looking to make financial gain rather than waiting for lunch or dinner to start, and secondly, money would provide more general benefit to the homeless than food would.

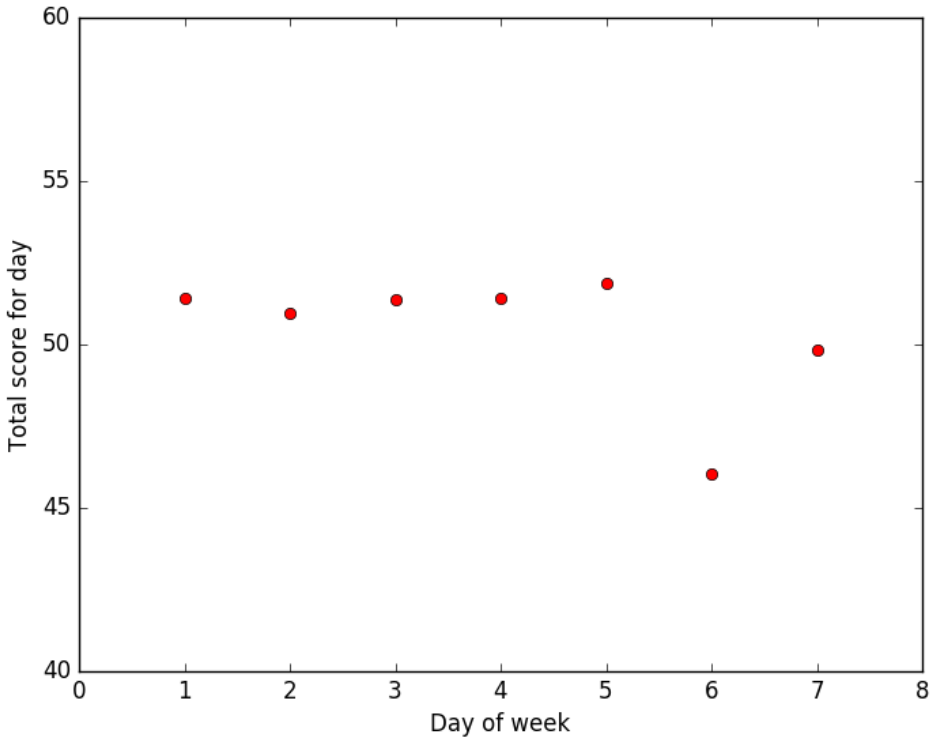
- The final score for each pedestrian counter location was considered a fairly good measurement of how beneficial the spot is for a homeless individual to spend his/her day at, and was used to further explore the information contained in the datasets.

7. Results:

1) **Given day of the week and meal of the day, find the best location in the City of Melbourne for the homeless to stay at:**

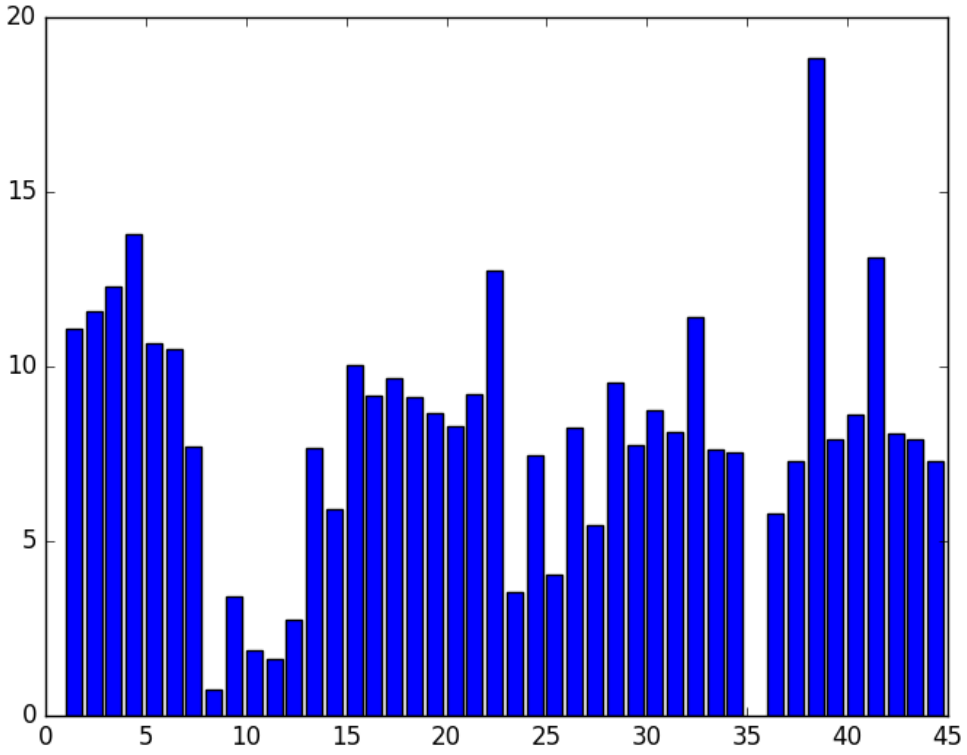
The function survive in explore.py does exactly this. Its first argument is an integer between [1, 7] that represents day of the week from Monday to Sunday. Its second argument is and integer in {0, 1, 2} that that represents breakfast, lunch and dinner. The function will take these two arguments and compute the the highest final score for given conditions and the id of the location that possess this highest score. It turned out that two locations in the City of Melbourne really stood out and constantly appeared in the top 3 list no matter the input. The first place was taken by sensor #38, and its location is Flinders St-Swanston St (West), followed by #4 at Town Hall (West) in the second place. Pedestrian counter #38 simply has tremendously more pedestrian count each day than any other counter in the dataset, making it impossible for them to surpass sensor #38 as the highest score, even thought other sensor locations may have an advantage in distance to community food service.

2) Find the best/worst day of the week for the homeless in the City of Melbourne:



The function score_days in explore.py computes the total final score of all pedestrian counter locations for every combination of day of the week and meal of the day, then sums up the total score for each day. The total score of all places for a day was considered a fairly good representation of how good the day was for the homeless. There are slight differences in scores from Monday to Friday, with Friday being the best of the week for the homeless. But the scores drop drastically in the weekend, with Saturday having the lowest total score of the week. The data shows that more community food services are available in the weekends than in weekdays. The severe drop in scores during the weekend may very possibly be caused by decrease in pedestrian counts, since people tend to rest at home or go on vacations during the weekend.

3) Find the best/worst spot for the homeless in the City of Melbourne:



My approach to this problem was very similar to the approach to finding the best/worst day of the week for the homeless. The function score_spots in explore.py will, for every possible combination of day of the week and meal of the day, calculate the total final score of a location in a week and return a list of pedestrian counter id's sorted in ascending order of their total scores. The total score of a location in seven days was assumed to be a decent measurement of how good the place was to a homeless individual. Not surprisingly, the location of sensor #38 Flinders St-Swanston St (West) was proved to be the best spot overall in the City of Melbourne

for the homeless, since it exceeded every other location of pedestrian counters in scores for every possible input. Whereas the worst spot in the City of Melbourne appeared to be the location of pedestrian counter #8 at Webb Bridge, with a total score dramatically lower than other locations in the dataset. The missing value in the graph was due to the fact that that sensor was ditched from the original data because it was a sensor for testing.

4) Limitations:

- The weight on score by pedestrian count and score by minimum distance to community food services when calculating the final score was debatable. The reason why I initially picked 6/4 was that I did not want equal weight on these two values, and also sensed money might be more important and desirable to the homeless after I actually talked to a homeless man. Thus the weight on the two values was purely based on personal experience, and can definitely be tweaked to improve the quality of the results.
- One of the major assumptions made at the initiation of the project was higher pedestrian counts will yield more income for the homeless. But the reality might be more complicated than this simple assumption, because the income of the homeless could also be affected by the financial status of pedestrians and pedestrians' willingness to give money. Information on these two factors is not reflected in the original dataset. The results could definitely be improved by including these two factors in calculation of scores.

8. Value:

The value of this project mainly comes from the data aggregation, data linkage and data normalization.

- The main method for processing Pedestrian Counts dataset was data aggregation. Each record in the original dataset simply contain the hourly pedestrian count registered by a sensor in an exact time interval. Without data aggregation to reveal the average pedestrian counts that could represent how many pedestrians pass by every locations in the dataset at any given time, the original dataset could be of little value.
- Data linkage was important when processing Pedestrian Counts dataset and Pedestrian Counter Locations dataset. Records in both dataset share the same set of ID's as their primary key. Data linkage thought ID's made possible connecting pedestrian counts to geographical coordinates.
- Data normalization was heavily applied when processing the meal time in the Community Food Map dataset. Since the values of meal time in the dataset were poorly formatted, they had to be converted into values of a consistent format, in my case, a tuple containing integers that indicate which meal of the day, for easy further processing. Otherwise, there was no way to process meal time without normalization.

9. Challenges & Reflections:

The most noticeable challenge so far lies not in processing and analyzing the data, but in justify the usefulness of the results to the homeless. Apparently the scoring system for different

locations in the City of Melbourne can be of use to the homeless if it is implemented in a web application or mobile application, provided homeless people have access to the Internet or smartphones. It is easy to think that homeless people do not own a mobile phone because they can hardly afford food, let alone electronic devices. But the truth is that most homeless people might have both a basic phone for receiving phone calls and then a smartphone which they cannot afford a contract on, but can use for accessing the Internet. (Spinks 2015) The real challenge for the homeless to use a smartphone lies in charging the battery, which can be a real struggle for homeless individuals. (Bell 2015) But with the increasing awareness of how technologies can be used to help the homeless the the rising availability of technologies, the result could be more and more useful to the homeless.

10. Question Resolution:

The results the the project showed given a date and meal time, what the best location in the City of Melbourne for the homeless is. It also showed on which day of the week the homeless need help the most. Firstly, the result would obviously interest the homeless individuals, since they are most likely to gain more financial income by following the result. Secondly, the result will likely be useful to the government. Because locations in the city that are deemed “good” by the results of the project naturally attract more homeless people even though the homeless may not consciously know the results of this project. With the results, the government can provide more help and supporting services to the homeless in areas that are more likely to attract homeless people, so that resources will not have to be blindly distributed.

11. Code:

Around 450 lines of Python code were written from scratch. The csv library was heavily used in the project, since all datasets were stored in csv files. The matplotlib library was used to plot graphs.

References:

Bell, A 2015, *Homeless rely on smartphones to survive but finding somewhere to re-charge is a challenge*, viewed 22nd May 2016,
< http://www.dailytelegraph.com.au/newslocal/news/homeless-rely-on-smartphones-to-survive-but-finding-somewhere-to-recharge-is-a-challenge/news-story/3cdbac970d1d78250a0c9a25b418d166 >

Spinks, R 2015, *Smartphones are a lifeline for homeless people*, viewed 22nd May 2016,
< http://www.theguardian.com/sustainable-business/2015/oct/01/smartphones-are-lifeline-for-homeless-people >