LA Neighborhood Score Midterm Report

Jameson Thai, Matheus Schmitz, Yeonsoo Park

Los Angeles is home to nearly 4 million residents with an abundance of recorded and freely available data covering multiple areas, all maintained by the City of Los Angeles. Exploring the different datasets and the data derived from them, this project aims to combine and assess different aspects about the quality of life in Los Angeles. We leverage this data to provide a comprehensive neighborhood infographic for Los Angeles residents. Additionally, this project can also function as a reference model for other neighborhood analysis of other cities across the United States. This team consists of three students, Matheus Schmitz with a background in Industrial Engineering and Product Management, along with Jameson Thai and Emma Park, each with a background in Computer Science specializing in Data Science.

Tackling this project, we have sourced official domains from verified authorities. These primarily come from the following websites: data.lacity.org, data.lacounty.gov, Geohub, cde, Zillow, USC's Socrata, and US Census Bureau. The LA City data contains a diverse range of information for the city of LA, however for this project we will primarily focus on arrest, crime, education, and HDI datasets. Geohub will be the main source for LA School information that we base our neighborhood analysis upon. Finally USC's Socrata and Zillow provides us data on housing prices throughout the USC area as well as LA. As per the Project Proposal, we have reduced our dataset to use these domains as the primary data sources.

Task	Person	Deadline	Status
Proposal slides	Emma	09/12	OK
Select datasets	Matheus	09/09	OK
Clean crime dataset	Matheus	10/12	OK
Clean school dataset	Emma	10/12	OK
Clean housing dataset	Jameson	10/12	OK
Upload data to MySQL	Jameson	10/16	OK
Write midterm report	Jameson	10/14	In Progress
MapReduce data to aggregate by neighborhood	Matheus	10/18	In Progress
Load data to Firebase	Matheus	10/18	In Progress
Create search function	ALL	11/16	Waiting
Create web view	Emma	11/16	Waiting

Regarding the project's progress, we have been actively tackling tasks to come closer to achieving this goal. As in the image above we delegated tasks according to three quality of life features. We have finished cleaning the three separate datasets from various sources into a standardized format. These three datasets are aggregated as CSV's to be stored on a MySQL database.

	A	B	C	D	E	F	G	H
1	DR_NO	DATE OCC	Crime_Code	LAT	LON	Neighborhood	Crime_Weighted	Crime_Weighted_No
2	191907191	2019-03-08	510	34.2991	-118.4211	Sylmar	0.1960784314	0.1137069485
3	190125334	2019-10-17	330	34.0363	-118.2314	Downtown Los Angel	0.303030303	0.2466509062
4	191920961	2019-12-22	510	34.2283	-118.4611	North Hills East	0.1960784314	0.1137069485
5	190604395	2019-01-09	510	34.0944	-118.318	Central LA	0.1960784314	0.1137069485
6	191310615	2019-04-30	510	33.993	-118.2411	South Los Angeles	0.1960784314	0.1137069485
7	191419522	2019-09-03	510	33.9759	-118.3897	Westchester	0.1960784314	0.1137069485
8	190123550	2019-09-21	330	34.0467	-118.2411	Downtown Los Angel	0.303030303	0.2466509062
9	191915118	2019-08-18	510	34.2985	-118.4205	Sylmar	0.1960784314	0.1137069485

The task of cleaning the datasets were equally divided between the team members. The majority of the datasets had an abundance of features and categories however, many of the datasets chosen included a longitude and latitude information where we fed into a Google Maps API to cluster the data into their appropriate neighborhoods. This was primarily done for the crime and school dataset as the housing dataset already had that information included. Using these groupings, we produced an average of a feature's score to be saved into a CSV that is later read by a Python program into a MySQL database.

id	neighborhood	size_rank	housing_score
1	South Los Angeles	4	442013.573529411750000
2	Southeast Los Angeles	8	367733.691176470600000
3	Hollywood	18	847758.705882352900000
4	Mid City	62	986030.014705882400000
5	Van Nuys	64	535259.676470588200000
6	Sun Valley	68	513386.485294117650000
7	Northridge	69	670616.470588235300000
8	Sylmar	74	483452.161764705900000
9	Boyle Heights	75	424736.161764705900000
10	North Hollywood	78	613276.544117647100000
11	San Pedro	79	583256.838235294100000
12	Koreatown	82	629124.544117647100000
13	Woodland Hills	83	780429.264705882400000
14	Sherman Oaks	85	951315.764705882400000
15	Pacoima	86	434265.852941176450000

The pipeline for getting the crime data to MySQL consisted of the following steps: First the data was downloaded as a CSV from the https://data.lacity.org page. The whole dataset contained 2 million records, covering years 2010-2019, using all of which

proved too burdensome, hence only records for the year of 2019 were kept, totalling about 200k records. Those records already contained latitude and longitude, hence the second step consisted of using the google maps API reverse geocode and obtaining the neighborhood in which the crime happened. Unfortunately the API has limitations, which stopped the queries after around 115k had been sent. This forced the team to resort to using a KNN algorithm to predict the neighborhoods for the rest of the data, which was step three. A train-test split showed this approach to be highly effective, as neighborhoods are by definition linearly separable points in a low (two) dimensional space, about as ideal a setting for a KNN algorithm as one could help for, resulting in a 98% accuracy. Step four was the creation of a feature called “Crime_Weighted_Norm”, which was obtained by using the “Crime Code” associated with each register. “Crime Code”indicates the crime committed, with lower crime class numbers representing more serious offenses. Thus “Crime_Weighted_Norm” was obtained by first dividing 100 (the lowest/most serious crime code) by the Crime Code of each data point, and then normalizing, resulting in a feature in the [0,1] range, with higher values representing mor e serious crimes. Other columns on the original dataset contained essentially the full police log on the crime, but no other relevant information for this project and were discarded. Lastly, step five was storing the final dataset into MySQL.

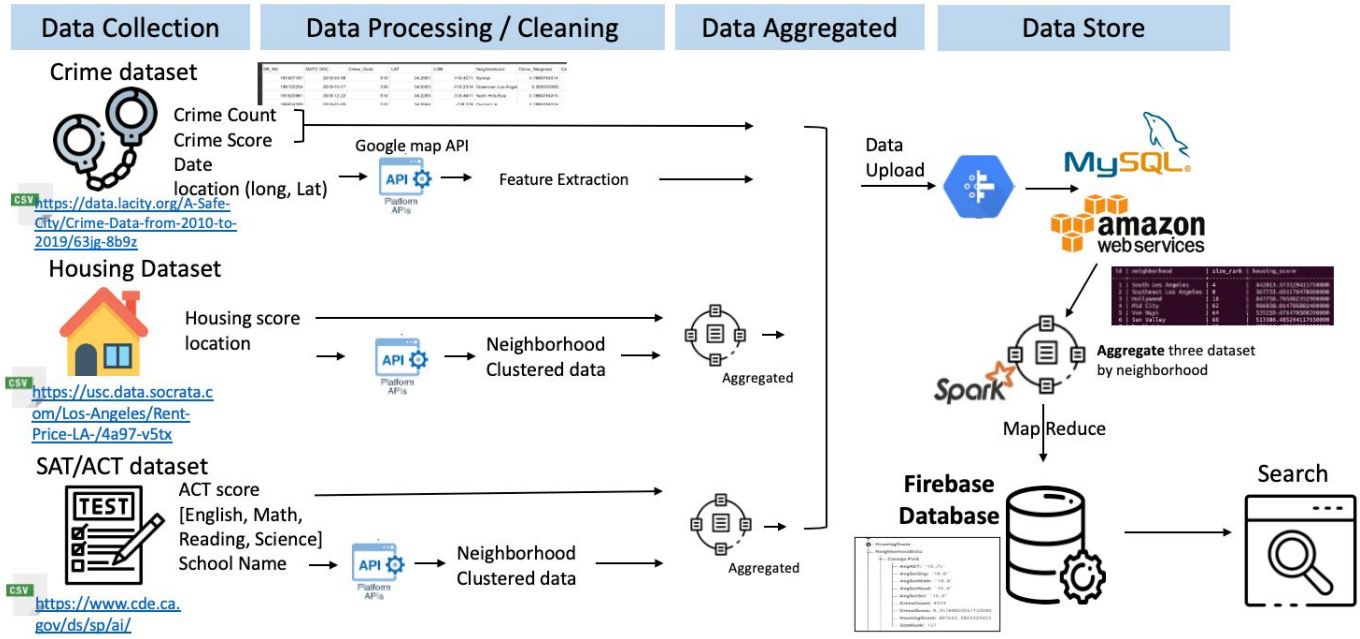
Neighborhoods can be extracted from the googlemaps python API using the reverse_geocode() method, which takes a tuple of (latitude, longitude) and extract all information related to the point in the globe. This information includes country, state, county, city, neighborhood and well as additional information for certain locations. As the data varies from one place to another, so does the structure of the output, and hence it’s necessary to explore all elements in it until the one containing neighborhood data is found. Then the neighborhood name can be extracted and appended to the register from whose (latitude, longitude) it originated. Below is an excerpt of the code used to obtain the neighborhoods using the google maps API:

```
In [*]: # Loop through each row in the crime dataframe
for idx, row in tqdm(df.iterrows(), total=len(df)):

    # Make a temporary dict (json) with data extracted with google maps' reverse_geocode
    tmp = gmaps.reverse_geocode((row.LAT, row.LON))

    # Take the length at [0]['address_components'] and create a range of attempts to find the neighborhood
    # [0]['address_components'] contains all address elements, eg: county, city, neighborhood
    # Each in one position in a list whose positions are denoted by the values in attempt_range
    try:
        attempt_range = range(len(tmp[0]['address_components']))

    # Some empty data points (Lat and long = 0) will have len(tmp[0]) == 0, in those cases just skip the data point
    except:
        continue

    # Loop through each possible position where the neighborhood data can be
    for attempt in attempt_range:

        # For each possible neighborhood location, check [0]['address_components'][attempt_range]['types']
        # To see if type == neighborhood
        if 'neighborhood' in tmp[0]['address_components'][attempt]['types']:

            # If yes, then extract the neighborhood name, putting it in a new column in the dataframe
            df.at[idx, 'neighborhood'] = tmp[0]['address_components'][attempt]['long_name']

            # And go to next iteration (no need to check the other positions in the tmp, as the answer has been found)
            continue

        # Else continue the loop until neighborhood is found
        else:
            continue

    # After each 20000 rows, save the dataframe as csv
    if idx%20000 == 0:
        df.to_csv('Crime_Data_2019_Neighborhoods.csv')

  0%|          | 356/216102 [01:12<10:43:24, 5.59it/s]
```

The MySQL database contains filtered data on an AWS EC2 instance. As seen in the image above, there is a primary key being ID and a secondary key set up as the neighborhood names with the neighborhood score and other extraneous information as table attributes shown in the image above. The extraneous information varied by dataset due to the what types of information that could be extrapolated from.

Architecture

Diagram



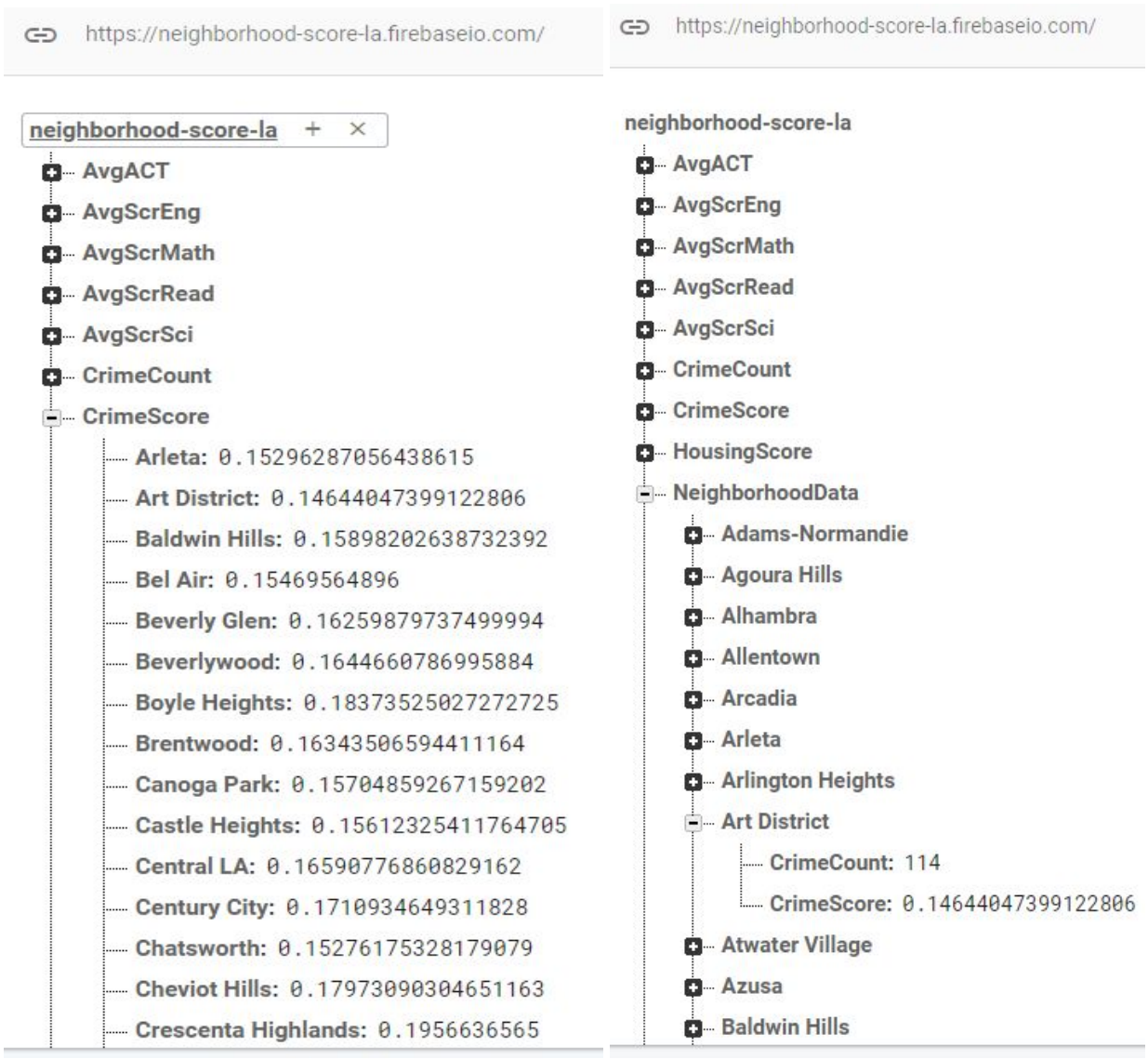
The architecture diagram above describes the pipeline process of our project. Our dataset is first cleaned by a python program that is specialized for each dataset source. For the SAT and Crime datasets, we do some initial clustering to form the neighborhood groups. The housing dataset on the other-hand already has the neighborhoods and just needs to extract the data regarding the City of Los Angeles. From there we aggregate where applicable and upload to a MySQL database containing all the filtered information. We also apply a MapReduce program to the MySQL database to aggregate and summarize the data into one neighborhood to be stored on Firebase.

In order to diversify the number of Spark use cases in the project, with the goal of enchanting our learning, we opted to have the crime data be saved on MySQL as individual data points (200k data points), instead of storing data already aggregated by neighborhood. This allows for Spark to be used to MapReduce the crime data and aggregate it by neighborhood in a more scalable fashion (and for extra learning). The Spark MapReduce pipeline took the “Crime_Weighted_Norm” feature that had already been extracted from the data prior to upload into MySQL, and averaged it by neighborhood, while also obtaining a second feature: the count of crimes per neighborhood.

Once the pipeline for the crime dataset was finished, the two other datasets, which had already been cleaned prior to being stored in MySQL, were loaded into Spark. By this point all datasets were already aggregated by neighborhood, and so they could be merged using Spark SQL. After the aggregation pipeline grouped all datasets together

with the join function, the resulting tabular dataset was then converted into JSON/dictionary format, so that the data were suitable for uploading to Firebase.

In order to ease querying into the data, the Firebase architecture created was inspired on the homework 1 architecture, with the feature of each neighborhood being stored twice: One time following the index of *{Feature: {Neighborhood_1: Value, Neighborhood_2, Value, …}}* and a second time using the Neighborhood as the main index *{NeighborhoodData: {Neighborhood_1: {Feature_1, Feature_2, …}, Neighborhood_2: {Feature_1, Feature_2, …}...}}*. The indexings can be seen in the figures below:



In summary, our team is on track to deploying an interactive web infographic. Many of the milestones are completed or very soon to be finished. Some challenges with the remaining tasks involve integrating the search functionality as well as visualizing the data and making the infographic interactive. Additionally we encountered a problem where there is a mismatch between the neighborhood datas. This is where some neighborhoods only have a score from one dataset that is not present in the others. Additionally there are some neighborhoods that are not within the city of Los Angeles. These outliers are checked on a case to case basis and are likely caused by how the Zillow dataset for housing may define the neighborhoods differently than the Geolocations fed into the Google Maps API to define the neighborhoods.

Below is a link of data sources we have extrapolated our data from.

Housing

https://www.zillow.com/research/data/

https://usc.data.socrata.com/Los-Angeles/Rent-Price-LA-/4a97-v5tx

https://data.census.gov

School

https://www.cde.ca.gov/ds/sp/ai/

https://geohub.lacity.org/datasets/70baf6da243e40298ba9246e9a67409b_0/data

Crime

https://data.lacity.org/A-Safe-City/Crime-Data-from-2010-to-2019/63jg-8b9z

Our Firebase dataset can be found at:

https://neighborhood-score-la.firebaseio.com/.json