Using Space to Define Opportunity

...

James Ades, Daniel Hartley, Bhanu Muvva, Carlos Monsivais

Data Sources

1. Census (Structure)
2. San Diego Police Department (Structured)
3. Reddit Data (Unstructured)
4. Google Geocode API (Unstructured)
5. Housing Data (Unstructured)
6. GeoJson Mapping Polygons

Project Technologies

- Google Cloud
  - Dataproc Clusters (Run Jupyter Notebooks)
  - Google Geocode API
  - Google Cloud Storage Buckets (Store Data)
- Neo4J Graph Database
- Python
- PostgreSQL

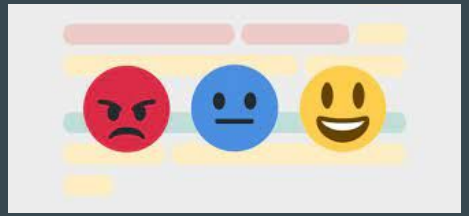
Reddit Data Scraping

- Extracted unstructured data from the San Diego subreddit.
  - Posts between January 1, 2020 and January 1, 2021
- The following fields were scraped:
  - Score, author, full link, date, number of comments, title, subreddit location, number of subscribers, title, image url, upvote ratio.
- Ended up scraping 18,709 posts and converting them from unstructured data coming back in JSON format to a structured format by putting into a dataframe and then storing it as a csv.
- Used a brute force method to keep trying to scrape a page until data is retrieved due to a request limit.

Reddit Sentiment Analysis and Text Similarity

- We wanted to see how people feel about their neighborhood, by using a sentiment score in their posts.
- Used Reddit title to calculate a sentiment analysis to see if the overall post was either positive or negative.
- Extracted neighborhood text from the post title where we are looking for a list of 120 different neighborhoods in San Diego.
- Used Spacy's phrasematcher NLP pipeline to match our list of neighborhoods to anything that is similar in the title of every post to extract what neighborhood people are talking .
    - Removed non alphabetical symbols
    - Removed stopwords
    - Used Spacy Token Patterns
    - Create NLP Pipeline to extract entities from text.
- Afterwards linking the neighborhood to the sentiment analysis to get a better sense of whether that neighborhood has a positive or negative associations.

spaCy

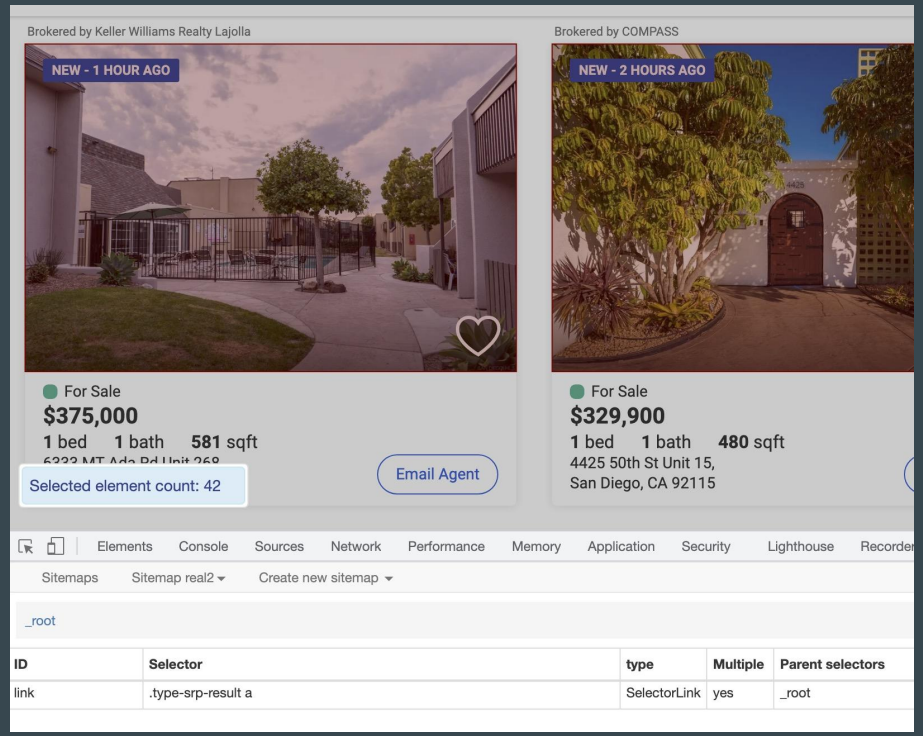


Crime Data Longitude Latitude

- Extracted data from the San Diego Police Department and read in 5 different files to join them including:
  - Stops, race, reason for stop, result, complaints
- Removed some low level crimes such as jaywalking, traffic violations, and dog unrestrained since these crimes are very minor.
- Combined the different address fields.
- Used the Google Geocode API to match the address fields to a longitude and latitude coordinates dataset.
- Using these coordinates we are able to map crimes on our shapefile polygons to know visually where crimes are happening in San Diego.

Scraping Housing Data

- Used chrome plugin webscraper.io for scraping realtor.com
- Works in browser, streamlines web scraping
- Set up to select each link on a page
- Scrapped the first 50 pages of listings for San Diego ~2000 houses
- Tags the information within each link to be scrapped
    - Price
    - Address
    - Number of beds
    - Number of baths
    - Area property
- Exports data as a csv file



Geocoding and Cleaning Housing Data

- The Googlemaps python library was used to match the housing address to lat/long coordinates
- Given the address scrapped from the webpage results are returned in json format
- Latitude and longitude values pulled from the geometry field of the json data
- These values were then added to the dataframe

- Majority of the housing data was scraped as strings
- Substrings like “bed” and “bath” needed to be removed
- Removed special characters like $ and + in order to change column data types
- Dropped unused columns

Neo4j Spatial Values and functions

- Neo4j supports only one type of spatial geometry, the *Point* with the following characteristics
  - Each point can have either 2 or 3 dimensions.This means it contains either 2 or 3 64-bit floating point values, which together are called the *Coordinate*.
  - Each point will also be associated with a specific Coordinate Reference System(CRS) that determines the meaning of the values in the *Coordinate.*
- Spatial functions
  - *Point.distance* returns a floating point number representing the geodesic distance between two points in same CRS
  - *point.withinBBox()* return value will be true if the provided point is contained in the bounding box, otherwise the return value will be false
  - point() returns a 2D point in the given CRS.
- Example Cypher Query:

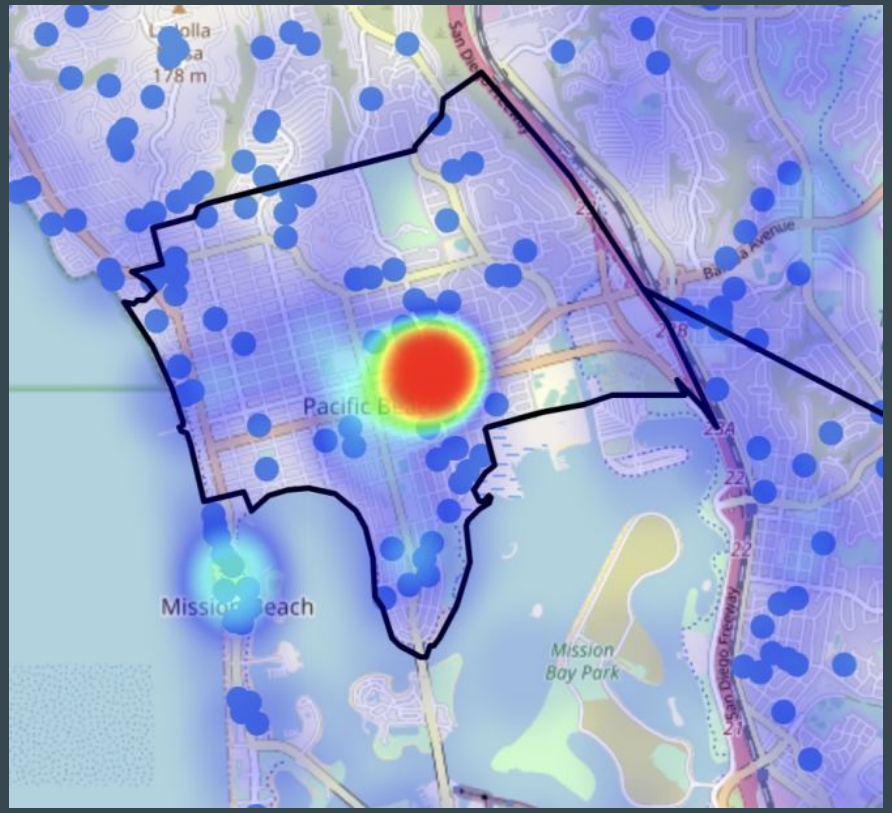
Cypher Query Copy to Clipboard Run in Neo4j Browser

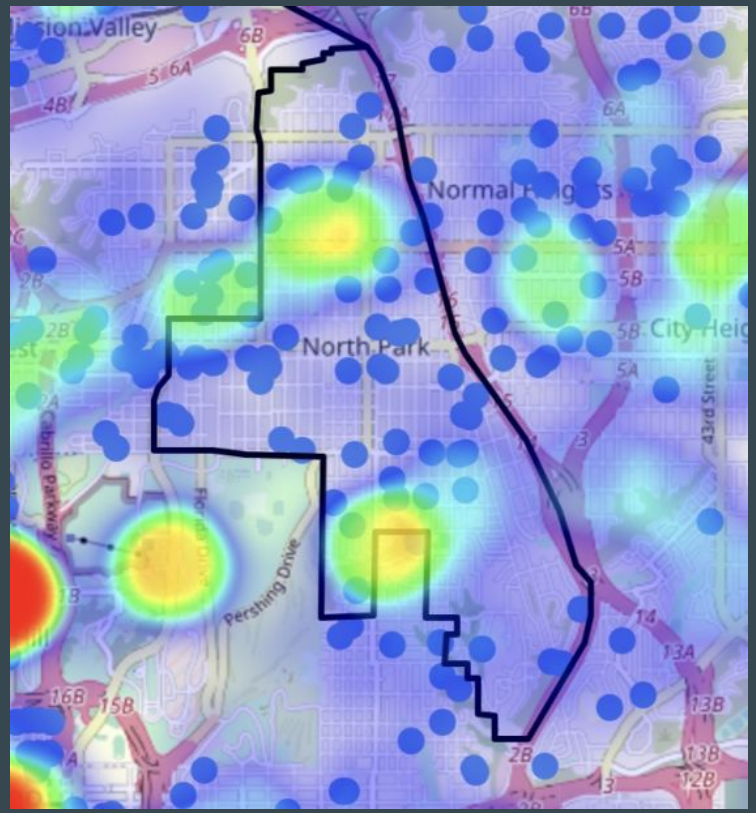
```
WITH
  point({latitude:toFloat('13.43'), longitude:toFloat('56.21')}) AS p1,
  point({latitude:toFloat('13.10'), longitude:toFloat('56.41')}) AS p2
RETURN toInteger(point.distance(p1, p2)/1000) AS km
```

Loading data neo4j and neomaps

- LOAD CSV from URLs
  - Cloud hosted versions of Neo4j can only access remote http(s) URLs.
  - Examples -
    - GitHub,Google Drive, Dropbox, Cloud provider Storage like Amazon AWS S3
- Node types and counts
  - ~2500 house nodes
  - ~25k crime incident nodes
  - ~120 neighbourhood nodes (with boundary information)
- **Neomap** : A Neo4j Desktop (React-based) application to visualize nodes with geographical attributes on a map.
  - We can visualize spatial data or nodes in the form of pins/markers or heatmaps
  - Layer type:
    - Simple : Define a node label and the properties holding the latitude and longitude attributes along with tooltip.
    - Advanced: Fetch the nodes via a Cypher query for extended configuration (for example using the WHERE clause)

Output with house,crime and neighbourhood layers in neomap





Future Research

- Assuming we had money for a more powerful Neo4J Server and more time:
  - Find better crime data instead of data that included instances
  - Run queries for all our data, we had speed issues due to scale of our data compared to the machine power.
  - Add homeless data to our analysis.
  - And look at other algorithms to get a better sense of how these features directly impact home prices.