Food Desert Detection

Sean Kunz

Problem

Def: A **food desert** is an area that has limited access to affordable and nutritious food

There does not currently exist a technical-based method of identifying a food desert. For example, the USDA currently classifies the Binghamton University campus as a food desert.

Solution: Create a database of grocery stores within a city and perform spatial/locational analysis to identify which neighborhoods lack access to healthy foods

Tools Used

Language: Python
NoSQL Database: MongoDB
Python Libraries:

- Tkinter (GUI)
- Python Image Library (GUI)
- Google Places API (ETL Pipeline)
- Google Geocoding API (ETL Pipeline)
- Pymongo (ETL Pipeline/GUI)

Implementation - Extract, Transform, Load

Extract: Data has to be gathered to generate the database. To accomplish this, I wrote a script that uses the Google Places and Geocoding APIs and dumps data into a MongoDB database. To fully capture all of the stores, coordinates are selected so that a search for stores can be performed at each latitude/longitude coordinate pair, ensuring that all stores are captured. Store information is then returned as JSON data.

Transform: As data is being returned from the API calls, it is being cleaned and modified so that only the necessary data will be inserted into the database.

Implementation - Extract, Transform, Load

Load: After the data is cleaned, it needs to be inserted into the database.

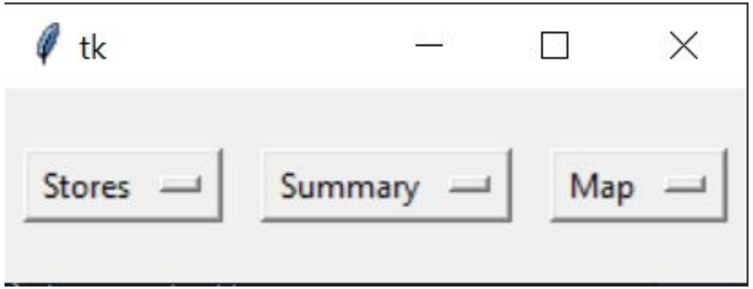
The query that actually inserted the data is as follows:

```
db.storeinfo.update({'_id':p['ID']}, store, upsert=True)
```

While appearing as an update query, this query actually functions as both an insert and an update, or an ‘Upsert’ as the flag indicates. The query checks if the ID is already in the database. If it is not in the database, then the store information will be inserted. Otherwise, it will be updated. This ensures that duplicates will not be inserted into the database.

Implementation - GUI

The GUI was implemented using Tkinter. The GUI presents the user with three options: select stores, select a city summary, or select a map.



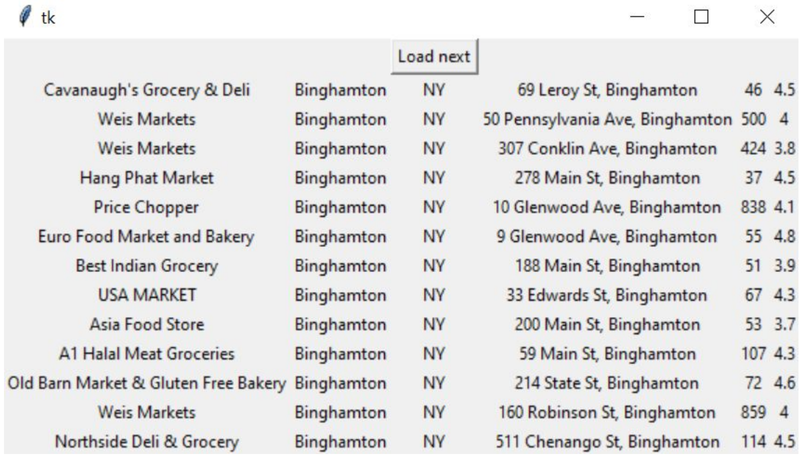
The GUI uses the equivalent of a ‘select distinct’ query to generate the options of the drop down menus. This way, the drop down menus don’t have to be hardcoded.

Implementation - GUI

The stores menu utilizes queries that select relevant information about the stores for a given city, or for all stores in the database. Twenty tuples are loaded for each page, but users can load more with the ‘Load Next’ button.

Query example: cursor = self.db.storeinfo.find({"City": value}, {"_id": 0})

Results:



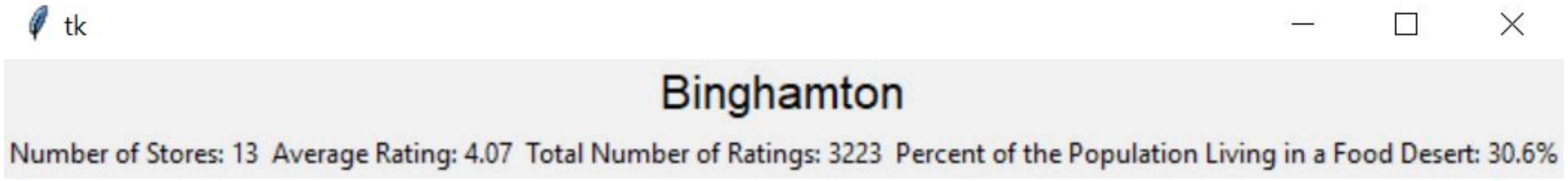
Cavanaugh's Grocery & Deli	Binghamton	NY	69 Leroy St, Binghamton	46	4.5
Weis Markets	Binghamton	NY	50 Pennsylvania Ave, Binghamton	500	4
Weis Markets	Binghamton	NY	307 Conklin Ave, Binghamton	424	3.8
Hang Phat Market	Binghamton	NY	278 Main St, Binghamton	37	4.5
Price Chopper	Binghamton	NY	10 Glenwood Ave, Binghamton	838	4.1
Euro Food Market and Bakery	Binghamton	NY	9 Glenwood Ave, Binghamton	55	4.8
Best Indian Grocery	Binghamton	NY	188 Main St, Binghamton	51	3.9
USA MARKET	Binghamton	NY	33 Edwards St, Binghamton	67	4.3
Asia Food Store	Binghamton	NY	200 Main St, Binghamton	53	3.7
A1 Halal Meat Groceries	Binghamton	NY	59 Main St, Binghamton	107	4.3
Old Barn Market & Gluten Free Bakery	Binghamton	NY	214 State St, Binghamton	72	4.6
Weis Markets	Binghamton	NY	160 Robinson St, Binghamton	859	4
Northside Deli & Grocery	Binghamton	NY	511 Chenango St, Binghamton	114	4.5

Implementation - GUI

The summary menu utilizes both the stores database and the maps database. Similarly to the stores option, information is selected from the collections to output data to the screen.

Query example: cursor = self.db.mapinfo.find({"City": value}, {"_id": 0})

Result:



Binghamton

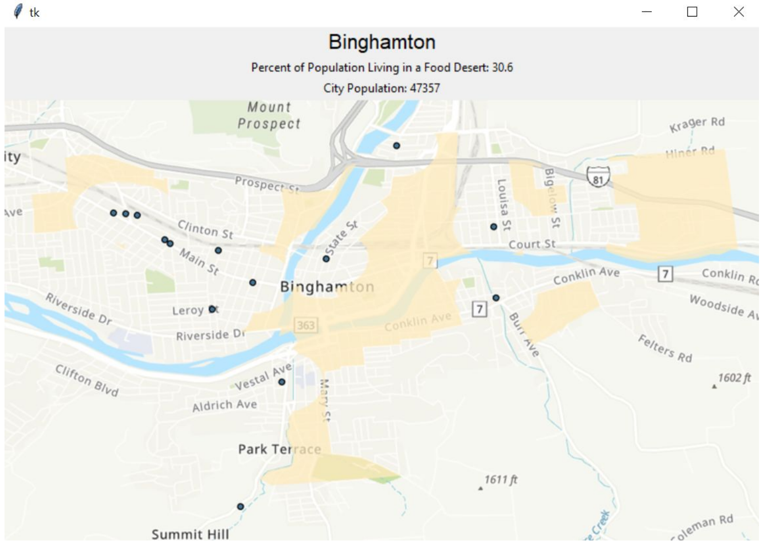
```
Number of Stores: 13  Average Rating: 4.07  Total Number of Ratings: 3223  Percent of the Population Living in a Food Desert: 30.6%
```

Implementation - GUI

The map menu loads a map from the map database (or rather the file path is stored in the database and fetched using the PIL)

Query example: cursor = self.db.mapinfo.find({"City": value}, {"_id": 0})

Result:



Binghamton

Percent of Population Living in a Food Desert: 30.6

City Population: 47357