Meal Mvrs

Food security, one click closer

Problem

Restaurants, often driven by factors like portion control and quality standards, frequently discard significant amounts of edible food, contributing to food waste. Simultaneously, countless people, including those experiencing homelessness and food insecurity, struggle to secure their next meal. This disparity highlights a critical need for solutions that can bridge this gap, such as initiatives that redirect surplus restaurant food to local food banks or connect restaurants with those in need, ultimately combating both food waste and hunger in our communities.

Solution

A portal website that connects consumers to food banks based on certain criterias

Facilitated and efficient process for restaurants to donate excess food to local food banks, targeting areas of high food insecurity

Step 1

- Accumulated the data of every food shelter in a 5 mile radius in Boston
- Created code that would be able to direct an individual to the nearest food shelter by converting every food shelter’s address into coordinates of latitude and longitude
- In addition, we did the same for the live location of the consumer.
- Coded a euclidean distance formula to calculate and return the lowest distance (the closest shelter)

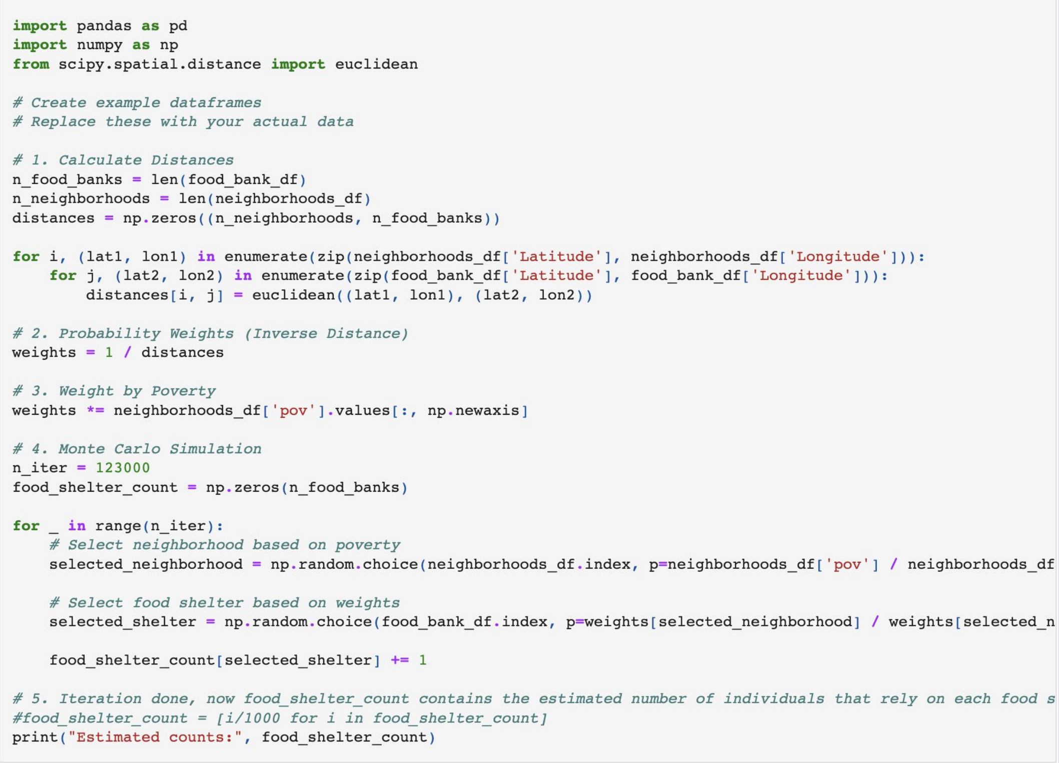
Step 2

- Primarily about connecting restaurants to food shelters
- Used poverty statistics with the assumption that those that would be more likely to actually need food shelters would be in more poverty stricken areas
- Found these by a data classification of the various neighborhoods of Boston: North End, South Boston, Mission Hill, etc.

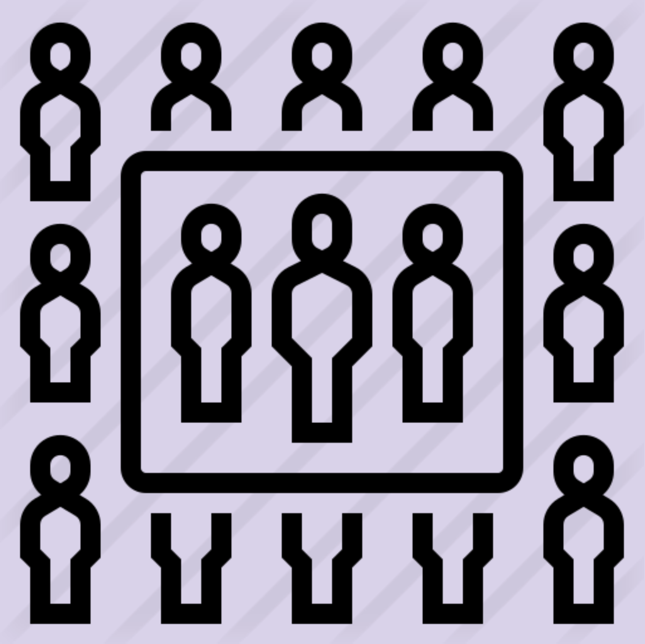
Step 2 (continued)

- We used a Monte Carlo analysis to estimate the load of individuals that would need food shelters across every neighborhood, so that one or two shelters would not be overrun by people
- Through this, we were able to also make our web app capable of telling restaurants where their excess food items and resources would be best reallocated

Monte Carlo Algorithm



```
import pandas as pd
import numpy as np
from scipy.spatial.distance import euclidean

# Create example dataframes
# Replace these with your actual data

# 1. Calculate Distances
n_food_banks = len(food_bank_df)
n_neighborhoods = len(neighborhoods_df)
distances = np.zeros((n_neighborhoods, n_food_banks))

for i, (lat1, lon1) in enumerate(zip(neighborhoods_df['Latitude'], neighborhoods_df['Longitude'])):
    for j, (lat2, lon2) in enumerate(zip(food_bank_df['Latitude'], food_bank_df['Longitude'])):
        distances[i, j] = euclidean((lat1, lon1), (lat2, lon2))

# 2. Probability Weights (Inverse Distance)
weights = 1 / distances

# 3. Weight by Poverty
weights *= neighborhoods_df['pov'].values[:, np.newaxis]

# 4. Monte Carlo Simulation
n_iter = 123000
food_shelter_count = np.zeros(n_food_banks)

for _ in range(n_iter):
    # Select neighborhood based on poverty
    selected_neighborhood = np.random.choice(neighborhoods_df.index, p=neighborhoods_df['pov'] / neighborhoods_df

    # Select food shelter based on weights
    selected_shelter = np.random.choice(food_bank_df.index, p=weights[selected_neighborhood] / weights[selected_n

    food_shelter_count[selected_shelter] += 1

# 5. Iteration done, now food_shelter_count contains the estimated number of individuals that rely on each food s
#food_shelter_count = [i/1000 for i in food_shelter_count]
print("Estimated counts:", food_shelter_count)
```



Future Goals

Creation of a mobile app with improved UI/UX (similar to delivery apps)

Accommodation to customers’ dietary needs

Widespread use of the app to allow for more restaurants and users to connect with various food banks