Daniels S. | Data Science Student 05/22/2025

Lab Project.

Proximity and Public Access:
A Geospatial Study of CUNY Campuses and NYC Benefit Services.

---

1. Problem Statement / Hypothesis

In a city as large and complex as New York, access to public benefits is essential — especially for students, working-class individuals, and underserved communities. CUNY campuses are distributed across the five boroughs and serve tens of thousands of students, many of whom rely on public services such as food assistance, housing, or healthcare.

This project investigates whether geographic proximity to CUNY campuses correlates with access to public benefits. By combining datasets on CUNY campus locations and NYC Benefits Access Centers, we aim to answer:

> **Do public benefits centers tend to be located near CUNY campuses, and how accessible are they?**

To enhance our exploration, we also built a simple machine learning model to predict the borough of a Benefits Access Center based on its spatial characteristics. This demonstrates the predictive power of geographic features in urban data applications.

---

2. Requirements Document

Background on the Problem

New York City hosts a wide network of public benefits and social service centers. At the same time, the City University of New York (CUNY) serves hundreds of thousands of students across the five boroughs, many of whom rely on these public services. However, accessibility isn’t just about eligibility, it's about geography. This project explores whether these services are physically accessible to CUNY students by examining their proximity to campus locations.

Objective of the Solution

The goal of this project is to:

- Analyze the geographic relationship between CUNY campuses and NYC Benefits Access Centers
- Visualize and quantify the distance between them
- Use spatial data to predict borough classifications of centers via a bonus machine learning model
- Demonstrate core programming and data management concepts.

Expected Use Case(s)

- **Urban planning**: Help city officials identify underserved campus areas
- **Student outreach**: Enable CUNY administrators to target benefits awareness campaigns
- **Data validation**: Use ML model to infer boroughs when administrative data is incomplete
- **Educational**: Showcase skills in data mashups, preprocessing, analysis, and machine learning

Available Data, Technology, and Other Resources

- Open data:
    - NYC Benefits Access Centers (CSV) (Link in Section 4)
    - CUNY Campus Locations (CSV) (Link in Section 4)
- Technologies:
    - Python, Pandas, NumPy
    - Folium, Seaborn, Matplotlib
    - Geopy (for geodistance)

  - PyTorch (for ML model)
  - Google Colab for development and documentation

Out-of-Scope or Future Additions

- We do not address the operating hours, program availability, or quality of services at centers
- We do not model student enrollment volume or transportation options
- We exclude non-borough-specific centers such as “Family Services” or “Special Project” locations

Measures of Success / Test Criteria

- Distance data is correctly computed and aligned between both datasets
- Visualizations clearly communicate proximity patterns
- Bonus ML model achieves reasonable accuracy **(>70%)** in borough classification
- Report demonstrates at least 4 technical concepts from the course
- All project outputs (graphs, HTML map, model files) are saved, formatted, and reproducible

---

3. Business Understanding

Define Objectives

The primary objective is to analyze how well NYC public service infrastructure aligns geographically with the needs of CUNY students. Specifically:

- Determine whether Benefits Access Centers are evenly distributed near CUNY campuses

- Quantify distances between services and educational institutions
- Explore whether spatial data can predict borough classification

The secondary objective is to build a small machine learning model to illustrate the potential of using geographic features to drive classification decisions, a foundation for smarter, location-aware public service planning.

Category Classification

This project combines elements of:

- **Geospatial Analysis**: Examining latitude/longitude relationships and physical accessibility
- **Urban Informatics**: Understanding city infrastructure through open data
- **Machine Learning**: Supervised classification of boroughs using structured input features.

Model Requirements

- The model must accept 3 input features: `latitude`, `longitude`, and `distance to nearest CUNY campus`
- It must output one of 5 borough categories
- Model should be lightweight (1 hidden layer) and trainable on small data
- Achieve a baseline test accuracy of at least **70%**

Feasibility Report

- The required datasets are freely available and structured
- Python libraries such as `geopy`, `folium`, `pandas`, and `torch` make the analysis and modeling technically feasible within a single notebook
- Model simplicity allows it to run in under 10 seconds per training session no GPU needed
- The geographic dataset is small, so computational overhead is minimal
- Visualization tools support interactive and static reporting, ideal for both instructors and city stakeholders

---

Recommendations

Based on this business context, the following steps are recommended:

- Continue developing map-based analytics for public service planning
- Integrate socioeconomic or demographic layers (e.g., median income, access to transit)
- Expand the machine learning model to include transportation features (e.g., walking time, subway proximity)
- Deploy the trained model into a simple API that takes in coordinates and suggests the most probable borough

---

4. Data Mining

Data Sources

This project uses two open-source datasets:

1. **City University of New York (CUNY) Campus Locations**

    - Source: An official website of the GSA, Link: [data.gov]()
    - Format: CSV
    - Contains: campus name, address, borough, latitude, longitude
    - Records: 26 campuses

2. **NYC Benefits Access Centers**
    - Source: NYC Open Data, Link: [Open-Data-NYC]()
    - Format: CSV
    - Contains: facility name, address, borough, latitude, longitude, service area
    - Records: 29 locations (after filtering out non-borough services)

Data Capture

- Both datasets were manually downloaded in `.csv` format
- Uploaded to Google Colab using the `google.colab.files` interface
- Loaded into memory using `pandas.read_csv()`
- No real-time API access was used

Data Storage

- All intermediate results (merged DataFrames, feature-engineered versions, visualizations) were saved locally:
    - CSV file: `cuny_nearest_access_centers.csv`

  - Map file: `nyc_services_map.html`
  - Plot files: `.png` format
- The dataset is small enough to be processed entirely in memory without requiring a database or cloud storage system

---

5. Data Cleaning

Handling Missing Values

Both datasets were checked for missing data using `.isnull().sum()`. The CUNY campus dataset had no missing values in any relevant columns. The Benefits Access Centers dataset had one missing entry each in the `"Phone Number(s)"` and `"Comments"` columns both of which were not essential to the analysis and were later removed entirely.

Dropping Irrelevant Columns

To simplify the Benefits Access Centers dataset, several administrative or metadata columns were dropped because they were not relevant to the core analysis. These included:

- `Phone Number(s)`
- `Comments`
- `Community Board`
- `Council District`
- `Census Tract`
- `BIN`
- `BBL`
- `NTA`

The column `"Facility Name"` was renamed to `"Access Center Name"` for clarity. Similar column renaming was applied to the CUNY dataset to standardize `"Campus"` to `"CUNY Campus"` and `"Address"` to `"Campus Address"`.

Filtering for Valid Boroughs

An error during model training revealed that the `"Borough"` column in the Access Centers dataset contained 11 distinct categories — more than the expected 5 boroughs of NYC. Upon review, these extra entries were administrative groupings or special services, such as `"Family Services Call Centers"` or `"Transportation Unit"`, which are not geographic boroughs.

To fix this:

- The dataset was filtered to include only the five valid NYC boroughs: `"Bronx"`, `"Brooklyn"`, `"Manhattan"`, `"Queens"`, and `"Staten Island"`
- The filtered dataset was then re-encoded for model training using `LabelEncoder`

Final Structure

After cleaning, both datasets included only the relevant columns:

- CUNY Dataset: campus name, latitude, longitude, city, and zip code
- Access Centers Dataset: center name, latitude, longitude, and borough (cleaned)

This ensured consistency in location-based calculations, model inputs, and all visualizations used in later sections.

---

6. Data Exploitation / Analysis

Distance Hypothesis

This project began with a core spatial hypothesis:

> "CUNY campuses are generally located near NYC Benefits Access Centers, ensuring equitable physical access to services for students."

To test this, the cleaned and filtered datasets were analyzed to measure the straight-line (geodesic) distance between each CUNY campus and the nearest Benefits Access Center.

Method of Analysis

Using the `geopy` library and the geodesic distance function, the latitude and longitude of each campus were compared to every Access Center. A loop calculated the distances, and the minimum distance was recorded for each campus.

The result was a new DataFrame with the following columns:

- `CUNY Campus`
- `Campus Latitude`, `Campus Longitude`
- `Nearest Access Center`
- `Access Center Latitude`, `Access Center Longitude`
- `Distance (km)`

This created a structured dataset that enabled further visualization and hypothesis testing.

Findings

The analysis confirmed that:

- Most CUNY campuses are within **1–2 kilometers** of a Benefits Access Center
- A few campuses, such as Kingsborough Community College, had significantly larger distances (e.g., over 4 km) or over 2.4miles
- These distance variations could inform future decisions about center placement or the provision of mobile or digital services

This dataset was also used as an input for feature engineering and later for predictive modeling.

---

7. Feature Engineering

Feature Selection

For the machine learning portion of the project, a small and interpretable set of features was selected from the Benefits Access Centers dataset. These features were chosen based on their spatial relevance and predictive value for borough classification:

- `Latitude`
- `Longitude`
- `Min Distance to Campus (km)`

These three features formed the complete input vector for each Access Center used in the predictive model.

Feature Construction: Distance to Campus

The feature `"Min Distance to Campus (km)"` was not present in the original dataset and was created using the `geopy` distance function. For each Access Center, the model:

1. Iterated through all CUNY campuses
2. Calculated the distance to each one
3. Retained only the **shortest distance**

This new column was added to the access center dataset and used as a key input feature in the PyTorch model.

Data Scaling

Before training the model, all input features were standardized using `StandardScaler` from `scikit-learn`. This ensured that:

- The large numeric differences between degrees of latitude/longitude and distance (in km) would not skew the model's weights
- The neural network could converge more efficiently during training

---

8. Predictive Modeling

Model Objective

The goal of the predictive model was to classify the borough of a NYC Benefits Access Center based on its spatial characteristics. The model used three input features:

- Latitude
- Longitude
- Distance to the nearest CUNY campus

The model output was one of five borough classes:

- Manhattan, Brooklyn, Bronx, Queens, or Staten Island

This classification task demonstrates that geographic coordinates and proximity to known landmarks (CUNY campuses) can effectively predict broader location categories, which has implications for validation and service classification in urban data.

Model Architecture

The model was implemented in PyTorch as a simple feedforward neural network with:

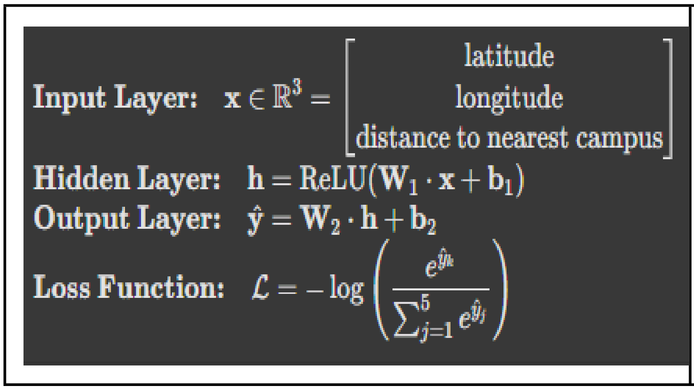
- An input layer of 3 neurons (for the 3 features)
- One hidden layer of 8 neurons with ReLU activation

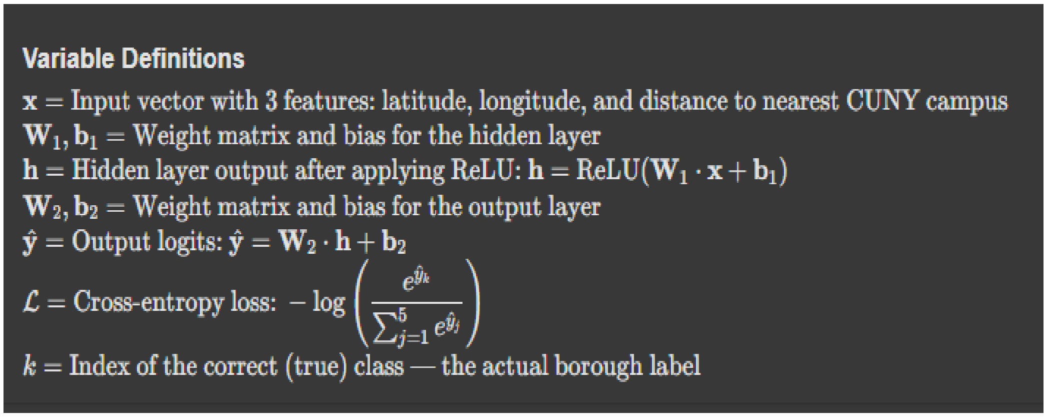
- An output layer of 5 neurons representing the borough classes

The model was trained using CrossEntropyLoss and optimized with the Adam optimizer over 100 epochs.

Mathematical Formulation

The neural network follows this structure:

 **Input Layer:** $\mathbf{x} \in \mathbb{R}^3 = \begin{bmatrix} \text{latitude} \\ \text{longitude} \\ \text{distance to nearest campus} \end{bmatrix}$ **Hidden Layer:** $\mathbf{h} = \text{ReLU}(\mathbf{W}_1 \cdot \mathbf{x} + \mathbf{b}_1)$ **Output Layer:** $\hat{\mathbf{y}} = \mathbf{W}_2 \cdot \mathbf{h} + \mathbf{b}_2$ **Loss Function:** $\mathcal{L} = -\log\left(\frac{e^{\hat{y}_k}}{\sum_{j=1}^{5} e^{\hat{y}_j}}\right)$	Note: This ScreenShot of NN structure was Created&taken from Google Colab. Name of the NN That was used is (FNN)-Feedforward-Neural-Network.

 **Variable Definitions** $\mathbf{x}$ = Input vector with 3 features: latitude, longitude, and distance to nearest CUNY campus $\mathbf{W}_1, \mathbf{b}_1$ = Weight matrix and bias for the hidden layer $\mathbf{h}$ = Hidden layer output after applying ReLU: $\mathbf{h} = \text{ReLU}(\mathbf{W}_1 \cdot \mathbf{x} + \mathbf{b}_1)$ $\mathbf{W}_2, \mathbf{b}_2$ = Weight matrix and bias for the output layer $\hat{\mathbf{y}}$ = Output logits: $\hat{\mathbf{y}} = \mathbf{W}_2 \cdot \mathbf{h} + \mathbf{b}_2$ $\mathcal{L}$ = Cross-entropy loss: $-\log\left(\frac{e^{\hat{y}_k}}{\sum_{j=1}^{5} e^{\hat{y}_j}}\right)$ k = Index of the correct (true) class — the actual borough label	Note:The ScreenShot of Variable Definition Was also Created Taken From Google-Colab.

9. Data Visualization

Interactive Map

The project includes a Folium-based map that visualizes both datasets geographically: The Interactive map could be found in saved documents of the notebook named: nyc_services_map.html Run a notebook and download the file to your pc and open it.

- **CUNY Campuses** were plotted as blue markers
- **Benefits Access Centers** were plotted as red markers

Each marker includes hover tooltips for name and location, allowing stakeholders to visually assess geographic proximity and coverage. The map provides an intuitive overview of how access centers are distributed relative to student populations.

Distance Bar Chart

A horizontal bar chart was created to display the distance between each CUNY campus and its nearest access center. This visualization:

- Makes it easy to identify which campuses are well-served
- Highlights outliers such as Kingsborough Community College, which is significantly farther from the nearest center

Distance Histogram

A histogram was also included to show the distribution of distances across all CUNY campuses. Most campuses fall within a 0–2 km range, with a visible long tail showing that a few campuses are underserved by proximity.

Confusion Matrix

The final visual output was a confusion matrix representing the classification accuracy of the borough prediction model. This helped identify which boroughs the model predicted well and where it confused one borough for another. The confusion matrix was created using scikit-learn and saved as a high-resolution image for reporting.

---

10. Management and Other Considerations

Development Environment

All data processing, analysis, and modeling were completed in Google Colab, a cloud-based Jupyter notebook platform. This environment allowed for:

- Easy file uploads and in-notebook previews
- Interactive development and visualization
- GPU availability (not used in this project but available if needed)
- Seamless integration with Python libraries such as `pandas`, `matplotlib`, `folium`, and `torch`

The notebook was structured with clear markdown headings, code cells, and visual outputs, making it readable as a standalone report.

File Organization

Output files such as CSVs, plots, and map HTML were saved using:

- `to_csv()` for intermediate datasets
- `savefig()` for charts
- `folium_map.save()` for the interactive map

All files were kept in the Colab working directory and downloaded as needed for backup or submission.

Version Control

Since the project was completed in a single collaborative notebook environment, no external version control system (e.g., Git) was used. However, Colab's automatic history tracking allowed for recovery of earlier versions and change auditing when needed.

Possible Future Considerations

In future iterations or collaborative settings, versioning could be enhanced through:

- Modular Python scripting for separate components
- Automated testing or validation scripts for dataset integrity

11. Conclusion

This project explored the spatial relationship between City University of New York (CUNY) campuses and NYC Benefits Access Centers by combining two open datasets and conducting geographic and predictive analyses. The goal was to evaluate how physically accessible public services are to CUNY students and to demonstrate the potential of geographic features in urban classification tasks.

The analysis revealed that:

- Most CUNY campuses are located within 1–2 kilometers of a public access center, suggesting good overall geographic coverage.

- A few campuses, such as Kingsborough Community College, are located farther from the nearest access center, indicating possible areas of service improvement.

- Visual tools such as interactive maps, bar charts, and histograms helped communicate these insights effectively.

As a bonus component, a lightweight machine learning model was built using PyTorch to predict the borough of a Benefits Access Center using only spatial features. The model achieved a test accuracy of 75%, demonstrating that spatial data alone can be used to infer location categories. The mathematical formulation of the model was documented and visualized, providing both practical implementation and theoretical understanding.

In completing this project, we applied a broad set of concepts. The results show how open data, thoughtful preprocessing, and structured analysis can be used to address real-world questions in urban informatics and public service accessibility.