Geographic Clustering for Neighborhood Boundaries: A Spatial Analysis of Chicago using Public Data

Joshua Benjamin Kuppersmith

A Senior Thesis Presented to the
Department of Applied Mathematics
In Partial Fulfillment for Honors in
Applied Mathematics in Data Science (AB)

Harvard University
Advised by: Pavlos Protopapas

March 29, 2019

Table of Contents

1 Abstract

Open data initiatives in cities around the world have enabled new efforts to understand and improve urban areas through data analysis. In order to develop actionable insights to improve cities, it is important to isolate differences between geographic areas throughout the city. Neighborhoods are typically used as a unit for spatial separation, where each neighborhood is internally similar, and different from outside areas. As such, neighborhood analysis is key to developing an understanding of complex urban dynamics, yet current neighborhood boundaries do not always adequately reflect similar areas of cities. This thesis proposes a new clustering algorithm to automatically generate neighborhoods with highly similar internal data profiles. Using a grid-model of a city, this new method of clustering, called Geographic K-Means, incorporates data accumulated within grid cells and builds clumps of neighboring cells with similar data trends. This method is optimized using hyper-parameter tuning to improve an Earth Mover's Distance-based measure of within-neighborhood homogeneity. The optimization uses regularization to enforce smooth neighborhood boundaries, helping us find an optimal balance between data similarity and realistic contiguous neighborhoods. In order to build and test this algorithm, we used Chicago as a case study due to its abundance of data. By generating new Chicago neighborhood boundaries, and increasing within-neighborhood crime homogeneity, we are able to see the relationship between crime and neighborhoods, and better detect sharp boundaries between areas of the city.

All code is available on Github at:

https://github.com/jkup11/Geographic-Neighborhood-Clustering

All raw data and cleaned crime data must be requested separately due to file size. If interested, contact Josh at joshkuppersmith@gmail.com.

2 Introduction

2.1 Motivation

Neighborhoods are a defining characteristic of cities. Large cities have a wide variety of areas, all serving different purposes. New York is well known for its diverse variety of neighborhoods, each having different attributes and personalities. Because of the difference in social attributes between neighborhoods, each can be analyzed and understood individually. The city as a whole is often too large to be analyzed in its entirety. Since each neighborhood is very different, neighborhood analyses are an important way to gain a better understanding of urban areas. As a result, spatial analyses attempting to better understand urban areas are often neighborhood-dependent.

Today, urban analyses are more powerful than ever. With wide availability of urban data through projects like Chicago's Open Data Initiative, data scientists are able to create more rigorous and complex models of cities. For example, crime analysis is one area where geospatial data science has made a considerable impact. Predictive policing, as described by Shapiro [Shapiro, 2017], is the use of spatial analytics to predict crime events and trends to help police deploy resources more effectively. Companies like Civicscape are using high-tech analytical frameworks to accomplish this goal. Micro-mobility businesses, such as ride and bike sharing companies, also use spatial predictive models to determine where to place their fleet and how to optimize their strategy in a city. Midgely explains how smart analytics are used to optimize performance in the bike-sharing business [Midgley, 2009], and cities like Chicago are becoming more and more reliant on these technologies with the boom of dockless bike sharing [Claffey and Lilia, 2018]. Geospatial analysis is not a simple task, and Andresen discusses the challenges associated with spatial heterogeneity in urban modeling, since urban areas are highly variable at a granular level [Andresen and Mallseon, 2013]. This work also implies that it is important to build more powerful urban models to capture this variance and better understand cities. Because of the complexity of cities, and the high stakes of urban modeling, spatial tools to help data scientists and researchers understand cities are extremely valuable.

Because neighborhoods are often crucial to understanding and modeling a city, it is important that neighborhood boundaries are meaningful. This raises a key issue: neighborhood boundaries are fluid. New developments, gentrification, and transportation can all change the identity of neighborhoods. In addition, neighborhood boundaries drawn by a city can be outdated, especially in light of these changing urban dynamics. Delmelle analyzes demographic changes in neighborhoods over time to propose several different key trends that are driving change in neighborhoods over the past 50 years [Delmelle, 2017]. O'Sullivan also highlights the way that neighborhoods influence change in cities, and how neighborhoods themselves change over time. This work suggests that the demographics of stationary neighborhoods irrefutably change over time, so it may be more useful to track how neighborhoods change spatially rather than analyzing change in stationary neighborhoods [O'Sullivan, 2009]. This work proposes a spatial graph-based model to represent urban neighborhoods, and uses data to track demographic change over time.

One possibility to address neighborhood change is to create dynamic neighborhood boundaries using crowdsourcing. Research suggests that there is variation in how individuals define their own neighborhood, but there are common understandings of core neighborhood identity [Coulton et al., 2011]. One proposed solution is that over time, people draw their understanding of neighborhoods, which are then aggregated into boundaries. Bostonography [Woodruff, 2012], a Boston-based mapping and spatial analytics blog, implemented this concept, available at https://github.com/wboykinm/hoodsproj. While this model allows for changes over time, it faces significant issues with individual and community bias, and a data-based model would be more objective.

By attempting to redraw neighborhood boundaries with more meaning, we can provide a tool to help people understand cities and be more effective in drawing conclusions about causation of events in a city. Neighborhoods drawn using data can keep up with the changing dynamics of a city and can help better differentiate neighborhoods.

2.2 Scope of Work

This thesis seeks to develop an algorithm to draw new neighborhood boundaries in urban areas that accurately reflect spatial homogeneity for some set of available data. For simplicity,

in this work, homogeneity will be measured using aggregate reported crime totals, but this method allows for any data to be used to calculate homogeneity, such as demographics from census data. By developing neighborhood boundaries with more internal spatial homogeneity than currently drawn neighborhood boundaries, one can more easily analyze differences between neighborhoods. Chicago is used as a test case for this algorithm, because it has extensive public data and is the author's home.

The first step is an exploratory data analysis of Chicago's available spatial data. Included in this step is pre-processing and cleaning of the data. In addition, an accurate grid representation of the city is generated, and data is accumulated onto the grid. Each row of the resulting database represents a small square within the city, and contains information about the square's location and the data within that area. From there, a metric of success is defined using Earth Mover's Distance and standard deviation to quantify crime homogeneity within each neighborhood. Then, a baseline for homogeneity is established using Chicago's 96 pre-defined neighborhoods. The first attempt to re-draw neighborhood boundaries using K-Means clustering reveals several issues with this method. This preliminary modeling leads to the proposal of the new algorithm, which uses the structure of K-Means with modified distance calculations to generate neighborhoods that are both **homogeneous** and **smoothly contiguous**. A simple Geographic K-Means algorithm is written and tested, then expanded upon to include weighted contributions of geographically accurate Haversine distance and vector distances of available data. The algorithm is then optimized by tuning the weighting hyperparameter, α, with Earth Mover's Distance (EMD) homogeneity used as the target metric. Regularization is introduced as a way to balance the target of neighborhood homogeneity and realistic boundaries. Regularization and EMD plots are used to select a reasonable range for α to create a map with smooth, homogeneous neighborhoods. Then, by visually inspecting the maps, we pick a realistic map that is homogeneous and contiguous. Finally, a smoothing step is run on the final result to re-assign outliers, resulting in an improved final model.

2.3 Previous Work

With the wide availability of detailed geospatial data and analytical tools, many researchers and businesses are using data science to analyze urban environments. Crime modeling of cities is a well-researched area, with an abundance of previous work. In many cases, crime analysis in cities is focused on detecting or predicting crime hot spots, or areas with particularly high crime density. These hot spots are useful in the predictive policing application [Perry et al., 2013], since these projects aim to efficiently allocate police resources to areas of high crime. Within this field, the strategy for locating hot spots varies greatly. In Hermann's spatio-temporal crime analysis [Herrmann, 2013], focus is placed on streets as a geographic unit in order to more effectively aid law enforcement. This work also introduces time as an important factor in predicting crime patterns. Another similar hotspot detection work is Hu et al.'s work [Hu et al., 2018] which utilizes Kernel Density Estimation as a way to analyze crime. Kernel Density Estimation is a method used to estimate a probability density function from sample data, so this method generates a spatial probability distribution which is easily evaluated for hotspots. All of these works present accurate ways to analyze crime and predict high density areas in a city. Because this thesis will specifically use Chicago's crime data to illustrate neighborhood building, these spatial crime analyses were useful to begin thinking about spatial relationships in urban crime, but do not directly address neighborhood-specific models.

In their paper, "Using High-Resolution Population Data to Identify Neighborhoods and Establish Their Boundaries", Spielman and Logan attempt to determine neighborhood boundaries in urban areas [Spielman and Logan, 2012]. They define neighborhoods as contiguous areas defined by social attributes and features that are different from the surrounding area. This definition helps to shape the goals of this thesis in re-defining appropriate and realistic neighborhoods. They calculate "egocentric signatures" to compare the similarities of geographic units given in the census. Then, using likelihood calculations, areas are clustered together, thus defining new neighborhoods. This paper helps to guide the notion of clustering together geographic units. However, this thesis will use uniform rectangular grid cells as the geographic unit, and will attempt to enforce a greater degree of neighborhood contiguity

to model reality more accurately. Hoodsquare [Zhang et al., 2013] also directly approaches the question of re-drawing neighborhoods using data. This work uses Foursquare check-in data to detect neighborhood boundaries, and compiles the algorithm and model into an application that allows users to explore cities, and have neighborhoods recommended to them. This paper uses spatial clustering for hotspot detection, and introduces a grid-based index for spatial homogeneity, which serves as the basis for this thesis's homogeneity metrics. This thesis explores this area by using a grid-based data representation as the basis for clustering. Other works similarly use Foursquare data, specifically to compare similar neighborhoods. One work recommends Earth Mover's Distance as the best way to quantify the difference between data in different neighborhoods [Le Falher et al., 2015], and we will similarly use this metric to calculate neighborhood homogeneity.

The geographic clustering method used in this thesis is also related to several other works that attempt to cluster geographic data. Murray et al.'s 2013 paper [Murray and Grubesic, 2013] explores using K-Means and other non-hierarchical clustering methods to cluster crime events and detect hot spots. While the framework for this thesis will focus on a grid-based crime model rather than individual incidents, this 2013 paper explains methods for spatial clustering and validates clustering as a good way to analyze geospatial datasets. DBSCAN (Density-based spatial clustering of applications with noise) is one clustering algorithm designed for spatial data [Ester et al., 1996]. It uses local density calculations to cluster areas with similarly high or low density, and it detects outliers, handles noise well, and determines the optimal number of clusters automatically. Like Murray et al., this is a valuable spatial clustering tool, but since our data has uniform spatial density, this is probably not the right algorithm to use for this modeling project. Instead, we need an algorithm that will handle the uniform spatial distances of data points (grid cells) and also use the density and distribution of other accumulated data to help determine neighborhood boundaries. These factors will likely need to be appropriately weighted to ensure successful neighborhood determination.

Grid-based clustering algorithms have also been extensively researched, with several popular models in use. STING [Wang, Wei et al., 1997] proposes using a statistical information grid to represent large spatial datasets in order to perform more computationally efficient clustering with higher performance. A similar grid-based approach is used in this thesis.

CLIQUE [Agrawal et al., 1998] is an algorithm used to find high density clusters in high-dimensional data using a grid representation of data, again illustrating the ability for a grid to simply and robustly represent spatial data. WaveCluster [Sheikholeslami et al., 2000] is another grid-based algorithm that uses wavelet transformation to find arbitrarily sized and shaped clusters without high sensitivity to noise and outliers. Grid-based clustering is its own subset of clustering analysis, with countless published algorithms with various applications and purposes, such as to reduce the influence of grid cell size [Lin et al., 2008] and to allow for efficient hierarchical clustering on grids [Schikuta and Erhart, 1997].

Other works focus on neighborhood analysis. Yuan et al.'s 2012 work analyzes neighborhoods in Beijing and uses clustering of point of interest and mobility data to determine functionality of different neighborhoods [Yuan et al., 2012]. This work assumes that roadways are a key factor in neighborhood development and uses road network data to define neighborhoods as areas between significant roads. This road-based neighborhood representation is different from the grid approach, but it enforces contiguous neighborhoods and is another reasonable framework for neighborhood building. It also validates the use of point of interest data to differentiate between neighborhoods with different functions, and presumably different crime profiles. Overall, this work seeks to better understand neighborhoods in an urban area, which is an outcome that this thesis seeks to improve. By re-drawing neighborhood boundaries, this kind of analysis can become even more powerful, since new neighborhoods are more homogeneous, creating greater differentiation between neighborhoods. While many of these works are relevant to the goals of this thesis, few works have attempted to re-draw neighborhood boundaries.

Examples from other fields also helped inspire the algorithmic approach in this thesis. Grid models are present across many fields, and algorithmic approaches to clustering grids are often similar to this geospatial model. For example, the Cellular Potts Model [Graner and Glazier, 1992] (further discussed in Section 7.2.4) uses a grid with clusters representing cellular organisms to model interactions of these organisms. The method uses random simulation to find a minimum energy grid configuration that results in smooth boundaries. This method inspires the regularization step in this thesis to generate neighborhoods with smooth boundaries. Image segmentation often uses grid-based models and clustering because pixels

form a natural grid. K-Means is one algorithm that has been used in this space [Dhanachandra et al., 2015], though this attempt does not account for the same spatial goals as this thesis.

3 Background

The broad idea of this project is motivated by previous work, and individual pieces of this complex problem are also driven by research. This thesis relies on various technical tools used in the field. This section aims to provide context and background to the methods used and developed while modeling the problem of re-drawing urban neighborhoods using geospatial data.

3.1 Geographic Features

As explained previously, Yuan et al.'s functional area analysis [Yuan et al., 2012] shows considerable evidence that geospatial datasets can reveal significant amounts of information about urban areas, by building a model to predict the function of different parts of a city. Several pieces of research use other geographic data to illustrate spatial relationships with crime. For example, Bogomolov's 2014 analysis uses human movement data and demographic data to predict crime, and validates the relationships between these data sets and crime trends [Bogomolov et al., 2014]. Groff is even more specific and uses point of interest data about bars to generate a geospatial crime modeling system and illustrate positive correlation between proximity of bars and higher crime [Groff, 2013]. This evidence suggests that geographic data like cell towers, points of interest, and transportation data can be useful in building a crime model, so they may also be successful in building neighborhood clusters that represent consistent crime. All of these types of features are available in Chicago and are collected and used in this project.

3.2 Data Pre-Processing and Spatial Grid System

Uniform grids are common in statistical geospatial work and rely on spherical trigonometry to accurately handle coordinates. Wang et al.'s work proposes a layered hierarchical grid to maximize information stored on the grid but minimize storage space and time complexity of lookup [Wang et al., 2018]. This work describes other spatial grid and clustering methods in use, and suggests that a grid-based model is a reasonable way to represent data across

a city. While it is very simple to lay out a grid of equal latitude and longitude offsets for each cell, this does not accurately represent geographic data. Because of the curvature of the Earth and how latitudes and longitudes project onto Earth's sphere, equal offsets in latitude and longitude do not represent equal geographic distance. Instead, the Haversine Formula must be used to calculate great circle distances given latitude and longitude, and find equally sized grid cells [Gellert, W. et al., 1989]. Another possibility considered was to represent distance with the Vincenty formula, which is slightly more accurately because it represents Earth as an oblate spheroid rather than a sphere [Vincenty, 1975]. Because all distances in this thesis are very small, the Haversine formula will suffice for calculating distance between two points given latitude and longitude coordinates, and is as follows:

$$a = \sin^2(\frac{\phi_2 - \phi_1}{2}) + \cos\phi_1 * \cos\phi_2 * \sin^2(\frac{\lambda_2 - \lambda_1}{2})$$

$$c = 2\tan^{-1}(\sqrt{a}, \sqrt{1-a})$$

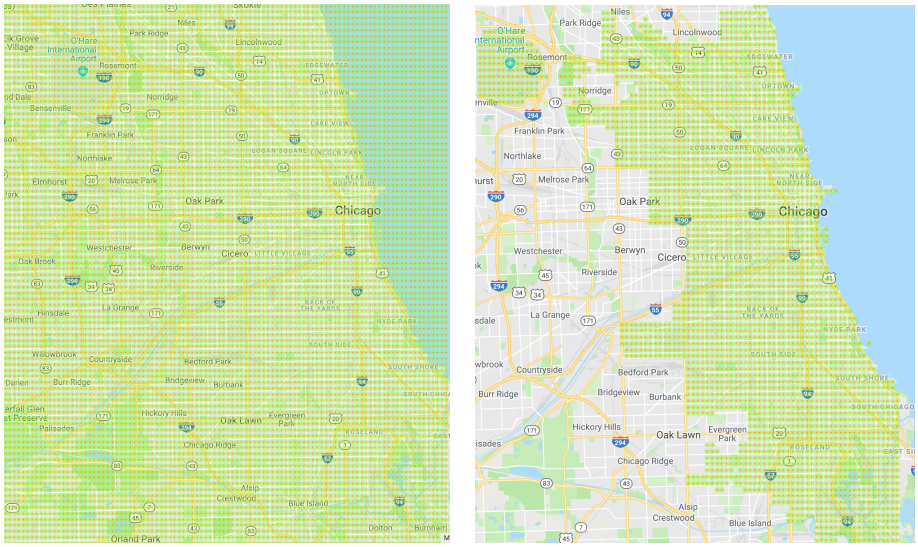
$$d = R * c$$

Where ϕ_1 and ϕ_2 are latitudes, λ_1 and λ_2 are longitudes, d is the distance in meters, and R is the constant for the Earth's radius, approximately 6371000m.

In the grid generation, the user picks the center point, desired distance offset between grid cells, and number of cells. In the current example, I selected 100x100 cells, each approximately 0.3km^2. This size was selected because it is large enough to minimize sparseness, and small enough to allow for precision in detecting neighborhood boundaries. This forms the basis of our units for clustering and analyzing the city.

The visuals below illustrate the initial generated grid, and how it changes once filtered by the city's boundary. The same filtering process is used to "cluster" the city by pre-defined neighborhood and police beat boundaries.

Figure 1: Spatial Grid Generation and Filtering



(a) Rectangular Dotted Spatial Grid (b) Chicago Filtered Spatial Grid

3.3 Principal Components Analysis

Principal Component Analysis, also known as PCA, uses linear algebra transformations on data to convert correlated features into a set of uncorrelated variables called principal components [Hotelling, 1992]. These principal components help to extract more meaning from correlated features, and they take into account initial scale of variables. Because many sets of features in this dataset are correlated (see Figure 6), like crime features and OSM data with cell towers, PCA is a great way to transform these features into a smaller and more informative set of features across the city.

PCA creates a list of p principal components $(Z_1, .., Z_p)$ which are all linear combinations of the original data with vector norms of 1. The principal components are pairwise orthogonal, and are ordered by the amount of variance in the data that each component explains, such that the data has more variance in the direction of vector Z_1 than Z_2.

Let matrix $\mathbf{Z}$ be composed of columns $Z_1, .., Z_p$, such that $\mathbf{Z}$ is an $n \times p$ matrix, where n is the number of rows in the original dataset. Matrix $\mathbf{X}$ is the original dataset, also an $n \times p$ matrix. Re-center each column in $\mathbf{X}$ to have mean 0 and no intercept, then calculate the eigenvectors of the matrix $\mathbf{X}^T\mathbf{X}$, and let $\mathbf{W}$ be the matrix of eigenvectors.

For more background on eigenvectors, please visit MIT Math's overview of Eigenvectors and Eigenvalues. $\mathbf{W}$ is a $p \times p$ matrix. Then, $\mathbf{Z} = \mathbf{XW}$, and the columns of $\mathbf{Z}$ are re-ordered based on explained variance [Jolliffe, 2002].

In this thesis, PCA is run using functionality in the `sklearn.decomposition` library. This library also allows for easy calculations of explained variance.

3.4 Clustering

Clustering is a subset of machine learning techniques used to group together similar data. Clustering algorithms have existed for decades, and clustering is a popular unsupervised learning technique because it explores inherent structure of data without pre-defined labels. "Algorithms for Clustering Data" introduced many methods including hierarchical and partition clustering, and many more sophisticated and accurate algorithms have been developed since this book was published [Jain, Anil and Dubes, Richard, 1988]. In this thesis, clustering is extremely important for grouping together cells with convincing similarity such that clustered neighborhoods are as homogeneous as possible. In our modeling, each generated cluster represents one neighborhood. K-Means clustering is a simple and reliable algorithm that will form the basis of the clustering in this project, and will be modified in order to perform accurate geospatial clustering. The K-Means clustering algorithm is described in depth in the next section. An update on this algorithm for geographic clustering will use a more appropriate distance function in each step of the algorithm. Along these lines, there is existing scholarship on spatial clustering methods, including CLARANS [Ng, R.T. and Han, Jiawei, 2002], developed in 2002. This algorithm is meant to group similar geographic data in an efficient way and builds functionality that allows for clustering of complex geographic objects. While this is not the method that will be used for geospatial clustering in this thesis, it provides good background on possibilities.

3.5 K-Means Clustering

One very common clustering algorithm is K-Means clustering. The goal of the algorithm is to optimize clustering of data points such that the sum of the distance between each data

point and its corresponding centroid is minimized. The observations are partitioned into k clusters. The algorithm is as follows:

Algorithm 1: K-Means Clustering

procedure K-MEANS(X, k) ▷ X: observations, k: # clusters
  Randomly initialize centroids $\mu_1, ..., \mu_k$ from X
  while $\Delta_\mu \neq 0$ **do** ▷ Iterate until convergence
    $\forall \mathbf{x_i} \in X$, set $C_i = \text{argmin}(||\mathbf{x_i} - \mu_\mathbf{j}||)$ for j in k ▷ C_i: label for x_i
    for $j \in k$ **do**
      $X_j \leftarrow \mathbf{x_i} \quad \forall \mathbf{x_i} \in X$ if $C_i = j$ ▷ Each X_j is a cluster
      $\mu_j = \frac{1}{|X_j|} \sum_{\mathbf{x} \in X_j} \mathbf{x}$ ▷ Update μ_j to the mean of their data
    end for
    $\Delta_\mu = ||\mu_\mathbf{old} - \mu_\mathbf{new}||$ ▷ Change in μ to check convergence
  end while
  return C_i ▷ C_i form the new clusters from X
end procedure

First, randomly initialize centroids as data points. Then, iterate until centroids converge. On each iteration, calculate distances between data points and centroids and assign each data point to the cluster corresponding to its nearest centroid. Then, update each centroid to the Euclidean mean of the data in its cluster. Calculate the difference between centroids iteration to iteration to check for convergence. The final cluster labels represent converged clusters.

3.6 Earth Mover's Distance

Earth Mover's Distance is a commonly used statistical metric which measures distance between two probability distributions and is proportional to the work required to change one distribution into the other. Concretely, if we have vectors x (length m) and y (length n) which both represent weighted probability distributions, then $F = (f_{ij})$ is an $m \times n$ matrix representing the flow between x and y, where f_{ij} "represents the amount of weight at x_i which

is matched to weight at y_i" [University, 1999]. We also define a distance matrix $D = (d_{ij})$, where distances can be calculated in many ways, the most common being Euclidean distance. Then, the formula for EMD is produced by solving an optimization problem to find the flow that minimizes the c,ost of altering distributions. The final result is as follows:

$$\text{EMD}(x, y) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}}{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij}}$$

This can be understood as work normalized by total flow. Additional work has been done with Earth Mover's Distance on vector fields, such as the work of Lavin, which presents a simple and novel way to use EMD to compare vector fields [Lavin et al., 1998]. This suggests that EMD can be used as a metric of similarity in this thesis, where crime counts across a neighborhood are represented as a distribution vector. The EMD between this distribution and a fabricated random distribution centered at the mean (μ) with noise up to $\pm\sqrt{\mu}$ is a measure of inhomogeneity. There are several libraries in Python that provide functionality for these calculations, which simplifies the process of making these calculations.

3.7 Normalization

Normalization allows us to find a better initial weighting of these distances in order to precisely tune hyperparameters. It plays a key role in the analysis of the developed clustering algorithm because features begin with highly variable magnitudes. K-Means clustering weighs features with higher magnitude more heavily, creating a need for normalization to isolate the influence of each individual feature. In addition, because of the way that the distance metric for the new clustering algorithm is calculated, it is important to normalize features so that geographic distance can be weighted appropriately in the algorithm.

In order to normalize a column of a dataset, represented by $\vec{x}$, we run the following vector transformation:

$$\vec{x}_{norm} = \frac{\vec{x} - \mu}{\sigma}$$

Where μ is the mean of the vector $\vec{x}$, and σ is the standard deviation of the vector $\vec{x}$. This vector transformation results in an updated vector $\vec{x}$ with a mean of 0 and a standard deviation of 1.

3.8 Regularization

Regularization is a technique used to add information to a modeling problem in order to avoid overfitting. Since clustering is an unsupervised learning method, this problem is not a standard application for regularization because there is no ground truth value. However, the concept of overfitting still applies when improper weighting of data makes clusters very homogeneous but non-contiguous, so they do not represent realistic neighborhoods. An example of this improper weighting is seen in Figure 18.

One standard regularization method is ridge regression, which uses an L2 regularization penalty term to minimize the size of the parameters, in order to shrink the parameters and prevent co-linearity [Hoerl and Kennard, 2000]. Lasso regression is similar, but uses an L1 regularization term and is often used to select features for a model [Tibshirani, 1996]. In either case, a specifically designed penalty term is used in the model to help find a better solution without overfitting.

In this thesis, a regularization term is designed to penalize non-contiguous neighborhoods since the model attempts to maximize neighborhood homogeneity, often at the expense of neighborhood contiguity. If a model clusters together cells using mostly crime data, homogeneity will be very high, but neighborhoods will be scattered across the map, and not contiguous. Although regularization is not built into this model because it is not a predictive model with pre-established truth, this regularization term can be plotted along with inhomogeneity to help select the most reasonable model. More details will be provided in sections 5.7 and 6.6.

3.9 Hyperparameter Optimization

In any machine learning model, certain parameters of the model can be tuned to improve the performance and outcome of the model. Ordinarily, with predictive models, this kind of

optimization is performed with cross validation. Data is split into training and test sets, a model is trained on the training set and deployed on the test set, and a metric of success is calculated on the test predictions. Grid search is the most common way of tuning multiple parameters at once. By running and evaluating the model multiple times on a grid of possible parameters, one is able to find the best set of parameters [Claesen and De Moor, 2015].

Since clustering is an unsupervised learning method, there is no definite truth of neighborhood identity for each cell, so it is not possible to define accuracy on a test set. Instead, we must define another metric of success: homogeneity. Using this metric, we can run the clustering algorithm with various parameters and determine which model yields the best results. In most clustering algorithms, k: the number of clusters, is an important parameter to choose. The weighting parameter for the new Geographic K-Means distance metric, α, is another parameter that must be carefully tuned.

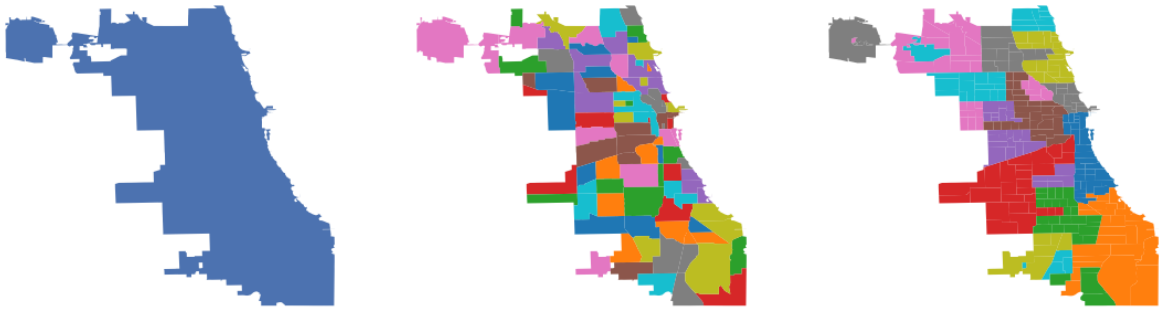
4 Data and Exploration

4.1 Data Overview

Chicago was selected as the test city for this thesis because of the quality and availability of data on the Chicago Data Portal. Chicago is at the cutting edge of urban data collection and has dedicated considerable resources toward the development of systems to record and store the city's data. The data is available to the public and used for urban planning. Chicago hired Brett Goldstein, a Senior Fellow in Urban Science at the University of Chicago and geospatial tech entrepreneur, as the city's first Chief Data Officer in 2011 then as the Chief Information Officer in 2012 [Goldstein, 2013]. Chicago's innovative integration of technology into the city's operations is crucial for generating high quality data that enables data scientists to make significant findings. Chicago is an example for other cities to follow to improve the field of geospatial data science. To begin this project, I performed an in-depth analysis of the data available for Chicago to gain some initial insight into the various available data sources.

The Chicago Data Platform provides shapefile datasets with pre-defined boundaries for the city. Figure 2 (a) is the outline of the city boundaries, which is used to filter out grid cells that are not in the city, greatly reducing the size of the dataset. The Neighborhood and Police Beat boundaries in Figures 2 (b) and (c) are used as baseline models and are useful to compare to the results of newly developed neighborhoods.

Figure 2: Various Chicago Shapefile Boundaries



(a) Chicago City Boundary (b) Neighborhood Boundaries (c) Police Beat Boundaries

Chicago's crime data is another important aspect of this project, since it is used in the example model to determine if the new algorithm can cluster together areas with similar crime trends. Chicago's crime data is available in `.csv` format, with a record for every crime recorded in Chicago since 2001. Each record is made anonymous by abstracting locations to the block level, thus each crime is given an approximate coordinate location. For each crime, the record includes the date and time, crime type and description, approximate location, and more. An important disclaimer on this crime data is that crime collection and reporting systems are deeply flawed and biased. Police reporting of crime can be biased because of differences in the way that neighborhoods are policed across economic and racial lines [Gaston and Brunson, 2018]. This bias is an extremely important issue, and more work must be done to understand how bias manifests itself in large crime data sets. This is an especially important consideration when deploying any kind of predictive model using crime data, which may reinforce these biases. Taking possible bias into account, this neighborhood model is not meant to be used to deploy resources in a city, rather it only attempts to find differences between areas and identify parts of cities that are clearly similar. The potential for algorithmic bias drives a need for better ways to combat bias in crime data, but is beyond the scope of this thesis.

Although this highly detailed Chicago crime dataset could merit its own separate analysis, the simple visualization of crime trends in Figure 3 illustrates the complexity of this dataset. Each crime has a unique trend, and determining the reasons for trends and significant spikes

and decreases in a specific crime type is a valuable way to try to understand crime in the city.

Figure 3: Crime Trends by Type for 2001-2018

Another way to visualize this dataset is with a map. The technique of mapping geospatial data is used throughout this project to understand trends over space, and was very important for developing the initial intuition and plans for the algorithm. The Google Maps API is very useful for plotting data and generating heatmaps, and the libraries Basemap and Geopandas were also used for mapping. Matplotlib and Seaborn plots with latitude and longitude on the axes were also used for simple graphs, like plotting clusters, differentiating each by color. A heatmap of all crime incidents in Chicago (Figure 4) illustrates hotspots of reported crime, and gives an initial understanding of parts of the city's layout.

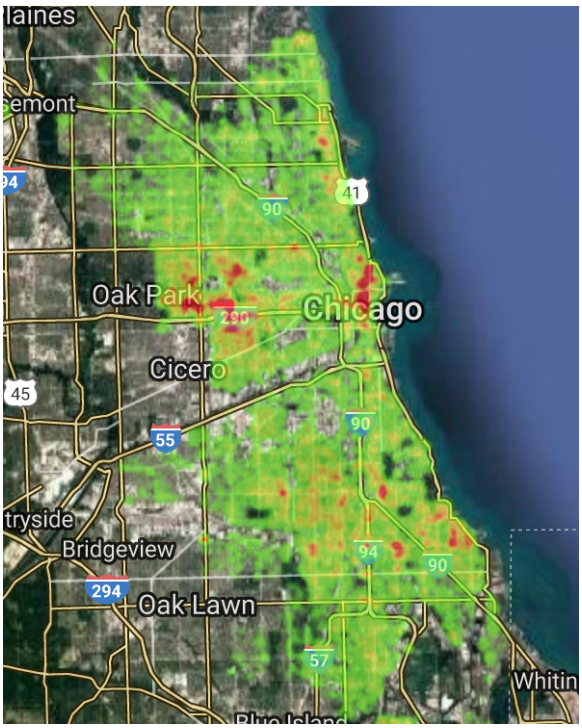
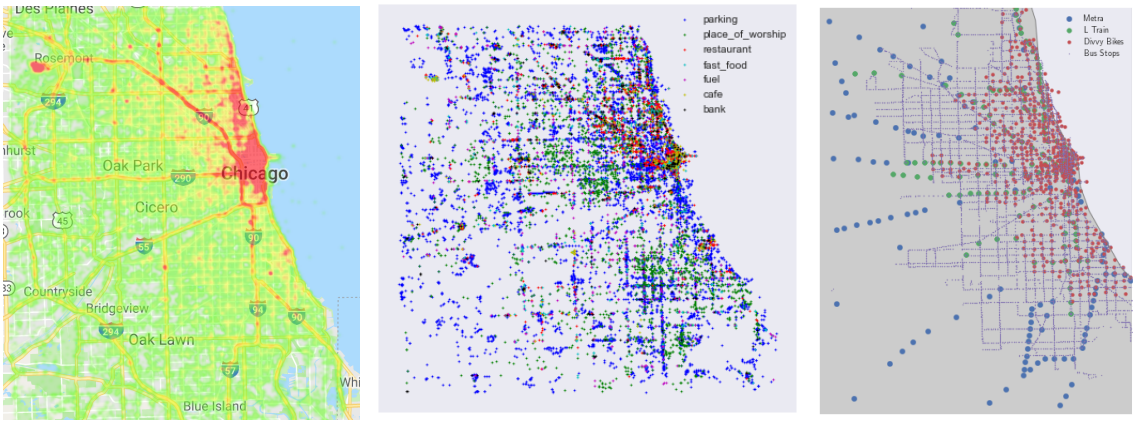


Figure 4: Heatmap of Chicago Crime

For this project, various sources of geospatial data were collected so that the accumulated dataset contains a rich set of information about Chicago's urban landscape. Point of interest data was collected from OpenCellID and OpenStreetMap. OpenCellID manages an enormous dataset of cell towers around the world, including well over 5 million towers in the United States. OpenStreetMap (OSM) is a crowdsourced map that relies on a virtual community of contributors to build a map of what exists and where points of interest are located. Together, these datasets presumably can represent population density and give information that can help to differentiate neighborhoods in a city, like residential areas, business areas, and commercial districts. In addition, the Chicago Data Platform provides rich data about public transportation and bike sharing, so the locations of these transportation hubs was added to the dataset in order to provide an even fuller collection of spatial features. Various visualizations of these datasets are shown in Figure 5.

Figure 5: Chicago Point of Interest Datasets

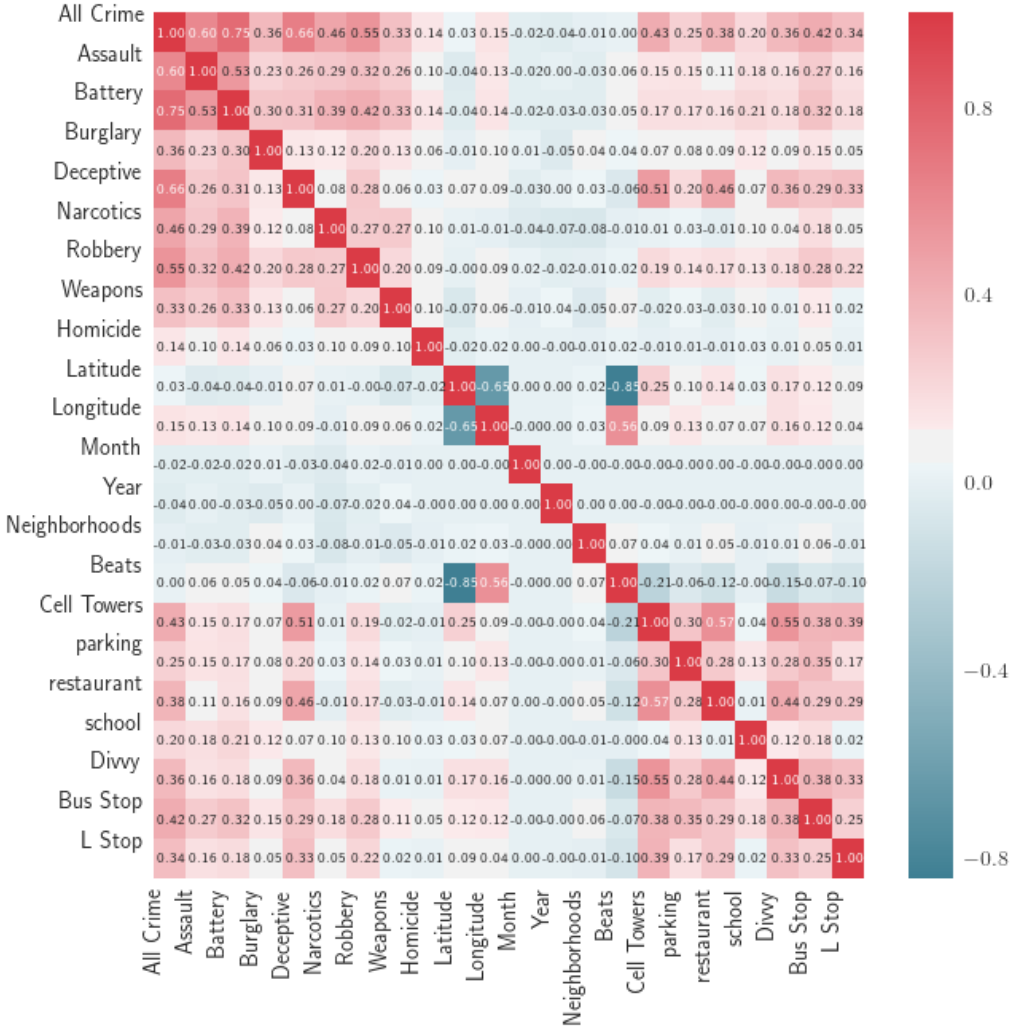


(a) Cell Tower Heatmap (b) OSM Points of Interest (c) Public Transportation

4.2 Feature Correlation and PCA

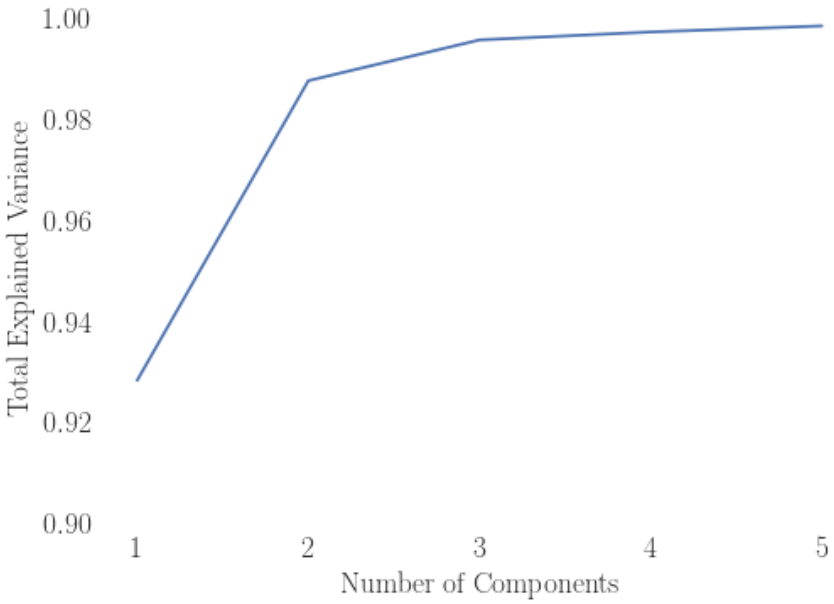
With these features collected, the grid is prepared for clustering analysis and is full of features that can help to differentiate neighborhoods from each other. Of course, some of these features are highly correlated, and Figure 6 illustrates the correlation between the most significant features in the data set. Several geographic features, including cell towers, parking, restaurants, Divvy bikes, and bus stops have reasonable positive correlation with particular types of crime such as Battery, Deceptive Practice, and Robbery. In addition, it is apparent that several of the crime metrics are highly correlated, suggesting that the crime data follows global trends as well as trends for individual crimes. In addition, many of the geographic features are highly correlated, which suggests that throughout the city, high and low feature density areas are consistent.

Figure 6: Grid Accumulated Correlation Between Key Features



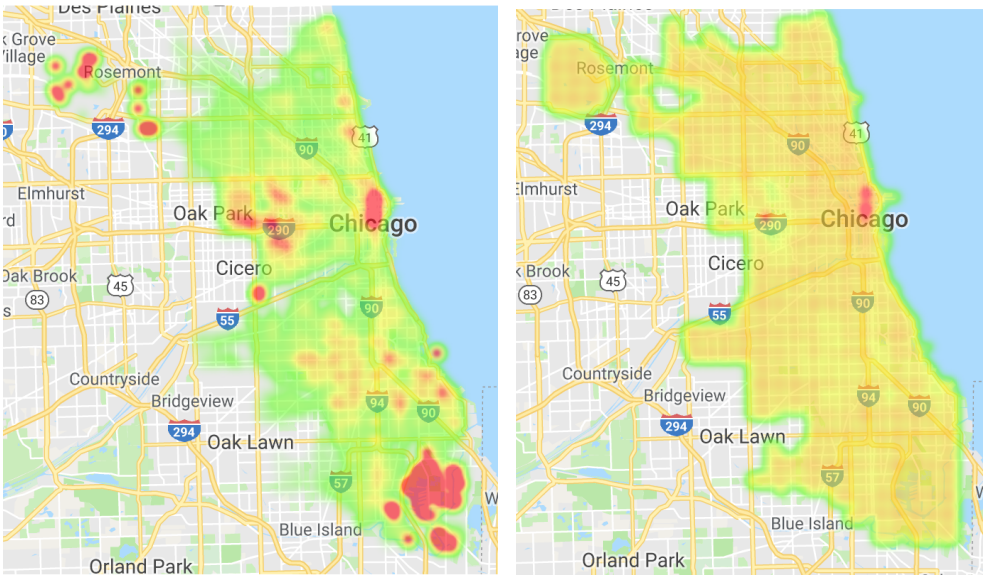
Because so many of these features are highly correlated, PCA is used for dimensionality reduction, in order to extract more informative features from this dataset. The first choice with PCA is to determine how many components are necessary for the analysis to account for most of the variance in the data. Figure 7 shows that after looking at only 2 principal components, nearly 99% of variance is explained. This means that these 2 features explain the trends in the data set extremely well.

Figure 7: Variance Explained by First 5 Principal Components



We can also visualize these first two principal components using weighted heatmaps that use the transformed PCA features as weights across the map to show areas of high value (see Figure 8). It seems that the first principal component shows areas of high crime, which makes sense because crime data has the highest magnitude of the original features. Principal Component #2 has low magnitudes, but highlights an area downtown in Chicago's loop, perhaps where there is a high density of features but low crime, since this would be orthogonal to the first principal component. It is useful that the various spatial trends of this dataset can be condensed to fewer features than we originally had. To interpret the PCA results, we can look at the coefficients of each feature vector, since the principal components are linear combinations of features. The first principal component has a very large positive coefficient for 'All Crime', as well as large positive coefficients for 'Theft', 'Narcotics', 'Battery', so this seems to be a good proxy for crime density in the city. The second principal component has large positive coefficients for 'Theft' and 'Deceptive Practice', and large negative coefficients for 'All Crime', 'Battery', and 'Narcotics', so this component represents another crime trend, perhaps where there is petty crime but not very much violent crime. Because the crime data is very rich in the dataset and has high magnitudes, these are the features that the PCA analysis highlights.

Figure 8: Principal Component Analysis Visualization



(a) Principal Component #1 (b) Principal Component #2

These analyses help to create initial ideas about the available data, and shape the desire to cluster the city into new neighborhoods based on the available data.

5 Methodology

The broad goal of this thesis is to develop a method of clustering a grid representation of Chicago with geographic data in order to generate intuitive and spatially homogeneous neighborhood boundaries. This section discusses the specific techniques used to achieve this goal.

5.1 Evaluation Metrics

The goal of the evaluation metric in this thesis is to measure homogeneity (or inhomogeneity) of neighborhoods by comparing the cells of a given neighborhood with each other. For simplicity, the evaluation metric compares the total crime counts in each cell. In the future, this metric could be multi-dimensional to account for counts of individual crime types as well.

The first metric considered is a calculation of standard deviation. For each neighborhood, we call the count of total crimes in the cells in the neighborhood $x_i, ..., x_n$, and $\bar{x}$ is the average crime count in the cell. Then, we calculate:

$$SD = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n-1}}$$

Iterating through the neighborhoods, we generate a vector of standard deviations for each neighborhood and we can take the mean or median to give information about overall homogeneity. Lower standard deviation in a neighborhood (or across all neighborhoods) correlates to a more homogeneous neighborhood. This calculation is performed using the standard deviation function in the Python package `numpy`.

The other metric used to measure homogeneity is Earth Mover's Distance, described in the EMD Background section. As a recap, Earth Mover's Distance is a metric used to measure distance between distributions such that similar histogram shapes are close and dissimilar distributions are far. Similar to the calculation of standard deviation, the EMD metric iterates through each neighborhood, and calculates a homogeneity value, returning a

vector of measurements which can be interpreted with mean and median. For each neighborhood, the vector of crime counts in each cell is compared against a fabricated vector which is the mean crime count (μ) $\pm$ noise (up to $\sqrt{\mu}$). EMD is the way that these distributions are compared, since they can both be interpreted as similar histograms. Because the comparison vector is generated randomly, the calculation is run 50 times and averaged in order to ensure consistent results that are not influenced by any one particular fabricated homogeneous vector.

EMD calculations are based on the definitions and `C++` implementation from Pele and Werman's 2009 Conference Paper, "Fast and robust Earth Mover's Distances" [Pele and Werman, 2009]. The Python package used is called `pyemd`, which is a Python wrapper for this `C++` implementation, and it is written by William Mayner.

5.2 Baseline - City Neighborhood and Police Beat Boundaries

For this simple baseline model, pre-defined neighborhoods and police beats are the areas that constitute neighborhoods. To generate the baseline model, we filter cells and determine which neighborhood and beat each cell is within. Then, the baseline model requires calculations of standard deviation and Earth Mover's distance, both simple calculations because these metrics are written into the functions `SD` and `EMD` (see file `Functions.py`).

The 'Police Beats' division contains ~260 distinct neighborhoods, while the 'Neighborhood' division has 96 distinct neighborhoods. Since this baseline will be tested against all future models, tests will be used to determine which baseline model is most appropriate.

5.3 Preliminary Modeling with K-Means

K-Means clustering, as described in Section 3.5, uses Euclidean distance as its distance metric and is a useful simple algorithm to cluster data. `Sklearn` contains excellent out-of-the-box functionality for K-Means clustering in Python, providing a simple way to perform initial tests on data.

Since this model is easy to run, it is used to test different combinations of features in clustering as well as the effects of geography and geospatial features, and manipulations

such as normalization. K-Means clustering provides a good naive model to run initial tests, and develop an understanding for the challenges associated with designing a meaningful algorithm for this problem.

5.4 Geographic K-Means Algorithm

This new clustering method will utilize distance calculations that are different from Euclidean distance for several reasons. First, the Haversine distance between coordinate sets must be used to account for accurately projected spatial distance. Additionally, this clustering method must prioritize contiguous results, since neighborhoods must be contiguous to adequately model reality. Based on our working definition of a neighborhood, it does not make sense for one neighborhood to contain cells all across the city. Therefore, the most important factor for clustering is geographic distance. For that reason, when we define the function to measure distance between cells during clustering, it must be structured as:

$$(1 - \alpha) * H + \alpha * F;$$

where H is the Haversine distance between cells and F is the Euclidean distance of the vectors containing other features considered in clustering. α is a parameter that can be adjusted to weight geographic distance and feature distance optimally, such that neighborhoods are both homogeneous and contiguous.

Whereas K-Means clustering selects centroids by choosing k random data points or k random points in the vector space, this algorithm uses a different method for centroid initialization. To initialize centroids, we first filter cells by each Chicago neighborhood boundary, and calculate the center of each neighborhood using Euclidean mean ($c_1, ..., c_k$ where $k = 96$). We then determine which cell in the data set is closest geographically to each c_i, and let centroid μ_i be the point closest to c_i. This method for centroid initialization takes advantage of the urban knowledge that goes into the city's neighborhood assignments by allowing these neighborhoods to influence the algorithm's initial configuration. Fixed centroid initialization also enforces repeatability of results because it is not a random initialization. This means that any two attempts to cluster the data with the same parameters will result in the same clusters. Finally, this method sets k strictly at 96, the number of neighborhoods

in Chicago. Although there are potential benefits to loosening restrictions on k and being able to optimize this parameter, this restriction ensures that comparisons to the baseline model are fair and valid. Restrictions on k will be discussed further in Section 7.2. The full algorithm is as follows:

Algorithm 2: Geographic K-Means Clustering

procedure GEO K-MEANS(X, k) ▷ X: observations, k: # clusters
  Filter cells by neighborhood, calculate each neighborhood's center $c_1, ..., c_k$
  Initialize centroids μ_i to the cell closest to c_i ▷ Result: centroids $\mu_1, ..., \mu_{96}$
  while $\Delta_\mu \neq 0$ **do** ▷ Iterate until convergence
    $\forall \mathbf{x_i} \in X$, calculate Haversine distance from each centroid's coordinates (H)
    $\forall \mathbf{x_i} \in X$, calculate Euclidean distance from each centroid's features (F)
    $\forall \mathbf{x_i} \in X$, add $(1-\alpha) * H + \alpha * F$ ▷ New weighted distance metric
    $\forall \mathbf{x_i} \in X$, set $C_i = \text{argmin}((1-\alpha) * H_j + \alpha * F_j)$ for $j \in k$
    for $j \in k$ **do**
      $X_j \leftarrow \mathbf{x_i} \quad \forall \mathbf{x_i} \in X$ if $C_i = j$ ▷ Each X_j is a cluster
      $\mu_j = \frac{1}{|X_j|} \sum_{\mathbf{x} \in X_j} \mathbf{x}$ ▷ Update μ_j to the mean of their data
    end for
    $\Delta_\mu = ||\mu_{\mathbf{old}} - \mu_{\mathbf{new}}||$ ▷ Change in μ to check convergence
  end while
  return C_i ▷ C_i form the new clusters from X
end procedure

Note that the centroid update is not consistent with the new distance metric, since it utilizes Euclidean mean. However, this centroid update is a very accurate approximation for the much more computationally expensive ideal calculation, so it is used for simplicity.

When running tests with this algorithm, our first attempts use only geography, then later trials include additional features. Because of the testing with PCA explained variance, and additional modeling performed separately, the first two principal components are used as features for this modeling. When running PCA, all geospatial features including crime, cell towers, OSM point of interest features, and transportation features are used, and the

resulting transformed vectors explain the variance in these data sources. We also experiment with normalization and optimization of parameters.

5.5 Proof of Convergence

Like the K-Means clustering algorithm, the Geographic K-Means clustering algorithm can be proven to guarantee convergence. The proof follows similar logic to regular K-Means, since the algorithm is very similar to K-Means, with small updates. The proof below has been adapted from Bottou and Bengio's 1995 work [Bottou and Bengio, 1995], and much of the inspiration for the proof is derived from Rebbapragada et al.'s 2009 paper, which proves convergence of an updated version of K-Means clustering called Phased K-Means [Rebbapragada et al., 2009].

This clustering algorithm has converged once it reaches an iteration where the cluster composition remains consistent. This guarantees that the centroids will not move and clusters are locked in position.

Some key notation will now be defined. At each iteration, there is a set of centroids W (either initial centroids, or the centroids calculated in the previous iteration). During each iteration, we calculate distances from centroids and assign new cluster labels C based on the minimums. These new clusters assignments are used to calculate updated centroids W', and the new set of cluster assignments are C'.

We will also define x_i to be a piece of data, w to be an individual centroid and $c(i)$ to be the cluster that x_i is assigned to at any time. Then, we can define the error function that the algorithm attempts to minimize, which is:

$$E(W, C) = \sum_i \frac{1}{2}(x_i - w_{c(i)})^2$$

This error metric iterates over each data point and sums the distance between the data point and the centroid in its assigned cluster. This algorithm minimizes the aggregate sums of these distances.

In order for K-Means to converge, it must be true that this error cannot increase during an iteration. Using our notation, this means that

$$E(W', C') \leq E(W, C)$$

$$\sum_i \frac{1}{2}(x_i - w'_{c'(i)})^2 \leq \sum_i \frac{1}{2}(x_i - w_{c(i)})^2$$

One way to show that this is true is to show that

$$E(W', C') \leq E(W', C) \leq E(W, C)$$

and this sequence follows the steps in the algorithm. Intuitively $E(W', C)$ is the error when we have calculated new centroids, but still have the old cluster assignments, and $E(W', C')$ is the error after each data point has then been re-assigned to the nearest cluster.

The second step in this inequality, $E(W', C') \leq E(W', C)$, is trivial because both of these terms have a constant set of centroids, but C' is calculated in order to minimize each individual distance between a data point x_i and its nearest centroid $w'_c(i)$, so the sum of these distances is less (or equal) after the updated assignment than before. This expression is a representation of this inequality:

$$\sum_i \frac{1}{2}(x_i - w'_{c'(i)})^2 \leq \sum_i \frac{1}{2}(x_i - w'_{c(i)})^2$$

The first step in the target inequality is to show that $E(W', C) \leq E(W, C)$. In both of these cases, the cluster assignments are the same, so we must show that when centroid positions are updated (W to W') by averaging the points in each cluster C (using Euclidean mean), they are closer on the whole to the set of data points in their clusters. By this definition of the update, W' are centroids that minimize distances between centroid and cluster data points. By calculating $\frac{\delta E}{\delta w'}$ for $w' \in W$, and setting equal to 0, we get:

$$w' = \frac{1}{|c|} \sum_{i \in c} x_i$$

Therefore by definition, $E(W', C) \leq E(W, C)$ holds.

We have shown that for each subsequent step in K-Means clustering, the error decreases: $E(W', C') \leq E(W', C) \leq E(W, C)$. Thus, for any finite set of data points and centroids (all will be finite in this thesis), K-Means clustering converges because the error function is non-increasing.

To extend this proof to this Geographic K-Means algorithm, a few small changes must be made. In this algorithm, rather than using Euclidean distance as the distance metric, we

use a weighted sum of Haversine distance and Euclidean vector distance. As stated above, Haversine distance is defined as:

$$H(\phi_1, \lambda_1, \phi_2, \lambda_2)$$

$$a = \sin^2(\frac{\phi_2 - \phi_1}{2}) + \cos\phi_1 * \cos\phi_2 * \sin^2(\frac{\lambda_2 - \lambda_1}{2})$$

$$c = 2\tan^{-1}(\sqrt{a}, \sqrt{1-a})$$

$$d = R * c$$

We define the above formula as H, and define feature vector Euclidean distance as F. If we let $\vec{a} = (a_1, a_2, ..., a_n)$ and $\vec{b} = (b_1, b_2, ..., b_n)$, then

$$F(\vec{a}, \vec{b}) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + ... + (a_n - b_n)^2}$$

Then, our weighted distance function for Geographic K-Means (where H is applied to latitudes and longitudes, and F is applied to features) is

$$(1 - \alpha) * H + \alpha * F$$

We want to show that this distance function is correlated with ordinary square Euclidean distance. Clearly $\alpha * F$ correlates with Euclidean distance because it is the same metric scaled down by a constant. Intuitively, Haversine distance correlates with Euclidean distance because it is a measure of distance on the ground of the spherical projection of Earth. Although the scales of these calculations are different, we know that if we take the Haversine and Euclidean distances between two sets of latitude/longitude coordinates a_1, a_2 and b_1, b_2 such that $a_1 > b_1$, $a_2 > b_2$, then do the same calculations on $(a_1 + c), (a_2 + c)$ and b_1, b_2, where c is a small positive constant, the distance will increase. Similarly, if c is small and negative, the distance will decrease. Therefore, Haversine distance and Euclidean distance preserve the change in distance despite different scales, and are monotone functions.

Because the new distance metric is a weighted sum of two distance functions that are correlated with ordinary Euclidean distance used to prove the convergence of K-Means, Geographic K-Means distance function and Euclidean distance are directly related. Note that we will call latitude and longitude *lat* and *lon* for reference, and spatial features are

$feat$. Therefore, we can redefine our error function with this new metric as

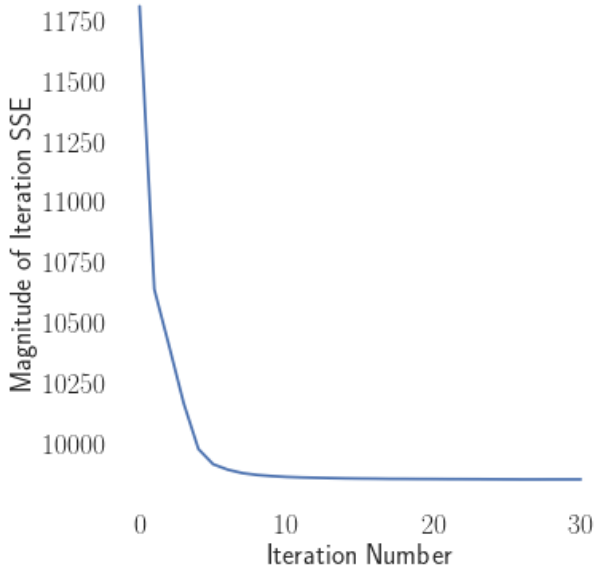
$$E(W, C) = \sum_i [(1-\alpha) * H(x_{i(lat)}, x_{i(lon)}, w_{c(i)(lat)}, w_{c(i)(lon)}) + \alpha * \frac{1}{2}(x_{i(feat)} - w_{c(i)(feat)})^2]$$

which is the sum of $\alpha * F$ and $(1-\alpha) * H$, and is highly correlated with Euclidean distance.

The Geographic K-Means algorithm only changes the error function used above in the proof of K-Means convergence by weighting factors. The new error function uses Haversine Distance, which is directly related to Euclidean distance, but uses trigonometry to better represent Earth's spherical shape and map projections. Therefore, centroid calculation does not change with this new metric, so convergence of Geographic K-Means holds from the proof of convergence for ordinary K-Means clustering.

Through trials of the K-Means algorithm on Chicago's data, we can calculate the target error and show that it weakly decreases every iteration, illustrating algorithmic convergence (see Figure 9 below).

Figure 9: Geographic K-Means Convergence



The centroid initialization technique also differs from the ordinary K-Means algorithm. Ordinarily, centroids are initialized at random from the data points, or are set to be randomly generated points in the vector space of the data. In this algorithm, the centroids are

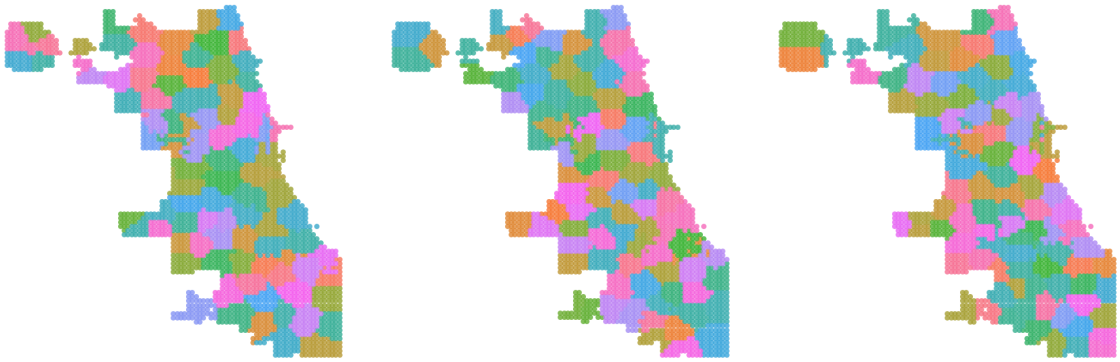
initialized to proxies for the center of neighborhoods defined by the City of Chicago, so they are not random. This change is trivial to convergence. The proof shows that this algorithm will converge given a random set of centroids $\mu_1, ..., \mu_k$. The non-random centroids specified by the updated algorithm $\mu_1, ..., \mu_{96}$ are a one observation subset of the possible centroids for which the algorithm will converge. Therefore, convergence still holds given this new strategy for centroid initialization.

5.6 Model Robustness

When algorithmically generating neighborhoods in a city, it is important to ensure that any results can be repeated and are robust to noise. If the results of a clustering trial vary greatly depending on the different initial sets of centroids, then those generated neighborhoods lose meaning. In other words, if a homogeneity result can be achieved in very different ways, and the model is highly sensitive to noise, then it is difficult to tell if a model truly represents better homogeneity in a city.

For this reason, the Geographic K-Means algorithm uses the strategy for initialization detailed in Section 5.4, the explanation of this algorithm. Prior to using this method for centroid initialization, centroids were initialized by randomly selecting k points in the dataset. Although this model was very flexible, the results were highly variable, and any three runs of the same model parameters could yield very different results due to the random initialization (see Figure 10 below).

Figure 10: High Variability, Lack of Robustness in Early Models



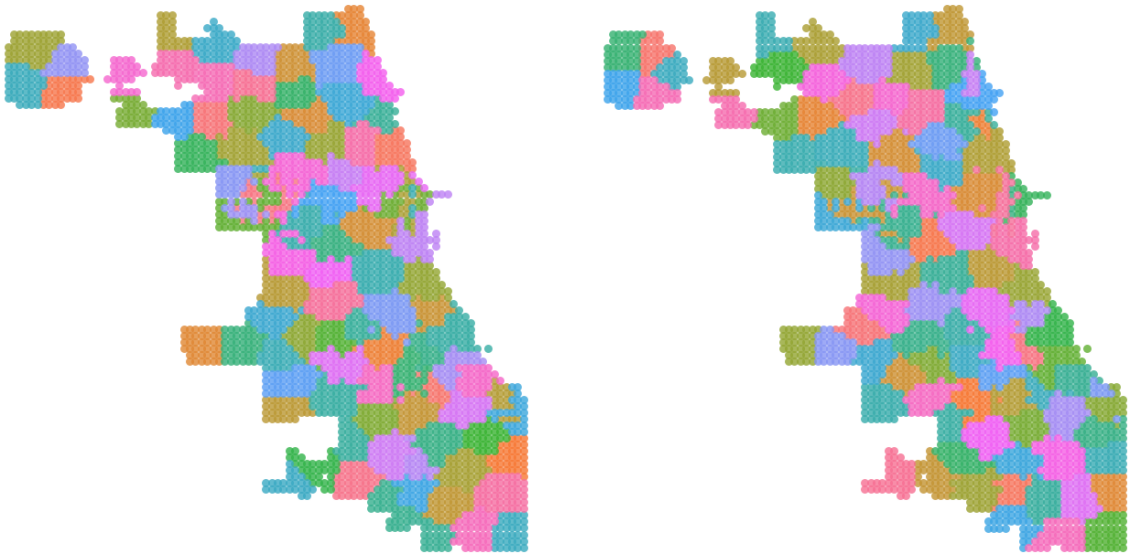
(a) Trial 1 (b) Trial 2 (c) Trial 3

These three trials had the same parameters and similar homogeneity and smoothness metrics, yet the neighborhoods boundaries are visually very different. By looking at the outcropping on the upper-left of the map (O'Hare Airport), we can see the high degree of variability. In order to make results more robust, we must update the method of centroid initialization to reduce randomness. One attempted method uses K-Means clustering to first generate evenly sized and spaced clusters from the cell locations, then calculate centers of each cluster, and set centroids to the the data points closest to these centers. Although this improved the similarity of clustering results run-to-run, the neighborhoods were still somewhat variable and sensitive to noise. Thus, the new strategy, which fixes k to 96 for Chicago and eliminates randomness was chosen. Another strength of this strategy is that it uses more data on Chicago by referencing the pre-defined neighborhoods. These neighborhoods were drawn by urban planners that understood the layout of the city, thus they do encode valuable information about the city's structure. With this new initialization technique, the initial state of the algorithm is improved, allowing the city's data to build on existing knowledge rather than starting from scratch.

A sensitivity analysis of the final centroid initialization technique reveals that the model is robust to noise. Because this is a greedy algorithm, there is no guarantee of reaching an absolute minumum solution each time, but we expect results to be similar despite noise. The following results were generated using this model, and for each centroid component x_i, we

added noise up to $\pm\frac{1}{2} * \sqrt{\text{var}(\vec{x_i})}$ where $\vec{x_i}$ represents each column of the data set.

Figure 11: Final Model Sensitivity Analysis



(a) Trial 1

(b) Trial 2

A close look at these neighborhoods shows that they are very similar, despite random noise in the initial centroids (see Figure 11). Therefore, this model is robust to noise, strengthening the significance of the results.

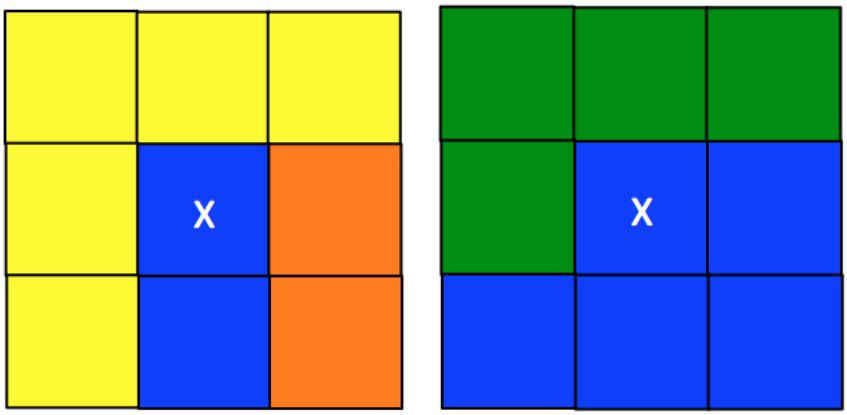
5.7 Regularization

The regularization parameter is meant to penalize clusters that do not look like neighborhoods. Thinking back to our definition of a neighborhood and looking at the Chicago neighborhood map, neighborhoods can be abnormally shaped (non-circular and non-rectangular), but they are typically not jagged and they do not contain isolated pieces.

This thesis will define a novel regularization parameter which relies on a cell's neighbors to penalize non-smoothness of neighborhood boundaries. Iterating through each cell in the grid, we can sum up individual contributions from each cell to the regularization term. In order to calculate an individual contribution, check the 8 total vertical, horizontal, and diagonal neighbors of a cell, and the regularization contribution is the number of cells that are

members of a different cluster. This means that a cell in the center of its cluster contributes 0 to the parameter, a completely isolated cell contributes 8 to the parameter, and a cell along a straight neighborhood boundary contributes 3 to the parameter. This ensures that more jagged and isolated that neighborhoods correlate to a higher regularization parameter. It also means that for higher values of k, the parameter will be greater since there are more boundaries (see Figure 12).

Figure 12: Illustrative Examples: Colored Neighborhood Identities



(a) Center Cell Contributes 7 (b) Center Cell Contributes 4

Regularization coupled with the inhomogeneity metric of Earth Mover's Distance form the key targets for optimizing a model. The goal is to minimize both inhomogeneity and the regularization parameter. When α is very small, geography dominates the clustering and we expect to end up with very uniformly shaped clusters. Inhomogeneity is high because little crime data is considered, but regularization is low because the neighborhoods are smooth. With a high value of α, geographic features dictate the clustering, leading to very homogeneous neighborhoods (low EMD) but a high regularization parameter because the neighborhoods are not contiguous. These two target metrics have opposite trends, so we have to find a range of α and k that reasonably satisfies both requirements- improved EMD over the baseline, while maintaining low regularization penalty and visually realistic neighborhoods.

5.8 Hyperparameter Optimization

For this problem, α (the weighting parameter) must be optimized to yield the best neighborhood results. As discussed in the previous subsection, we must optimize this parameter to minimize both within-cluster EMD and the regularization penalty term.

First, for each of 12 α values in the range $(0, 1)$, we cluster with this value of α using the normalized first two principal components as features. Using the results of these models, we plot α against EMD and α against the regularization parameter. Since we want to minimize both of these metrics, we visually inspect these plots and pick a range of α values that seem reasonable and well-performing.

The next step is to visually inspect the maps generated from each of these clustering models in the acceptable range and determine which is the most homogeneous without violating the mandate that neighborhoods must be contiguous. Although the regularization parameter helps to measure this, the human eye is the best judge for this target (with the exception of machine vision algorithms that could handle this problem).

Finally, if the final chosen model necessitates smoothing, we can then smooth the clusters by detecting outliers (cells that contribute 7 or 8 to the regularization parameter), and reassigning them to one of the neighboring clusters that is most similar.

6 Results and Discussion

The results of the previously described models are analyzed by looking at homogeneity of clusters for each different method, as well as visual smoothness and the regularization value. The results are presented and explained in this section.

6.1 Choosing the Evaluation Metrics

In testing to select an evaluation metric for homogeneity, Standard Deviation (SD) and Earth Mover's Distance (EMD) were considered, as well as an implementation of Shannon Entropy [Shannon, 1948] for measuring within-neighborhood information. The Shannon Entropy implementation did not follow all expected trends in testing, so it was dropped as a possible method. Standard Deviation and EMD, however, followed expectations and both proved to be useful measures of homogeneity.

The following histograms (Figure 13) show the aggregated crime counts in each of 3 different neighborhoods of relatively similar size, and illustrate the functionality of SD and EMD.

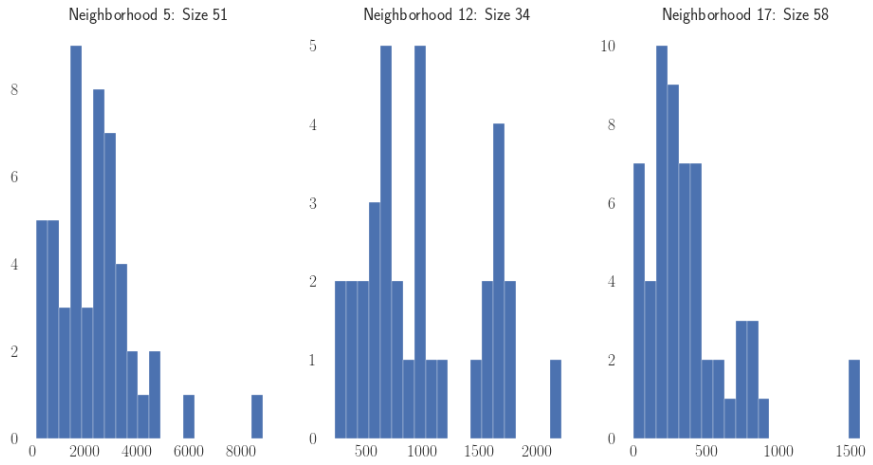


Figure 13: Sample Neighborhoods to Illustrate SD and EMD

The results of the homogeneity calculations for this neighborhoods are as follows:

	Std Dev	EMD
Neighborhood 5	1525	1082
Neighborhood 12	515	415
Neighborhood 17	313	206

Both metrics penalize neighborhood 5 for having the widest distribution. Another key attribute is the punishment of multi-modality. Neighborhoods 12 and 17 are on a similar scale of crime magnitude, but neighborhood 12 has a multi-modal distribution with three large peaks. Both metrics penalize this attribute of the distribution. EMD penalizes multi-modality more heavily, as illustrated by the fact that neighborhood 12's EMD is more than double that of neighborhood 17. Therefore, since both metrics are very fast to calculate and EMD has all of the desired attributes of a homogeneity metric, EMD will be considered as the main homogeneity metric throughout this thesis.

The other choice we need to make is whether to use mean or median to represent the vector of within-cluster EMDs (or SDs). When calculating baselines for this thesis on Chicago's Neighborhoods, we generate the following histograms (Figure 14 below) from the calculated vectors of the within-neighborhood EMD and SD.

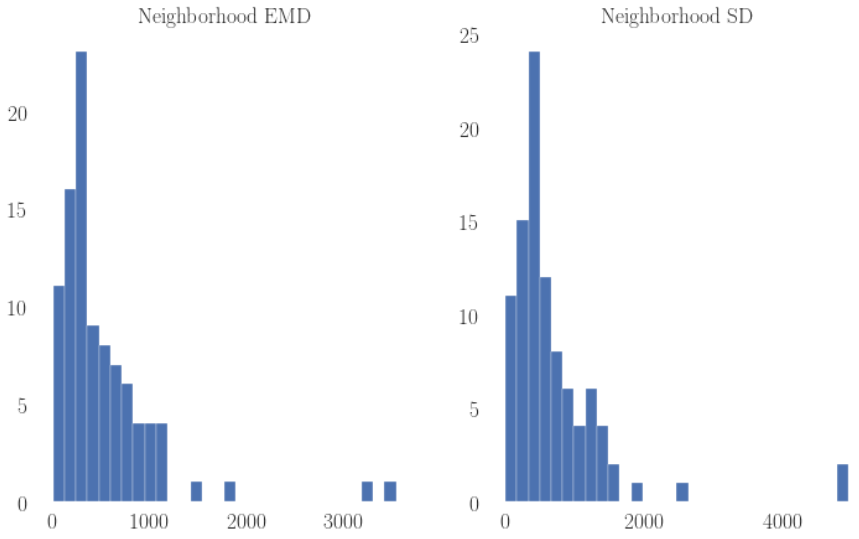


Figure 14: Histograms of EMD and SD in Chicago Neighborhoods

These distributions look very similar, illustrating the fact that they both measure the

same phenomenon, and both are useful. We notice also that these distributions have heavy right tails and a few neighborhoods with big outlier values. For that reason, I believe that a calculation of mean will be skewed higher than it should be, so median is a better measure of the homogeneity.

6.2 Baseline Model

For this project, the baseline is defined as the homogeneity of neighborhoods that are pre-defined by the City of Chicago. The Standard Deviation of crime counts within the neighborhood and Earth Mover's Distance are applied to the clusters to obtain metrics of homogeneity. For this baseline model, we observe the following results, which are medians over all neighborhoods:

	Std Dev	**EMD**
Neighborhoods	464.9	348.5
Police Beats	530.2	402.6

Additionally, Figure 14 shows a histogram of the within cluster EMD and SD for both neighborhood clusters. Figure 15 is the same plot for police beats:

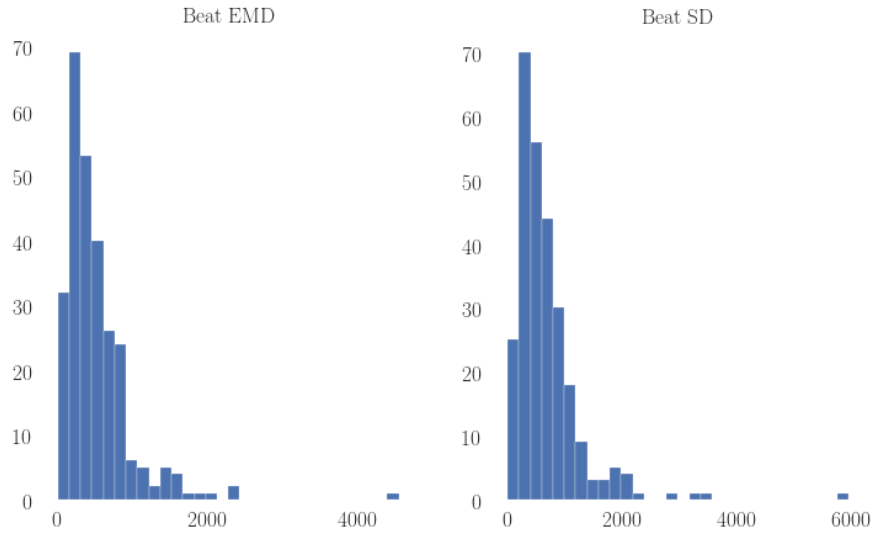


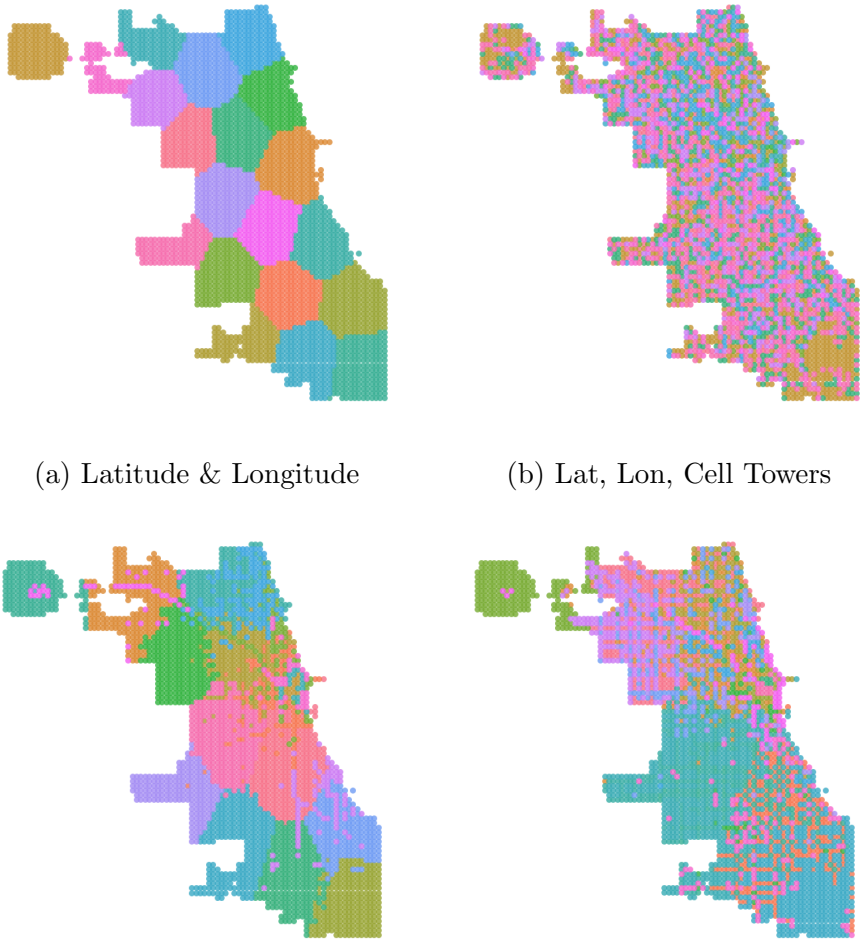
Figure 15: Histograms of EMD and SD in Chicago Police Beats

Again, the distribution of Earth Mover's Distance and Standard Deviation are very similar. For police beats, these distributions seem to be slightly less normal than the neighborhood baselines. In addition, the baseline model needs to be one that we can use to compare to future models, and it seems that $k \sim 260$ for police beats is too large to compare to clustering models. Instead, $k = 96$ neighborhoods is a reasonable number of clusters to use as a point of comparison. For these reasons, neighborhoods are selected as the baseline that will be used for comparison throughout modeling.

6.3 Naive K-Means Results

Tests with `sklearn`'s K-Means clustering functionality provided initial insight for how to build the geographic clustering method. The following preliminary modeled neighborhood maps (Figure 16) were generated with this method with $k = 20$.

Figure 16: Results of Naive K-Means Trials on Multiple Subsets of Data



(a) Latitude & Longitude

(b) Lat, Lon, Cell Towers

(c) Norm. Lat, Lon, Cell Towers

(d) Norm. with more Features

In the first model (a), K-Means was run using only the latitude and longitude of each cell as features. As expected, this generates very uniformly sized and evenly distributed neighborhoods. In the second model (b), latitude, longitude, and cell towers were used and the map is highly chaotic. In the third model (c), latitude, longitude, and cell towers were again used but they were normalized prior to clustering. In this model, we see that the cell towers manipulate the clusters so that they are not just uniform blobs, but more interesting neighborhood shapes. The final model (d) again normalizes the features, but uses more features like restaurants, bus stops, and parking lots (all taken from the OpenStreetMap data). This model is again very chaotic, with little resemblance of neighborhoods. This

exercise illustrates that balance of the geography and the spatial features is challenging and important. In model (a), geography is weighted too heavily, model (b) weights features too heavily, and model (c) weights them both better than the others. Normalization puts features on the same scale, evening out their effect, but in model (d), the four features overpower latitude and longitude in absolute scale and lead to poor results. This leads to the conclusion that geographic distance and feature distance must be handled separately in a Geographic K-Means algorithm with a weighting factor. Normalization improves the initial scale of our data features, which enables more precise tuning of the weighting parameter.

6.4 Preliminary Geographic K-Means Results

To start, some preliminary tests were run using the Geographic K-Means function to fix errors and profile the code so that we could speed up the algorithm. These crucial speed improvements allowed this model to be scaled and optimized without significant time issues, so this preliminary testing played an important role in the development of this project. Although this clustering map, shown in Figure 17, is naive and based on only latitude and longitude-based Haversine distance, it shows $k = 96$ relatively uniform clusters.

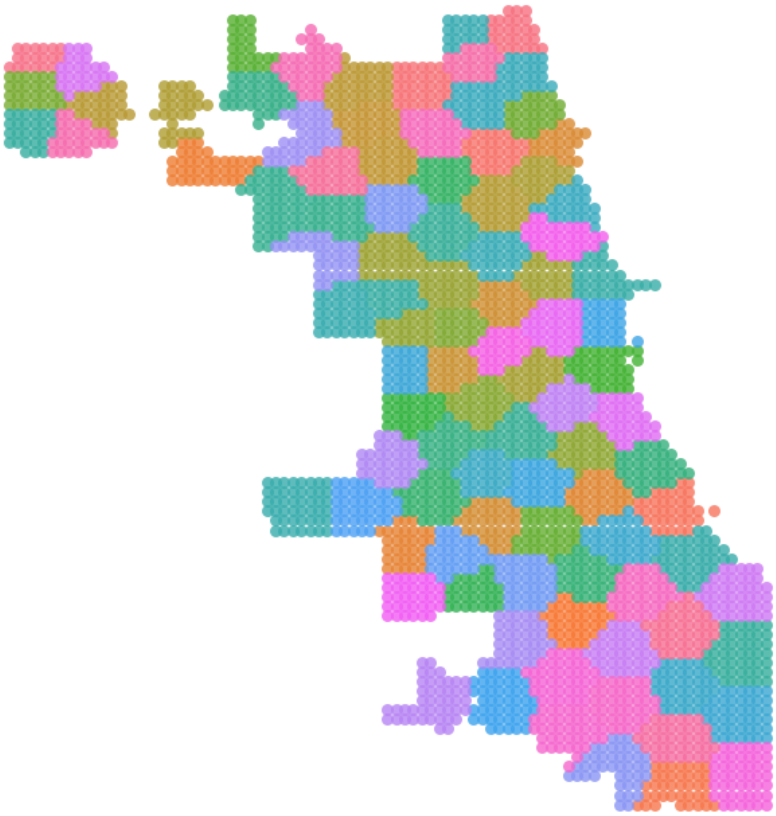


Figure 17: Clustering with Haversine Distance Only

These results look similar to the first K-Means test, with larger k. The neighborhoods are smooth and uniform. The **median EMD for this model is 402** and **median SD is 526**, so this model performs worse (i.e. less homogeneous) than the baseline model. This makes sense because city neighborhood boundaries encode some information about the surrounding area since city governments have an understanding of the city's layout, while this model essentially represents randomness, with the benefit of geographically close areas being clustered together into contiguous neighborhoods. In Geographic K-Means, we leverage the knowledge that cities use to build these neighborhoods by initializing our centroids to the center of the pre-defined neighborhoods.

6.5 Feature Based Clustering Results

Once this testing was complete and the algorithm was running fast enough to handle large trials, more features were introduced so that the algorithm handled the entire distance metric, $(1 - \alpha) * H + \alpha * F$ where H is Haversine distance and F is vector distance of geographically accumulated features. In these examples, we use 'PCA 1' and 'PCA 2', the first two components of the PCA analysis (which explain nearly 99% of the data's variance), as the features. These are extremely informational vectors about crime and data trends across the city, and they together capture the major trends that are detectable in Chicago's public data. All of these models use $k = 96$ due to the selection of initial centroids from Chicago's pre-defined neighborhoods, and this also makes it easier to compare performance with the initial baseline model. We also use $\alpha = 0.5$ for simplicity (even weighting betwen H and F).

Trial 1: Geographic K-Means on Latitude, Longitude, Principal Component #1 and Principal Component #2

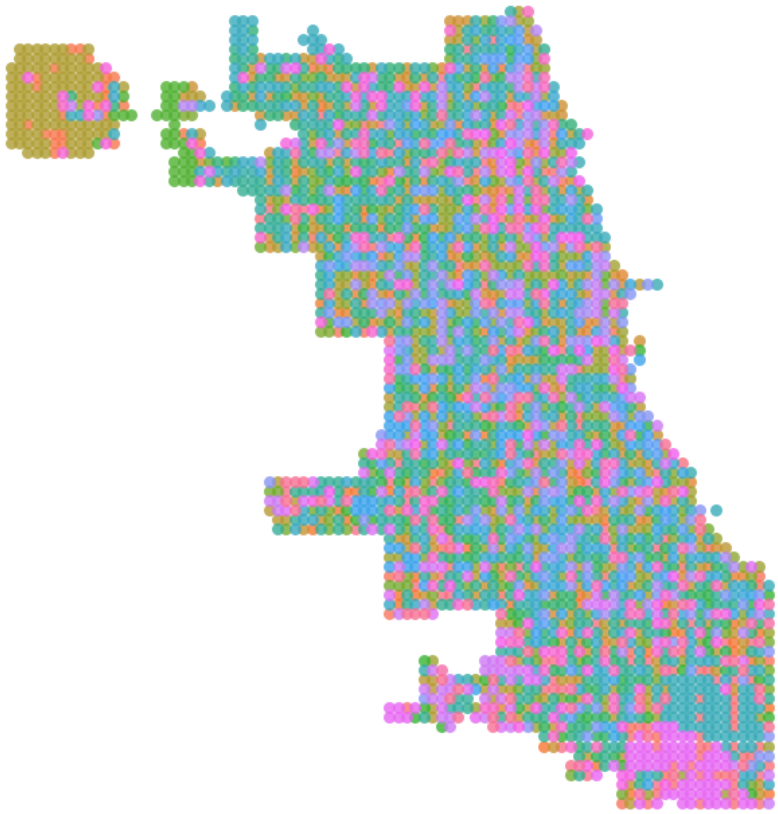


Figure 18: Simple Geographic K-Means Trial

This model has **median EMD of 6.7** and **median SD of 20.8**. This is a vast improvement over the baseline and random geographic model.

The map looks chaotic, and given the extreme homogeneity of this model, it seems that the feature data overwhelmed geography in this model. While the EMD and SD metrics look very good, this model does not represent realistic neighborhoods, and we need to better balance H and F. In this model, the Regularization parameter would be extremely large. We will use normalization and α tuning to find this balance.

Trial 2: Geographic K-Means on Latitude, Longitude, Principal Component #1 and Principal Component #2 with all Features Normalized

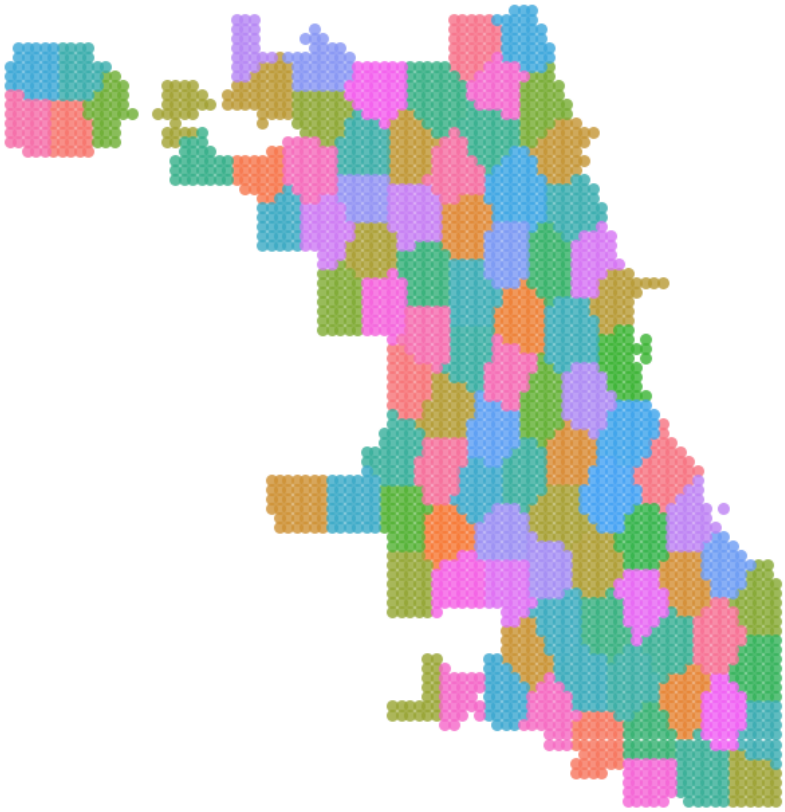


Figure 19: Normalized Geographic K-Means

This model has **median EMD of 387** and **median SD of 495**. This is significantly worse than the baseline model and slightly better than pure Geographic K-Means with no features.

In this model, all features (including latitude and longitude) are normalized prior to running clustering. The clusters are very uniform and smooth, illustrating that geography was weighted more heavily than features. The Haversine distance on normalized latitude and longitude clearly influences the model more than the normalized feature vector distances. Because latitude and longitude are handled differently than other features in this algorithm's distance function, it doesn't make intuitive sense to normalize everything. In addition, Haversine distance is implemented specifically to handle latitude and longitude projections, so manipulating these features prevents Haversine distance from having its desired effect of enforcing geographic accuracy.

Trial 3: Geographic K-Means on Lat, Lon, PCA Component #1 and PCA Component #2 with PCA Features Normalized

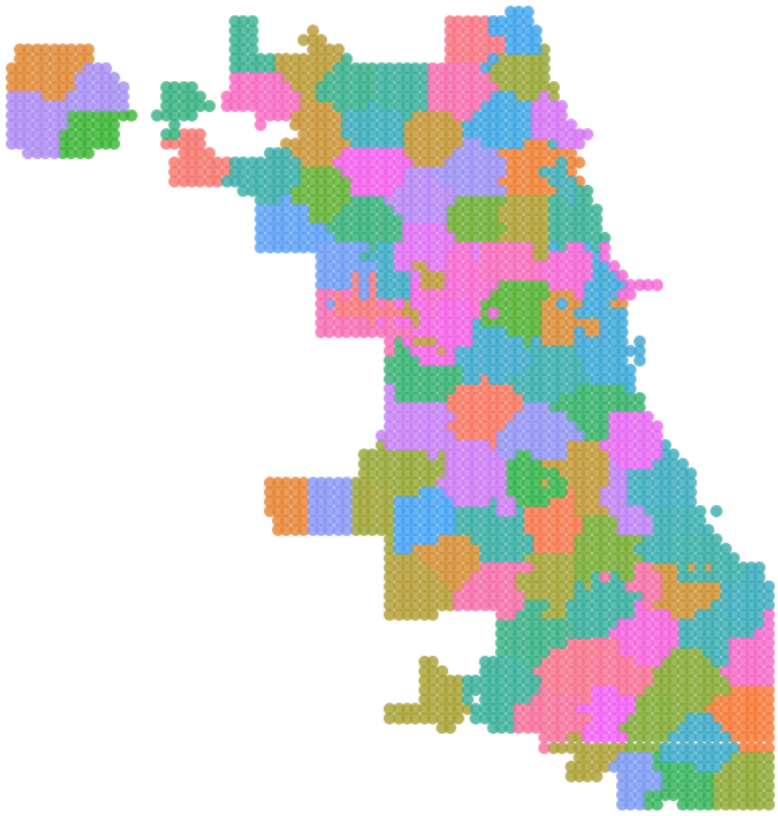


Figure 20: Geographic K-Means with all Features but Lat/Lon Normalized

This model has **median EMD of 300** and **median SD of 405**. This is an improvement over the baseline model.

This model normalizes all features except for latitude and longitude, and the results are better than any previous model. The clusters are more homogeneous than the baseline model, and we can observe visually that the neighborhood boundaries are interesting and non-uniform, but still represent mostly realistic neighborhoods. There is still work to do to tune α to get the best model, but this model seems to achieve the goal of using normalization to find a better initial balance between geography and accumulated features so that tuning is more effective.

6.6 Picking Optimal Alpha

Now, to optimize α, we generate models for many α values between 0 and 1, and plot the regularization penalty and Earth Mover's Distance to determine a range of possible values (see Figure 21 below).

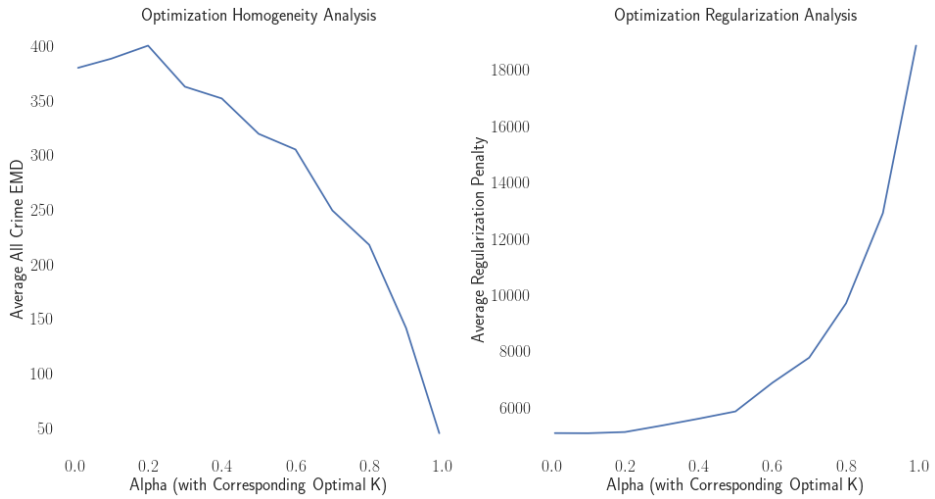


Figure 21: EMD and Regularization Penalty for α with Optimal k Value

We want to minimize both EMD and regularization, so there is a good range of possible α values in the middle of these plots. Based on this plot, the acceptable range is 0.3-0.7. With α below 0.3, EMD is very high so these models do not improve over the baseline model. With α above 0.7, regularization penalty skyrockets, and the corresponding maps start to become very random and disobey the rules of contiguous neighborhoods. Looking at the maps, it seems that $\alpha = 0.6$ is a good map to choose because it improves significantly over the baseline, and the neighborhoods are realistic. Figure 22 illustrates the map for this chosen model.

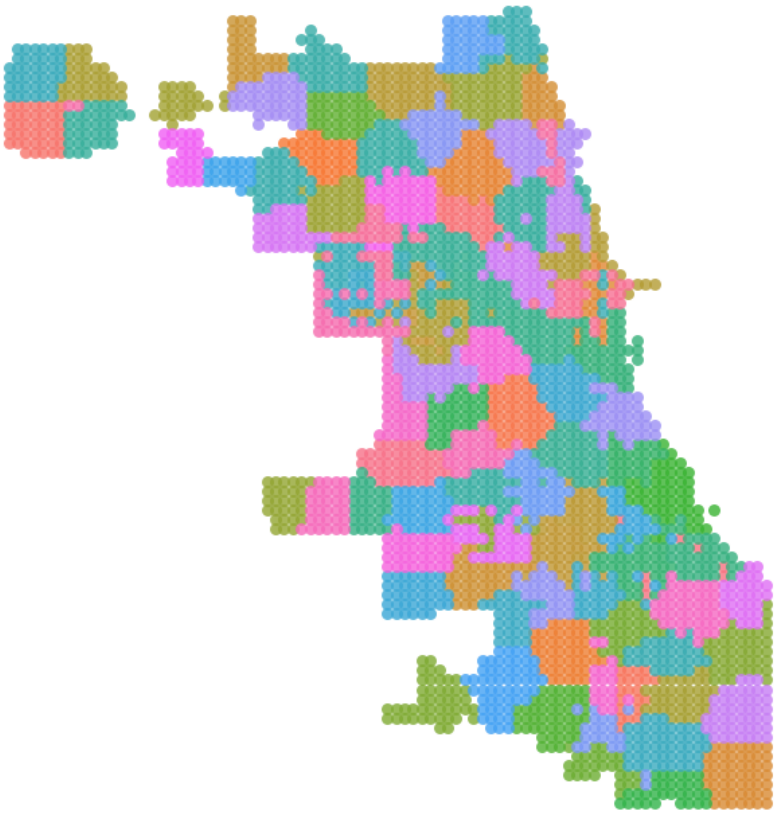


Figure 22: Clustered Neighborhoods for α=0.6, $k = 96$

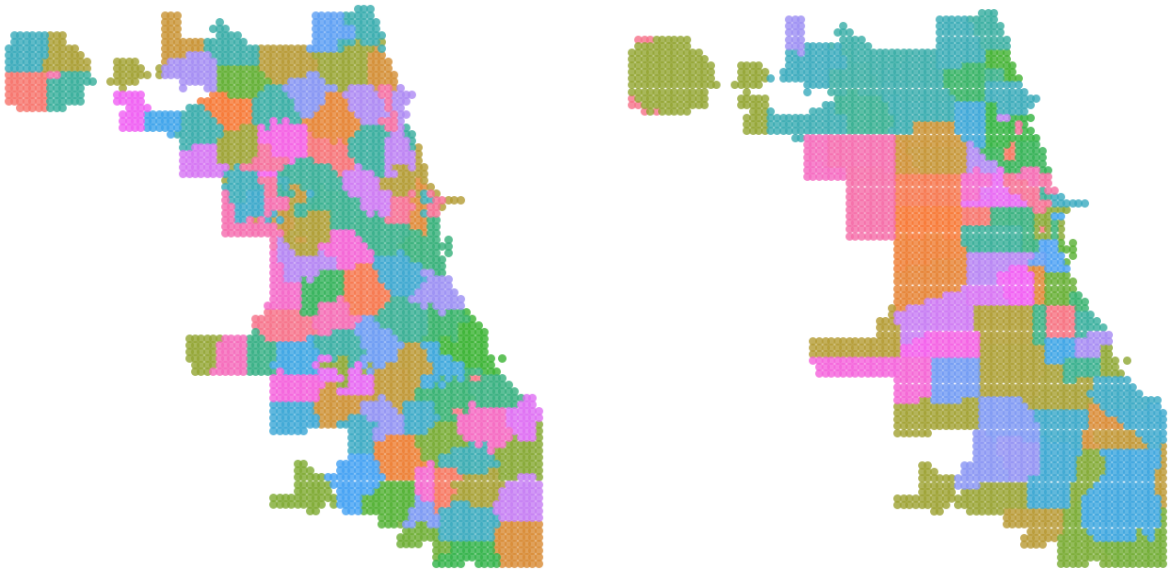
This model has **median EMD of 296** and **median SD of 386**. This is a significant improvement over the baseline model, which has median EMD of 349 and median SD of 465.

6.7 Boundary Cleaning

Once this final model is chosen, there is one last step to re-assign any outliers on the map so that the clusters represent real neighborhoods better. This process uses code very similar to the regularization step, since outliers are defined as points that contribute 7 or 8 to the regularization parameter, and these points are re-assigned to the most similar neighboring cluster.

Running this code on the model with $\alpha = 0.6$ and $k = 96$, we see that 97 of the 3177 cells are re-assigned, and the resulting map in Figure 23 (a) is generated.

Figure 23: Final Model Compared to Baseline Model



(a) Final Smoothed Clustering Model

(b) Initial Baseline Neighborhoods

This model has **median EMD of 305** and **median SD of 407**. This is still a significant improvement over the baseline model, which has median EMD of 349 and median SD of 465. This is a 12.6% EMD improvement over the baseline model and a 12.5% SD improvement over the baseline. This model also performs only slightly worse than the model pre-smoothing neighborhoods. The regularization parameter drops from 6764 pre-smoothing to 5964 post-smoothing, so it is clear that this step makes a significant impact on smoothness. Visually, we can see that many outliers are clustered better and the neighborhoods are more contiguous and less jagged. This smoothing step helps to enforce the rules of neighborhoods that we defined, and this final model represents a useful set of neighborhoods based on Chicago crime data that are more internally homogeneous than the neighborhoods that the City of Chicago defines. Although the neighborhoods look very different from Chicago's initial neighborhoods, which have higher variability of size, we can see that many different neighborhoods in the new model appear to be subset of pre-existing neighborhoods, and some of the smaller Chicago neighborhoods are brought together into one larger neighborhood in the new model. There are advantages and disadvantages of each model, but the spatial even-

ness and improved data-based homogeneity of the Geographic K-Means clustering model is compelling as a strong model that can be used to update Chicago's neighborhoods.

7 Conclusion

7.1 Insights

This thesis attempts to create a scientific process for re-drawing neighborhood boundaries in urban areas. More meaningful neighborhood design can help to facilitate more significant and insightful neighborhood-based studies and can help to differentiate a city into its key areas. This can then provide a better understanding of cities and the driving factors of urban phenomena. An improved understanding can help city governments, police departments, and businesses make better decisions to drive change. The goal of this thesis is to provide a template and methodology for how to use data to generate these neighborhoods. Although the Chicago example is illustrative and powerful because of the quality of data in the city, the template and the code for this thesis could easily be extended to another city. Another data scientist could follow the steps for data processing and modeling, and run a completely new model in a new place, with different data for clustering and different objectives. As such, this thesis may empower data scientists and researchers to use a new unique tool for urban neighborhood analysis, and in the future, I encourage others to build off of this algorithm to make it even more powerful. It seems very possible that this work, or variants on this subject, could be used to make policy and business decisions, and improve conditions in real cities. This algorithm can enable city governments to better understand and respond to city-wide trends, such as rising violent crime or shifting demographics in neighborhoods. There are also a multitude of possible applications of the algorithm outside of urban planning. Some compelling applications include using voter data to generate optimal gerrymandering districts, using consumer data to better target customers with ads throughout a city, and using real estate data to find boundaries in housing markets in a city.

This project was replete with many interesting challenges and surprises. The central challenge is the trade-off between boundary smoothness and maximized homogeneity in the process of neighborhood generation. Maximizing homogeneity is simple and requires no geographic input, but it results in nonsensical neighborhoods with no contiguity. Similarly, generating smooth and uniform neighborhoods is a simple task that can be achieved with

a naive K-Means clustering model on only latitude and longitude grid locations. Neither of these alone are sufficient to generate a successful model of urban neighborhoods. To navigate this trade-off and find a working solution, we tuned the α weighting parameter and utilized a regularization term to monitor smoothness. The final neighborhood model contained more homogeneous neighborhoods than those that the City of Chicago provides for our baseline, and the neighborhood shapes are interesting and non-standard, as neighborhoods are in reality. Although it is difficult to test if the neighborhoods truly represent internal social uniformity, the crime metrics suggests that this algorithm returns more meaningfully homogeneous regions. Another goal of the model was to generate reproducible and robust results, which was a challenge because of the random nature of K-Means clustering. Although the final neighborhood-based centroid initialization is imperfect because it limits flexibility of k, it creates consistent and high-performing results, especially on the final clustering model. This final model achieves the goals initially set out in this thesis and illustrates the possibility for successful extension to other cities and other objectives. My hope is that these results inspire others to continue building and improving this algorithm, and use it to as a powerful tool for urban geospatial analysis.

7.2 Future Work

Although the objectives of this thesis were met, there were many interesting concepts that arose throughout the process that suggested new areas to be considered for future work.

7.2.1 Frontend Visualization Platform

One piece of software that could be built to better illustrate the results of this work is an interactive front end platform to deploy and visualize the clustering models built by this algorithm. I envision a map-based web app with a backend database containing pre-processed grids for one or several cities. The web app would contain an options panel where a user can make model selections including city, grid size, α for weighting, number of neighborhoods k, and accumulated features to use, then deploy the model. The backend would query for the correct data and run the algorithm to generate clusters (speeding up the implementation

of the algorithm would be an important step), then render the results on the map. This would allow users without data science skills to utilize this algorithm for the purposes of urban planning or trying to understand the city. It would also allow even more testing of combinations of hyperparameters and feature sets that were not attempted in this thesis.

7.2.2 Further Clustering Testing

There are also aspects of the algorithm that could be improved in future works. While this algorithm is based on K-Means clustering and alters the distance functions, there are many other clustering algorithms that could be altered to perform the same neighborhood clustering. K-Means (a partition clustering algorithm) was used because it is easy to interpret, allowing for very intuitive hyperparameter optimization and successful progression through models. However, other partition clustering algorithms could be used, and hierarchical clustering would also be an interesting method to try to use for modeling this problem.

7.2.3 Allowing Flexibility for K

One drawback of the final selected model is that it relies on the city's pre-defined neighborhoods in order to initialize the centroids for clustering. For Chicago, this sets $k = 96$ immediately, with no attempt to optimize k for homogeneity and smoothness. While this works in this example, and the method for initializing centroids strengthens robustness and model intuition, it would be beneficial if future work could address how to extend this model to allow for better tuning the parameter k.

In most clustering problems, tuning k is an important step. Two important methods for optimizing the number of clusters in a clustering algorithm are the elbow plot and the silhouette method. To generate an elbow plot, run the clustering algorithm many times for different values of k and plot the value of k against the sum squared error of each data point from its centroid in the final converged state [Kodinariya and Makwana, 2013]. In this plot, we look for an "elbow", where the rate of decrease declines, indicating that the marginal benefit of raising the value of k is not significant. We select k to be the point at which this "elbow" occurs. The silhouette method is another way to select an optimal value of k. For each point in the converged clusters, we calculate a silhouette score between $[-1, 1]$ which

represents how similar the data point is to its own cluster as compared to others [Rousseeuw, 1987]. A high value indicates a well-clustered point, so we pick the value of k that maximizes the average silhouette score across clusters.

In order to allow for this tuning of k, there must be another way to initialize centroids that does not rely on the baseline neighborhood boundaries. The new method must maintain robustness, yet allow for any number of clusters. One method attempted in this thesis is to use K-Means clustering on geographic data and select initial centroids to be the center of these clusters. This method is discussed in Section 5.6 and is a possibility for further testing in the future. Other possibilities include using the neighborhood based initialization and randomly adding or subtracting centroids to achieve the desired k. More work must be done to find the best way to enforce model robustness and use k as a parameter to improve results and optimization.

7.2.4 Cellular Potts Model

This modeling question can specifically be re-worked using the Cellular Potts Model, which has many similar goals to this algorithm. The Cellular Potts model comes from computational biology and addresses the interaction and behavior of cellular organisms. Graner and Glazier first proposed the model in 1992 to simulate how different types of cells would arrange themselves when mixed [Graner and Glazier, 1992]. This approach represents 2-dimensional space with a discrete grid, like our grid-based approach. Grid cells can be classified to a particular organism, which makes up a contiguous cluster of cells. The algorithm uses a random simulation to find a greedy lowest energy configuration of the cells. In this case, energy is calculated by a Hamiltonian operator that includes adhesion energy between organisms and volume constraints. Then, we randomly select grid cells and attempt to change the classification of the cell, and only accept the change if the resulting energy state (based on the Hamiltonian) is lower than the previous state. To read more on current Cellular Potts modeling or run your own simulations, look at CompuCell3D's Documenation.

Though this model comes from a field very different from geospatial analysis, the grid-based algorithm has clear applications. This algorithm could be used to find an approximate lowest energy configuration for neighborhood classifications on the city's grid. In order to

successfully use this algorithm, a Hamiltonian operator would have to be carefully designed to support the goals of the algorithm. The Hamiltonian would have to include some measure of homogeneity, such as EMD, as well as smoothness, like the regularization term. The Hamiltonian could calculate a weighted average of these terms, and perhaps others, so that the random steps attempt to maximize both contiguity and homogeneity.

Once a Hamiltonian is designed, it would be easy to run random simulations to find a greedy optimal grid configuration. This algorithm could be run on the initial neighborhoods defined by the City of Chicago in order to improve upon this initial state. Another possibility is to run Geographic K-Means clustering and use the Cellular Potts model to improve this result, like a smoothing step. This model could be very beneficial to improving the procedure for re-drawing neighborhood boundaries, and future work is needed to implement and test its functionality.

7.2.5 Data Processing

In addition, the clustering and neighborhood building could be improved with alterations to the data and data-processing. Given the sparsity and significant variance of some of the data features, it would be valuable to attempt data smoothing over the grid. For example, each cell's crime counts and features could be averaged with the 8 surrounding cells, or a weighted average could be used, so that the distribution of features is smoother. By using these smoothed features, the neighborhoods generated might better represent reality, since outliers which can detract from the model would be less prominent. The data could also be accumulated onto a grid with smaller cells. The cells used in this thesis are approximately 300m x 300m, which is fairly small, but even smaller cells would provide a more precise model. In addition, the neighborhood boundaries would be smoother, since the large cells often result in a more jagged appearance. This would be much more computationally expensive in both pre-processing and modeling, but by running the code with smaller cells on a powerful server, and using parallelization to speed up code, it is possible. The neighborhood analysis would be even more precise with smooth data and smaller units of area.

8 Acknowledgements

Thank you so much to my advisor, Dr. Pavlos Protopapas, for guiding me through this process and providing insight in every step of this project. Thank you for supporting me in pursuing a topic that excites me and for many brainstorming sessions in which we imagined different ways to approach and solve this problem. Your creativity and expertise were crucial for helping me define my problem and advance past difficulties.

Thank you to Professor Chris Rycroft for inspiring my interest in Applied Mathematics and modeling through AM50 during my Freshman year of college. This course completely changed my interests at a time when I was still considering pre-med. Your advising through the years has kept me on track, and enabled me to achieve my academic goals. Finally, thank you for encouraging me to pursue a Senior Thesis as a culmination of my academic experience.

A big thank you also to James Dreben and the team at Zoba Inc., where I first became interested in geospatial data analytics. Your willingness to teach me and allow me to contribute early in my education was crucial to my development and sparked my passion for data science. Being a member of Zobas team has been such a meaningful personal and academic experience which has inspired me to further my knowledge and contribute tools to the spatial analytics field.

Finally, thank you to my family and friends who encouraged and supported me throughout this process.

References

[Agrawal et al., 1998] Agrawal, R., Gehrke, J., Gunopulos, D., and Raghavan, P. (1998). Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, SIGMOD '98, pages 94–105, New York, NY, USA. ACM.

[Andresen and Mallseon, 2013] Andresen, M. and Mallseon, N. (2013). Spatial heterogeneity in crime analysis. In *Crime Modeling and Mapping Using Geospatial Technologies*, pages 3–24. Springer Netherlands : Imprint: Springer.

[Bogomolov et al., 2014] Bogomolov, A., Lepri, B., Staiano, J., Oliver, N., Pianesi, F., and Pentland, A. (2014). Once upon a crime: Towards crime prediction from demographics and mobile data.

[Bottou and Bengio, 1995] Bottou, L. and Bengio, Y. (1995). Convergence properties of the k-means algorithms. In *Advances in neural information processing systems*, pages 585–592.

[Claesen and De Moor, 2015] Claesen, M. and De Moor, B. (2015). Hyperparameter search in machine learning. *arXiv preprint arXiv:1502.02127*.

[Claffey and Lilia, 2018] Claffey, M. and Lilia, C. (2018). City of Chicago pilot project for dockless bike rental starts today to serve residents in far south side.

[Coulton et al., 2011] Coulton, C., Chan, T., and Mikelbank, K. (2011). Finding place in community change initiatives: Using gis to uncover resident perceptions of their neighborhoods. *Journal of Community Practice*, 19(1):10–28.

[Delmelle, 2017] Delmelle, E. C. (2017). Differentiating pathways of neighborhood change in 50 us metropolitan areas. *Environment and planning A*, 49(10):2402–2424.

[Dhanachandra et al., 2015] Dhanachandra, N., Manglem, K., and Chanu, Y. J. (2015). Image segmentation using k-means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science*, 54:764–771.

[Ester et al., 1996] Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise.

[Gaston and Brunson, 2018] Gaston, S. and Brunson, R. K. (2018). Reasonable suspicion in the eye of the beholder: Routine policing in racially different disadvantaged neighborhoods.

[Gellert, W. et al., 1989] Gellert, W., Gottwald, S., Hellwich, M., Kastner, H., and Kustner, H. (1989). Haversine distance.

[Goldstein, 2013] Goldstein, B. (2013). Open data in Chicago: Game on. In *Beyond Transparency: Open Data and the Future of Civic Innovation*. Code for America Press.

[Graner and Glazier, 1992] Graner, F. and Glazier, J. A. (1992). Simulation of biological cell sorting using a two-dimensional extended potts model. *Physical review letters*, 69(13):2013.

[Groff, 2013] Groff, E. (2013). Measuring a place‘s exposure to facilities using geoprocessing models: An illustration using drinking places and crime. In *Crime Modeling and Mapping Using Geospatial Technologies*, pages 269–298. Springer Netherlands : Imprint: Springer.

[Herrmann, 2013] Herrmann, C. (2013). Street-level spatiotemporal crime analysis: Examples from Bronx County, NY (20062010). In *Crime Modeling and Mapping Using Geospatial Technologies*, pages 73–104. Springer Netherlands : Imprint: Springer.

[Hoerl and Kennard, 2000] Hoerl, A. E. and Kennard, R. W. (2000). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 42(1):80–86.

[Hotelling, 1992] Hotelling, H. (1992). Relations between two sets of variates. In *Breakthroughs in statistics*, pages 162–190. Springer.

[Hu et al., 2018] Hu, Y., Wang, F., Guin, C., and Zhu, H. (2018). A spatio-temporal kernel density estimation framework for predictive crime hotspot mapping and evaluation. 99:89–97.

[Jain, Anil and Dubes, Richard, 1988] Jain, Anil and Dubes, Richard (1988). *Algorithms for Clustering Data*. Prentice Hall.

[Jolliffe, 2002] Jolliffe, I. T. (2002). *Principal component analysis*. Springer series in statistics. Springer, New York, 2nd ed. edition.

[Kodinariya and Makwana, 2013] Kodinariya, T. M. and Makwana, P. R. (2013). Review on determining number of cluster in k-means clustering. *International Journal*, 1(6):90–95.

[Lavin et al., 1998] Lavin, Rajesh Batra, and Lambertus Hesselink (1998). Feature comparisons of vector fields using earth mover's distance.

[Le Falher et al., 2015] Le Falher, G., Gionis, A., and Mathioudakis, M. (2015). Where is the Soho of Rome? measures and algorithms for finding similar neighborhoods in cities. In *Ninth International AAAI Conference on Web and Social Media*.

[Lin et al., 2008] Lin, N. P., Chang, C.-I., Chueh, H.-E., Chen, H.-J., Hao, W.-H., et al. (2008). A deflected grid-based algorithm for clustering analysis. *WSEAS Transactions on Computers*, 7(4):125–132.

[Midgley, 2009] Midgley, P. (2009). The role of smart bike-sharing systems.

[Murray and Grubesic, 2013] Murray, A. and Grubesic, T. (2013). Exploring spatial patterns of crime using non-hierarchical cluster analysis. In *Crime Modeling and Mapping Using Geospatial Technologies*, pages 105–124. Springer Netherlands : Imprint: Springer.

[Ng, R.T. and Han, Jiawei, 2002] Ng, R.T. and Han, Jiawei (2002). CLARANS: A method for clustering objects for spatial data mining. 14(5):1003–1016.

[O'Sullivan, 2009] O'Sullivan, D. (2009). Changing neighborhoods: A framework for spatially explicit agent-based models of social systems. *Sociological Methods & Research*, 37(4):498–530.

[Pele and Werman, 2009] Pele, O. and Werman, M. (2009). Fast and robust earth mover's distances. In *2009 IEEE 12th International Conference on Computer Vision*, pages 460–467.

[Perry et al., 2013] Perry, W., McInnis, B., Price, C., Smith, S., and Hollywood, J. (2013). *Predictive policing the role of crime forecasting in law enforcement operations.* RAND Corp.

[Rebbapragada et al., 2009] Rebbapragada, U., Protopapas, P., Brodley, C., and Alcock, C. (2009). Finding anomalous periodic time series. *Machine Learning*, 74(3):281–313.

[Rousseeuw, 1987] Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

[Schikuta and Erhart, 1997] Schikuta, E. and Erhart, M. (1997). The bang-clustering system: Grid-based data analysis. In *International Symposium on Intelligent Data Analysis*, pages 513–524. Springer.

[Shannon, 1948] Shannon, C. E. (1948). A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423.

[Shapiro, 2017] Shapiro, A. (2017). Reform predictive policing. 541(7638).

[Sheikholeslami et al., 2000] Sheikholeslami, G., Chatterjee, S., and Zhang, A. (2000). Wavecluster: A wavelet-based clustering approach for spatial data in very large databases. *The VLDB Journal*, 8(3-4):289–304.

[Spielman and Logan, 2012] Spielman, S. E. and Logan, J. R. (2012). Using high-resolution population data to identify neighborhoods and establish their boundaries. *Annals of the Association of American Geographers*, 103(1).

[Tibshirani, 1996] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288.

[University, 1999] University, S. (1999). The earth mover's distance (EMD).

[Vincenty, 1975] Vincenty, T. (1975). Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. *Survey Review*, 23(176):88–93.

[Wang et al., 2018] Wang, Q., Phillips, N. E., Small, M. L., and Sampson, R. J. (2018). Urban mobility and neighborhood isolation in America's 50 largest cities. 115(30).

[Wang, Wei et al., 1997] Wang, Wei, Jiong Yang, and Richard Muntz (1997). STING : A statistical information grid approach to spatial data mining.

[Woodruff, 2012] Woodruff, A. (2012). Crowdsourced neighborhood boundaries, part one: Consensus.

[Yuan et al., 2012] Yuan, J., Zheng, Y., and Xie, X. (2012). Discovering regions of different functions in a city using human mobility and POIs. In *Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining*, KDD '12, pages 186–194. ACM.

[Zhang et al., 2013] Zhang, A. X., Noulas, A., Scellato, S., and Mascolo, C. (2013). Hoodsquare: Modeling and recommending neighborhoods in location-based social networks. In *2013 International Conference on Social Computing*, pages 69–74. IEEE.