Impact of Demographics and City Investment on Median House Price Trends in Chicago, 2010-2015

Quantitative Methods

MSc in Smart Cities and Urban Analytics

Center for Advanced Spatial Analytics

University College London

Municipal governments attempt to spend their money in a way that will produce the best outcomes for the communities they serve. In this paper the author introduces a method of using K-means clustering as a tool for exploratory data analysis when assessing the impact of public investments on community revitalization. The clustering method will be used to determine whether demographic characteristics of a neighborhood or public and private investment strategies are better predictors of median house price trends in Chicago, IL during the years 2010-2015.

Background

Assessing the cost-benefit trade-off for the investment of public funds requires the identification of a goal or desired outcome (Maass, 1966). House prices are commonly used to assess the desirability of living in an area (Ding et al., 2000; Galster et al., 2006; Wachter, 2004), and will be employed in this study to determine whether a neighborhood is considered to be improving or declining.

Investment in improving the built environment can improve property prices, and these impacts will be greater for property values nearby the investment (Ding et al., 2000; Simons et al., 1998). It has also been noted that the number of businesses can impact house prices in a region (Ding and Knaap, 2002). Research has shown that there are increasing returns to additional investment after some threshold amount has been invested in the area ("The Impact of CDBG Spending on Urban Neighborhoods," n.d.), and that small amounts of investment below this threshold have unappreciable impacts (Galster et al., 2006). The choice of strategies employed as part of a public investment program has a profound effect on outcomes in the target area (Wachter and Gillen, 2006), and it is therefore important to develop methods of evaluating the effectiveness of government investment policy.

K-means clustering is a popular clustering algorithm (Jain, 2010), and has been previously applied to identify housing submarkets (Bourassa et al., 1999).

Data Sources and Descriptions

Data for 94 neighborhoods in Chicago are available from the Trulia API (Developer.trulia.com). Median house prices were extracted for each month from 2010-01-01 until 2015-11-01. Estimates of the total population count, number of White and Black community members, measure of wealth inequality measured as a Gini coefficient, and total annual income were obtained for the years 2010-2015 from the American FactFinder download center, all of which were measured at the Census Block level (Bureau, n.d.). Data on public and private investments were obtained from the City of Chicago Open Data Portal (CoC ODP) ("City of Chicago | Data Portal," n.d.). The CoC ODP also provided shapefiles of the neighborhoods. After accounting for differences in the availability of data from Trulia and the CoC ODP, 69 neighborhoods were included in the analysis.

The City of Chicago investment data included records of private business permits, as well as public investments from the Chicago Microlending Institute (CMI), the Micro-Market Recovery Program (MMRP), and Tax-Increment Financing (TIF) Projects. Each of these investments was geo-tagged with latitude and longitude coordinates. The CMI loans average at $10,000 per loan, and are purposed to help small enterprises have easier access to capital. The MMRP are specifically purposed towards increasing property values and improving the environment for private investment by investing in rehabilitation of properties. The TIF projects leverage increases in property values in particular districts to fund public and private projects in the district.

Methodology

The R Statistical programming language and Python scripting language are used in the analysis. The R language was used to overlay the business permits, CMI, MMRP, and TIF records onto the neighborhood shapefile. R was also used to aggregate the census block data to the neighborhood shapefile. The resulting dataframes were then exported, and the remainder of the analysis is performed using Python. Python was used to clean the data and to create functions to perform the K-means clustering, included in Appendix A. Data cleaning included the conversion of monthly median house prices to quarterly measures of the percent difference in price from the 2010-01-01 baseline.

The K-means clustering algorithm works in four steps. First, it randomly assigns each data point in the set to one of a pre-determined number of groups. Next, it calculates the centroid of the data points for each group. Third, each data point's group is updated according to the centroid it is closest to. Finally, the second and third steps are repeated until there is no longer any differences between assignments in consecutive cycles of the algorithm. Silhouette analysis (SA) is then used to determine the appropriateness of the clustering process. The SA measures the extent to which points are closer to their cluster than any other, and will return a value between -1 and 1. Values closer to 1 reflect that points are much closer to their assigned cluster than any other, and therefore that the clustering process has resulted in distinct groups.

The clustering assignments resulting from a K-Means process are highly dependent on the random assignments made in the first step. To make the results more reliable, the clustering process is repeated multiple times using the function createClusters(). After this was used to generate a list of 10 clustering assignments for the same data, a function createMatrix() was used to calculate the proportion of times that each of the 69 neighborhoods were clustered with each other. The result is a symmetric, square matrix with a diagonal of 1's, as each neighborhood was always clustered with itself in all 10 of the clustering assignments.

The neighborhoods were clustered in this manner according to four data sets: their quarterly median house prices; the demographic profile as measured by percent white, percent black, gini coefficient, and per capita income; the public and private investment profile as measured by the 2015-11-01 count of per capita businesses, and cumulative per capita investments in the CMI, MMRP, and TIF programs; and the combined demographic and investment profiles including all variables from the previous two data sets. Clustering assignments were only accepted if they satisfied a cutoff silhouette value. The number of clusters used and cutoff silhouette value were determined for each data set based on observation of SA values for repeated applications of K-means to the data set with different numbers of clusters.

The result of the above is four square matrices: priceMatrix, demoMatrix, investmentMatrix, and investmentAndDemoMatrix. These matrices were then compared with each other with a function matchMatrices() to determine whether the clustering of neighborhood price dynamics was more similar to the clustering of neighborhoods by demographic characteristics, by investment characteristics, or a combination of the two.

	0	1	2	3	4	5	6	7	8	9	...	60	61	62	63	64	65	66	67	68	69
0	1.0	0.3	0.8	0.1	0.1	0.1	0.3	0.7	0.5	0.6	...	0.1	0.8	0.1	0.1	0.0	0.1	0.4	0.1	0.8	0.1
1	0.3	1.0	0.2	0.3	0.4	0.4	0.5	0.2	0.6	0.2	...	0.4	0.2	0.6	0.4	0.2	0.4	0.3	0.5	0.2	0.5
2	0.8	0.2	1.0	0.3	0.2	0.3	0.5	0.6	0.5	0.7	...	0.3	0.8	0.2	0.2	0.0	0.0	0.5	0.2	0.8	0.1
3	0.1	0.3	0.3	1.0	0.5	0.7	0.5	0.1	0.4	0.2	...	0.5	0.2	0.4	0.5	0.4	0.4	0.3	0.4	0.2	0.4
4	0.1	0.4	0.2	0.5	1.0	0.8	0.5	0.2	0.4	0.2	...	0.7	0.2	0.5	0.7	0.5	0.3	0.4	0.6	0.1	0.5
5	0.1	0.4	0.3	0.7	0.8	1.0	0.7	0.1	0.4	0.2	...	0.8	0.2	0.6	0.6	0.3	0.3	0.3	0.6	0.2	0.6
6	0.3	0.5	0.5	0.5	0.5	0.7	1.0	0.2	0.5	0.3	...	0.6	0.4	0.6	0.4	0.2	0.2	0.4	0.6	0.3	0.6
7	0.7	0.2	0.6	0.1	0.2	0.1	0.2	1.0	0.3	0.7	...	0.1	0.8	0.1	0.1	0.1	0.0	0.7	0.2	0.8	0.1
8	0.5	0.6	0.5	0.4	0.4	0.4	0.5	0.3	1.0	0.4	...	0.3	0.3	0.3	0.3	0.1	0.2	0.6	0.3	0.4	0.3

Figure 1: an excerpt from priceMatrix

Results

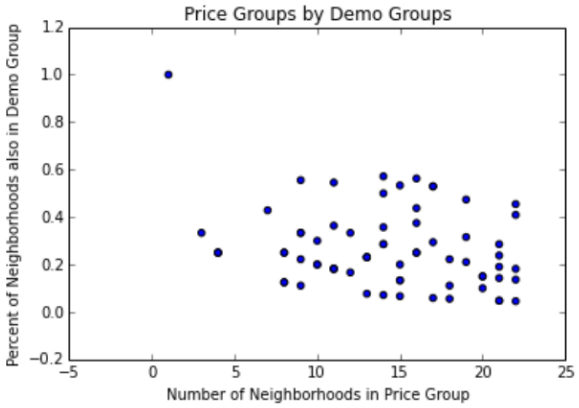

Figure 2a

Figure 2b

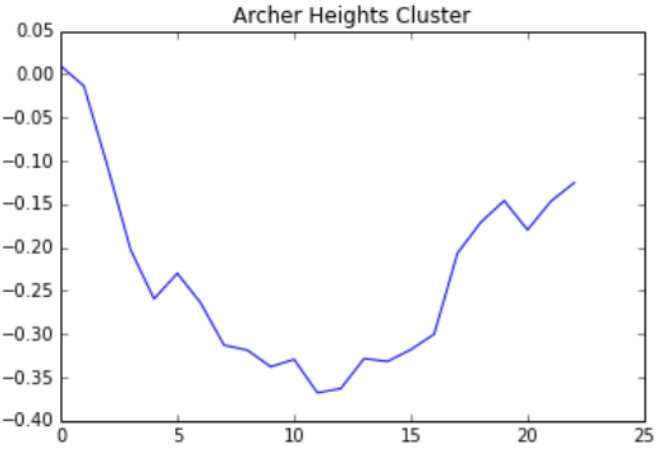

Figure 2c

Archer Heights Cluster Data

Cluster	1
Total	34997.33
White	11019.78
Black	14338
AggIncome	5.723223e+08
Gini	0.4024919
PerCapitaIncome	16864.26
PerWhite	0.3745338
PerBlack	0.3098847
TIF	1.987778e+07
MicroMarket	677.1111
MicroLending	1.666667
Businesses	393
PerCapitaTIF	671.156
PerCapitaMicroMarkets	0.02603773
PerCapitaMicroLending	3.159145e-05
PerCapitaBusinesses	0.01354079

Figure 2d

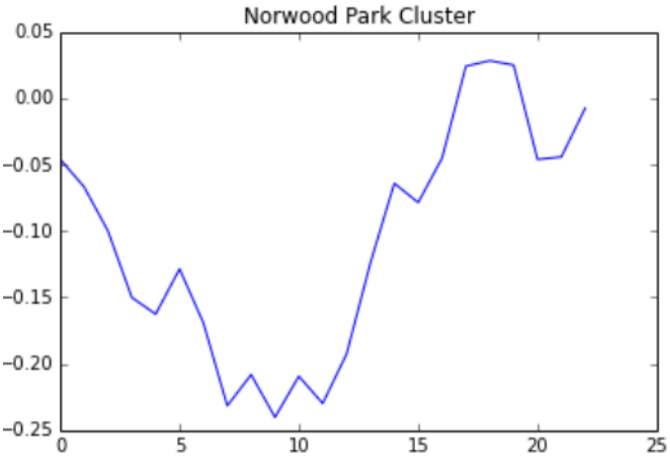

Figure 2e

Norwood Park Cluster Data

Cluster	50
Total	30247.45
White	19930.18
Black	2916.273
AggIncome	9.576019e+08
Gini	0.462943
PerCapitaIncome	27721.77
PerWhite	0.5053407
PerBlack	0.2521011
TIF	5.053491e+07
MicroMarket	112.7273
MicroLending	1.181818
Businesses	531.8182
PerCapitaTIF	4165.069
PerCapitaMicroMarkets	0.03480311
PerCapitaMicroLending	0.0001129548
PerCapitaBusinesses	0.01869489

Figure 2f

Discussion

Two neighborhoods were considered to be in the same price, demographic, investment, or investment and demographic group if they were clustered together at least 6 out of the 10 instances created by the createClusters() function. The graphs above display the pairwise relationships between the group assignments of neighborhoods. Each point on figures 2a and 2b represents a neighborhood.

Figure 2a and 2b show 69 neighborhoods, and the relationship between their price groups and their demographic groups and investment groups, respectively. The number of neighborhoods in a neighborhood's price group is displayed on the x-axis. This value could fall between 1 and 69, with a value of 1 meaning that the neighborhood was typically alone in its price group, or was not consistently grouped with any other neighborhoods. Higher values indicate that there is a larger group of neighborhoods that were consistently grouped as having similar price dynamics to the neighborhood in question. The y-axis displays the percent of neighborhoods in a neighborhood's price group that are also in the neighborhood's demographic group. We can interpret a point at [3, .333] as meaning that of the 3 neighborhoods that this neighborhood is grouped with in 6/10 clustering assignments based on price dynamic, 1 of them is also grouped with this neighborhood in 6/10 clustering assignments based on demographic profile.

The analysis suggests that the city investment strategies are better predictors of neighborhood price dynamics than demographic features of the neighborhoods. Figures 2a and 2b do not give any indication of whether increased investment leads to better outcomes in housing prices. To gain some insight here, we look at two neighborhoods, Archer Heights and Norwood Park. Neighborhoods in Archer Height's cluster did not recover to their 2010 levels by 2015, while neighborhoods in Norwood Park's cluster did. All of the per capita investment measures were higher for Norwood Park than Archer Heights. This suggests that increased city investment may lead to increased ability of neighborhoods to recover from falls in their house prices.

Conclusions

The ways that cities choose to invest in their neighborhoods has more effect on the dynamics of house prices than can be explained by demographic trends.

Further research is necessary to determine whether this is because neighborhoods with falling prices receive more investment from targeted city programs, or whether the city investments are leading to improved median house price outcomes. Further research should also work to identify the magnitude of each investment strategy's contribution to improvements in neighborhood median house price trends.

This method is useful in helping to narrow the focus of city planners to the programs or investment strategies that matter most in affecting particular trends that they are interested in influencing. They can then determine which neighborhood types are most consistently influenced by their programs, and which others are not.

```
# Import libraries
import pandas as pd
import random as rd
import matplotlib.pyplot as plt
import numpy as np

# Create a function to determine which cluster the row of data is closest to
# centCoords is a list of the Cluster Centroid coordinates
def calc_dist(rowVals, centCoords):
    rowVals = np.array(rowVals)[0:len(rowVals)-1]
    # Store distances from each cluster in a list, dists
    dists = []
    for i in centCoords:
        # np.linalg.norm calculates the distance between two vectors, here: between "rowVals" and "i"
        dists.append(np.linalg.norm(rowVals-i))
    # Return the index of the minimum value in dists
    return np.argmin(dists)

def calc_dist2(rowVals, centCoords):
    rowVals = np.array(rowVals)[0:len(rowVals)-1]
    print "rowVals ", rowVals
    # Store distances from each cluster in a list, dists
    dists = []
    for i in centCoords:
        # np.linalg.norm calculates the distance between two vectors, here: between "rowVals" and "i"
        dists.append(np.linalg.norm(rowVals-i))
    # Return the index of the minimum value in dists
    return np.argmin(dists)

# Create a function to calculate the centroids of all the clusters
def calc_centroids(data):
    # Group the data by cluster, calculate the mean of each column, then undo the grouping with reset_index()
    # This returns more columns than we want, so we only take columns 1-4 that have the mean values along each
    # of the four dimensions we're measuring on the flowers.
    means = data.groupby("Cluster").mean().reset_index()
    # print means
    means = means.as_matrix()[:,1:]
    return means

# Crete a function to determine if the new groups are equal to the old groups or not.
# This is our stopping criteria
def returns_diffs(df1,df2):
    df1 = np.array(df1)
    df2 = np.array(df2)
    diffs = pd.DataFrame({"diffs":df1-df2})
    # Choose only the entries that are not the same
    diffs = diffs[diffs["diffs"]!=0]
    return diffs

def standardize(column):
    mean = np.mean(column)
    std = np.std(column)
    col = np.array(column)
    col = (col-mean)/std
    return col

# Silhouette analysis:

# Create a function to calculate the mean distance of a point to all others in it's cluster
def avgDist(row,group):
    row1 = np.array(row)[0:len(row)-4]
    group["dists"] = group.apply(lambda row: np.linalg.norm(row1-row[0:len(row)-4]), axis=1)
    return np.mean(group["dists"])

# Create a function to determine the 2nd closest cluster to a point
def findNearCluster(row,centCoords):
    row = np.array(row)[0:len(row)-4]
    dists = np.array([])

    for i in centCoords:
        # np.linalg.norm calculates the distance between two vectors, here: between "row" and "i"
        dists = np.append(dists,np.linalg.norm(row-i[1:]))

    # argsort() lists the indexes in order. Only works for an np.array.
    distsOrder = dists.argsort()
    # Grabs the index of the second lowest value - i.e. the nearest neighbor
    # print distsOrder
    nearest_cluster_num = distsOrder[1]

    return nearest_cluster_num
```

```
# Create a function to apply the silhouette criteria to a clustering
def silhouette(data):
    # calculate average distance to all in group
    data["InGroup"] = 1
    data["NearGroup"] = 1

    data["InGroup"] = data.apply(lambda row: avgDist(row,data[data["Cluster"]==row["Cluster"]]), axis=1)

    # calculate average distance to all in nearest cluster
    data["NearGroup"] = data.apply(lambda row: avgDist(row, data[data["Cluster"]==
                    findNearCluster(row,calc_centroids(data.drop(["InGroup","NearGroup"], axis=1)))]), axis=1)

    # calculate the silhouette value for the row
    data["Sil"] = data.apply(lambda row: ((row["NearGroup"]-row["InGroup"])/
                    max(row["NearGroup"],row["InGroup"])), axis=1)
    # A value close to 1 means that data is well clustered
    # A value close to 0 means that data are in between clusters
    # A value close to -1 means that data should be in different clusters

    # return the mean of all the silhouette values.
    return np.mean(data["Sil"])
def createClusters(numClusters, threshold, size, originalData, origNames, std = True):
    # Create a list of clustering assignments, along with their silhouette value that exceed the threshold
    clusters = []
    count = 0
    while(len(clusters)<size):
        work = originalData
        rands = pd.Series([random.randint(0,numClusters) for i in range(len(work))])
        if std:
            work = work.apply(lambda col: standardize(col), axis=0)
        work["Cluster"] = rands

        if len(work["Cluster"].unique()) != numClusters:
            print "Not equal at assign"

        flag = False
        while not flag:
            if len(work["Cluster"].unique()) != numClusters:
                print "Not equal during process", len(work["Cluster"].unique())
            work["NewGroups"] = work.apply(lambda row: calc_dist(row,calc_centroids(work)), axis=1)
            flag = work["NewGroups"].equals(work["Cluster"])
            work["Cluster"]=work["NewGroups"]

        if len(work["Cluster"].unique()) == numClusters:
            print count
            check = silhouette(work)
            if(check > threshold):
                work = pd.concat([origNames, work], axis=1)
                clusters.append(work)
                print "Success", check
            count = count + 1
        print count, len(work["Cluster"].unique())
    return clusters
def createMatrix(clusters):
    arrayLen = len(clusters[0])
    matrix = []

    for index in range(arrayLen):
        neighs = np.repeat(0.0,arrayLen)
        for df in clusters:
            group = df[df["Cluster"]==df.iloc[index]["Cluster"]]
            for ix in group.index.values:
                neighs[ix] = neighs[ix] + 1

        matrix.append(neighs/len(clusters))

    return pd.DataFrame(matrix)

def matchMatrices(matrix1, mat1Data, mat1Com, matrix2, mat2Data, mat2Com, threshold):
    threshold = threshold
    matchPercents = []

    for index in range(len(matrix2)):


        #print "Price group: ", mat1Data.iloc[index][mat1Com]
        Cluster1 = mat1Data[matrix1[str(index)]>=threshold]

        #print "Demo group: ", mat2Data.iloc[index][mat2Com]
        Cluster2 = mat1Data[mat1Data[mat1Com].str.upper().isin(mat2Data[matrix2[str(index)]>=threshold][mat2Com])]

        #print ""

        matches = Cluster1[Cluster1[mat1Com].isin(Cluster2[mat1Com])]
        #print "Matches:", len(matches[mat1Com])
        #print matches[mat1Com]
        #print ""

        matchPercents.append([len(Cluster1),float(len(matches[mat1Com]))/len(Cluster1)])

    return pd.DataFrame(matchPercents)
```

Bibliography

Bourassa, S.C., Hamelink, F., Hoesli, M., MacGregor, B.D., 1999. Defining Housing Submarkets. J. Hous. Econ. 8, 160–183. doi:10.1006/jhec.1999.0246

Bureau, U.S.C., n.d. American FactFinder - Download Center [WWW Document]. URL http://factfinder.census.gov/faces/nav/jsf/pages/download_center.xhtml (accessed 1.4.16).

City of Chicago | Data Portal [WWW Document], n.d. . Chicago. URL https://data.cityofchicago.org/ (accessed 1.4.16).

Developer.trulia.com,. "Trulia - Welcome To The Trulia API". N.p., 2016. Web. 4 Jan. 2016.

Ding, C., Knaap, G.-J., 2002. Property values in inner-city neighborhoods: The effects of homeownership, housing investment, and economic development. Hous. Policy Debate 13, 701–727. doi:10.1080/10511482.2002.9521462

Ding, C., Simons, R., Baku, E., 2000. The Effect of Residential Investment on Nearby Property Values: Evidence from Cleveland, Ohio. J. Real Estate Res. 19, 23–48. doi:10.5555/rees.19.1.lkq702h122626576

Galster, G., Tatian, P., Accordino, J., 2006. Targeting Investments for Neighborhood Revitalization. J. Am. Plann. Assoc. 72, 457–474. doi:10.1080/01944360608976766

Jain, A.K., 2010. Data clustering: 50 years beyond K-means. Pattern Recognit. Lett., Award winning papers from the 19th International Conference on Pattern Recognition (ICPR)19th International Conference in Pattern Recognition (ICPR) 31, 651–666. doi:10.1016/j.patrec.2009.09.011

Maass, A., 1966. Benefit-Cost Analysis: Its Relevance to Public Investment Decisions. Q. J. Econ. 80, 208–226. doi:10.2307/1880690

Simons, R., Quercia, R., Levin, I., 1998. The Value Impact of New Residential Construction and Neighborhood Disinvestment on Residential Sales Price. J. Real Estate Res. 15, 147–161. doi:10.5555/rees.15.2.7788757q04786454

The Impact of CDBG Spending on Urban Neighborhoods [WWW Document], n.d. URL http://www.urban.org/research/publication/impact-cdbg-spending-urban-neighborhoods (accessed 1.4.16).

Wachter, S., 2004. The Determinants of Neighborhood Transformations in Philadelphia-Identification and Analysis: The New Kensington Pilot Study.

Wachter, S.M., Gillen, K.C., 2006. Public investment strategies: How they matter for neighborhoods in Philadelphia.