Introduction

In this Capstone project I will be identifying planning permission applications in Cork City from 2008 onwards. This data is in csv format and has been obtained from the City Council's open data website https://data.corkcity.ie.

I will be looking at what neighbourhoods these applications take place in. Here the Foursquare data will help us define these neighbourhoods by the amenities available. I will also be using the supplemental data in this source to provide further useful comparisons between the segmented neighbourhoods once the clustering has been mapped.

The utility of this project lies in a number of features. On the most basic level this is will let us identify the basic neighbourhood types that attract the highest density of planning permission. Additionally the data contains a number of extra pieces of information which can be used to supplement the analysis after clustering. Primarily these applications record whether an application was successful, whether it was for one-off housing and the square footage of the application. We will be able to find the average values for each neighbourhood (or in the case of application success we can see which neighbourhoods are easiest and hardest to obtain permission for).

This information would be of significant interest to many parties. From building contractors and developers to county planners. It would help local businesses identify changing land use patterns and help banks identify which areas are going to experience significant growth.

Data

The data I am using is a csv file I obtained from https://data.corkcity.ie.

It is reasonably large measuring 4,975 rows by 34 columns. However of these the majority are unimportant to our investigation. As part of data preparation the following columns were kept and the rest dropped:

1. Application Number
    a. As Primary Key
2. Latitude
3. Longitude
4. OneOffHouseFlag
    a. This signals if the permission was for a one off house
5. NumberOfResidentialUnits
    a. This signifies how many homes are being built
6. AreaOfSite
    a. The total area of the property in km2
7. Decision
    a. Whether the application was accepted for rejected

The new data frame looks like this:

ApplicationNumber	Latitude	Longitude	OneOffHouseKPI	NumResidentialUnits	AreaOfSite	Decision
832737	51.87518	-8.49607	No	0	0.043	Refused
832738	51.89802	-8.41491	No	0	0.074	Granted (Conditional)
832739	51.88336	-8.48256	No	0	0.02	Granted (Conditional)
832740	51.88769	-8.47358	Yes	1	0.042	Refused
832741	51.91723	-8.46982	No	0	0.85	Granted (Conditional)

In my Jupyter notebook it looks like this

```
[11]: planp = pd.read_csv('planningpermission2.csv')
      planp.head()
```

[11]:

	_id	ApplicationNumber	Latitude	Longitude	OneOffHouseKPI	NumResidentialUnits	AreaOfSite	Decision
0	3	832737	51.875175	-8.496067	No	0	0.043	Refused
1	4	832738	51.898020	-8.414907	No	0	0.074	Granted (Conditional)
2	5	832739	51.883356	-8.482564	No	0	0.020	Granted (Conditional)
3	6	832740	51.887688	-8.473576	Yes	1	0.042	Refused
4	7	832741	51.917226	-8.469818	No	0	0.850	Granted (Conditional)

Additionally, mindful of the limits on API calls that could be made to Foursquare we adjusted the dataset to come within the call limit taking a random sample of 870 applications (roughly the calls we had left on the day).

Methodology

The first step was to run the various applications through the Foursquare API to assemble a list of most common nearby venues. The radius was set to 500m as this is appropriately sized for a city of Cork's size. The limit was reduced to 20 for ease of computation.

The initial API results (with one return per venue type) were first aggregated to application level with the most common venue for each application type being listed. Then these individual results were transposed onto a larger table with the application numbers serving as primary key:

	ApplicationNumber	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	832737	Convenience Store	Bar	Gas Station	Bus Stop	Discount Store	Electronics Store	Farmers Market	Fast Food Restaurant	Department Store	Gastropub
1	832738	Trail	Market	Gourmet Shop	Harbor / Marina	Wine Shop	Field	Furniture / Home Store	Fried Chicken Joint	French Restaurant	Food Court
2	832739	Plaza	Chinese Restaurant	Supermarket	Convenience Store	Department Store	Discount Store	Electronics Store	Farmers Market	Fast Food Restaurant	Gas Station
3	832740	Soccer Stadium	Park	Fish & Chips Shop	Fast Food Restaurant	Department Store	Discount Store	Electronics Store	Deli / Bodega	Farmers Market	Convenience Store
4	832741	Coffee Shop	Soba Restaurant	Restaurant	Shopping Mall	Fast Food Restaurant	Supermarket	Grocery Store	Pizza Place	Shopping Plaza	Department Store

This was done first through the use of one hot encoding on the venue types returned and the frequency of these dummies in each application then recorded:

```
# one hot encoding
cork_onehot = pd.get_dummies(Cork_venues[['Venue Category']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe
cork_onehot['ApplicationNumber'] = Cork_venues['ApplicationNumber']

# move neighborhood column to the first column
fixed_columns = [cork_onehot.columns[-1]] + list(cork_onehot.columns[:-1])
cork_onehot = cork_onehot[fixed_columns]

cork_grouped = cork_onehot.groupby('ApplicationNumber').mean().reset_index()

num_top_venues = 5

for hood in cork_grouped['ApplicationNumber']:
    print(hood)
    temp = cork_grouped[cork_grouped['ApplicationNumber'] == hood].T.reset_index()
    temp.columns = ['venue','freq']
    temp = temp.iloc[1:]
    temp['freq'] = temp['freq'].astype(float)
    temp = temp.round({'freq': 2})
    print(temp.sort_values('freq', ascending=False).reset_index(drop=True).head(num_top_venues))
    print('\n')
```

```
832737
                venue  freq
0  Convenience Store  0.25
1        Gas Station  0.25
2                Bar  0.25
3           Bus Stop  0.25
4        Art Gallery  0.00


832738
              venue  freq
0             Trail   0.4
1     Gourmet Shop   0.2
2           Market   0.2
3  Harbor / Marina   0.2
4      Art Gallery   0.0


832739
                venue  freq
0               Plaza  0.25
1   Convenience Store  0.25
2  Chinese Restaurant  0.25
3         Supermarket  0.25
4                Park  0.00
```

Finally these sub-categories were aggregated through a custom function:

```
def return_most_common_venues(row, num_top_venues):
    row_categories = row.iloc[1:]
    row_categories_sorted = row_categories.sort_values(ascending=False)

    return row_categories_sorted.index.values[0:num_top_venues]

num_top_venues = 10

indicators = ['st', 'nd', 'rd']

# create columns according to number of top venues
columns = ['ApplicationNumber']
for ind in np.arange(num_top_venues):
    try:
        columns.append('{}{} Most Common Venue'.format(ind+1, indicators[ind]))
    except:
        columns.append('{}th Most Common Venue'.format(ind+1))

# create a new dataframe
application_venues_sorted = pd.DataFrame(columns=columns)
application_venues_sorted['ApplicationNumber'] = cork_grouped['ApplicationNumber']

for ind in np.arange(cork_grouped.shape[0]):
    application_venues_sorted.iloc[ind, 1:] = return_most_common_venues(cork_grouped.iloc[ind, :], num_top_venues)
```

From here we could use k means clustering to identify which neighbourhoods were the most similar to each other. We aimed for four clusters in this method and it returned the following table:

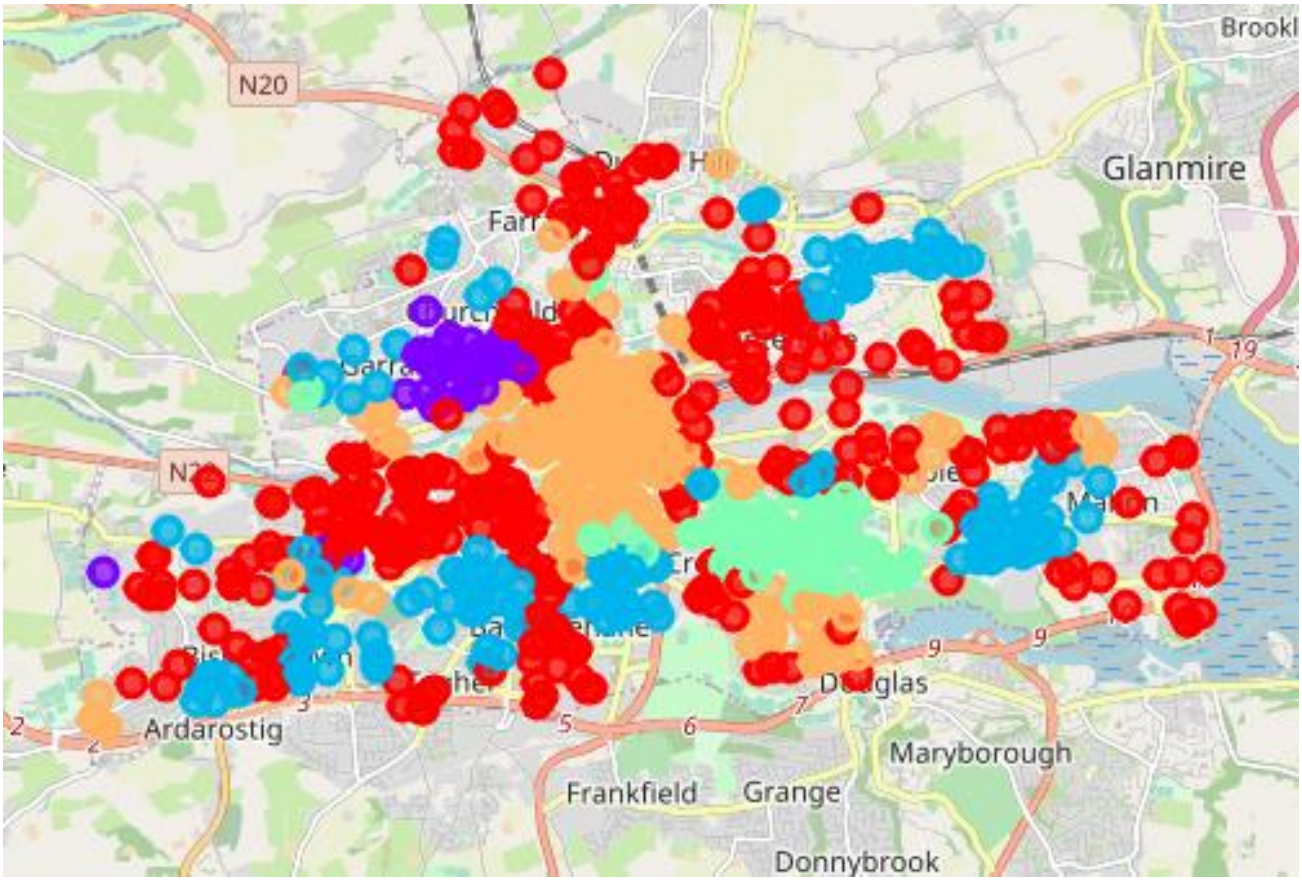
	_id	ApplicationNumber	Latitude	Longitude	OneOffHouseKPI	NumResidentialUnits	AreaOfSite	Decision	Cluster Labels	1st Most Common Venue	2nd Most Common Venue
0	3	832737	51.875175	-8.496067	No	0	0.043	Refused	0.0	Convenience Store	Bar
1	4	832738	51.898020	-8.414907	No	0	0.074	Granted (Conditional)	0.0	Trail	Market
2	5	832739	51.883356	-8.482564	No	0	0.020	Granted (Conditional)	2.0	Plaza	Chinese Restaurant
3	6	832740	51.887688	-8.473576	Yes	1	0.042	Refused	3.0	Soccer Stadium	Park
4	7	832741	51.917226	-8.469818	No	0	0.850	Granted (Conditional)	0.0	Coffee Shop	Soba Restaurant

Finally we made to map the clusters using Folium and a simple line of code:

```
# finally we will create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=11)

# set color scheme for the clusters
x = np.arange(kclusters)
ys = [i + x + (i*x)**2 for i in range(kclusters)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(cork_merged['Latitude'], cork_merged['Longitude'], cork_merged['ApplicationNumber'], cork_me
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Results

The map produced was striking:



Discussion

The most striking thing about the results was emergence of a clear city centre area (orange), city periphery area (red), north-west city centre (purple), south-east suburbia (green). Investigating these areas more specifically we can see that the popular venues confirm these land uses.

The city-centre area is dominated by pubs and restaurants:

	ApplicationNumber	NumResidentialUnits	AreaOfSite	Decision	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue
9	832747	0	1.240	Granted (Conditional)	4	Restaurant	Brewery	Bar	Supermarket	Pub
15	832753	0	0.060	Granted (Conditional)	4	Brewery	Restaurant	BBQ Joint	Café	Liquor Store
16	832754	0	4.130	Granted (Conditional)	4	Café	Fast Food Restaurant	Sporting Goods Shop	Health Food Store	Italian Restaurant
18	832756	0	0.028	Granted (Conditional)	4	Pub	Restaurant	Burger Joint	Vegetarian / Vegan Restaurant	Performing Arts Venue
19	832757	0	0.028	Granted (Conditional)	4	Café	Pub	Restaurant	Burger Joint	Vegetarian / Vegan Restaurant

The suburban area meanwhile saw more parks and sports areas:

[76]:

	ApplicationNumber	NumResidentialUnits	AreaOfSite	Decision	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue
3	832740	1	0.042	Refused	3	Soccer Stadium	Park	Fish & Chips Shop	Fast Food Restaurant	Department Store	Discount Store
20	832758	1	0.034	Granted (Conditional)	3	Fast Food Restaurant	Pub	Grocery Store	Park	Wine Shop	Fried Chicken Joint
29	832770	0	0.014	Granted (Conditional)	3	Pub	Fast Food Restaurant	Grocery Store	Field	Furniture / Home Store	Fried Chicken Joint
41	832783	0	0.025	Granted (Conditional)	3	Soccer Stadium	Park	Fish & Chips Shop	Fast Food Restaurant	Department Store	Discount Store
50	832793	0	0.026	Granted (Conditional)	3	Garden Center	Stadium	Grocery Store	Fast Food Restaurant	Fish & Chips Shop	Furniture / Home Store

The city periphery area has a wider array of services such as coffee shops, walking trails and even a marina:

	ApplicationNumber	NumResidentialUnits	AreaOfSite	Decision	Cluster Labels	Common Venue	Common Venue	Common Venue	Common Venue	Common Venue	Com V
0	832737	0	0.043	Refused	0	Convenience Store	Bar	Gas Station	Bus Stop	Discount Store	Electr
1	832738	0	0.074	Granted (Conditional)	0	Trail	Market	Gourmet Shop	Harbor / Marina	Wine Shop	
4	832741	0	0.850	Granted (Conditional)	0	Coffee Shop	Soba Restaurant	Restaurant	Shopping Mall	Fast Food Restaurant	Superm
8	832746	1	0.000	NaN	0	Coffee Shop	Platform	Gym	Hotel	Train Station	
12	832750	2	0.020	Refused	0	Hotel	Restaurant	Bar	Fish & Chips Shop	Liquor Store	Wine

Finally the north inner city area contains a number of fast food places and some bars:

[74]:

	ApplicationNumber	NumResidentialUnits	AreaOfSite	Decision	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Mo Commo Venu
5	832742	0	1.000	Granted (Conditional)	1	Soccer Field	Pizza Place	Fried Chicken Joint	Field	Garden Center	Furniture Hom Sto
24	832763	0	0.035	Granted (Conditional)	1	Pizza Place	BBQ Joint	College Theater	Café	Bar	Brewe
26	832767	0	0.064	Granted (Conditional)	1	Pizza Place	Fried Chicken Joint	Wine Shop	Fast Food Restaurant	Furniture / Home Store	Frenc Restaura
37	832779	0	0.081	Granted (Conditional)	1	Fried Chicken Joint	Museum	Wine Shop	Field	Furniture / Home Store	Frenc Restaura
80	832828	1	0.030	NaN	1	Playground	Asian Restaurant	Pizza Place	Fried Chicken Joint	Field	Garde Cent

The least apt grouping was the blue group which contains a disparate collection of resources:

[75]:

	ApplicationNumber	NumResidentialUnits	AreaOfSite	Decision	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue
2	832739	0	0.020	Granted (Conditional)	2	Plaza	Chinese Restaurant	Supermarket	Convenience Store	Department Store
6	832744	0	NaN	Granted (Conditional)	2	Convenience Store	Fish & Chips Shop	Shopping Mall	Grocery Store	Supermarket
7	832745	0	0.700	Granted (Conditional)	2	Playground	Sporting Goods Shop	Gourmet Shop	Bar	Supermarket
10	832748	0	0.026	Refused	2	Supermarket	Gastropub	Convenience Store	Fish & Chips Shop	Garden Center
11	832749	0	0.000	Refused	2	Chinese Restaurant	Supermarket	Locksmith	Wine Shop	Garden Center

Conclusion

This demonstrates that planning applications should distinguish strongly between suburban, city centre, inner-city areas but also that localisation can be taken too far and that many amenities and characteristics are common across urban areas (as seen in the blue areas of the map).

There is further advantage to this data with further investigation. We can see for example that suburban areas seem to see the highest rate of rejection while city centre areas see the smallest properties with applications. These simple insights can be easily expanded upon.