Applied Data Science Capstone Project: Comparing Helsinki Neighborhoods

Table of Contents

Introduction

In this project, we will be comparing the neighborhoods of my hometown - Helsinki, Finland. We will be using availability of services and some socio-economic data from different kinds of open data sources to compare Helsinki neighborhoods.

Business problem

I designed this study for people who are planning moving to Helsinki, Finland from abroad. In particular, I am thinking about ICT experts moving to Finland – there is a dire shortage of people with skills in machine learning, full-stack development and test automation, for example. The idea is to help a professional select a place to live during their stay in Finland.

When moving to a city in a foreign country, you really do not have that much real insight on what the different neighborhoods really are like. This study will provide that insight.

We will be using the following data to analyze the neighborhoods:

- Availability of different kinds of services in an area
- Occupational status of people living in an area
- Median income of households in an area
- Housing information
    - Average size of dwellings in an area
    - Relative share of apartments in an area
    - Relative share of houses in an area

The business problem we are answering in this study is the following:

- What kind of areas there are in Helsinki, Finland, in terms of available services and key socio-economic data?

As stated, this information should be helpful for people relocating to Helsinki area.

Data used in the study

Postal code data

We will fetch Helsinki postal code data from Helsinki Regional Infoshare (HRI) ReST service. No API keys needed. The URL is the following – a simple GET:

https://hri.fi/data/api/action/datastore_search?resource_id=cbc11e4a-f695-4efa-93d7-9446066a07dd&limit=84

We limit ourselves to first 84 postal codes - we would get postal codes for all of Finland, if we did not limit ourselves. The response is "in Finnish". The response is the following – note the strange coding of postal codes ('100' instead of '00100' which is the norm in Finland" – interesting data points marked in red):


```
{
  "help": "https:\/\/hri.fi\/data\/api\/3\/action\/help_show?name=datastore_search",
  "success": true,
  "result": {
    "include_total": true,
    "resource_id": "cbc11e4a-f695-4efa-93d7-9446066a07dd",
    "fields": [
      {
        "type": "int",
        "id": "_id"
      },
      {
        "type": "numeric",
        "id": "Postinumero"
      },
      {
        "type": "text",
        "id": "Postitoimipaikka"
      },
      {
        "type": "text",
        "id": "Postitoimipaikka_Ru"
      },
      {
        "type": "text",
        "id": "Nimi"
      },
```

```
      {
        "type": "text",
        "id": "Nimi_Ru"
      },
      {
        "type": "text",
        "id": "Kunta"
      },
      {
        "type": "numeric",
        "id": "Kunta_nro"
      }
    ],
    "records_format": "objects",
    "records": [
      {
        "_id": 1,
        "Postinumero": 100,
        "Postitoimipaikka": "HELSINKI",
        "Postitoimipaikka_Ru": "HELSINGFORS",
        "Nimi": "Helsinki Keskusta - Etu-T\u00f6\u00f6l\u00f6",
        "Nimi_Ru": "Helsingfors centrum - Fr\u00e4mre T\u00f6l\u00f6",
        "Kunta": "Helsinki",
        "Kunta_nro": 91
      },
      {
        "_id": 2,
        "Postinumero": 120,
        "Postitoimipaikka": "HELSINKI",
        "Postitoimipaikka_Ru": "HELSINGFORS",
        "Nimi": "Punavuori",
        "Nimi_Ru": "R\u00f6dbergen",
        "Kunta": "Helsinki",
        "Kunta_nro": 91
      },
      …
      {
        "_id": 84,
        "Postinumero": 990,
        "Postitoimipaikka": "HELSINKI",
        "Postitoimipaikka_Ru": "HELSINGFORS",
        "Nimi": "Aurinkolahti",
        "Nimi_Ru": "Solvik",
        "Kunta": "Helsinki",
        "Kunta_nro": 91
      }
    ],
    "limit": 84,
    "_links": {
      "start": "\/api\/action\/datastore_search?limit=84&resource_id=cbc11e4a-f695-4efa-
93d7-9446066a07dd",
      "next": "\/api\/action\/datastore_search?offset=84&limit=84&resource_id=cbc11e4a-
f695-4efa-93d7-9446066a07dd"
    },
    "total": 172
  }
}
```


There are, in fact, many other postal codes in Helsinki region – there are many “special” postal codes. HRI data is limited to actual - let us say physical - postal codes, which is exactly what we want. As an example this is the postal code list of Post Finland – as we can see, there are many more postal codes here. The physical postal codes for Helsinki region are those with numbers such as 00100, 00110 and so on, up to 0990. Eighty-four of these are currently in use – that is why we limit ourselves to 84 postal codes above.

https://www.verkkoposti.com/e3/postinumeroluettelo?streetname=&postcodeorcommune=Helsinki&_ga=2.59104429.1032254499.1543841837-1839900633.1543660209

Occupational data

We'll fetch occupational data from Statistics Finland public data service. We are quite lucky, since Statistics Finland has got data all kinds of interesting data already sorted according to postal code area! For each postal code area, we will fetch (in parenthesis the data point in question – which is a Finnish abbreviation, sorry about that…

- Employed ("Pt_tyoll")
- Unemployed ("Pt_tyott")
- Children ("Pt_0_14")
- Students ("Pt_opisk")
- Pensioners ("Pt_elakel")
- Other ("Pt_muut") – this includes mothers and fathers taking care of their children, for example

We will use this data to calculate the relative shares of these groups in an area.

No API keys are needed. To fetch this data, we need just a URL, and do a POST request describing our parameters and data fetched in JSON format. The data can also be browsed via a web interface in English:

http://pxnet2.stat.fi/PXWeb/pxweb/en/Postinumeroalueittainen_avoin_tieto/Postinumeroalueittainen_avoin_tieto__2018/paavo_8_pt_2018.px/?rxid=39840011-c10c-4e00-8cd1-7015d2e09479

URL:

http://pxnet2.stat.fi/PXWeb/api/v1/en/Postinumeroalueittainen_avoin_tieto/2018/paavo_8_pt_2018.px

Example post request:

```
{
  "query": [
    {
      "code": "Postinumeroalue",
      "selection": {
        "filter": "item",
        "values": [
          "00100"
        ]
      }
    },
    {
      "code": "Tiedot",
      "selection": {
        "filter": "item",
        "values": [
          "Pt_tyoll",
          "Pt_tyott",
          "Pt_0_14",
          "Pt_opisk",
          "Pt_elakel",
          "Pt_muut"
        ]
      }
    }
  ],
  "response": {
    "format": "JSON"
  }
}
```

Example response – note that we'll get description of each data, too! The output is a little bit strange mix of both Finnish and English, let us not care about that. Also note that data is actually a list – the response is

not structured according to postal code, as one might expect. We'll need to process the response further to get to tabular format.


```
{
  "columns": [
    {
      "code": "Postinumeroalue",
      "text": "Postal code area",
      "comment": "2018 postal code areas\r\n",
      "type": "d"
    },
    {
      "code": "Tiedot",
      "text": "Data",
      "type": "d"
    },
    {
      "code": "Paavo - Open data by postal code area 2018",
      "text": "Paavo - Open data by postal code area 2018",
      "type": "c"
    }
  ],
  "comments": [
    {
      "variable": "Tiedot",
      "value": "Pt_tyovy",
      "comment": "The labour force comprises employed and unemployed people who were
either employed or unemployed during the last week of the year. Information about being
in the labour force is based on data obtained from various registers.\r\n"
    },
    {
      "variable": "Tiedot",
      "value": "Pt_tyoll",
      "comment": "Employed labour force is defined as people aged 18 to 74 who were
gainfully employed during the last week of the year.\r\n"
    },
    {
      "variable": "Tiedot",
      "value": "Pt_tyott",
      "comment": "Unemployed labour force comprises people aged 15 to 64 who were
unemployed on the last working day of the year.\r\n"
    },
    {
      "variable": "Tiedot",
      "value": "Pt_0_14",
      "comment": "Children aged 0 to 14.\r\n"
    },
    {
      "variable": "Tiedot",
      "value": "Pt_opisk",
      "comment": "Students are defined as persons who study full-time and are not
gainfully employed or unemployed. The definition is based on a person\u2019s situation in
September.\r\n"
    },
    {
      "variable": "Tiedot",
      "value": "Pt_elakel",
      "comment": "Pensioners are defined as persons who according to the Social Insurance
Institution or the Finnish Centre for Pensions receive a pension or have some other
pension income. If a pensioner is working while receiving pension, he or she is
considered employed.\r\n"
    },
    {
      "variable": "Tiedot",
      "value": "Pt_muut",
      "comment": "Others include all other persons outside the labour force except for
children, students and pensioners. This group also includes conscripts.\r\n"
    }
```

```
],
"data": [
  {
    "key": [
      "00100",
      "Pt_tyovy"
    ],
    "values": [
      "10735"
    ]
  },
  {
    "key": [
      "00100",
      "Pt_tyoll"
    ],
    "values": [
      "9871"
    ]
  },
  {
    "key": [
      "00100",
      "Pt_tyott"
    ],
    "values": [
      "864"
    ]
  },
  {
    "key": [
      "00100",
      "Pt_0_14"
    ],
    "values": [
      "1773"
    ]
  },
  {
    "key": [
      "00100",
      "Pt_opisk"
    ],
    "values": [
      "1258"
    ]
  },
  {
    "key": [
      "00100",
      "Pt_elakel"
    ],
    "values": [
      "3295"
    ]
  },
  {
    "key": [
      "00100",
      "Pt_muut"
    ],
    "values": [
      "807"
    ]
  },
  {
    "key": [
      "00120",
```

```
        "Pt_tyovy"
      ],
      "values": [
        "4187"
      ]
    },
    {
      "key": [
        "00120",
        "Pt_tyoll"
      ],
      "values": [
        "3837"
      ]
    },
    {
      "key": [
        "00120",
        "Pt_tyott"
      ],
      "values": [
        "350"
      ]
    },
    {
      "key": [
        "00120",
        "Pt_0_14"
      ],
      "values": [
        "794"
      ]
    },
    {
      "key": [
        "00120",
        "Pt_opisk"
      ],
      "values": [
        "434"
      ]
    },
    {
      "key": [
        "00120",
        "Pt_elakel"
      ],
      "values": [
        "1246"
      ]
    },
    {
      "key": [
        "00120",
        "Pt_muut"
      ],
      "values": [
        "365"
      ]
    }
  ]
}
```


Housing data

We'll fetch the income data much in the same style as we fetched the occupation data, from Statistics Finland ReST service. This time, we want the following data per postal code area:

- Average floor area (“Ra_as_kpa”) – average house/apartment size, in square meters, in area
- Dwellings in small houses (“Ra_pt_as”) – number of houses in area
- Dwellings in blocks of flats (“Ra_kt_as”) – number of apartments in area

We will use the latter two to calculate the relative shares of houses vs apartments in an area.

Web URL:

http://pxnet2.stat.fi/PXWeb/pxweb/en/Postinumeroalueittainen_avoin_tieto/Postinumeroalueittainen_avoin_tieto__2018/paavo_6_ra_2018.px/table/tableViewLayout2/?rxid=39840011-c10c-4e00-8cd1-7015d2e09479

Service URL:
http://pxnet2.stat.fi/PXWeb/api/v1/en/Postinumeroalueittainen_avoin_tieto/2018/paavo_6_ra_2018.px

Example POST data:


```
{
  "query": [
    {
      "code": "Postinumeroalue",
      "selection": {
        "filter": "item",
        "values": [
          "00100",
          "00120"
        ]
      }
    },
    {
      "code": "Tiedot",
      "selection": {
        "filter": "item",
        "values": [
          "Ra_as_kpa",
          "Ra_pt_as",
          "Ra_kt_as"
        ]
      }
    }
  ],
  "response": {
    "format": "json"
  }
}
```


Example answer:


```
{
  "columns": [
    {
      "code": "Postinumeroalue",
      "text": "Postal code area",
      "comment": "2018 postal code areas\r\n",
      "type": "d"
    },
    {
      "code": "Tiedot",
      "text": "Data",
      "type": "d"
    },
    {
      "code": "Paavo - Open data by postal code area 2018",
```

```
      "text": "Paavo - Open data by postal code area 2018",
      "type": "c"
    }
  ],
  "comments": [
    {
      "variable": "Tiedot",
      "value": "Ra_as_kpa",
      "comment": "Average floor area (m2)  is the total floor area of all dwellings
divided by their number.\r\n"
    },
    {
      "variable": "Tiedot",
      "value": "Ra_pt_as",
      "comment": "Dwellings in small houses are dwellings in detached small houses
(detached or semi-detached houses) or terraced and attached houses (comprising at least
three attached houses).\r\n"
    },
    {
      "variable": "Tiedot",
      "value": "Ra_kt_as",
      "comment": "Dwellings in blocks of flats are dwellings in residential blocks. They
include buildings with at least three flats of which at least two are located on top of
each other.\r\n"
    }
  ],
  "data": [
    {
      "key": [
        "00100",
        "Ra_as_kpa"
      ],
      "values": [
        "66.2"
      ]
    },
    {
      "key": [
        "00100",
        "Ra_pt_as"
      ],
      "values": [
        "2"
      ]
    },
    {
      "key": [
        "00100",
        "Ra_kt_as"
      ],
      "values": [
        "11766"
      ]
    },
    {
      "key": [
        "00120",
        "Ra_as_kpa"
      ],
      "values": [
        "69.2"
      ]
    },
    {
      "key": [
        "00120",
        "Ra_pt_as"
      ],
      "
```

```
      "values": [
        "7"
      ]
    },
    {
      "key": [
        "00120",
        "Ra_kt_as"
      ],
      "values": [
        "4414"
      ]
    }
  ]
}
```

Income data

We will again fetch this from Statistics Finland. This time we fetch the following from a different data set

- Median household income ("Tr_mtu")

Average household income, purchasing power and income categories would also we available here, but we will we use this data point only. We will utilize this either as min-max scaled or normalized.

Web URL:

http://pxnet2.stat.fi/PXWeb/pxweb/en/Postinumeroalueittainen_avoin_tieto/Postinumeroalueittainen_avoin_tieto__2018/paavo_5_tr_2018.px/table/tableViewLayout2/?rxid=39840011-c10c-4e00-8cd1-7015d2e09479

Service URL:

http://pxnet2.stat.fi/PXWeb/api/v1/en/Postinumeroalueittainen_avoin_tieto/2018/paavo_5_tr_2018.px

Example POST data:

```
{
  "query": [
    {
      "code": "Postinumeroalue",
      "selection": {
        "filter": "item",
        "values": [
          "00100",
          "00120"
        ]
      }
    },
    {
      "code": "Tiedot",
      "selection": {
        "filter": "item",
        "values": [
          "Tr_mtu"
        ]
      }
    }
  ],
  "response": {
    "format": "json"
  }
}
```

Example answer:

```
{
```

```
  "columns": [
    {
      "code": "Postinumeroalue",
      "text": "Postal code area",
      "comment": "2018 postal code areas\r\n",
      "type": "d"
    },
    {
      "code": "Tiedot",
      "text": "Data",
      "type": "d"
    },
    {
      "code": "Paavo - Open data by postal code area 2018",
      "text": "Paavo - Open data by postal code area 2018",
      "type": "c"
    }
  ],
  "comments": [
    {
      "variable": "Tiedot",
      "value": "Tr_mtu",
      "comment": "Median income of households (€) is obtained by listing households by
the amount of disposable monetary income. Median income is the income of the middle
household. An equal number of households remain on both sides of the middle
household.\r\n"
    }
  ],
  "data": [
    {
      "key": [
        "00100",
        "Tr_mtu"
      ],
      "values": [
        "38139"
      ]
    },
    {
      "key": [
        "00120",
        "Tr_mtu"
      ],
      "values": [
        "40165"
      ]
    }
  ]
}
```


Postal code data

I will use Bing Maps REST Services for geocoding. I already used this for Toronto assignment, and found out that the service is reliable and fast. We will be using geocoder library, which takes care of the details – I just need an API key, which I created according to Microsoft instructions. With a developer account, you can make up to 99500 API calls per day free.

Getting started:

https://msdn.microsoft.com/en-us/library/ff701702.aspx

Creating Bing Maps keys:

https://msdn.microsoft.com/library/ff428642.aspx

Once you have the Bing keys, you need to install geocoder library, which can then utilize a number of geocoding services:

```
!conda install -c conda-forge geocoder --yes
import geocoder
```

With this installed, and your API key available, this would get the address of my home postal area using Bing:

```
address = '00270, Helsinki, Finland'
g = geocoder.bing(address, key=BING_KEY)
print('Address: {}, latitude: {}, longitude: {}'.format(address,g.latlng[0],
g.latlng[1]))
➔ Address: 00270, Helsinki, Finland, latitude: 60.1942901611328, longitude:
24.9036903381348
```

Venues/services data

As required in the assignment, we will be using Foursquare for some purpose – we will be finding venue data from Foursquare, just as we did with New York and Toronto assignments. We are interested in the following data points, which I have marked in red in the response:

- Venue name
- Venue coordinates (lat, long)
- Venue's first category name

We will be calculating the relative frequencies different categories of venues for each postal code area.

To use Foursquare APIs, you need a "Client ID" and "Client Secret". These can be gotten from Foursquare developer site. You also need to specify a version for the api, such as "20180605".

Note that some of the API calls are "regular", and some are "premium". You can use many times more "regular" calls than "premium" calls with a free developer account. We will be using just one regular call, namely "explore", which will return venues within a specified radius from a certain coordinate.

For example, the following call would return us a large JSON data structure showing venues around the address we just geocoded – namely, the venues around the place I live. I put a limit of 2 to keep the answer short:

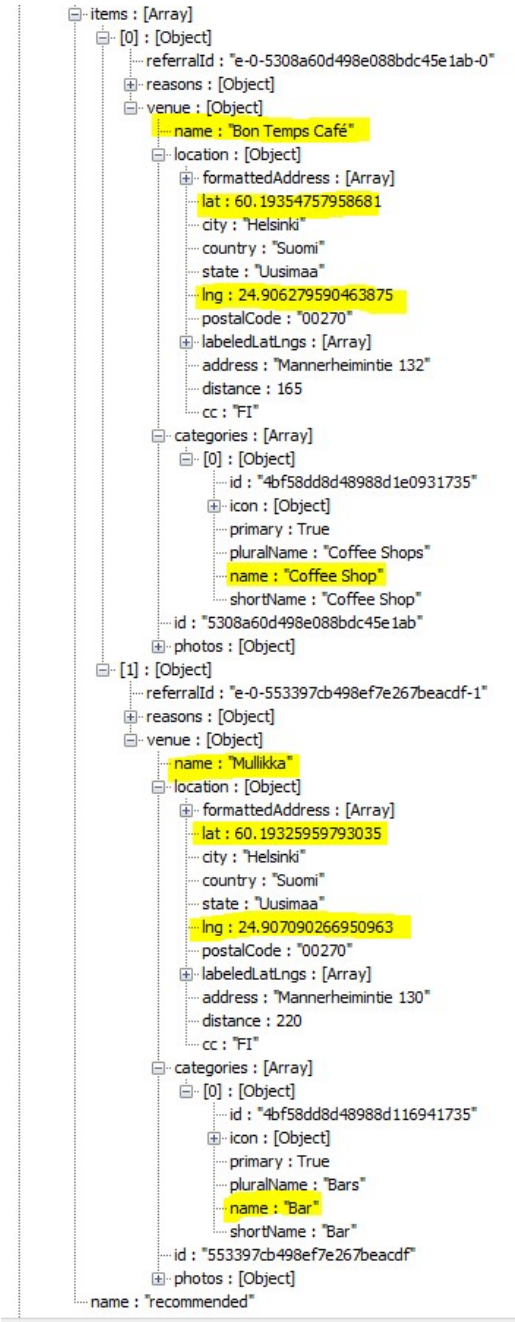
```
url =
'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{
}&radius={}&limit={}'.format(
            FOURSQUARE_CLIENT_ID,
            FOURSQUARE_CLIENT_SECRET,
            "20180605",
            60.1942901611328,
            24.9036903381348,
            500,
            2)
```

The answer is a little messy JSON file – but I can immediately see that my favourite local Café (Bon Temps Café) and bar (Mullikka) can be found.

```
{'response': {'suggestedFilters': {'filters': [{'name': 'Open now', 'key': 'openNow'}],
'header': 'Tap to show:'}, 'totalResults': 20, 'headerFullLocation': 'Meilahti,
Helsinki', 'headerLocationGranularity': 'neighborhood', 'groups': [{'type': 'Recommended
Places', 'items': [{'referralId': 'e-0-5308a60d498e088bdc45e1ab-0', 'reasons': {'items':
[{'reasonName': 'globalInteractionReason', 'type': 'general', 'summary': 'This spot is
popular'}], 'count': 0}, 'venue': {'name': 'Bon Temps Café', 'location':
{'formattedAddress': ['Mannerheimintie 132', '00270 Helsinki', 'Suomi'], 'lat':
60.19354757958681, 'city': 'Helsinki', 'country': 'Suomi', 'state': 'Uusimaa', 'lng':
24.906279590463875, 'postalCode': '00270', 'labeledLatLngs': [{'lat': 60.19354757958681,
'label': 'display', 'lng': 24.906279590463875}], 'address': 'Mannerheimintie 132',
```

'distance': 165, 'cc': 'FI'}, 'categories': [{'id': '4bf58dd8d48988d1e0931735', 'icon': {'prefix': 'https://ss3.4sqi.net/img/categories_v2/food/coffeeshop_', 'suffix': '.png'}, 'primary': True, 'pluralName': 'Coffee Shops', 'name': 'Coffee Shop', 'shortName': 'Coffee Shop'}], 'id': '5308a60d498e088bdc45e1ab', 'photos': {'groups': [], 'count': 0}}}, {'referralId': 'e-0-553397cb498ef7e267beacdf-1', 'reasons': {'items': [{'reasonName': 'globalInteractionReason', 'type': 'general', 'summary': 'This spot is popular'}], 'count': 0}, 'venue': {'name': 'Mullikka', 'location': {'formattedAddress': ['Mannerheimintie 130', '00270 Helsinki', 'Suomi'], 'lat': 60.19325959793035, 'city': 'Helsinki', 'country': 'Suomi', 'state': 'Uusimaa', 'lng': 24.907090266950963, 'postalCode': '00270', 'labeledLatLngs': [{'lat': 60.19325959793035, 'label': 'display', 'lng': 24.907090266950963}], 'address': 'Mannerheimintie 130', 'distance': 220, 'cc': 'FI'}, 'categories': [{'id': '4bf58dd8d48988d116941735', 'icon': {'prefix': 'https://ss3.4sqi.net/img/categories_v2/nightlife/pub_', 'suffix': '.png'}, 'primary': True, 'pluralName': 'Bars', 'name': 'Bar', 'shortName': 'Bar'}], 'id': '553397cb498ef7e267beacdf', 'photos': {'groups': [], 'count': 0}}}], 'name': 'recommended'}], 'suggestedBounds': {'ne': {'lat': 60.1987901656328, 'lng': 24.91272666883028}, 'sw': {'lat': 60.189790156632796, 'lng': 24.89465400743932}}, 'warning': {'text': "There aren't a lot of results near you. Try something more general, reset your filters, or expand the search area."}, 'headerLocation': 'Meilahti'}, 'meta': {'code': 200, 'requestId': '5c056c30db04f56ae9b81d70'}}

This picture shows the structure and the information we need a little bit better:



Methodology

The main statistical method we are using to perform the actual analysis is K-means clustering – we wish to cluster Helsinki neighborhoods to six different clusters according to their service offering and key socio-economic data.

In order to do cluster analysis, all data vectors should be of uniform scaling. We use 0..1 scaling for all data points. This means we will have to calculate relative shares of different occupational groups and housing types as well as do min-max normalization of average apartment size and median household income.

Exploring and preparing postal codes

Fetching postal codes from Helsinki Region Infoshare (HRI) was painless. The only point to note was the coding of postal code data in the service (possibly a feature of the JSON interface) – all other data sources use “00100” style of postal codes, that is, a string. HRI interface uses *numeric* postal codes, that is, in this

case, 100. So we have to make that 100 a string, and add “00” to get “00100” are required by other APIs. This is crucial, since I used postal code as the index of the DataFrame throughout, and the different sub-DataFrames are joined using postal codes as index.

It is worth noting, too, that all postal codes are not necessarily used. For example, postal code “00110” is not currently used. It was used in the past, but not at the moment. It may be used again in the future. I happen to know that all serious information systems in Finland need to know the “history of postal codes in Finland” – they can split (this is tricky, since this is then based on street address), or one postal code area can be combined with another at some point in time.

For the purposes of this study, it is sufficient to know that we are using the *current postal codes* in Helsinki, Finland. There are 84 postal codes currently in use in Helsinki. The end result is this DataFrame with postal codes and neighborhood names:

PostalCode	NeighbourhoodName
00100	Helsinki Keskusta - Etu-Töölö
00120	Punavuori
00130	Kaartinkaupunki
00140	Kaivopuisto - Ullanlinna
00150	Eira - Hernesaari

Exploring and preparing occupation data

Reading data from Statistics Finland (SF) postal code oriented database was made quite easy by the very nice web interface they are providing to their open data – once have to data you need on the screen, you can (from a different tab) see the ReST call you need to make in your program in order to get the data! I find this nice – you do not have to guess the search parameters, since they are shown to you.

That was the easy part. The more difficult problem for me was that SF ReST services do not return the data in any smart structure (for example, grouped by postal code), but in a list. That is, data points of postal code 00100, then 00120 and so on.

Luckily, Pandas provides DataFrame.pivot operation, which does just the trick. That is, we can move from this…

	PostalCode	OccupationCategory	NumberInThisOccupation
0	00100	Pt_vakiy	17868
1	00100	Pt_tyovy	10735
2	00100	Pt_tyott	864
3	00100	Pt_0_14	1773
4	00100	Pt_opisk	1258
5	00100	Pt_elakel	3295
6	00100	Pt_muut	807
7	00120	Pt_vakiy	7026
8	00120	Pt_tyovy	4187

…to this in one command:

OccupationCategory	Pt_0_14	Pt_elakel	Pt_muut	Pt_opisk	Pt_tyott	Pt_tyovy	Pt_vakiy
PostalCode							
00100	1773	3295	807	1258	864	10735	17868
00120	794	1246	365	434	350	4187	7026
00130	170	252	71	111	70	932	1536
00140	892	1494	429	508	383	4494	7817

What is left is translating the names into English, calculating the relative share of each group in a row and combining this with postal code data. We get this:

PostalCode	NeighbourhoodName	EmployedR	UnemployedR	ChildR	StudentR	OthersR	PensionerR
00100	Helsinki Keskusta - Etu-Töölö	0.552440	0.048355	0.099228	0.070405	0.045165	0.184408
00120	Punavuori	0.546114	0.049815	0.113009	0.061771	0.051950	0.177341
00130	Kaartinkaupunki	0.561198	0.045573	0.110677	0.072266	0.046224	0.164062
00140	Kaivopuisto - Ullanlinna	0.525905	0.048996	0.114110	0.064987	0.054880	0.191122
00150	Eira - Hernesaari	0.569631	0.057641	0.098075	0.060652	0.054414	0.159587

Exploring and preparing median household income

We will use another Statistics Finland data set for this. We use median income of households in postal code area as a data point. Handling of data follows the same principles as with the occupational data. The data point is then standard scaled and combined to our master data frame, resulting in this:

PostalCode	NeighbourhoodName	EmployedR	UnemployedR	ChildR	StudentR	OthersR	PensionerR	MedianHouseholdIncomeNorm
00100	Helsinki Keskusta - Etu-Töölö	0.552440	0.048355	0.099228	0.070405	0.045165	0.184408	0.135282
00120	Punavuori	0.546114	0.049815	0.113009	0.061771	0.051950	0.177341	0.311709
00130	Kaartinkaupunki	0.561198	0.045573	0.110677	0.072266	0.046224	0.164062	0.701486
00140	Kaivopuisto - Ullanlinna	0.525905	0.048996	0.114110	0.064987	0.054880	0.191122	0.397746
00150	Eira - Hernesaari	0.569631	0.057641	0.098075	0.060652	0.054414	0.159587	-0.284276

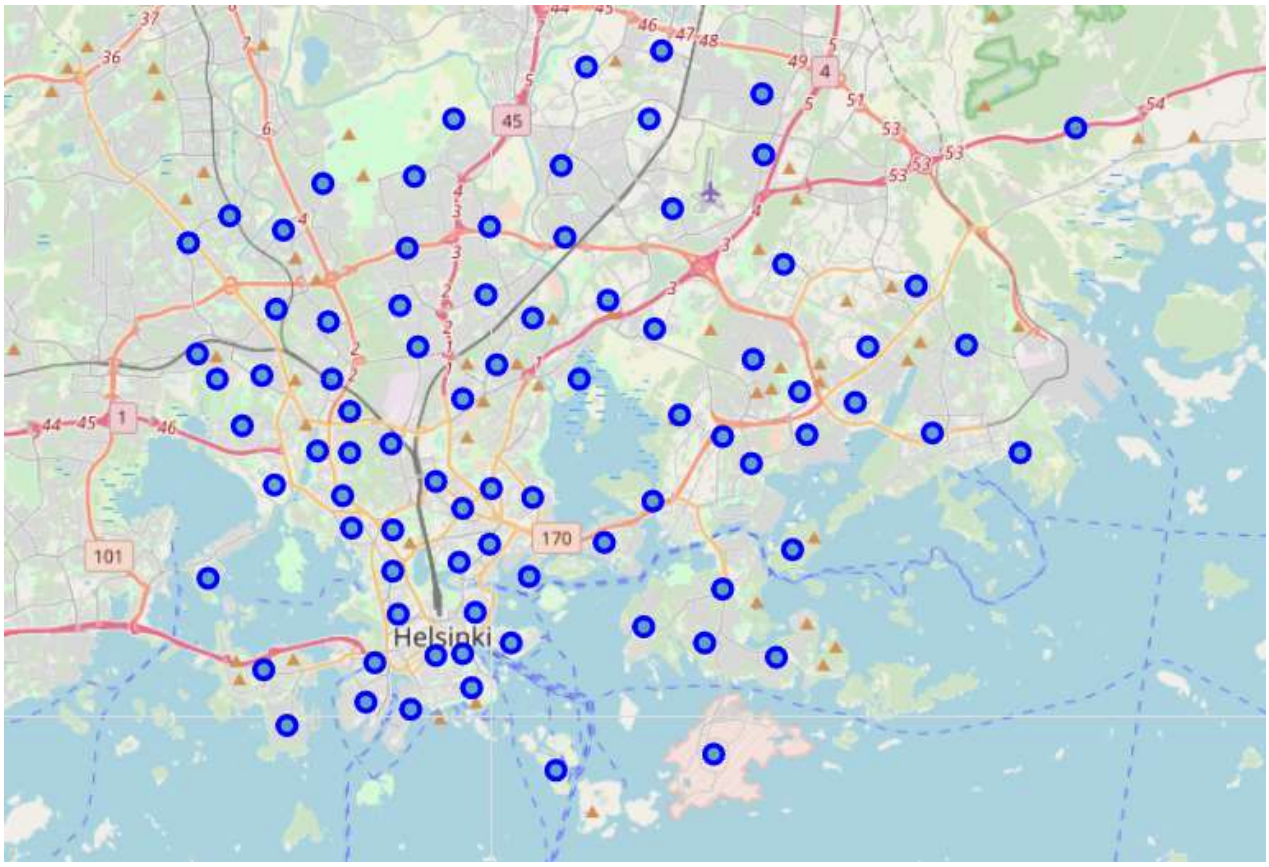
Exploring and preparing housing type and average size data

We continue the same process, adding housing type and average apartment size information, which is min-max scaled. We get the following – as we can see, there are no houses in some postal code areas in downtown Helsinki. This is also confirmed by experience – there really are no houses in Kaartinkaupunki, just apartment buildings (and public buildings, but they are not included here – they could very well be).

PostalCode	NeighbourhoodName	EmployedR	UnemployedR	ChildR	StudentR	OthersR	PensionerR	MedianHouseholdIncomeNorm	AverageFloorSizeR	DwellingsHouseR	DwellingsApartmentR
00100	Helsinki Keskusta - Etu-Töölö	0.552440	0.048355	0.099228	0.070405	0.045165	0.184408	0.135282	0.408390	0.000170	0.999830
00120	Punavuori	0.546114	0.049815	0.113009	0.061771	0.051950	0.177341	0.311709	0.426897	0.001583	0.998417
00130	Kaartinkaupunki	0.561198	0.045573	0.110677	0.072266	0.046224	0.164062	0.701486	0.462060	0.000000	1.000000
00140	Kaivopuisto - Ullanlinna	0.525905	0.048996	0.114110	0.064987	0.054880	0.191122	0.397746	0.455891	0.002580	0.997420
00150	Eira - Hernesaari	0.569631	0.057641	0.098075	0.060652	0.054414	0.159587	-0.284276	0.342998	0.004730	0.995270

Exploring postal code location data

Fetching postal code average locations was easy and efficient using Bing API as described in Data section. I mapped to postal codes on a Folium map to make sure that things look good – and they do. All postal codes are in proper places. The point to the farthest to east in no outlier – as it happened, the city of Helsinki bought a large area of land from the neighboring Sipoo some years ago. As we can see from the picture below, postal codes are actually a very good way to divide Helsinki to smaller areas.



Since some of the postal code locations are quite close to each other, there might be some overlap in venues. We actually could filter out the results of Foursquare venue searches based on postal code (postal code in included in venue locations in Foursquare data, but I chose not to do that. The services are near, if they are near, despite their postal code. When you walk the streets, you actually do not care about the postal code areas, they are arbitrary. From service consumption point of view, they are meaningless. Therefore, we stick to venues within a certain distance of the (mathematically calculated) central point of postal codes.

Postal code locations were then added to the master data frame, which now looks like this (getting wider and wider):

PostalCode	NeighbourhoodName	EmployedR	UnemployedR	ChildR	StudentR	OthersR	PensionerR	MedianHouseholdIncomeNorm	AverageFloorSizeR	DwellingsHouseR	DwellingsApartmentR	Latitude	Longitude
00100	Helsinki Keskusta - Etu-Töölö	0.552440	0.048355	0.099228	0.070405	0.045165	0.184408	0.135282	0.408390	0.000170	0.999830	60.172020	24.925289
00120	Punavuori	0.546114	0.049815	0.113009	0.061771	0.051950	0.177341	0.311709	0.426897	0.001583	0.998417	60.164116	24.939407
00130	Kaartinkaupunki	0.561198	0.045573	0.110677	0.072266	0.046224	0.164062	0.701486	0.462060	0.000000	1.000000	60.164246	24.949619
00140	Kaivopuisto - Ullanlinna	0.525905	0.048996	0.114110	0.064987	0.054880	0.191122	0.397746	0.455891	0.002580	0.997420	60.157925	24.953203
00150	Eira - Hernesaari	0.569631	0.057641	0.098075	0.060652	0.054414	0.159587	-0.284276	0.342998	0.004730	0.995270	60.153954	24.929998

Exploring venue data

Venue data was fetched from Foursquare with a radius of 500. This was then one-hot encoded, grouped by postal code area and venue type, and relative shares of venue types were calculated.

At this stage, we notice that one of the postal code areas (00890, far to the east) has no Foursquare venue data. We drop this postal code area from further analysis.

We then add this to our master data frame, resulting in really wide data set with lots of different venue categories. We notice that postal code area 00230 does have some Foursquare data, but no relevant information in Statistics Finland dataset, so we drop this postal code, too, from further analysis.

It can be seen from venue data that there is no proper Foursquare venue coverage outside downtown Helsinki. In many postal code areas, the most common venue type is “Bus Stop” – up to 75%. In addition, in downtown Helsinki, there are no bus stops, which really does not hold true. In some areas, there are only a few venues. Quality of Foursquare data in Helsinki is low. This is disappointing, because it means that Foursquare data does not really differentiate Helsinki suburbs at all. Luckily, we have to socio-economic data! If I had the choice (which I do not have in this assignment), I would definitely skip Foursquare data in a proper analysis.

Cluster analysis

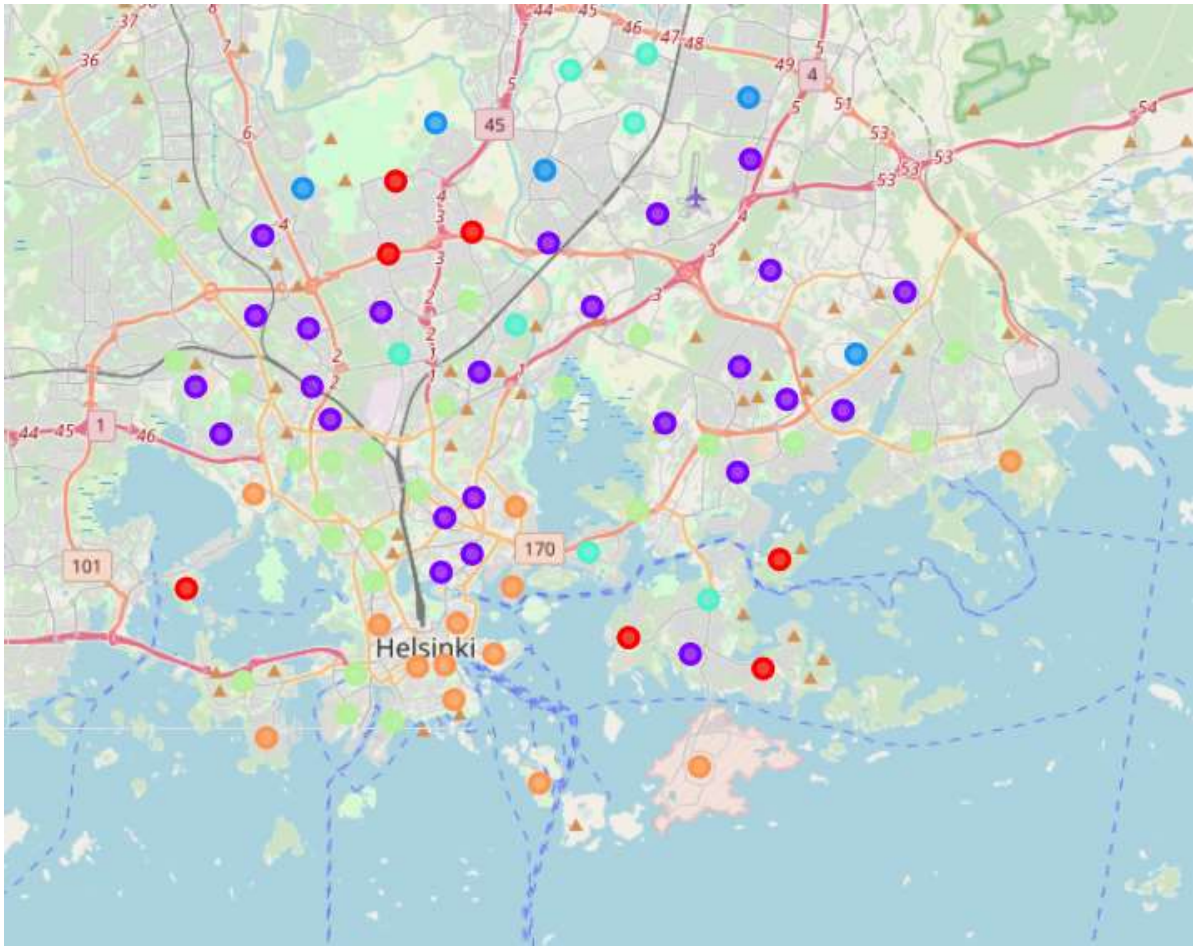
I selected K-means clustering as the main machine learning method to use. Its use is appropriate in this context – it answers our business problem. I tried DBSCAN, but that did not work out very well (only 1 cluster). I also tried hierarchical clustering, but got max three different clusters).

With K-means clustering, I am able to select the number of clusters. As a long-time Helsinki dweller, I *know* that given the data we have, we should get more variation. There are some areas with large houses and wealthy people. There are some areas with mainly smaller apartments, with lots of students and perhaps unemployed. Central Helsinki is much more expensive than the suburbs, apartment prices can be up to four times as much per square meter as they are in the suburbs.

I used K-means clustering with 6 clusters to see the variation. I got the results I was actually expecting with this data set.

Results

This shows the main results of the clustering to six different clusters



Discussion

Do these results make sense? I have been living in Helsinki for almost 30 years (I moved here to study when I was 20), and the results are very much aligned with be everyday experiences. I can rely on the results. For example, the regions in southern tip of Helsinki peninsula are expensive to live in, and there are many big flats.

Correlations exist between various variables used. Most of these feel quite natural. For example, median income and average house size have a correlation of 0.867, and median income and unemployment -0,732. Number of children in area and have 0.594 correlation to DwellingsHouseR – that is, many kids live in houses rather than apartments. Another interesting thing is that children and median income have 0.561 correlation – meaning that in Helsinki it well-to-do people who have children. Being a student or a pensioner do not have that great effect on anything (except pensioners and employment have a -0.653 correlation, which is quite natural). Correlations between venue types and other data fall between -0.25 and 0.25, majority being under 0.1. This is the correlation matrix between the socio-economic variables:

	EmployedR	UnemployedR	ChildR	StudentR	OthersR	PensionerR	MedianHouseholdIncomeNorm	AverageFloorSizeR	DwellingsHouseR	DwellingsApartmentR
EmployedR	1.000000	-0.246359	-0.405733	-0.208461	-0.205343	-0.652993	-0.021108	-0.389472	-0.336947	0.336947
UnemployedR	-0.246359	1.000000	-0.360091	0.114670	0.450477	0.113408	-0.732189	-0.565060	-0.409933	0.409933
ChildR	-0.405733	-0.360091	1.000000	0.182867	-0.308608	-0.252621	0.560558	0.586388	0.594327	-0.594327
StudentR	-0.208461	0.114670	0.182867	1.000000	-0.201543	-0.215605	-0.041624	0.046404	0.117908	-0.117908
OthersR	-0.205343	0.450477	-0.308608	-0.201543	1.000000	0.198215	-0.313530	-0.145860	-0.378389	0.378389
PensionerR	-0.652993	0.113408	-0.252621	-0.215605	0.198215	1.000000	-0.119505	0.179074	0.079686	-0.079686
MedianHouseholdIncomeNorm	-0.021108	-0.732189	0.560558	-0.041624	-0.313530	-0.119505	1.000000	0.867169	0.753957	-0.753957
AverageFloorSizeR	-0.389472	-0.565060	0.586388	0.046404	-0.145860	0.179074	0.867169	1.000000	0.830113	-0.830113
DwellingsHouseR	-0.336947	-0.409933	0.594327	0.117908	-0.378389	0.079686	0.753957	0.830113	1.000000	-1.000000
DwellingsApartmentR	0.336947	0.409933	-0.594327	-0.117908	0.378389	-0.079686	-0.753957	-0.830113	-1.000000	1.000000

Our business problem was the following:

- What kind of areas there are in Helsinki, Finland, in terms of available services and key socio-economic data?

To answer this question we can characterize the clusters formed as follows:

- Cluster 0 (Red): Large houses, very well-to do people, not that much services
- Cluster 1 (Purple): Small apartments, medium-to-low income, basic services
- Cluster 2 (Light blue): Large houses, medium income, basic services
- Cluster 3 (Cyan): Half apartments, half houses, medium plus income, decent services
- Cluster 4 (Lime): Apartments, medium income, good services
- Cluster 5 (Orange): Large apartments, well-to-do people, good services

We can imagine further data to color the picture a little more:

- Proper data to cover the practically missing data in Helsinki suburbs – we use Statistics Finland “workplace structure” data
- We could also add educational data of people from Statistics Finland
- One could also consider including data on native languages / countries of origin of people living in each area

Conclusion

So – what would we propose the IT expert planning moving to Helsinki? I would recommend places with good service levels. Orange, lime and cyan would be my first choices.

As can be seen from this study, Helsinki postal code areas can be effectively categorized given some socio-economic data and Foursquare venue data.

It was worth seeing that even though Foursquare is a big name in business, the quality of Foursquare data in Helsinki is quite poor. It does cover downtown Helsinki quite properly, but farther in the suburbs data is patchy, even non-existent. Some other data sources, on the other hand, fail on some postal codes. Deficiencies with data seem to present in all data sources.

The lesson learned is the following: always check our data sources when doing your studies. Do not trust a data source to contain all the data you would expect. Never trust a data source blindly.