Applied Data Science Capstone Project: Comparing Helsinki Neighborhoods

Table of Contents

Introduction 1
Business problem 1
Data used in the study 2
Postal code data 2
Occupational data 3
Housing data 7
Income data 9
Postal code data 11
Venues/services data 11

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.

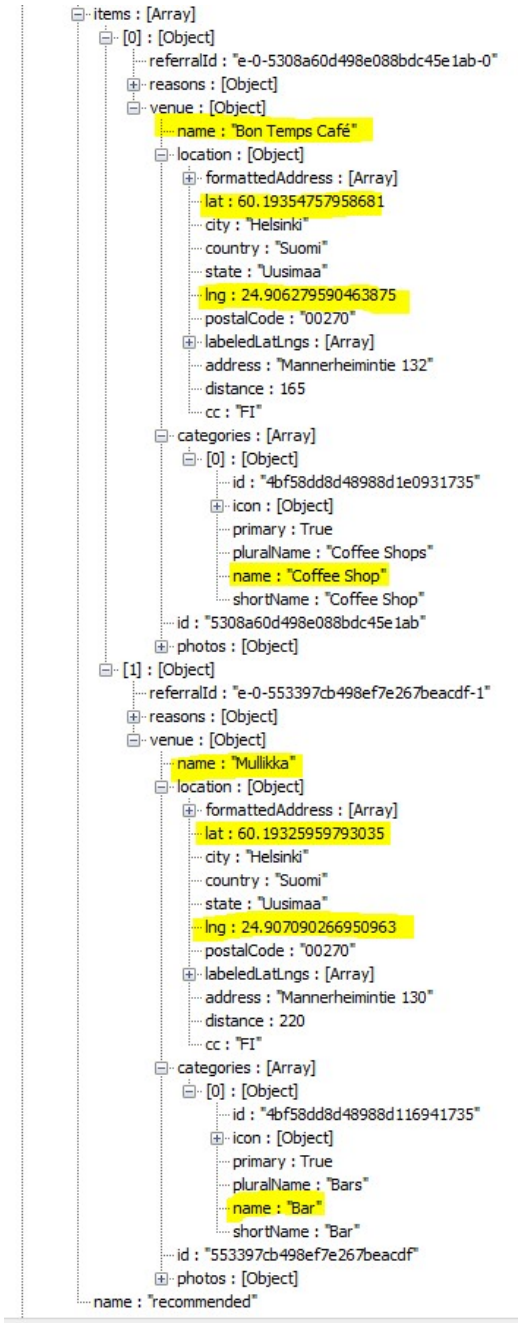
For example, the following call would return us a large JSON data structure showing venues around the address we just geocoded – namely, the venues around the place I live. I put a limit of 2 to keep the answer short:

```
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:


items : [Array]
[0] : [Object]
referralId : "e-0-5308a60d498e088bdc45e1ab-0"
reasons : [Object]
venue : [Object]
name : "Bon Temps Café"
location : [Object]
formattedAddress : [Array]
lat : 60.19354757958681
city : "Helsinki"
country : "Suomi"
state : "Uusimaa"
lng : 24.906279590463875
postalCode : "00270"
labeledLatLngs : [Array]
address : "Mannerheimintie 132"
distance : 165
cc : "FI"
categories : [Array]
[0] : [Object]
id : "4bf58dd8d48988d1e0931735"
icon : [Object]
primary : True
pluralName : "Coffee Shops"
name : "Coffee Shop"
shortName : "Coffee Shop"
id : "5308a60d498e088bdc45e1ab"
photos : [Object]
[1] : [Object]
referralId : "e-0-553397cb498ef7e267beacdf-1"
reasons : [Object]
venue : [Object]
name : "Mullikka"
location : [Object]
formattedAddress : [Array]
lat : 60.19325959793035
city : "Helsinki"
country : "Suomi"
state : "Uusimaa"
lng : 24.907090266950963
postalCode : "00270"
labeledLatLngs : [Array]
address : "Mannerheimintie 130"
distance : 220
cc : "FI"
categories : [Array]
[0] : [Object]
id : "4bf58dd8d48988d116941735"
icon : [Object]
primary : True
pluralName : "Bars"
name : "Bar"
shortName : "Bar"
id : "553397cb498ef7e267beacdf"
photos : [Object]
name : "recommended"