**Analyzing the Effects of Fintech on the Banking Industry with Foursquare Developers API**

**Ian Zelaya**

**14/10/2020**

**Introduction**

In the last couple of years, the fintech industry has revolutionized how the banking industry interacts with its customers. This trend began with movements such as the PayPal company, where transfers of money are made via online. Now, more and more traditional banking services are being transferred to a digital form. Some countries, such as Singapore, have further deepened the usage of fintech to the point where most transactions, either for groceries or clothing, are made through digital devices. In the past, banks required to invest a great portion of their money into infrastructure to maintain clients from different areas of a city. Now that fintech services are available, many people prefer to adapt to the technology rather than wasting time in line. This suggests that banks are now able to reduce their investments on infrastructure, gain their customers via online, and focus on more fintech. If this argument is true, there should be evidence of a decreased amount of bank establishments.

**Objective**

The objective of this project, besides understanding how fintech has affected the banking industry, is to use data in the Foursquare API. From the API, we will access information of all banking establishments in different cities. The best approach to evidence if fintech has significantly reduced the amount of bank establishments is through time series data. In this approach, the data would be easier to relate it with fintech innovation. Unfortunately, Foursquare API lacks time series data. As an alternative, we will try to evaluate three different hypotheses:

* Divergence between countries with high and low fintech innovation
* Divergence between big and small cities
* Divergence between neighborhoods of high and low income

The first hypothesis assumes that countries which are more developed in fintech will have lower levels of bank establishments than countries with lower fintech. The second hypothesis assumes a divergence between big and small cities, as in theory, big cities should become early adopters of fintech. At last, the third hypothesis assumes a divergence between high versus low income neighborhoods for the same reason as hypothesis two.

**Data Section**

The project will focus on finding divergence in the amount of bank establishments per population of a city or neighborhood. Our hypotheses will be measured in the following manner:

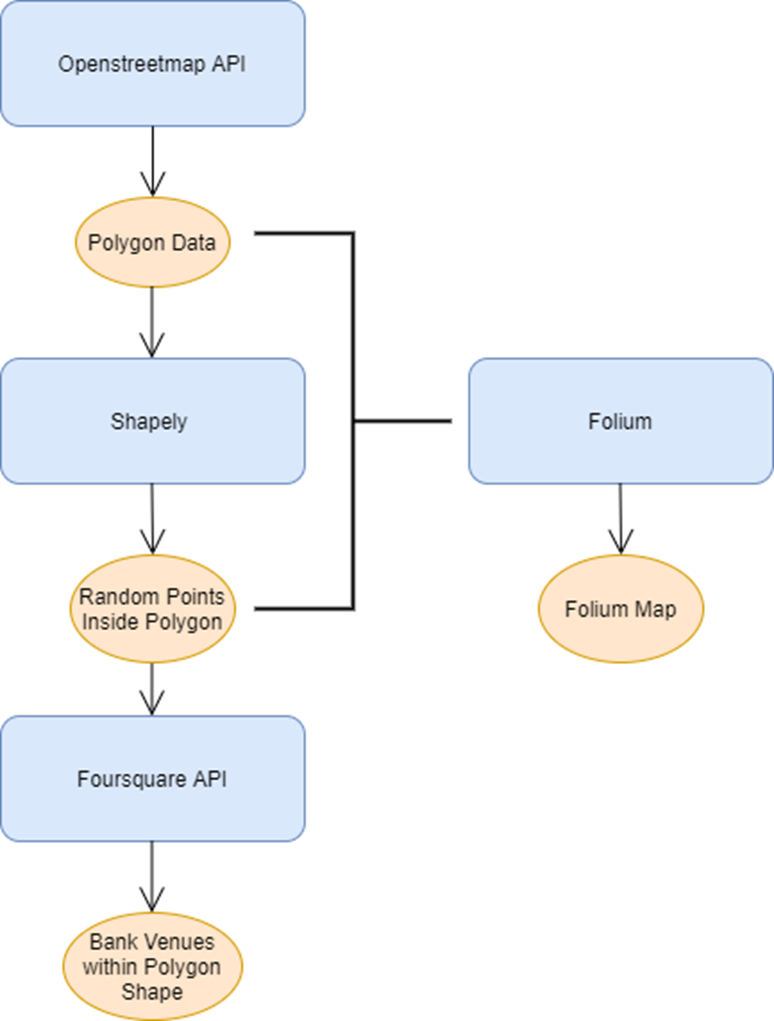
* Part I, Geographic difference: Bank establishments per city population
* Part II, Big and small cities: Bank establishments per city population
* Part III, High and low income: Bank establishments per neighborhood population

Foursquare API requires of a location input in order to return a list of venues close to that location. Fortunately, Openstreetmap API provides all the location data required to run Foursquare API. For part II and part III, we will also need to find data related to the population and income of cities and neighborhoods. After some research, I found out that Statistical Atlas displays this information through its website. The data they use is extracted from the United States Census Bureau, which is a reliable source. Web scraping will be necessary to extract the information from the Statistical Atlas Website.

**Methodology**

## **Part I: Countries with High and Low FinTech Innovation**

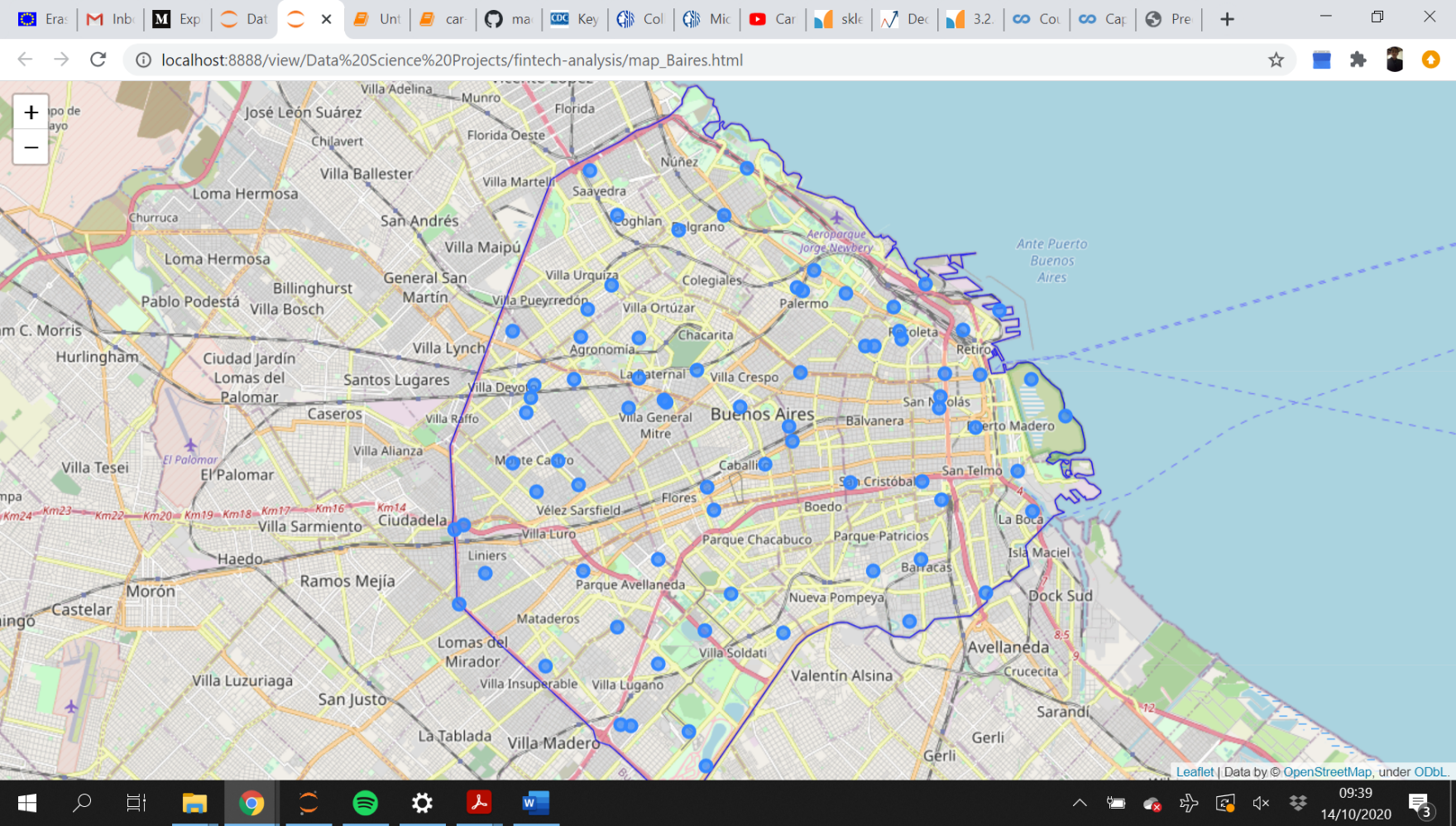
The first analysis will be the divergence between countries of high and low fintech innovation. We will use the capital city as a representation for the countries to analyze. There are 195 countries in the world. We could implement an analysis which considers every country of the world but that would imply a deeper data mining as some developing countries have insufficient or hard to access data. For that reason, it is necessary to find a way to sample countries which have high fintech and low fintech. The Institute for Financial Services Zug (IFZ) of the Lucerne University of Applied Sciences has conducted a reliable and comprehensive [research](https://blog.hslu.ch/retailbanking/files/2019/03/IFZ-FinTech-Study-2019_Switzerland.pdf) to identify the regions which have the highest levels of fintech. They evaluated the political, economic, technological, and social factors in order to create a fintech ranking. In the top of the ranking, Singapore, Zurich, and Amsterdam were rated the highest. From the 33 cities analyzed, Buenos Aires, Mumbai and Sao Paulo were rated the lowest. For this project, we will use the three highest rated countries as samples for the high fintech countries and the bottom three as samples for the low fintech countries.

Now that we have defined the cities to analyze, we must determine the way to extract the number of banks establishments per city. To execute this task, we will make usage of GeoJSON data. A GeoJSON file is a format for encoding a variety of geographic data structures. In our case, we can create GeoJSON files that represent the boundaries of a city or neighborhood through data structures such as Polygons and Multipolygons. Fortunately for us, Openstreetmap API has already constructed these Polygons and Multipolygons for cities and neighborhoods. We will extract this data through the Geocoder library as it has an easy implementation to manipulate the Openstreetmap API data. The following diagram illustrates the data flow process and will detail the steps in the next ****paragraph. Here is the diagram:

First, we begin by extracting the city's polygon through the Openstreetmap API. Using the polygon, we will create random points inside the city. The reason for creating these random points is that Foursquare API has a limit of 100 venues to return for any location given. So, for example, if we analyze a radius of 5KM around a location and there are more than 100 venues in that radius, we won't be able to extract all the venues. By generating random points inside the polygon of a city, we will be able to extract all the banks venues within. The Shapely library has a function named "bounds" which serves perfectly to implement this random data. Basically, the bounds function creates random points within maximum X and Y values. In this case, the X and Y values are the maximum and minimum latitude and longitude of a polygon. Finally, we are ready to input the random points into Foursquare API and return the bank venues. Note that many of the random points will return repeated bank establishments, but this will not be a problem as a simple filtering of duplicated longitude and latitude will solve the issue. We will also be able to generate a geographic map with Folium Library to visualize the random points or bank venues inside the polygon.

The analyses of this project can be considered as an iterative analysis for each city or neighborhood. As a result, the generation of functions to automate processes will become useful. The function "random\_point\_generator" executes the process from the data flow diagram up to the Shapely random points generation. The function requires as input a name or list of names of cities and has the option to specify the number of random points to generate. It also accepts data in a neighborhood format, but that will be explained in part III of this project. As output, the function generates a Python dictionary with keys for each city specified in the list. At the same time, "random\_point\_generator" saves the Folium map and the Polygon GeoJSON file in case the Openstreetmap API returns an error (which is common) while executing the for loop. Once we have the Python dictionary with the random points for each city, we continue the process by using the random points to extract the bank venues. However, before continuing the analysis, it is a good idea to save Python dictionary if we ever want to use it for further analysis. The function "save\_dict\_maps" saves a dictionary to the local file in a pickle format and declares a variable with the dictionary for immediate usage. The pickle format is a Python-specifit data format and is one of the highest and fastest compression formats to use in Python. The disadvantage, however, is that the pickle format is not available for usage in other programming languages. In this case, the pickle format is the appropiate option to save data files as we will use it only internally. For verification purposes, we will display the maps generated for each city with their respective random data points. Here we can see the effectiveness of using Shapely's "bounds" function as every data point is generated inside the polygon. As seen in the pictures below, the random points cover the area inside the polygon.

The following is an example of the map of Buenos Aires with its respective random points:

****

Next, the "venue\_extractor" function will convert the random points into the desired bank venues. To implement this process, the function applies a nested for loop to apply the analysis to each data point from each city. Inside the nested for loop, the function executes a get request from the Foursquare API to extract all bank venues within a radius of 5KM from the specific data point. The API responds by sharing the bank name, location and more features related to every bank venue. We will only extract the bank name, latitude and longitude to later eliminate duplicated bank venues by comparing repeated latitude and longitude values.

An interesting library function used for "venue\_extractor" was the "from\_iterable" function from the itertools library. With the "from\_iterable" function, we are able to convert a group of Python lists into a single list. In our case, we use this function to group the list of bank venues for each city. Similar to the function "random\_point\_generator", this function also accepts dictionaries for neighborhood analysis, but that will later be explained in part III. We now have the bank establishments from all cities and converted them into dataframes for each city. However, there is duplicated data as many of the random points generated have common bank venues in their radius. To solve this problem, we use the "drop\_duplicated" function from Pandas. We will then have a dataframe with unique bank establishments and now we are able to count the number of establishments with the "len" built-in function of Python. Then, we construct a new dataframe with the number of banks venues, population and banks per million of population. Here we sort values with pandas and analyze the results.

## **Part II: Big and Small Cities Analysis**

In part II, we will analyze the difference of bank venues in big versus small cities. To continue with the analysis, we must first determine an appropiate filter to extract a sample of cities which will represent both groups. I decided to implement this analysis by controlling the country factor. In other words, I will select cities from the same country to avoid any data disturbance from differences between countries. The United States is an attractive option due to the extensive public data of the country and at the same time has an ample category of cities. Now that we selected the country to analyze, we will proceed to start part II.

To begin, we need a list of cities within the U.S. and extract the biggest and smallest cities. To extract this list, we will use the list of cities of the U.S. published at Wikipedia, and their respective population. The library BeautifulSoup is a Python package for parsing HTML documents and will be of importance for our data extraction. With BeautifulSoup, we can extract sections, subsections or items from the HTML document. In our case, we want to extract the table section from the website which contains the list of cities, their population and more related data. Once we have the table defined within a variable in Python, we can further analyze the table to find items with characteristics. After reading the HTML code on the website (right-click and inspect option with Google Chrome), we can find the code that differentiates the html elements we want to extract from any other html elements. For the cities name data, this code is a hyperlink attribute and for the population data, the unique code is the style attribute. Having the unique attributes for each data feature, we are now able to apply a for loop in the table and extract the city name and population for each row (or each city). After converting all the data into a dataframe, we order the list of cities by the population column and extract the top 3 and bottom 3. As we can see in the modified dataframe, the top 3 cities are New York, Los Angeles and Chicago. For the bottom 3, we obtained Clinton, Bend and Woodbridge:

|  | **City** | **Population** |
| --- | --- | --- |
| **0** | New York | 8336817.0 |
| **1** | Los Angeles | 3979576.0 |
| **2** | Chicago | 2693976.0 |
| **3** | Clinton | 100471.0 |
| **4** | Bend | 100421.0 |
| **5** | Woodbridge | 100145.0 |

## Now we will use the list of cities as input for the "random\_point\_generator" and save the random points generated for each city. In the same way as part I, we visualize the random points to confirm that they are within the polygon. Then, we will use the dictionary of random points for the "venue\_extractor" function and extract the bank venues for each city.

## **Part III: Neighborhoods of High and Low Income**

## In this section, we will evaluate the differences between neighborhoods of high and low income. For this occasion, we will use a research paper published by the Federal Reserve Bank of New York to decide what cities to analyze. The research paper focuses on analyzing the income gaps in different cities of the U.S. and unsurprisingly, cities with high levels of urbanization demonstrated the highest levels of income gap. In particular, New York, Washington, Chicago, Houston, and Los Angeles were among the cities with the highest income gap.

Then, we will apply this function to the rest of cities. Once extracted the data, we will go ahead and analyze the neighborhoods with highest and lowest income. Pandas has a useful function called "sort\_values" which orders the rows of a Dataframe in accordance to a particular column values. In our case, the ordering of rows will be in accordance to the "Income" column.

New-York's neighborhoods with highest median household income:

|  |  |  |  |
| --- | --- | --- | --- |
| **City** | **Neighborhood** | **Population** | **Household Income** |
| New-York | Sutton-Place | 21830 | 154995 |
| New-York | Battery-Park | 13386 | 164676 |
| New-York | Carnegie-Hill | 15600 | 179100 |
| New-York | Tribeca | 19794 | 193906 |
| New-York | DUMBO | 3033 | 225120 |

**Results**

For part I, as we can in the final dataframe, by excluding Mumbai and Zurich, there seems to be a difference in bank venues for cities of high and low fintech innovation. Sao Paulo and Buenos Aires averaged a ratio of 134 while Singapore and Amsterdam averaged a ratio of 111. The results are tempting at most, as we would need further analysis to make statistical conclusions. The reason for such a low ratio from Mumbai could be their overpopulation and, for Zurich, the reason for a high ratio could be their small but concentrated area as a financial center.

Final dataframe from part I:

| **Cities** | **Banks** | **Population (Million)** | **Banks/Population** |
| --- | --- | --- | --- |
| **Mumbai, India** | 754 | 18.400000 | 40.978261 |
| **Amsterdam, Netherlands** | 89 | 0.821000 | 108.404385 |
| **Singapore** | 665 | 5.850000 | 113.675214 |
| **Buenos Aires, Argentina** | 382 | 2.890000 | 132.179931 |
| **Sao Paulo, Brazil** | 1654 | 12.180000 | 135.796388 |
| **Zurich, Switzerland** | 68 | 0.402762 | 168.834200 |

For part II, we can see a reduced number of banks per population for big cities in relationship to small cities. Except for Clinton, the trend seems to support hypothesis 2. The data also suggests that small cities have an abundance of banks which could reduce the incentive to adapt to fintech.

|  |  |  |
| --- | --- | --- |
| **Cities** | **Banks** | **Population** | **Banks/Population (per Million)** |
| **New York** | 1679 | 8336817.0 | 201.395808 |
| **Clinton** | 21 | 100471.0 | 209.015537 |
| **Los Angeles** | 1308 | 3979576.0 | 328.678231 |
| **Chicago** | 1170 | 2693976.0 | 434.302310 |
| **Bend** | 53 | 100421.0 | 527.778054 |
| **Woodbridge** | 108 | 100145.0 | 1078.436267 |

For part III, we can see a clear difference between neighborhoods of high and low income. The higher the income of a neighborhood, the higher the number of bank establishments. This would go against the third hypothesis. A reason for this result may be that high-income neighborhoods are closer to business sections of the city, while low income neighborhoods are farther away.

|  |  |  |
| --- | --- | --- |
|  | **Neighborhood** | **Household Income** | **Bank Establishments/Population (per Thousand)** |
| **5** | West Farms | 22652 | 5.539246 |
| **2** | Morris Heights | 23172 | 2.440096 |
| **3** | Mott Haven | 23195 | 1.946245 |
| **0** | Carnegie Hill | 179100 | 8.205128 |
| **4** | Tribeca | 193906 | 6.214004 |
| **1** | DUMBO | 225120 | 32.970656 |

**Discussion**

After the research, it is clear that more analysis is required to make more solid conclusions. In particular, it will be of great importance to analyze time series of a city, neighborhood or country and compare the growth of fintech investment with the number of bank establishments. In this way, we will understand how the number of banks has fluctuated across time and how this relates to fintech investment. At the same time, Foursquare API has an insufficient number of free requests to develop an automated program based on the code from this project. For this reason, an issue was that while testing the code, the limited free requests blocked further requests and it was required to wait for a day or create a new account. At the same time Openstreetmap fails frequently. Fortunately, there exists better geodata APIs with bound data such as Google API.

**Conclusion**

* The results for hypothesis I suggest that fintech innovation may influence the number of bank establishments in a city, but other factors such as overpopulation will also have an influence on the number of banks per capita.
* The results for hypothesis II suggest that the size of a city affects the number of bank establishments per capita. The bigger the city, the lower the number of banks. However, this result may be due to overpopulation and not because of fintech innovation.
* The results for hypothesis III suggest that the level of income of a neighborhood causes a change in the number of bank establishments per capita. The higher the income in a neighborhood, the greater number of banks establishments. However, this result may be due to a closer distance of high-income neighborhood to the business sector of a city in comparison with low-income neighborhood.