Group Project Report

Violazioni ai regolamenti comunali 2017

Germinario Federica, Micozzi Eleonora, Pisani Alessandro

Torino, 06/06/2019

OVERVIEW

For our project, we chose to develop one of the ideas suggested by the professor. We used the "Violazioni regolamenti" dataset, which contains data on the fines issued in the metropolitan area of Turin in the year 2017.

We decided to answer the question **"In which neighbourhoods of Turin were more fines issued, and what type of fines?"**

Therefore, we decided to create a visualization with a **choropleth map** to represent the density of crimes in each of Turin's 24 neighbourhoods. We also included a **bar graph** which models the number of fines per neighbourhood. Finally, we concentrated on the 3 types of fines that were more frequently found, and modelled them on a map on 3 layers, so that they could be visible both alone or together in combination.

METHOD

We split the design in various parts, and we each developed a certain section.

Design choice

After looking at the dataset, we chose to divide the visualization into 3 main parts:

- A choropleth map, which models the number of fines for each neighbourhood, showing the different densities in various areas;
- A bar graph, to represent more clearly the number of fines in each neighbourhood;
- Three maps to represent the densities of fines regarding the 3 more frequent types found in the dataset.

We decided to use choropleth maps because they were the best way to give an immediate impression of the different density of fines, based on the intensity of the colour. We believe they delivered the message effectively, and also offered an overall picture.

We chose to use a bar graph because, having to report data for 24 different neighbourhoods, any other representation(i.e. pie chart) would turn out to be too crowded, while with the bar graph we believe we were able to maintain the design clear while representing all the data needed.

We believe this approach exhaustively answered to the question we posed, and was clear enough without overcrowding the web page.

Data Handling

We started with the handling of the data. The dataset we were using was divided in 2 CSV files containing data for the fines given throughout the year 2017.

Since we decided to create a choropleth map, we needed to transform the data we had to suit our needs. In fact, the creation of the choropleth map required us to have the coordinates of each of the places where the fines were issued. This was quite difficult since we only had the street name and street number of the address, but no postal code (CAP) or coordinates information.

We had some trouble doing this: in the beginning, we thought of mapping the streets into neighbourhoods using Excel. We tried doing so with the aid of an additional CSV file, which contained the names of all the streets found in each neighbourhood. The problem with this approach is that it was not at all reliable, because there is no official document containing the streets for each neighbourhood, so we were using data from the web which was not trustworthy.

Also, we could only find data for the more important streets, but in our dataset there were many minor street, and we had no information on those. Hence, at the end, we had many rows with no information, and we would have had to try and figure them out by hand.

Therefore, we decided to change approach and proceeded to use a GIS (Geographic Information System) software.

We found that the university provides the students with a license for "ESRI ArcGIS Platform", which is an application for geocoding. It takes in a CSV file, and based on country, city, address and street number, it is able to return the coordinates of the address (longitude and latitude). This solved our problem. We exported the coordinates in another CSV file, and we were able to work on them to create the map.

Choropleth map

In order to obtain our choropleth map we used Python and a package called Folium which allows to generate maps for the browser through the use of leaflet.js, which is an open source javascript library for interactive maps. Folium builds on the data wrangling strengths of the Python ecosystem and the mapping strengths of the leaflet.js library; this allowed us to manipulate our data in Python, then visualize it in on a Leaflet map via folium.

Our first idea was to use choropleth maps provided by Plotly, however we found out that Plotly does not provide a map containing Turin districts, so we needed to find another solution. In order to solve this issue we used "GeoJson", which is a format for encoding a variety of geographic data structures. We created a set of polygons to outline the borders of the districts of the city of Turin and to plot them on our choropleth map. We did it through the website www.geojson.io which uses a kind of GUI to create this data structure: this is how we created the file "*MAP.json"*.

We found each district area from the web site "www.quartieri.torino.it/" and we replicated them with as much precision as possible. However this choice for the geojson file structure lead to a discrepancy between the district division taken from the quartieri.torino web site with respect to the one provided by ESRI ArcGIS Platform. So we decided to keep the division provided by the website that we had already implemented, and use a function to check how many of the couples (longitude, latitude) were inside each district.

At this point we needed another file, a .csv file, with only two columns, the name of the district and the associated number of violations. To create this file we used a javascript script which, by using a function called “inside”, checks if a certain couple of points (longitude, latitude) fall inside the previously created polygons and, if this is the case ,returns a 1, otherwise it returns a 0.

Therefore we looped through our .csv file containing all the information regarding latitude, longitude, number of violations and so on, and for each couple of coordinates we checked in which of the districts present in the MAP.json file they were. Then we created an associative array containing as keys the names of the district and as values the number of violations, so we updated a counter starting from one associated to each value every time we found a match with the function “inside”. However we believe it is important to highlight the fact that the counter was not updated by one at each match, but it was updated of a value contained in the 15^{th} column of our csv file, because in each day and in each address we had cases of more than one violations. Eventually, starting from our associative array, we created a new .csv file containing the couple [District, no. Violations].

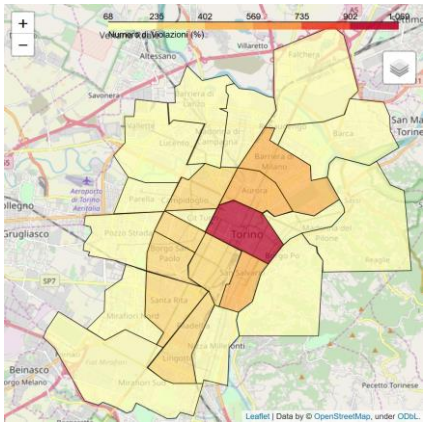
For the creation of our map we first used the folium library, setting the coordinates of the center point of our map and the initial zoom, in order to have a good visualization of our data.

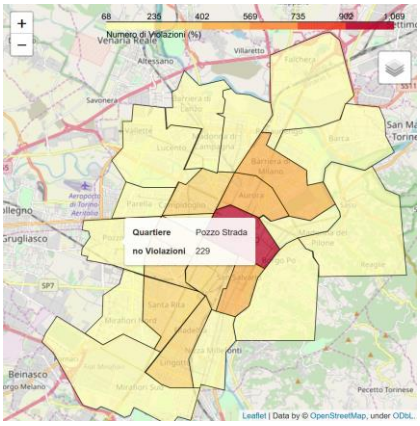
Then we used an analysis tool named “pandas”, which is an open source library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language: we basically used it to analyse the csv file.

Using the folium method named “choropleth” we created our choropleth map based on the polygons present in the “MAP.json” file and the data present in the csv file mentioned above, we also added a tooltip for each district in order to have a small pop-up box when the user moves the mouse pointer over a district, visualizing with more accuracy the information associated to them. To do so, however, we had to modify our file “MAP.json” in order to add to the property field what we wanted to display, so we read the json file and the csv file, we converted them into python dict and we added to the geojson dict the fields needed and then created a new geojson file updated with these fields.

Before plotting our map we also added a control layer which allows to activate and deactivate interactively the layers associated with the map, such as the choropleth layer or the tooltip layer. We added also a zoom control to zoom in and out and a legend associated with the map.

Eventually we created an html page displaying our map, obviously embedded with javascript code as well. So our goal at this point was to include this map inside our web page. The result we got was the following:

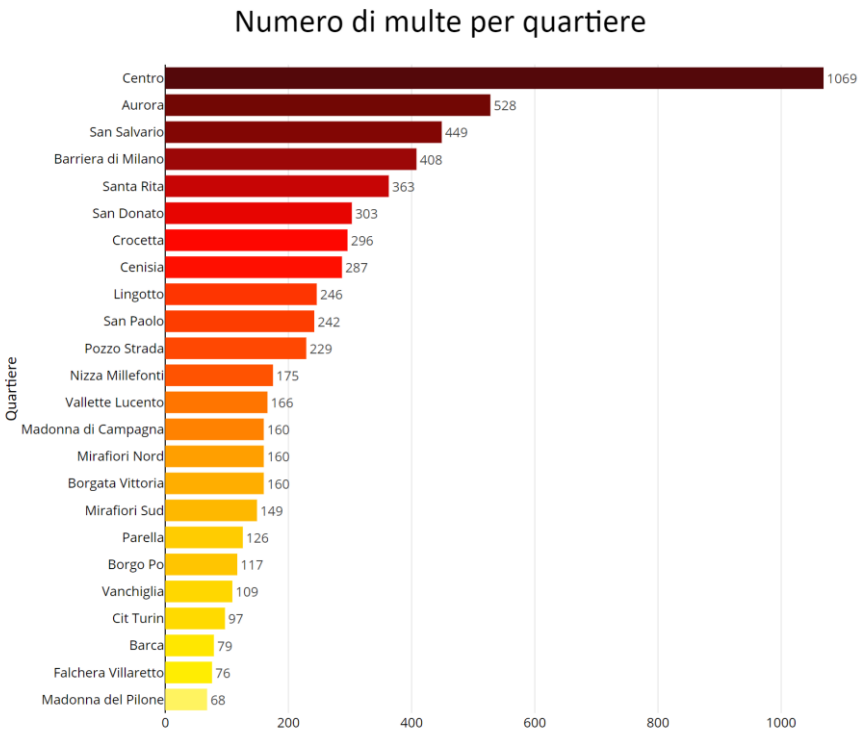




In the second picture we hover our cursor over ‘Pozzo Strada’ district.

Bar graph

We also included in our design a bar graph which contains information on how many fines were issued in each neighbourhood.



To produce the graph, we used the “Plotly” library for JavaScript.

We first created two arrays, one containing the name of the neighbourhood and one containing the number of fines issued in that neighbourhood. Hence, for each position, the arrays contained the name of a neighbourhood and the respective number of fines. We sorted the arrays in discending order, to create a bar graph going from the longest bar to the shortest.

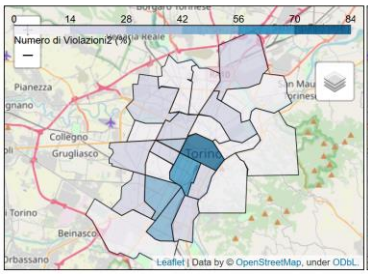
Then, we used “Plotly” to plot the bar graph, with the neighbourhoods on the y axis and the number of fines on the x axis. We chose the horizontal configuration for the bar graph since this way we were able to fit all the names of the neighbourhoods on the graph horizontally, instead of having them vertical or diagonal, which would have been less readable. Also, we chose colours whose intensity gradually decreases from top to bottom, to further emphasize the difference in the number of fines.

Other maps

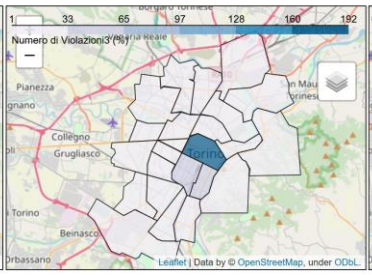
We decided to add other three maps to out page, each containing a specific violation. We chose the most recurrent violations and among them the most interesting to be studied. In order to implement them we used the same logic explained above, we only modified the javascript piece of code used to generate the csv file containing the couple [District, no. Violations], in order to add filters to filter only the three most present and interesting district violations. In addition we changed the colour of the choropleth map in order to underline the difference between the first graphs containing all violations with respect to those containing only a specific one.

The obtained result is the following:

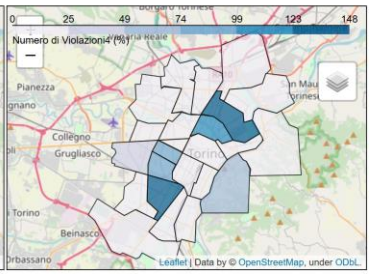
Sanctions and late payment fees



Fee evasion



Driving ban for motor vehicles



CONCLUSIONS

In conclusion we were able to answer to our question through the analysis and visualization of the dataset. Going more into detail, we noticed that strickingly the number of violations decreases radially from the center towards the suburbs of Turin.

Based on this it can be seen how it is dispelled the myth of a large number of violations in the suburbs.

However from the visualization performed with the three smaller maps it can be seen how the trend is not equal for all kind of violations, indeed having regard to the 'Driving Ban for motor vehicles' map it can be seen how the trend is reversed.

WEBSITE REFERENCE

https://plot.ly

https://pro.arcgis.com/en/pro-app/help/data/tables/export-tables.htm

https://pro.arcgis.com/en/pro-app/help/data/geocoding/tutorial-geocode-a-table-of-addresses.htm

https://python-visualization.github.io/folium/

http://geojson.io/#map=2/20.0/0.0

https://pandas.pydata.org/

http://www.quartieri.torino.it/