The Austin Project

Business Intelligence

Authors: Mikaela Pisani, Marcin Grzechowiak, Roger Valdez

Introduction

Different datasets from Austin Local Government and the U.S Government were merged together in Python and visualized in the Tableau Software. Five different data sets were used in the project:

- Mixed Beverage Gross Receipts
- Real-Time Traffic Incident Reports
- Housing Market Analysis Data by Zip Code
- Austin Water - Residential Water Consumption
- Food Establishment Inspection Scores

In order to merge mentioned above datasets, the common column, zip-codes were used.

Analysis of Data

What questions can these data potentially answer?

This data could be useful in a few different perspectives. The first perspective being that of a business wanting to expand in Austin, Texas. For instance, a large alcohol distributor could choose a location where alcohol sales are the highest based on the Zip code. For a person interested in moving to the Austin area information about housing affordability based on occupation could be useful for a social perspective. In this case a tech worker observing the high amounts of units for sale or rent could influence their decision to move the area. Lastly, it is believed the local city government can benefit the most by information related to the zip codes. With the information provided about different types of crashes and where they occur, could help transportation by looking at the areas most affected in order to prevent or reduce traffic accidents. The city government could potentially find use with the information about income and affordability by locating areas that are low in income and are not affordable. Having this information available will allow the city government to make decisions to help affected areas. For improvements we would suggest to the local government to collect more data on the suburbs of Austin because most of the data collected is based on the city center and not so much on the suburbs.

What are the potential valuable data items exists within the data?

Almost every CSV file found contained a column related to zip codes of Austin Texas. We found this to be the most valuable variable because we wanted to relate the data to a location. The lowest level of granularity in the data set are Zip Codes. This is because the data contains sensitive information such as income and it is understandable that in order to protect people's privacy data is aggregated to this level. From the Zip code, patterns were seen in areas such as traffic, income, rental, and alcohol sales. Due to the amount of csv files, finding a commonality among them allowed for a merge on the common zip code column. This data became extremely valuable when looking at specific locations and later visualizing them in map graphs. Another major valuable aspect of the data has to do with traffic accidents. In this case traffic reports are pulled in real-time and each time the code is run new data is provided. Having this valuable information will give a precise up-to-date account of traffic incidents to Austin government officials to make decisions that would benefit the citizens and the city.

How might they be applied for direct business application and indirect business applications?

- For direct business application: An insurance company looking for the highest number of crashes within a zip code or as a local business looking for the most amount of traffic that passes through a given (area based on traffic incidents) to gain more exposure to vehicles that pass by. Also, to alcohol distributors looking to target areas where the most amount of alcohol sales occur. They could decide to partner with restaurants or bars in the zip code areas that have the highest amount of sales. Restaurant owners looking to move or expand to a new location could find the area with the high food scores to base that decision on.
- Indirect business application: The city government can indirectly affect businesses by fixing or providing infrastructure to areas with high traffic incidents. If the city is able to provide new roads or fix current traffic hazards the businesses could be affected by gaining more customers who before avoided the areas due traffic accidents.

What do you suggest as potential usages for different variables within the dataset?

With the dataset there is potential usage for many different variables. Each variable can be analyzed separate as well as in combination with others. For instance, from the many different categories of traffic incidents the most problematic type of accident can be identified in order to focus and provide a method to reduce them. In addition, the traffic incidents can be visualized in a map to see in which zones where the most amount of accidents occur. Different columns of the dataset can be combined to create new value showing the relationship between them. For example, combining income, race, or zone

to observe if there is a relation between them and to analyze if low income is more prevalent in a certain zone or if race plays a part in either.

Data Cleaning

The process of data cleaning was completed in python. In order to complete the task, several functions were created. These functions are mostly located in the “data_cleaning.py” file.

What is the overall quality of the data?

In general, the quality of the data was good. However, some missing values were spotted in the dataset. The biggest problem was that after merging the data many values became NA. The most common approach to solve this issue was to input zeros.

What variables contained missing data?

In the first .csv file missing values were:

- One Zip-Code value (only one zip-code were missing)
- Homes affordable to people earning less than $50,000
- Owner units affordable to average retail/service worker
- Owner units affordable to average teacher
- Owner units affordable to average tech worker
- Owner units affordable to average artist

What kinds of missing value exists in the dataset and which variables are they related to?

Among variables mentioned above, all of the columns were numeric data, except the column zip code, which were a key value. Numeric columns were filled by zeros, because any other method, such as a median imputation would not make sense. In the case of key value, the entire row was deleted. It is acknowledged that deleting an entire row for the whole zip code is a big loss, however, without information about zip code an entire row had no meaning in further analysis and merging was impossible.

What methods did you use to clean the missing data?

As mentioned above in order to clean the data one row had to be dropped (zip code). In the case of the rest of variables zeros were imputed. In order to explain why this method was used, a simple example should be considered. The data set contains columns as different type of accidents. It is possible that certain type of accidents never happened in some zip codes area. Thus, imputing any value in this place would be incorrect. For this reason, the most reasonable solution was inputting zeros.

Data Merging

What were the common elements between both datasets?

The common elements and lowest level between the datasets were zip codes in Austin, Texas. Only one dataset contained a lower level than zip codes which were longitude and latitude (data with accidents). However, in order to merge this data with other information, accidents have been grouped and summed by zip codes.

Were there any issues with multilevel measurement in the final dataset?

The most common and significant issue with merging data was when the data were grouped by zip codes and at the same time by some categories. For example, for each zip code data contains several different types of accidents. In order to store this information correctly, the groups (type of accidents) had to be transformed into columns. The problem and solution can be seen on the figure below.

	zipcode	Issue Reported	Total incidents
0	76574	COLLISION	1
1	76574	Crash Urgent	1
2	76574	TRFC HAZD/ DEBRIS	1
3	78602	COLLISION	2
4	78610	BLOCKED DRIV/ HWY	2
5	78610	COLLISION	160
6	78610	COLLISION WITH INJURY	44
7	78610	COLLISION/PRIVATE PROPERTY	3
8	78610	COLLISN/ LVNG SCN	32
9	78610	Crash Service	10
10	78610	Crash Urgent	24
11	78610	ICY ROADWAY	1
12	78610	LOOSE LIVESTOCK	157
13	78610	TRFC HAZD/ DEBRIS	166
14	78610	Traffic Hazard	25
15	78610	Traffic Impediment	1
16	78610	VEHICLE FIRE	8
17	78610	zSTALLED VEHICLE	6
18	78612	BLOCKED DRIV/ HWY	1
19	78612	COLLISION	17
20	78612	COLLISION WITH INJURY	4

Transformation: row groups into columns

Issue Reported	zipcode	AUTO/ PED	...	VEHICLE FIRE	zSTALLED VEHICLE
0	76574	NaN	...	NaN	NaN
1	78602	NaN	...	NaN	NaN
2	78610	NaN	...	8.0	6.0
3	78612	NaN	...	NaN	NaN
4	78613	NaN	...	NaN	10.0
5	78615	NaN	...	1.0	NaN
6	78616	NaN	...	NaN	NaN
7	78617	1.0	...	19.0	56.0
8	78619	NaN	...	NaN	NaN
9	78620	NaN	...	NaN	NaN
10	78621	NaN	...	4.0	NaN
11	78626	NaN	...	NaN	2.0
12	78628	NaN	...	NaN	3.0
13	78634	NaN	...	NaN	NaN
14	78640	NaN	...	NaN	NaN

It can be seen that in the original data set (on the left) all ‘Issue Reports’ were stored in a one column and zip-codes were repeated. In order to store data correctly after merging, each type of accidents has to be stored as a new column. The problem which arise in this method is that data contains big amount of NaN values after transformation. As mentioned in data cleaning section, those rows were filled in with zeros.

What variables are more valuable combined than being in separate datasets?

After merging the datasets, the local government has more precise picture about the whole city. Before each file was considering one certain aspect of the city, for example, only water consumption per postal code. After merging the data, relation between water consumption and median household income or race can be compared. Moreover, the merged data set contains a median score of restaurants in a given zip code. The local businesses can analyze where the best potential location is by comparing restaurants in the area that have a low score and by targeting this location. The thought behind this is customers will more than likely choose the restaurant with the higher food score if it is surrounded by restaurants that have low food scores. Thanks to merging all

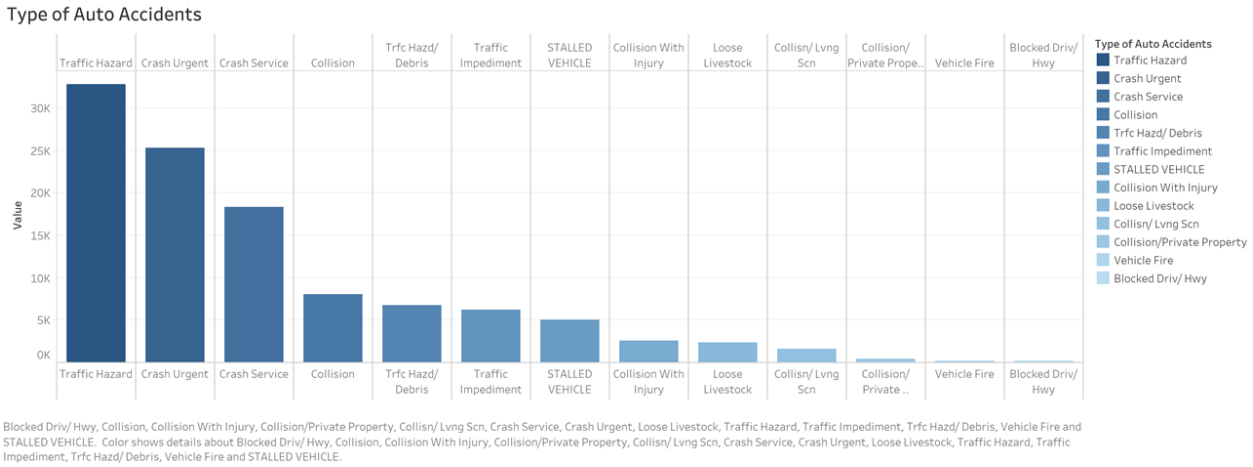
the data sets together, based on the most common race group in the area the best type of restaurant can be chosen. For example, in the area with a high concentration of Hispanic people, opening a Mexican restaurant would probably the best business strategy. Furthermore, the analysis might be done faster because more variables are gathered in only one file and there is no need to change csv files for different analysis.

It is believed that the biggest advantage of the data set is real traffic information. Companies which develop applications about the traffic issues could benefit from the dataset. Every time the code is ran, new accidents are added to the dataset, giving more precise information about the traffic accidents.

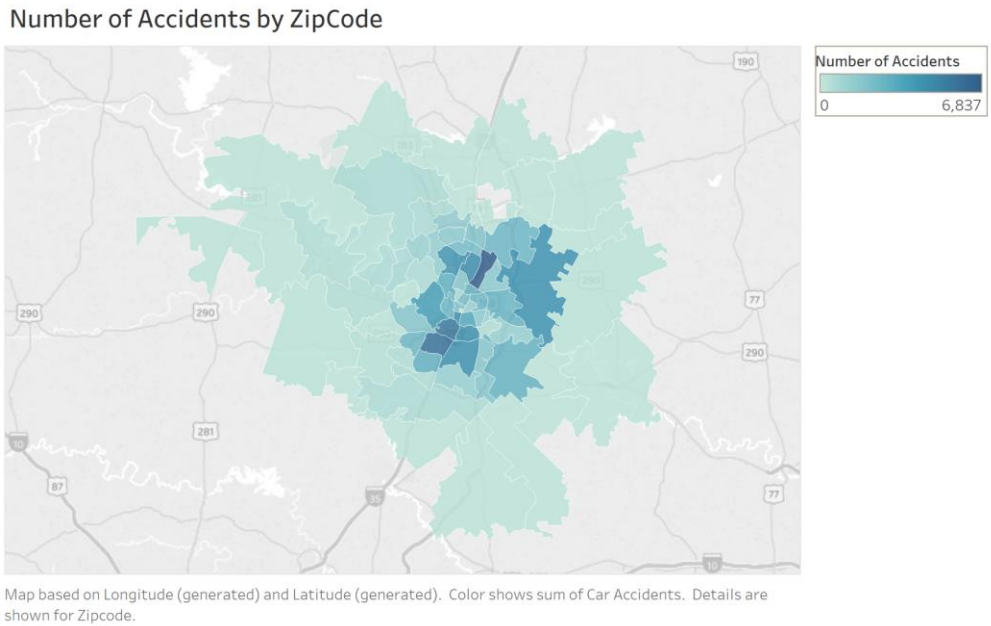
Analysis of Visualizations

How well does your visualization adhere to the principles and characteristics of a good Visualization?

In the case of good visualization, the graphs shown adhere to the association characteristic in the sense that each graph presented has similar categories that are grouped together. For example, the bar graph labeled “Types of Auto Accidents” (graph below) explains different types of auto accidents that occurred in the Austin, Texas area. The Y-Axis is the amount of accidents that occurred, and each category is shown next to one another to account for ordered perception. This is displayed in this manner so the user can make comparisons and visualize which category contains the most amount of incidents. Also, the darker blue is uniform showing a larger value throughout the entire dashboard to account for similar association when moving from one graph to the next. In fact, the user does not have to look at the legends while scrolling through the Tableau Dashboard, thanks to intuitively used colors. Among the whole dashboard darker blue means ‘more’ while the lighter blue means ‘less’. Used colors allow the user to better and faster understand relationships on the maps and graphs.



The map labeled “Number of Accidents by Zip Code” (a map below) is an example of a good visualization characteristic relating to quantitative perception. The reason behind this is the map displays a total of accidents by summing each type of car accident and creating a new attribute with the total summation to display on the map. This visualization technique allowed a better understanding of the location where the most amount of accidents occur, which makes sense because in this case we assume that the most amount of car accidents would occur towards the more populated city center and the map shows this.

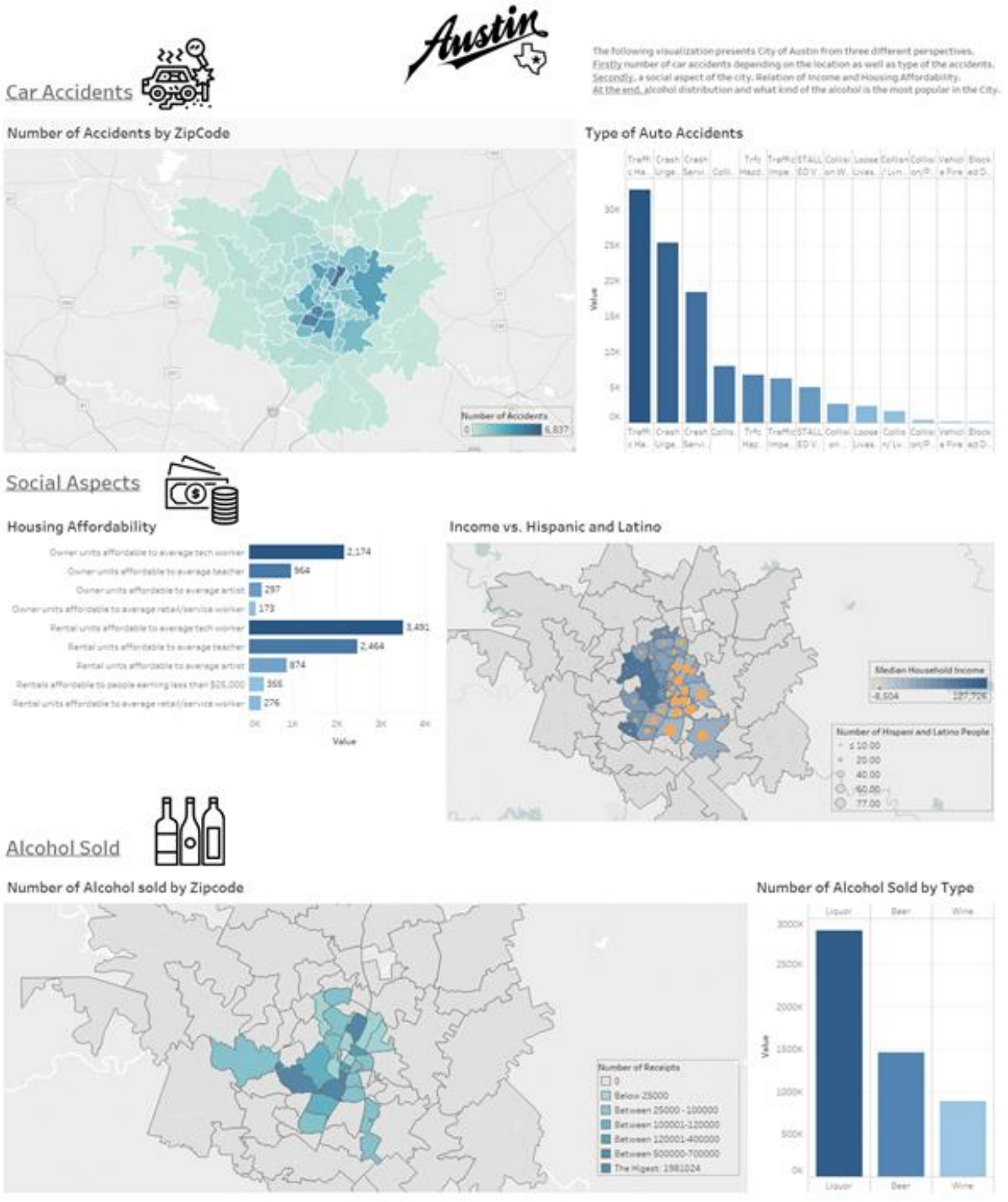


How well does your visualization adhere to the concept of natural processing? Are there things in your graph that are necessary but do not have a natural processing correlate?

In terms of natural processing, graphs were displayed in two-dimensions in order to allow for easier and faster interpretation. It was decided not to complicate visualization techniques with graphs that have more than two-dimensions and unnecessary over the top features. With this decision, bar and map graphs that are common and simple were chosen to best display the data of the Austin, Texas city. Map graphs from a top-down view of the entire city helped separate zip code boundaries and allowed for individual points to be plotted within them. From an outside user's perspective interpreting a colored map with darker shades of blue as higher values is a normal natural processing method. With the graphs presented there are not any unnatural processing techniques used, this is due to the data provided as well as a group decision not to complicate the graphs.

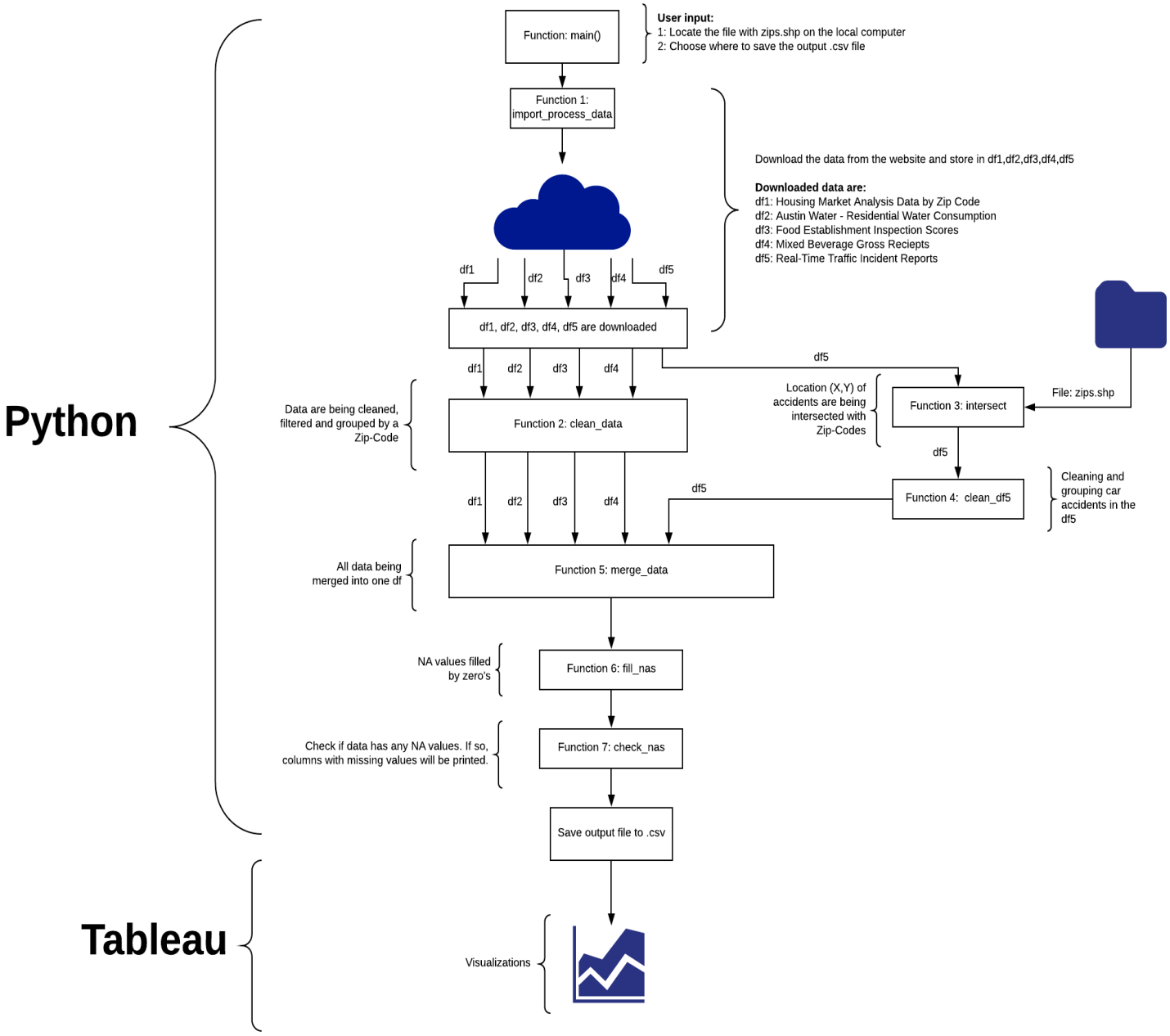
Copies of your visualizations:

Below presented is the screenshot of Tableau Dashboard. However, for better experience with interactive graphs, please visit the Tableau Public here: https://public.tableau.com/profile/grzechowiak#!/



Flow diagram:

On the diagram it can be seen that the project contains many functions which are used to process the data.



Instructions for code

Before running any line of the code the user has to make sure that following packages are installed:

- Pandas
- Numpy
- Geopandas
- Shapely
- rtree

While the two first libraries are the most common, the user should look closer into the last three mentioned packages. The best option would be to open Anaconda Prompt and use: *pip install geopandas* and *pip install shapely*. From previous experience, rtree might cause some problems depending on the operating system, thus the following command line should be tried: *conda install -c conda-forge rtree.* In case the packages are not installed, the terminal (after double clicking *main.py*) will display what imports are necessary.
On the Mac system there was observed an issue with *'import os'* and the solution was to use: *brew install spatialindex.*

The code is designed and divided into 6 files which have other functions. One of the functions, the main() function, calls the others. In order to run the code, the file called *main.py* has to be double clicked. All instructions are displayed in the console. The user has to input two paths:

- 1. A path to the folder with Zip-codes. The path should look similar to: *"C:\Users\user_name\Desktop\BI-project\files\Zipcodes".* Note that the filename is stored in a variable and should not be included in the above path.
- *2.* User has to input a path where the new, merged and cleaned output will be stored.

If there is any error, user can go to the place where file is required and try to solve the problem. The file is required in the function called: *main.py* in the line 38*: path_zip=input('Enter the (1) First path: ').* User can set up the file path and store it as a variable. The *'input'* should be deleted and the path inserted into variable '*path_zip*'. However, this approach should be taken only when the approach with inserting the path as an input via console does not work. The code was tested many times and the only issue could be related to different operating system. As such setting a path in main function should solve the problem.