KU LEUVEN

FACULTY OF SCIENCE

MDA project

AED analysis and reallocation

Elouan Debels (r0793605)

Academic year 2023-2024

General approach and plan

The key points of the approach and plan are briefly summarized here.

Data was given on the distribution of cardiac arrests and the distribution of AED's. The goal of this analysis was to find the AED's which were poorly placed and thus underperforming, and to reallocate these poorly placed AED's, such that their performance would increase and more lives could be saved.

First, we need to define what is considered underperforming. There are 3 conditions that define an underperforming AED, if any hold the AED is considered underperforming:
-the cardiac arrest is already reached within 3 minutes by a professional

-it takes longer than 3 minutes to reach the AED from the spot of the cardiac arrest (summing both ways, so the actual distance itself is 1.5 minutes from the spot)

-there is already another AED closer to the cardiac arrest

(The criterion value of 3 minutes was selected based on the fact that survival rate is high if the person is reached within this time. It can easily be changed in the code itself if this is required)

These underperforming AED's would then be set aside, and a new placement for them needs to be found.

A clustering approach was used in order to find locations where more cardiac arrests occur than elsewhere. We decide, based on an exploratory plot, that density in distribution is the most important factor for finding clusters. DBSCAN was used because it allows us to make use of density in distribution, and also allows a minimum size for clusters to be specified, which helps in avoiding creating clusters based on coincidental close occurrence of cardiac arrests. Then, for each cluster found, we assign an amount of AED's to them, proportional to the size of this cluster. The assigned amount of AED's are used in order to cover as much area as possible within each cluster.

Synthetic data was generated in order to help evaluate the performance of our solution. The metric used to judge the validity of these datasets is simple: the amount of AED's considered valid, and the amount of cardiac arrests covered. Two synthetic datasets were created, based on different methods. The plots of these datasets are also inspected in order to see if they are similar enough to the original dataset, and thus useful.

We also implement an option for the user make use of a budget and cost. If no cost and budget are specified, we assume that there are no constraints, and utilize rational values for the parameters.

We implement our solution, and test the performance, measured as the amount of valid AED's and the amount of cardiac arrests covered, of the new setup of AED locations on both the original dataset and the synthetic datasets. All tests show massive improvements in both the amount of valid AED's and the amount of cardiac arrests that are covered now (approximately 3 times more valid AED's and cardiac arrests covered on the original dataset; and more moderate estimates of improvement using the synthetic datasets).

Notice that this is not a normal classification task, prediction task or clustering task. No simple objective performance criterion could be used, which makes it so that grid searching parameters or pitting different models against each other were not really an option. Therefore, rational parameters and methods needed to be selected, based on sound reasoning.

The analysis was conducted in four parts: an initial data inspection, data cleaning and handling, an exploratory analysis and the actual analysis. For the initial data inspection a simple R script was used. There was also data inspection during the data cleaning, as these two naturally coincide. For the data cleaning and handling a separate notebook was used. The exploratory analysis and actual analysis both have their own notebooks. Each part will be discussed in the logical order, i.e. the order in which they were conducted. There are some very minor differences in the order discussed here and the order in the notebooks, done so for the sake of the continuity of this report, but they should not hinder the reading and understanding.

Initial data inspection

R was used for an initial inspection of the data. This is because it is very easy to inspect the entire table, compared to pandas, where you can only easily see a part of the data. We simply load all given datasets in, and inspect them. This was to get a better idea of how the data was structured, what was stored where, and other such basic but vital information.

From this simple inspection, it already became apparent that the data was not clean and needed heavy processing in order to be made ready for the analysis. An example is that there were a lot of variables that were unnecessary, and variables containing the same information in different tables were named differently (for example the coordinates of the intervention, in the “interventions_bxl” table the latitude of the intervention was under the variable “latitude_permanence”, while in the “interventions_bxl2” table it was under the variable “Latitude Permanence”). The values in the dataset itself were also often inconsistent, for example values in different languages for the “type” variable from the “aed_locations” table.

Data cleaning

More thorough data inspection took place in the data cleaning notebook. As the name suggests, the data was also cleaned here, and made ready to be used for the further analysis. It also became apparent how unclean the data is (e.g. missing values, names of variables that are not consistent across tables, incorrectly inputted data, ...). The different steps are briefly summarized here, for each group of data (AED data, hospital data and interventions data) separately.

Generally speaking, the unclean data is usually not problematic. However, there are some cases where it can cause problems, for example for the AED's, it is not exactly clear which ones are actually publicly available. However, this does have massive influence on the results of our analysis, since we cannot consider a non-available AED to be valid for our purposes, and we also don't know if this AED is actually to be moved in the first place, since it may be privately owned. The assumption was often made that the AED is useful, which may not always be justified in reality. These are problems on the level of the data itself, that cannot be fixed by the analysis

For each group of data, we now summarise which steps were taken.

<u>AED data:</u>

Missing values were imputed if this was valid (for example, some instances had missing id's, but were valid for the rest, so for these instances an id was generated).

Duplicate instances were removed.

Instances with duplicate values were either changed so that each value was unique, or if that was not valid, the instance was removed.

For some AED's (approximately 14%), the exact street number it was located on was missing. These AED's were not removed from the analysis. This lack of street number impacts the exact coordinates the AED receives during the geocoding.

There were two variables ("public" and "available") that indicated whether or not the AED was available to the public, i.e. if it could be used by anyone. The information for these variables was not always consistent, with the value indicating that it is available for one variable and then the other variable indicating it is not available. We are optimistic, and only consider those instances unavailable for which both variables indicate that it is so. We would ask the stake holder for more information if possible.

There was a variable "hours", indicating at which hours the AED is available. We consider the AED's available during the office hours to be always available, since this covers a large portion of the day anyway.

We added the exact coordinates (latitude and longitude) of the AED's based on their address. For this we used the geolocating service Nomatim. For the instances where the exact street number was missing, we took as street number 0. This means that the latitude and longitude are not actually accurate for these instances, which is somewhat problematic. Some coordinates could also not be fetched, for example because the address name was not properly spelled. For 2825 instances this was the case, given this large amount it is impossible to manually correct all instances.

Some AED's already got filtered out of the dataset, and were not used in the remainder of the analysis. These removed AED's were kept in a separate table, called "removed_AED", which contains

the id, the reason for removal, the province it is located in and the municipality it is located in. The reasons for removal in the data cleaning notebook are the following:

-the address was missing

-the AED was not available

-the coordinates could not be fetched

There was also some missing data for the type variable, that is actually relevant, namely for 10060 (out of the 15227) instances the type is missing. We made the assumption that all AED's are useable, regardless of type. This assumption is not necessarily valid, and should be further verified in a real world scenario.

We also mention here that throughout the entire project, AED's which were removed or considered invalid were stored in a table called removed_AED. The reason for their removal is also stored.

Hospital data:

ambulance_locations, mug_locations and pit_locations contain the data on the hospitals.

We remove all instances where the hospital is not permanently staffed, based on the "occasional_permanence" variable.

We add locations to the hospitals, using the Nomatim service again. Since the amount of instances is somewhat limited, here we could actually manually correct instances for which the address was not complete or the geocoding service could not find coordinates.

We remove all variables except the coordinate variables from the datasets. Then we put all instances in a big table, called "all_hospital_locations", and drop duplicates.

before moving on to the rest of the data, we verify whether or not the coordinates of the hospitals match the coordinates of the permanence coordinates, stored in the interventions tables (including cad9). Before we can do so, we need to correct the latitude and longitude variable values for some tables, since the values were not stored with the correct amount of decimals, and this incorrect amount of decimals is not equal across different instances and different tables. More details on how this was corrected are available in the notebook. After correcting, we notice that the coordinates in the "permanence" variable for the interventions does not align fully with the coordinates we fetched for the hospitals. Considering the inconsistencies in the values in the "permanence" variable from the interventions tables, we will not rely on these but we will use the coordinates from the hospitals we fetched. The "permanence" variables get dropped from the interventions table.

Interventions:

The interventions get merged into one big table, to make handling it easy. First the variables are renamed. Variables that are not needed are dropped. The time variables were not formatted the same across all tables, so all these variables were formatted to one format. There were some problems with the time values for the "interventions_bxl" table, more details are available in the notebook. The province variable for the "interventions_bxl" and "interventions_bxl2" tables with as value "BRU", to prevent the information from getting lost. Finally, the tables were merged into a big table named "card_arrest"

We created a variable "time_difference_seconds", which contains the difference between the value in the "T0" and the "T3" variables. This gives the time needed to reach the place of intervention, from call to arrival

The "card_arrest" was filtered to only keep interventions related to cardiac problems.

We look at the "abandon_reason" variable in order to filter out interventions that were false alarms. For example, interventions for which the value is "Geannuleerd" ("cancelled") are not kept and the instances are removed.

Instances for which the coordinates are missing are dropped.

Duplicates get removed. First by simply removing instances that are completely the same (by using the .drop_duplicates() method from pandas). It became apparent however that there were still duplicates present, but for which there some small differences in a variable (for example instances where the timing variables were slightly different, but identical for the rest). We removed these duplicates by dropping instances for which the exact latitude, longitude and T0 values were the same, since the chance that these three variables contain the same value is extremely small (and thus they must be accidental duplicates).

We reformatted the latitude and longitude again (to the correct amount of decimals), using the function defined earlier.

The "T0" and "T3" variables get dropped, since we don't need them anymore.

We notice some problems with the data regarding the cardiac cases. A substantial amount of cardiac arrests (9847 out of the 56284) does not contain the time needed for intervention. One possible way to generate time needed for intervention is to calculate the distances between the closest hospital and the location of occurrence of the cardiac arrest, and then estimate the timing based on this. However, this would likely produce very inaccurate timings, since for example not all roads can be traversed at the same speed, and some locations are harder to reach. Therefore, we do not generate any timings values ourselves. Given that the average and median times to reach the place of intervention is high (14 minutes and 58 seconds, 12 minutes 26 seconds respectively), we consider missing values as not reached in time by a professional.

After all that is done, we store four tables, which contain all the data needed for the further analysis, in .csv format:

"card_arrest_processed.csv", which contains all the data on the cardiac arrests

"hospital_coordinates_processed.csv", which contains the coordinates of the hospitals

"aed_locations_coordinates_processed.csv", which contains all data on the AED's

"removed_AED.csv", which contains all data on the AED's which have already been filtered out.

These tables have the following variables:

<u>card_arrest_processed:</u>

eventType_trip, latitude_intervention, longitude_intervention, parsed_T0, parsed_T3, time_difference_seconds, province

hospital_coordinates_processed:

latitude, longitude

aed_locations_coordinates_processed:

id, address, number, municipality, latitude, longitude, postal_code, province

removed_AED:

id, reason, province, municipality

Exploratory analysis

We provide some exploratory insights on the data in this notebook. We also created some simple plots, in order to further get a grip on the data, and gain some more insight.

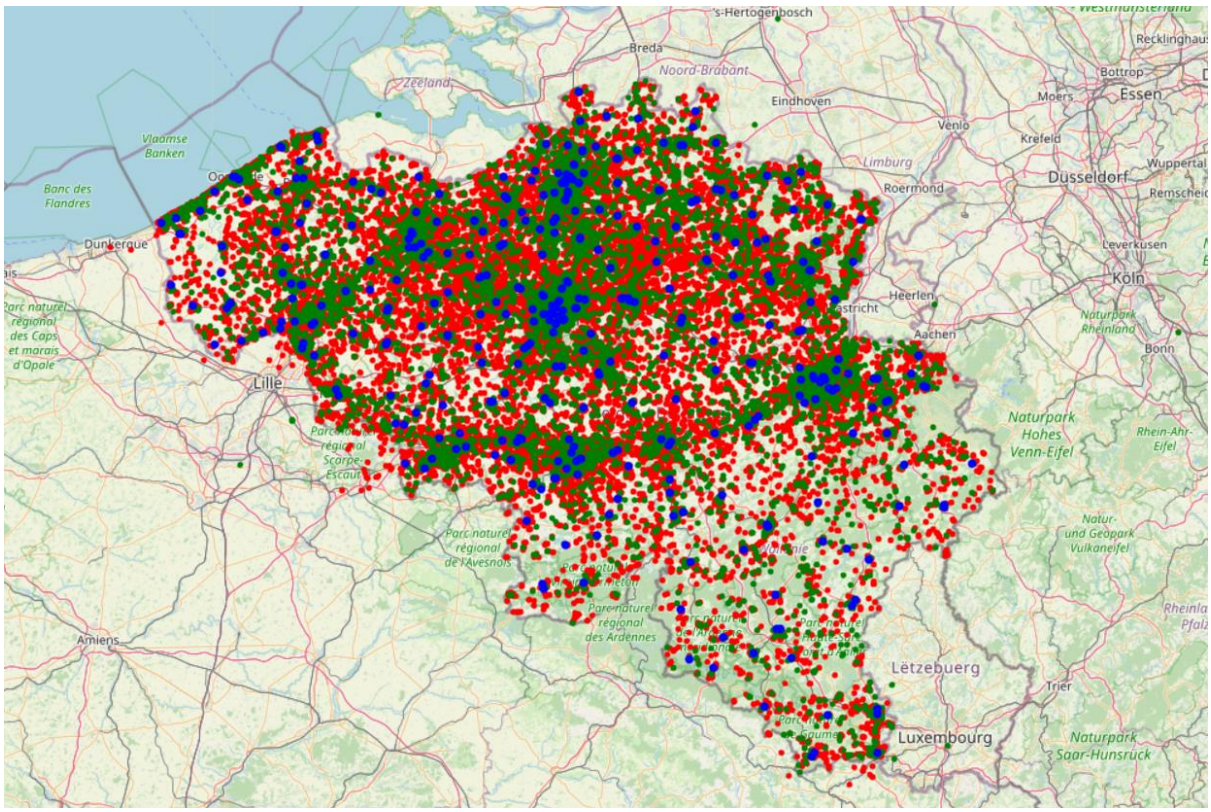
We note the following:

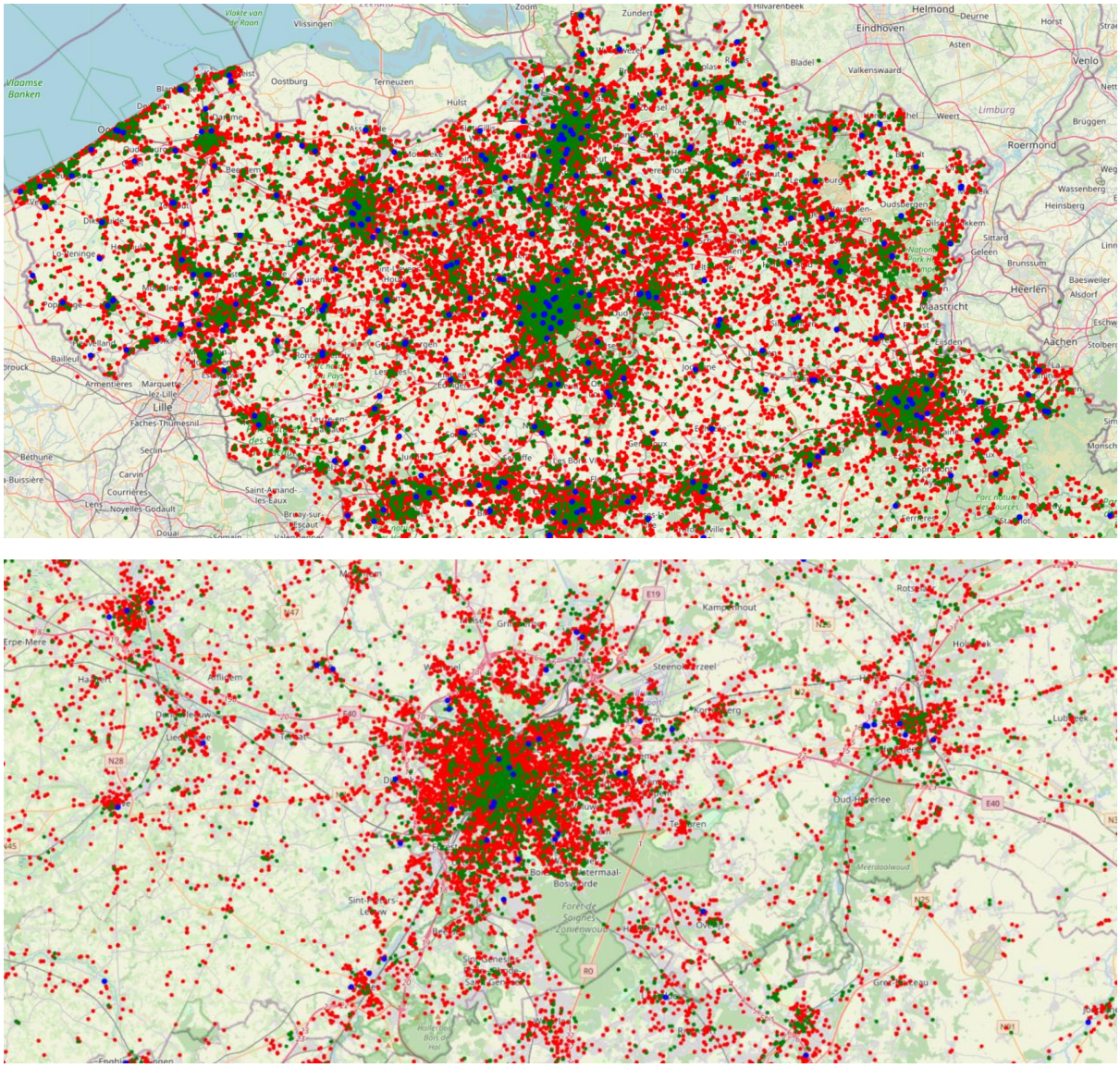
There are 3576 AED's removed from the analysis, equal to around 23.48% of the original amount of AED's, during the cleaning. 2825 (18.55% of the total) because the coordinates could not be retrieved via Nomatim, for example because the provided address had abbreviations or misspellings. 749 (4.91%) because they were considered to be unavailable to the public, and thus not useable. 2 were removed simply because the address was missing.

Plots:

We used folium to plot the different datapoints we collected. Folium creates an interactive map based that can be zoomed in on and dragged around. Screenshots are provided here, but the actual map itself, stored as a .html file, is a lot clearer. It is therefore recommended to also take a look at those.

We plotted the locations of the cardiac arrests, the valid (based on the filtering thus far) AED's and the hospital locations. (red dots are cardiac cases, greed dots are AED's, blue dots are hospitals) (available as exploratory_map.html)





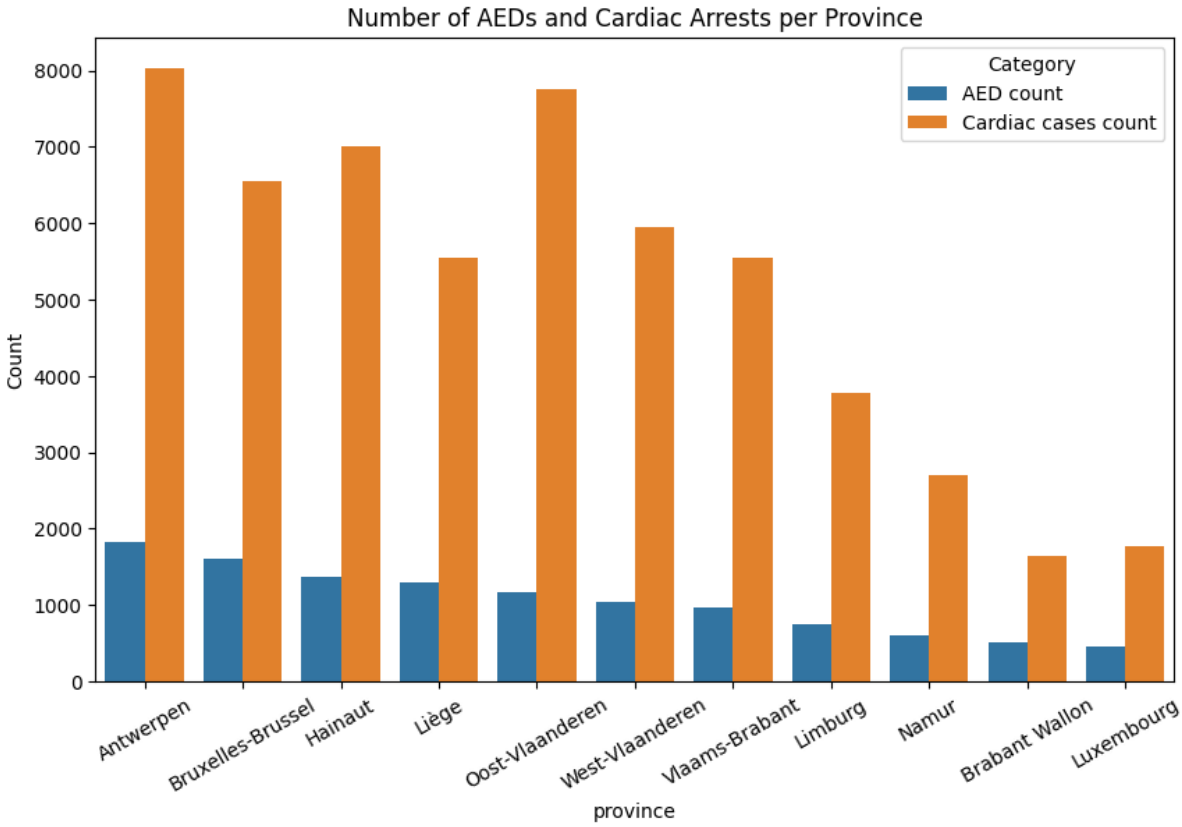
We notice from this plot that some AED's are incorrectly mapped, because they lie outside of the bounds of Belgium. We approximately remove these by taking the outermost coordinates of Belgium (see code for details).

It is apparent that within cities the density of cases is highest, with bigger cities like Brussels city and Antwerp City the highest density, regardless of where the city is located. This makes sense since cities are simply more densely populated. The coastline also contains a high density of cardiac arrests, perhaps caused too by typically higher average age of residents there (which also would have more cardiac problems). The more Southern part of Wallonia has less densely located cardiac arrests compared to the rest of the map, which also makes sense since it is less densely populated. In general the distribution of cardiac cases seems to match the population densities of different areas. Flanders is generally more densely populated, and the cardiac cases are more densely populated there. One last thing to notice is that there seems to be a belt of more densely spread cases which follows the Meuse river.

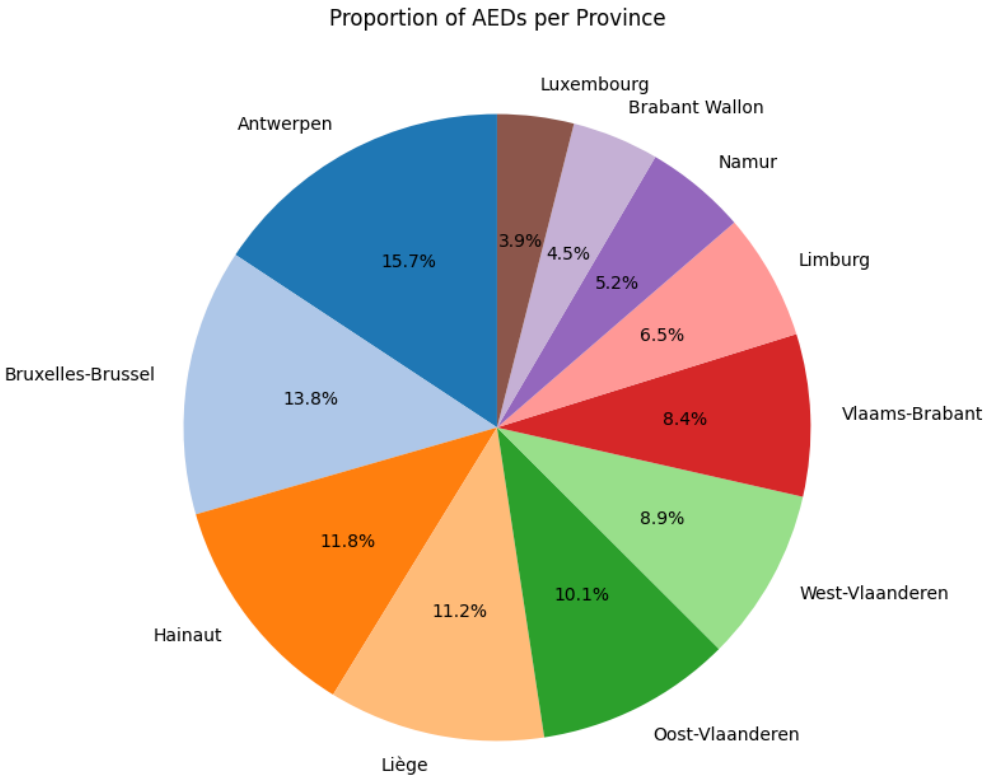
The spread of the AED's seems to match the spread of the cardiac cases pretty well, although when looking at big cities, for example Brussels City, there seem to be significantly less AED's compared to cardiac cases, compared to for example a smaller town like Overijse.

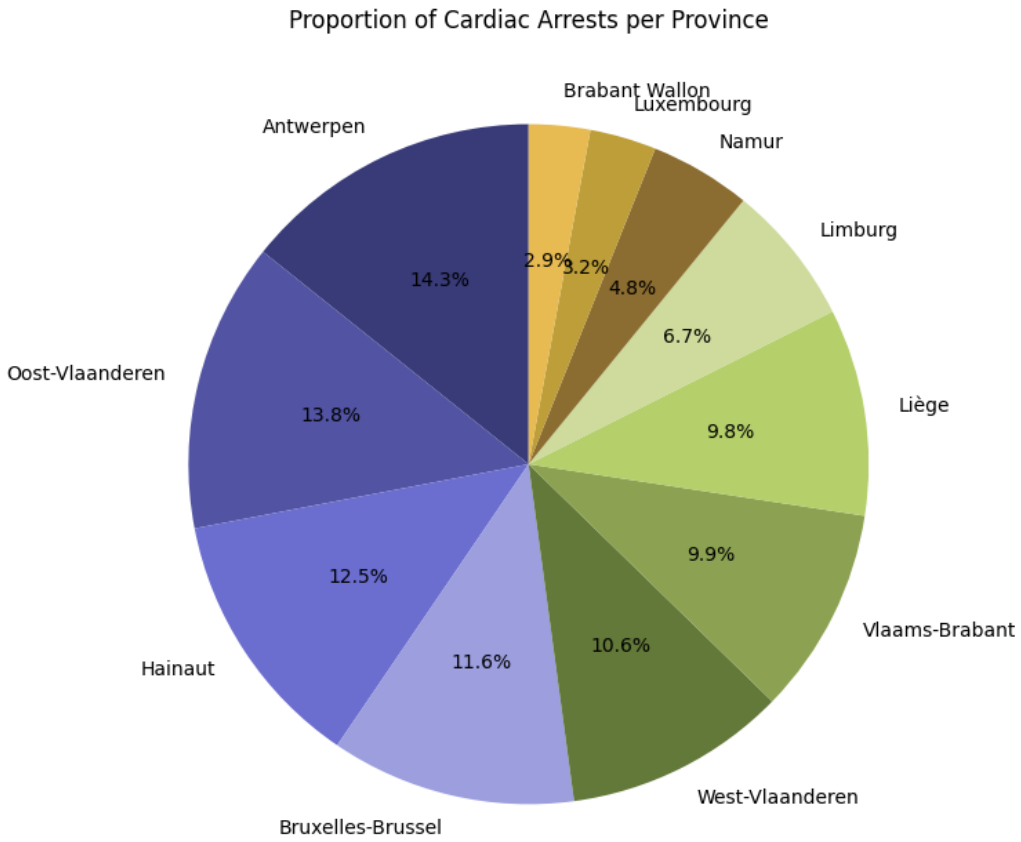
Finally, when looking at the spread of the cardiac problems, we notice that there are naturally formed clusters. This indicates that a cluster analysis is suited in order to find the optimal places for the AED's.

We make a plot showing the absolute count of cardiac cases and AED's available, for each province (also including Brussels).



We also create pie charts, showing the proportion of AED's and the proportion of cardiac cases respectively for each province.


Proportion of AEDs per Province
Luxembourg
Brabant Wallon
Antwerpen
Namur
Limburg
3.9%
4.5%
15.7%
5.2%
6.5%
Bruxelles-Brussel
13.8%
8.4%
Vlaams-Brabant
8.9%
11.8%
West-Vlaanderen
10.1%
11.2%
Hainaut
Oost-Vlaanderen
Liège


Proportion of Cardiac Arrests per Province
Brabant Wallon
Luxembourg
Antwerpen
Namur
Limburg
2.9%
3.2%
14.3%
4.8%
6.7%
Oost-Vlaanderen
13.8%
Liège
9.8%
9.9%
12.5%
Vlaams-Brabant
10.6%
11.6%
Hainaut
West-Vlaanderen
Bruxelles-Brussel

From these plots we notice that there are some discrepancies between amount of cardiac arrests and amount of AED's available in the province visible in the pie charts, some examples:

-Brussels has the second most AED's while having the fifth most cardiac arrests

-East-Flanders has the second most cardiac arrests while having the fifth most AED's.

However, these plots only show the amount of AED's, not how useful they actually are.

Processing

The main analysis occurs in this notebook.

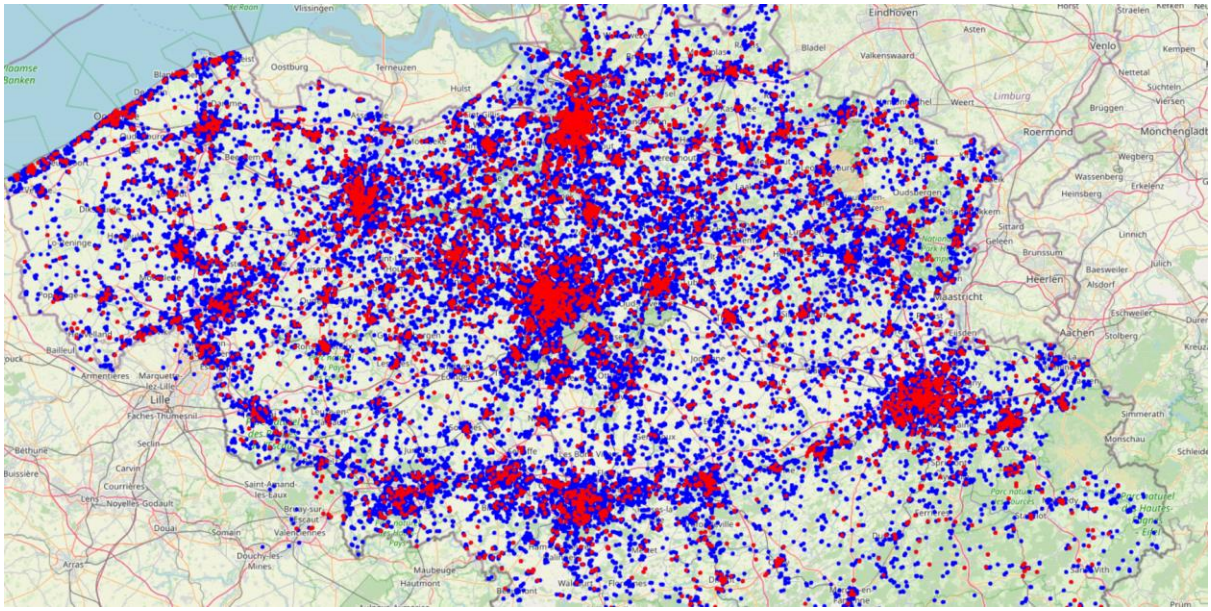
Which data to use

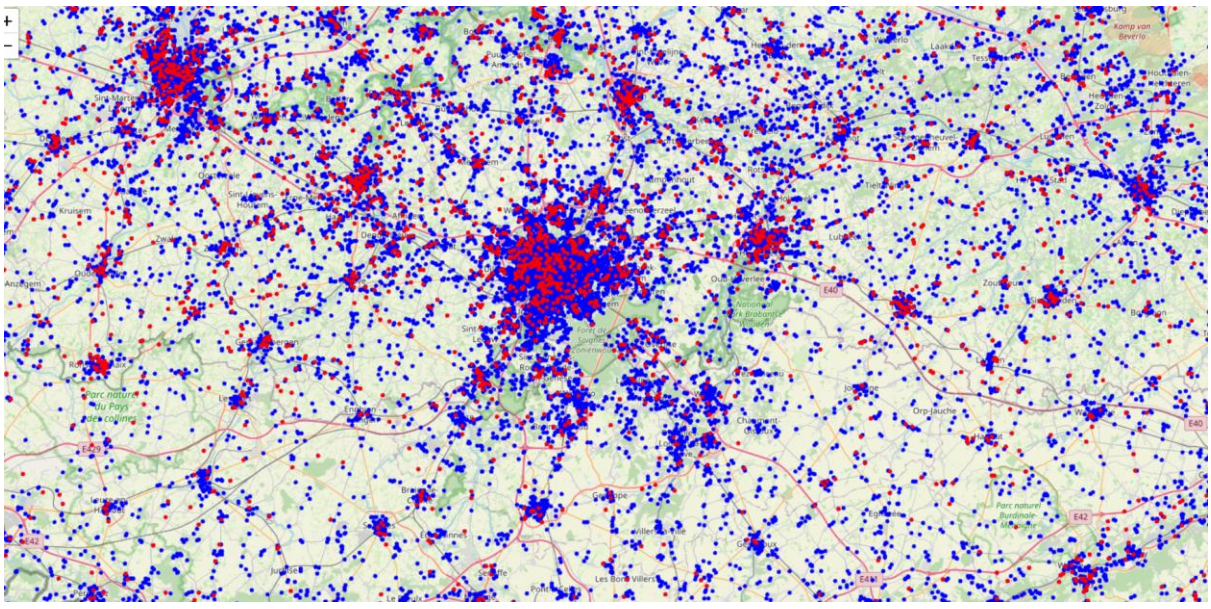
For the analysis, we will use all cardiac cases (so not only cardiac arrests). There are two reasons for this:

-an AED might actually be useful for more than just cardiac arrests, so we would want to also use these in the analysis. This would need to be verified by the stake holder.

-we assume that the distribution of all cardiac problems is similar enough to the cardiac arrests

To test the assumption that the distribution is similar, both groups (all cardiac cases and specifically the cardiac arrests) are plotted. From the plot, we notice that the distribution is indeed very similar, justifying the assumption and further use of all cases. (red dots: specifically cardiac arrests; blue dots: all cardiac cases) (plot available as cardiac_specific_general.html)





Using all cardiac cases is to be preferred because of the simple fact that there are more observations (56284 compared to 6429). This means our analysis will be less subject to random variations in the data, making our solution more robust.

Distance to use

It is hard to find the exact time a person with cardiac arrest can survive without receiving aid. This exact value influences the results of the analysis drastically however. In a real analysis the stakeholder would be contacted to provide an estimate of this time. We will assume that a person can go for 3 minutes without receiving aid.

People can run at about 10 km per hour, so the distance ran in 3 minutes would be 500 meter. Since it's unlikely that you can run in a straight line from occurrence of cardiac arrest to AED (e.g. the AED is located on the other side of the building it is attached to), we add another 15% in time needed to reach the AED. This translates to the total running distance to be approximately 435 meter. Since the person would also need to run back and forth, the distance between cardiac arrest and AED needs to be 217.5 meter or less, for it to be a useful AED.

We remove the cardiac arrests that are reached within 3 minutes from the analysis, since these do not need an AED in order to be helped, they are already helped in time by professionals. We will use the remaining cardiac arrest for the rest of the analysis. 98 cases get removed from the analysis this way.

How the underperforming were found

The following is an explanation on how the underperforming AED's were found. There are 3 conditions that define an underperforming AED, if any hold the AED is considered underperforming:
-the cardiac arrest is already reached within 3 minutes by a professional

-it takes longer than 3 minutes to reach the AED from the spot of the cardiac arrest (summing both ways, so the actual distance itself is 1.5 minutes from the spot)

-there is already another AED closer to the cardiac arrest

The first rule is already applied by excluding all cardiac arrests that were reached by a professional from the analysis in the first place, because then the AED cannot be considered close to it, since it is not present in the analysis in the first place.

Now observe the fact that each cardiac arrest should only have one single AED it is closest to.

We want to calculate the distances between cardiac arrests and AED's, and use these to decide which follow the two rules. Brute forcing it (simply calculating the distance between every cardiac arrest and every AED) would result in practically infinite time needed to finish the calculations. Therefore, the following solution was invented:

Instead of looking at the AED's, we look at the cardiac arrests. We create an empty table containing the distance between the AED and cardiac arrest, id of the cardiac arrest, and the id of the AED. Now, for each cardiac arrest, we look for all AED's within a radius of X meters. For all AED's within this distance, we append to the empty table the calculated distance, the id of the AED and the id of the cardiac arrest. When all that is done, we know that each cardiac arrest should only occur once at

most in the table, since there should only be one AED it is closest to, hence if the cardiac arrest id does occur more than once, we can remove the instances with the higher distance value. Then we end up with a table that contains only those AED's which follow the rules apply (it is within range X of a cardiac arrest and it is the closest one to the cardiac arrest); all the other AED's that are not in this table do not follow the rules, and are hence invalid.

The following can further elucidate this process:

A table is created, containing cardiac arrest id, AED id, and the distance between the AED and the cardiac arrest. For each cardiac arrest, we check the area within a radius of X for the presence of AED's.

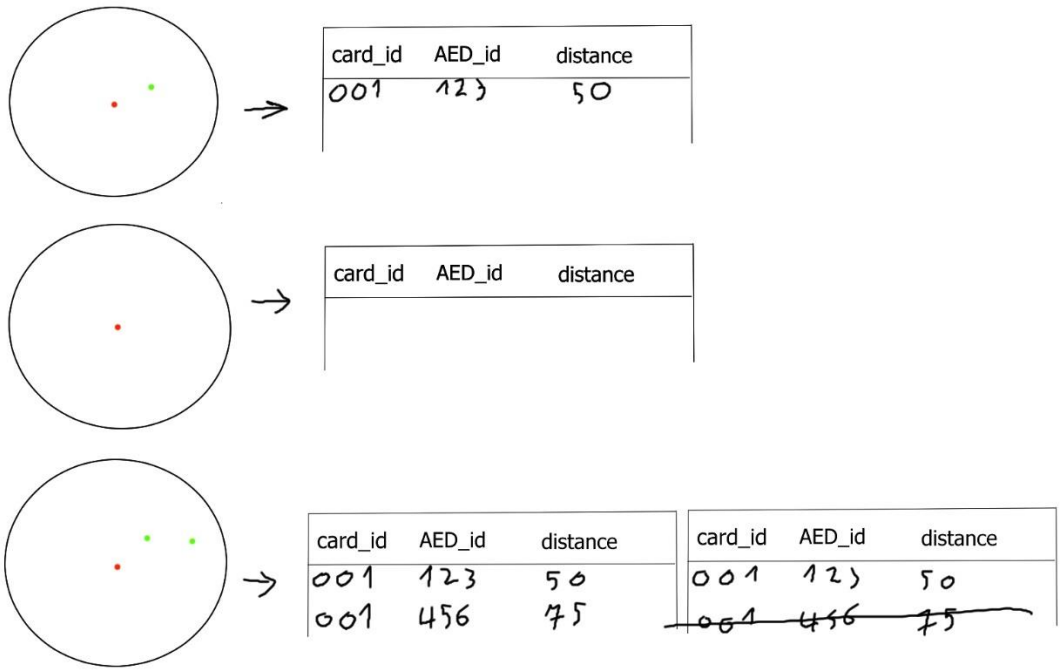
There are three possible situations:

-there is one single AED within a radius of X near the cardiac occurrence

-there are no AED's within a radius of X near the cardiac occurrence

-there are multiple AED's within a radius of X near the cardiac occurrence

For each cardiac arrest



In the first situation, the id of the cardiac arrest and the AED get appended to the table, along with the distance between the two.

In the second situation, nothing gets appended to the table.

In the third situation, we append both AED's, where for each instance, we append the id of the cardiac arrest, the id of the AED and the distance. Now, we make use of the fact that only one single AED can be closest to a cardiac arrest, and for each card_id in the table, we only keep the instance

where the distance is smallest. This way, we create a table in which for each cardiac id, the AED which is closest is kept.

This table then contains the AED's which follow the rules: all AED's are within distance of a cardiac arrest, and all AED's are closest to at least one cardiac arrest. The first rule (the cardiac arrest is already reached within 3 minutes by a professional) is already followed by removing those cardiac arrests from the calculations beforehand.

To speed up the process even more, a KD-Tree is constructed for the AED coordinates. This allows for efficient spatial queries to find AEDs within a certain distance of a cardiac arrest, without having to calculate the AED and all cardiac arrests in the dataset.

Since coordinates are being used (latitude and longitude), the haversine formula is used in order to calculate the actual distances themselves.

The exact implementation and more details are present in the notebook.

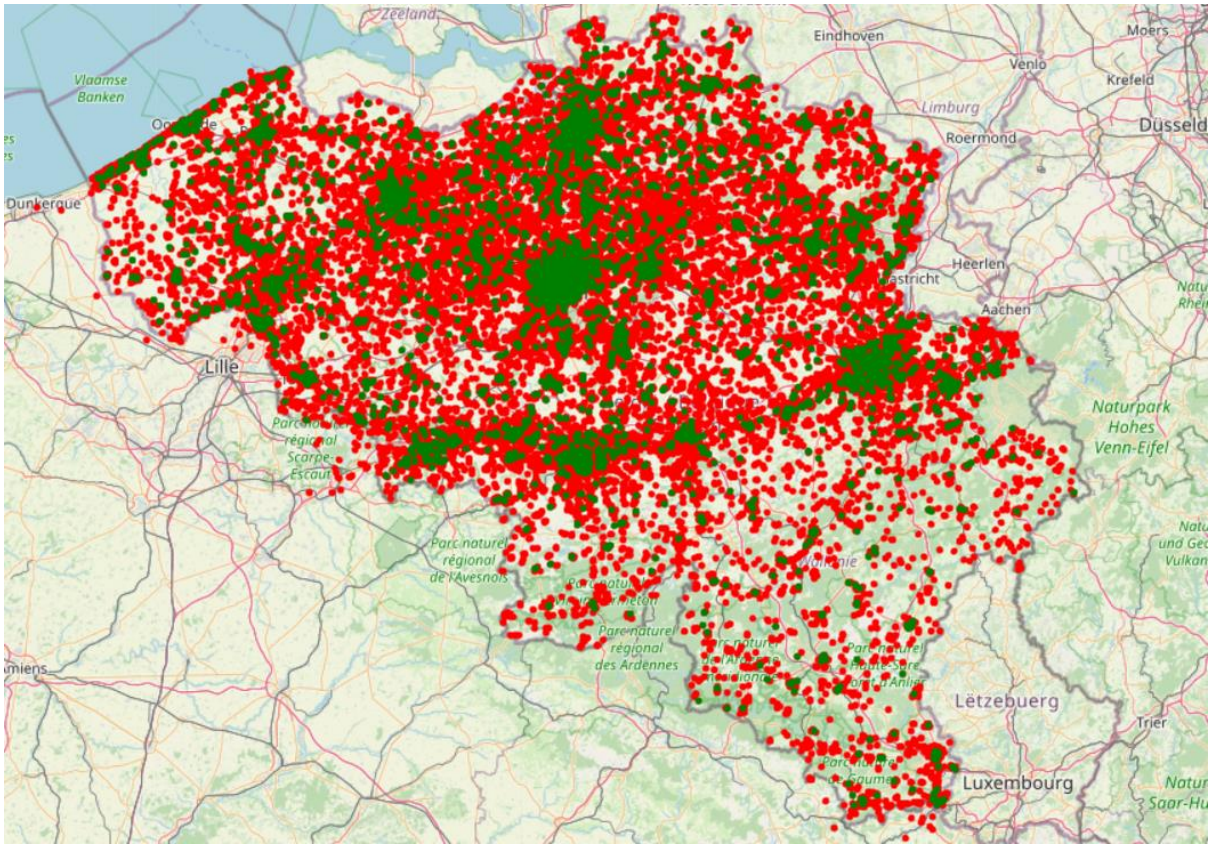
Results initial analysis

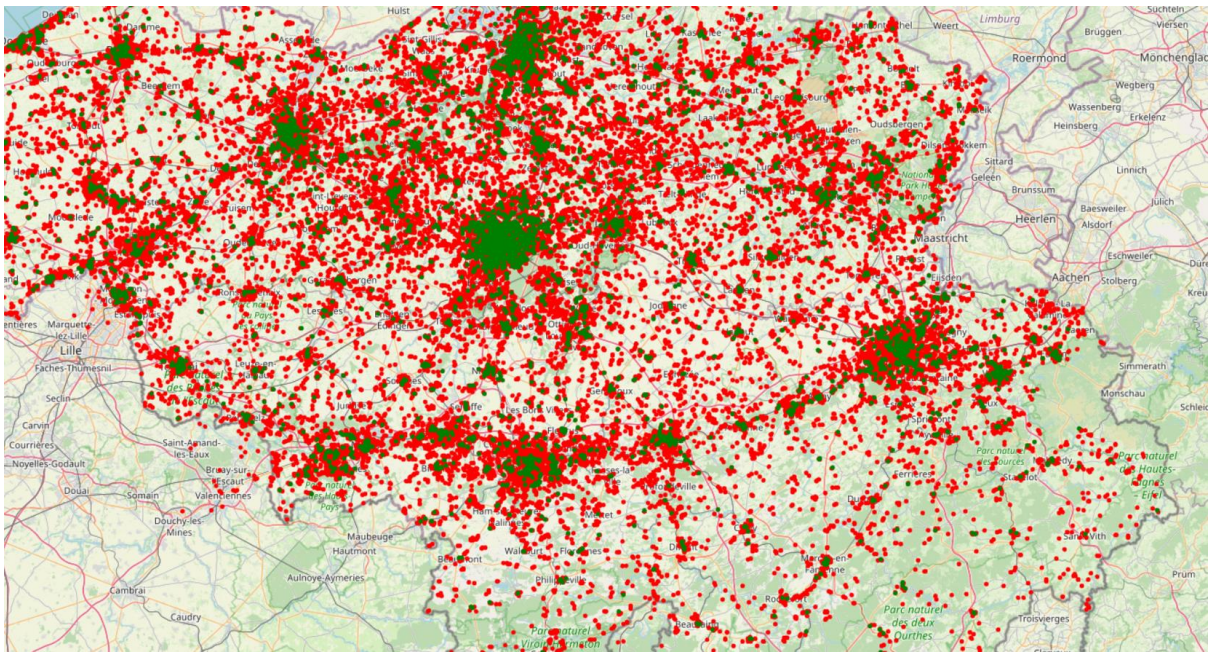
Using our method, we find that with the current setup, 4668 AED's are valid out of the 11 649 tested. 13219 cardiac cases are covered by them. This shows that there is room for improvement.

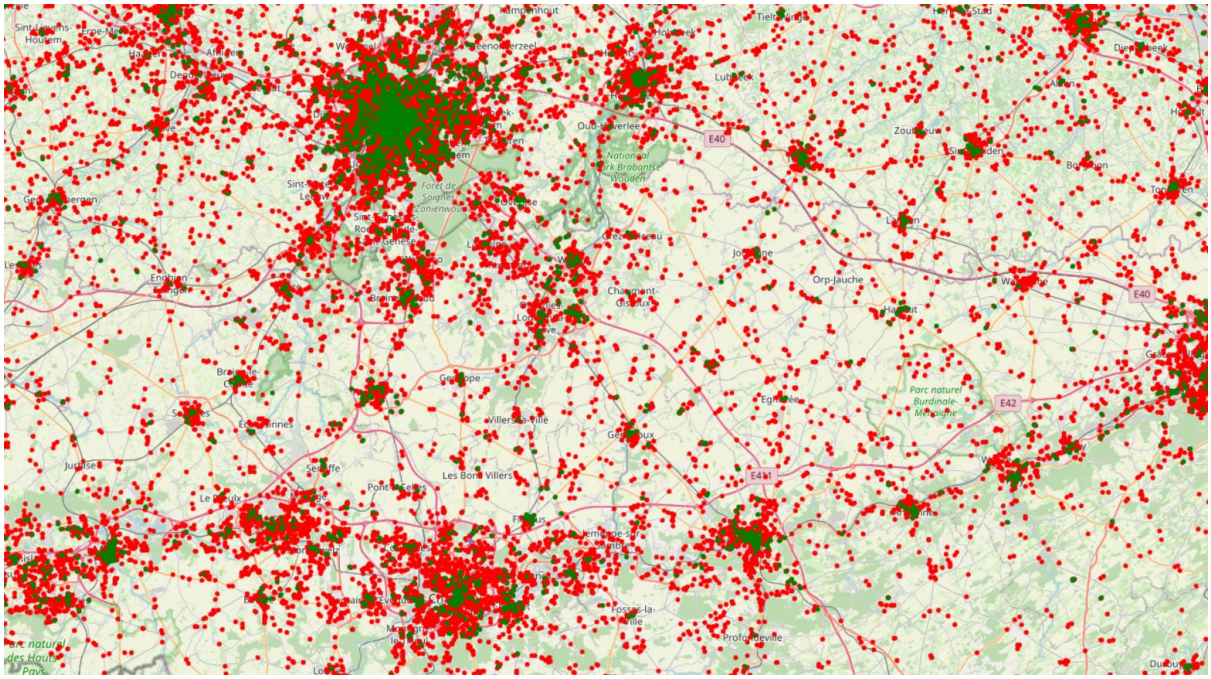
We append the underperforming AED's to the removed_AED table.

We derive a function from the code used here, that can be used to calculate the amount of valid AED's and the amount of cardiac arrests covered using distance value X. This will be used for the rest of the notebook, whenever these calculations need to be made.

We plot the valid AED's and cardiac arrests on the map. Perhaps we can learn something from this. As expected, the valid AED's lie in or close to a city, where the population density is higher. There are also no valid AED's in the less densely populated south of Wallonia. This again illustrates the importance of density of occurrence for an AED to be useful. (green dots: valid AED; red dot: cardiac case) (plot available as valid_aeds.html)


Zeeland
Eindhoven
Moers
Venlo
Krefeld
Vlaamse Banken
Limburg
Roermond
Düsseld
Dunkerque
Heerlen
Aachen
Lille
Naturpark Hohes Venn-Eifel
Parc naturel régional Scarpe-Escaut
Parc naturel régional de l'Avesnois
Parc naturel régional des Ardennes
Lëtzebuerg
Trier
Luxembourg


Hulst
Limburg
Roermond
Heerlen
Maastricht
Aachen
Lille
Valenciennes
Maubeuge
Cambrai
Parc naturel des deux Ourthes



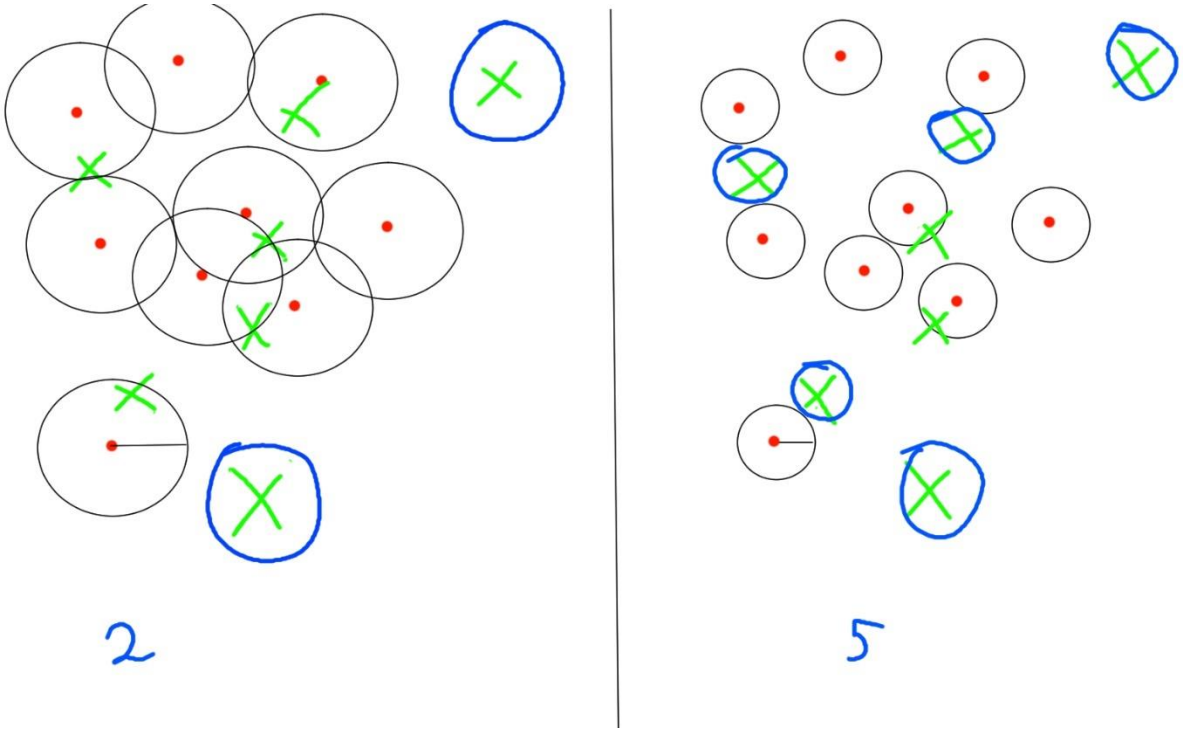
Budget – cost implementation

For the calculations above, we have not kept into account any budget constraints. For the value of X, we used a rational choice (the distance for which an AED would be valid). However, an option to make use of a budget is also implemented. Suppose the stakeholder only has enough budget to move a fixed amount of AED's, called Y. Then we want to find the Y worst performing AED's, where worst performing means having the greatest distance to it's closest cardiac arrest.

This can be done by finding the distance value X defined earlier, for which there will be Y AED's considered underperforming. The function created to calculate the number of valid AED's relies on the input data for the AED locations, the input data for the cardiac arrests, and the X distance value. The function outputs the amount of valid AED's according to our rules. Given that the AED's to move equals the total AED's available minus the AED's considered valid, we can manipulate X while keeping the input data the same, in order to search for the X that provides the correct amount of AED's to move, so that it matches the desired amount Y.

This method also makes sure that it is the Y worst performing AED's that get removed first. This is so because we consider the worst performing AED's to be the ones that are furthest away from all cardiac arrests (more exactly, the AED's with the greatest distance value to the cardiac arrest they are closest to), thus we need to find the X distance value for which there will be Y AED's that have the same or a greater distance value to it's closest cardiac arrest. So, stated differently, if we can find the X value for which Y AED's are to be moved according to our function, we automatically have found the Y worst performing AED's. The following explanation and image can make this more clear:

Given a certain cost per AED moved and a total budget, we know how many AED's we can move (i.e. total budget divided by cost per AED). Now, observe the fact that by manipulating X, we can change the amount of valid AED's, with for larger X, the amount of valid AED's will rise, and for lower X, the amount of valid AED's will diminish. The following illustration makes this clear:



On the left a greater value for X is selected (which is equivalent to changing the radius for each cardiac arrest), which makes it so that only 2 AED's are considered invalid (which are furthest away from their closest cardiac arrest point and thus the most underperforming), and hence need to be moved. On the right the value for X is smaller, making it so that more AED's are considered to be invalid compared to left and need to be moved. Using this, we can find X such that the amount considered invalid by our method matches the amount that can be moved using our budget.

To find the value of X, we make use of an iterative approach rather than an exact one (more specifically a binary search). The results will be accurate enough for our purposes.

Using this solution, we can effectively use solution and code, which were originally designed for a rational value for X, as opposed to for a fixed amount of AED's to be moved.

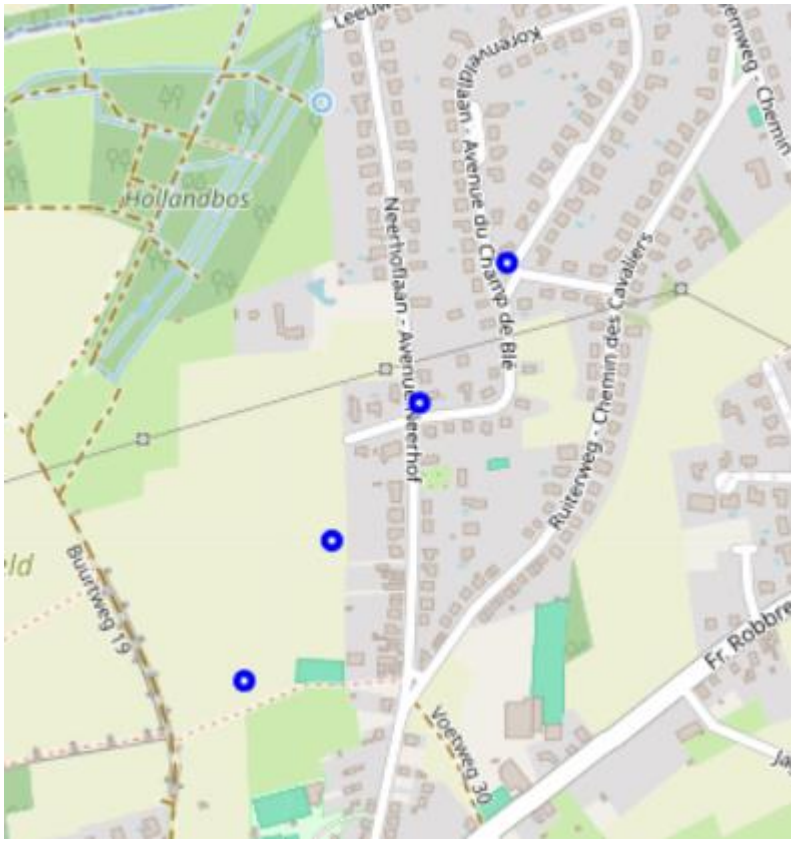
Note that when using this method, X loses it's interpretative value, and becomes just a parameter to be found. Also note that for the DBSCAN and the evaluation of the performance (see further), we would still use our rational value for X, since it still is this value for X which decides if an AED is valid or not; the budget value for X was merely a parameter to be found.

Synthetic data

We create synthetic cardiac arrests, in order to test the performance of our solution. If we only test on the real dataset, we risk overestimating the performance of our solution. Suppose a person is walking on the street when he has his cardiac arrest. It is purely coincidental if it happens 5 minutes earlier or 5 minutes later. This randomness in exact location of occurrence is what we try to simulate in our bootstrapped data set. To generate these synthetic instances, two methods were used:

-K-means was used to split the original dataset up into clusters, and then generate data using these clusters

-using an uniform distribution to add perturbation to the original dataset, in such a way that most instances are about a street further. Below plot shows points moved by 0.0015 in latitude and longitude (this is what is meant by “one street further”). (image available as coord_testing.html)



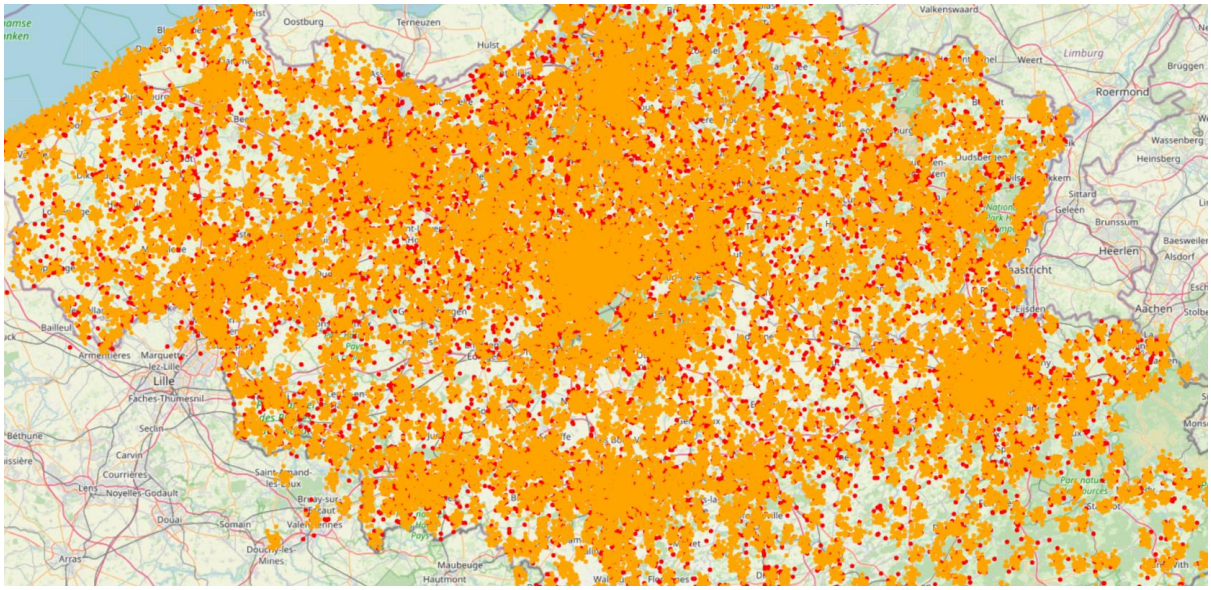
Both methods were tested for their validity, by both looking at a plot of the synthetic dataset against the original dataset, and comparing the performance of the original setup to the synthetic setup.

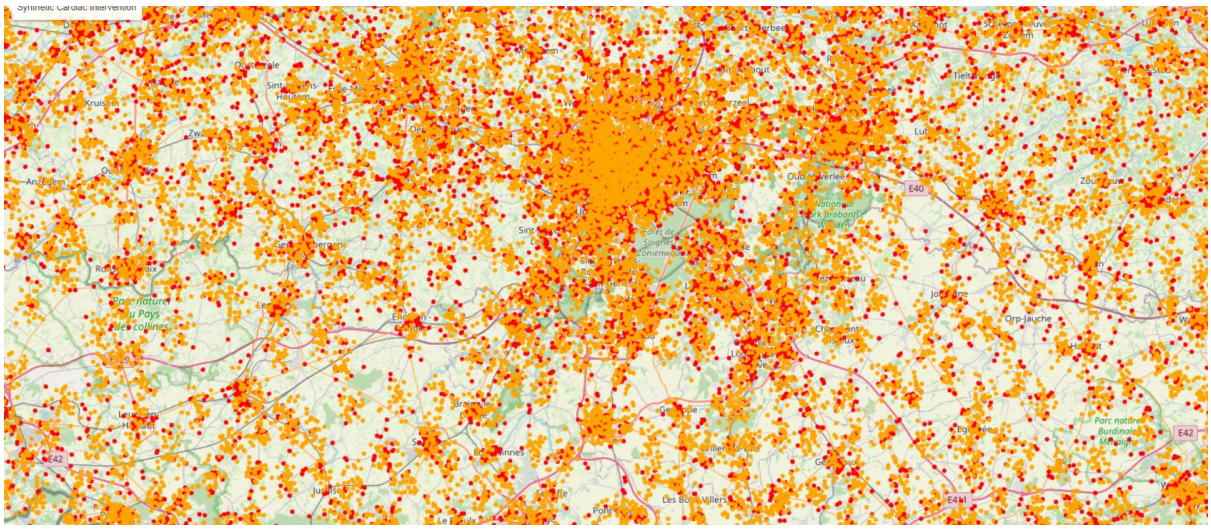
Note that we did not use DBSCAN or another density based clustering method in order to generate the synthetic data. This was deliberate, since we will make use of DBSCAN in our analysis; generating instances based on DBSCAN would then have the opposite effect, i.e. cause us to overestimate the effectiveness of our solution.

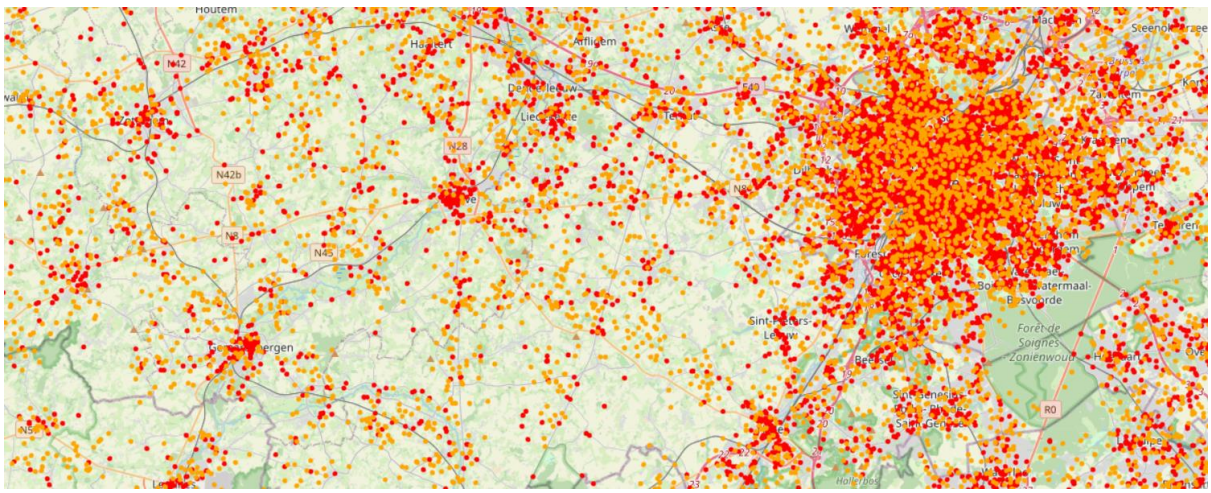
We prefer using a synthetic dataset over a simple train-test split because the amount of instances is relatively small, even with more than 50 000 instances, since they are spread over a wide area. It would be hard to make a valid split; keep too many in the train set, and the test set will be too small and too much subject to randomness, making results for this test set skewed. Put too many in the test set, and the train set will not be realistic anymore, because again it would be subject to randomness, making the results of the analysis itself skewed. This is amplified further by the fact that we make use of DBSCAN, where we define the minimum cluster size to be 3, which would mean potentially a lot of smaller clusters get eliminated by creating a train-test split simply because some of the points that would belong to the cluster get set aside for the testing set.

The plot of the k-means generated synthetic dataset shows that the data generated is more or less realistic, although we notice that it is not able to capture the different densities well (random places

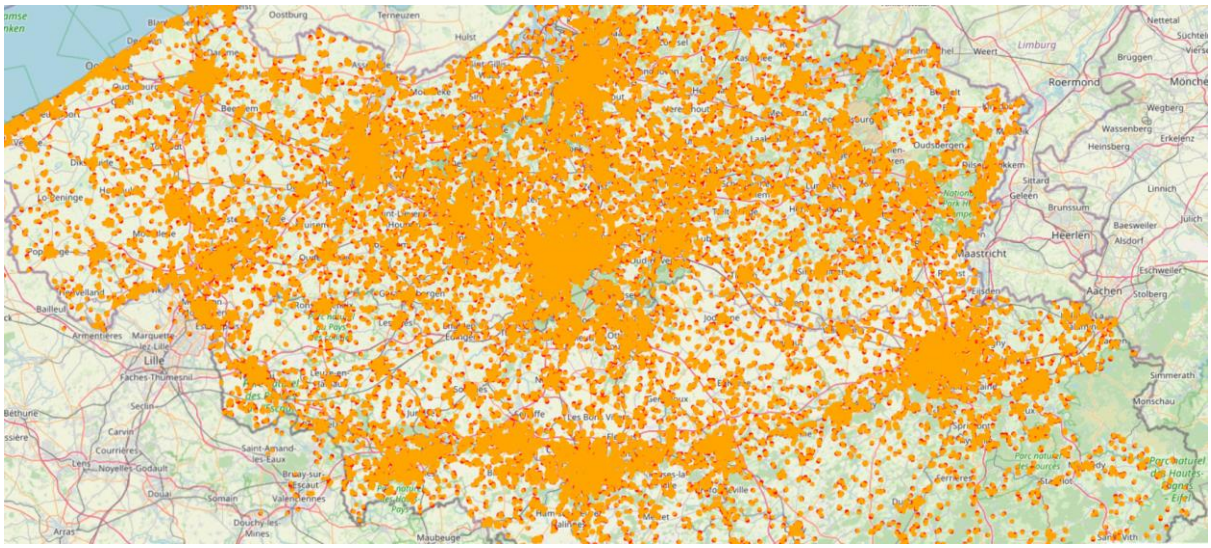
are as dense as big cities). (orange are the synthetic instances, red the original cardiac arrest locations) (orange dots: synthetic instances; red dots: original cardiac cases) (available as synthetic_Kmeans.html)

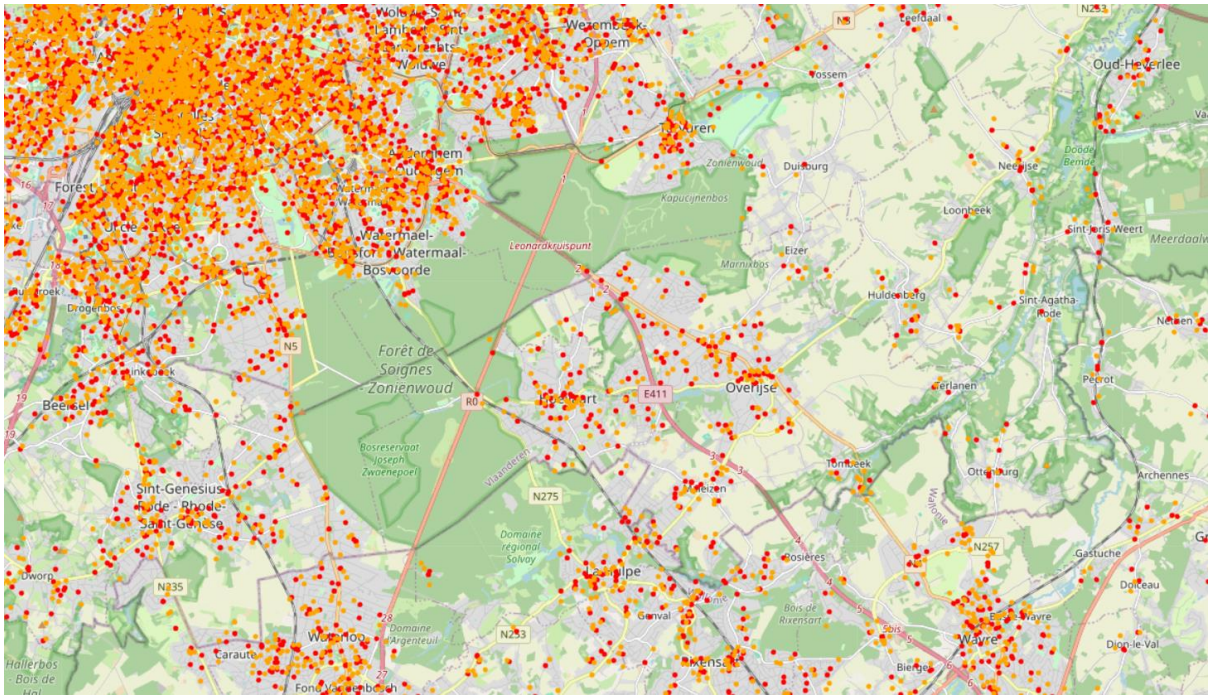






The plot of the uniformly generated synthetic dataset shows that the generated data is very realistic. This makes sense, since the structure (density, distribution) is the same, only the exact locations are changed. (orange are the synthetic instances, red the original cardiac arrest locations) (available under cardiac_arrests_map_belgium_uniform.html)





We also check the validity of the synthetic datasets by comparing the performance of the original setup on these datasets to the performance on these synthetic datasets. If the synthetic data is realistic, then the performance should be similar (i.e. same amount of valid AED's and same amount of cardiac interventions within range).

For the k-means synthetic data, 3077 AED's are deemed valid and 4023 cardiac arrests are covered, compared to the 4668 and 13219 of the original solution. This shows that the k-means synthetic dataset is somewhat realistic.

For the uniformly generated synthetic instances, the amount of AED's is 4821 and the amount of cardiac arrests is 11735, which comes close to the original performance, providing further support for the validity of this method for generating synthetic instances.

Placing the AED’s

There is a limited amount of AED's compared to the overall size that should be covered. Identifying locations where cardiac arrests occur more often compared to the rest of Belgium and placing AED's is key to finding the optimal locations for placement. We make use of a clustering solution, since as we observed earlier, the cardiac arrests tend to occur in denser clusters in certain areas. We make use of 2 rules in order to identify the spots where an AED would likely perform well:

-density: for a cluster to be valid, the density needs to be above a certain threshold. This is because density is a very important factor when considering to place an AED somewhere, since density decides whether or not the AED will actually be close to the occurrence (since we cannot know the exact location of an occurrence in the future, this is all we can use). This means that it doesn't matter if the method finds a perfectly shaped cluster, if all the cardiac arrests within this cluster are far from each other, placing effective AED's will not be possible.

-minimum number of occurrences per cluster: in order to avoid finding clusters consisting of cardiac arrests which lie coincidentally close to each other without there being a real pattern, a minimum amount of cardiac arrests needs to be in the cluster. This minimum amount should be small, since the chance that two or more lie close to each other is small anyway.

Before conducting the actual analysis, we first exclude all cardiac arrests which are already reached by a professional, like before, and all cardiac arrests which are covered by a valid AED. This is because these cardiac arrests do not need more coverage, and if they are not excluded, they will influence the outcome.

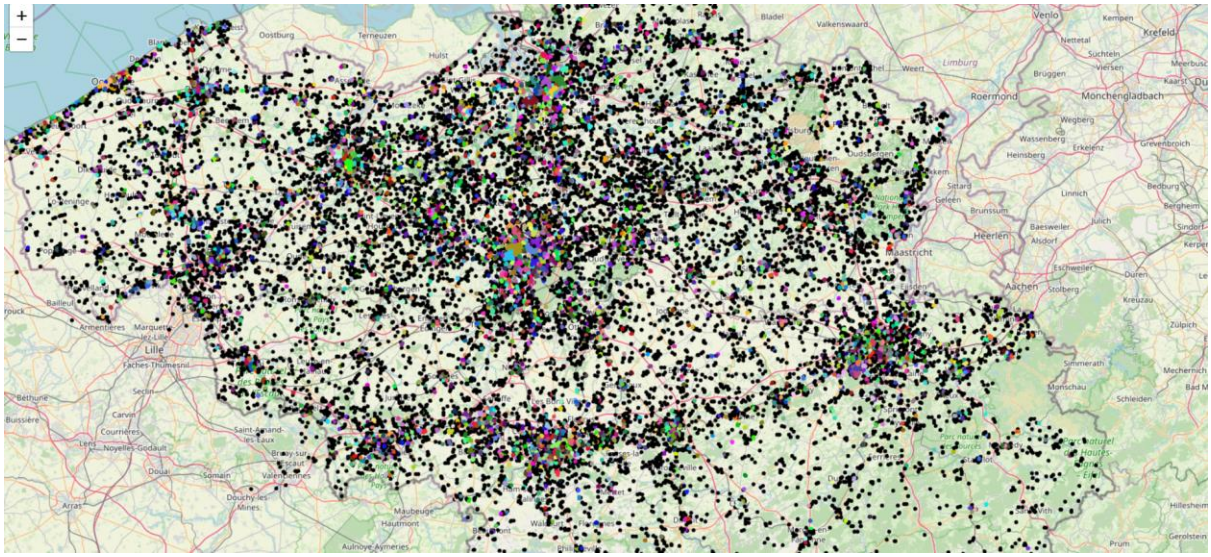
We will also only reallocate the AED's for which we are certain they are actually underperforming, and can be reallocated. We will only move the AED's that are underperforming, or that are not available to the public (although we would need to verify if these last ones are actually privately owned or not, since if they are, they cannot be reallocated)

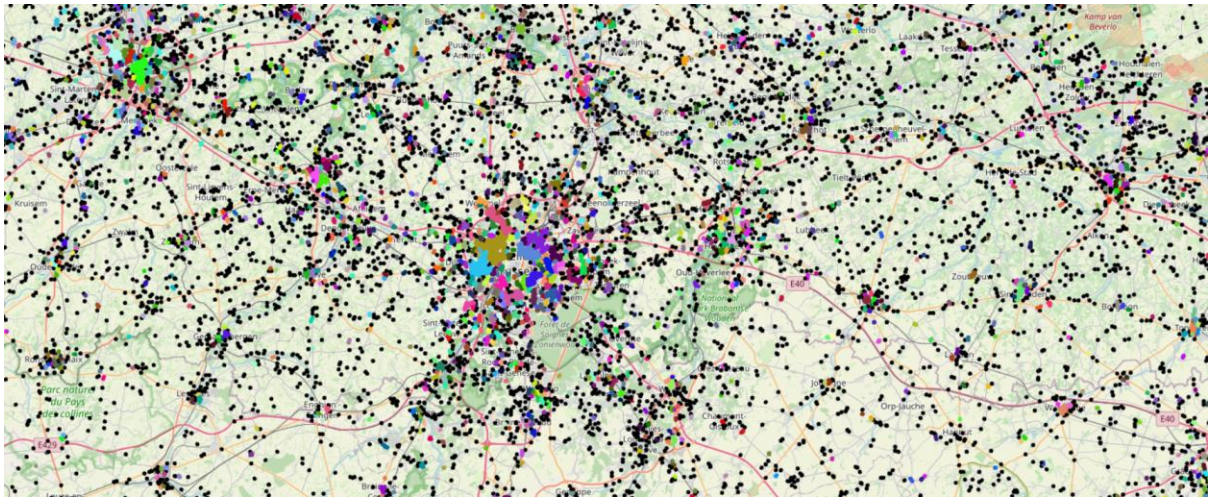
We use DBSCAN to find our initial clustering. This is so because it is able to make use of both our rules, namely the clusters are above a certain density, and the clusters contain a minimum amount of instances.

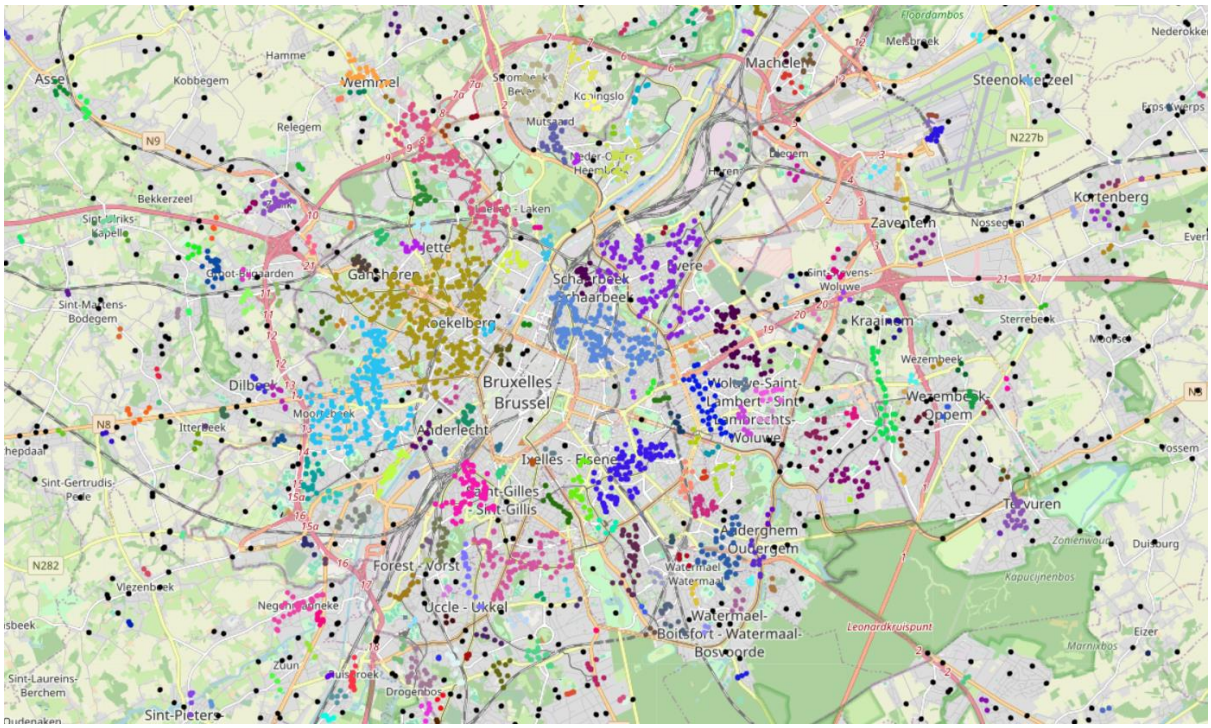
However, using DBSCAN we cannot make use of our haversine formula directly. Only Euclidean distance can be used. This is not a real problem, since the distances used should be small anyway (in 3 minutes time, the distance is approximately only 217.5 meter; it does not really matter if it is a couple meter more or less due to this inaccuracy). We calculate an approximate distance for the coordinates, expressed in degrees, which will be used.

We use the approximate distance an AED is valid in, calculated before, expressed in degrees, as our value for the epsilon parameter. As minimum amount per cluster we select 3, based on earlier reasoning.

3522 clusters get created, with the biggest cluster containing 382 instances. In total, 25129 out of 56284 cardiac cases get assigned to a cluster. We plot the clustering solution. Brussels contains a lot of instances in clusters. Notice however that the centre of Brussels city contains almost no instances, while for example Schaerbeek does contain instances. This aligns with the fact that the centre already has a lot of AED's, thus not needing any more there. For the rest, this solution finds clusters in cities and towns, but not really outside of that. We also generate a .txt file (size_counts_output.txt), in order to easily inspect all clusters and their counts. (plot: each cluster gets it own color. Points in black are noise points, i.e. points not included in any cluster) (plot available as dbscan_plus_card_arrest.html)


Lille
Aachen
Maastricht
Venlo
Roermond
Mönchengladbach
Heerlen
Eindhoven


Leuven
Tienen
Zoutleeuw
Orp-Jauche


Asse
Kobbegem
Wemmel
Hamme
Relegem
Machelen
Steenokkerzeel
Bekkerzeel
Sint-Ulriks-Kapelle
Jette
Ganshoren
Schaarbeek
Schaarbeek
Evere
Zaventem
Kortenberg
Nossegem
Sint-Martens-Bodegem
Koekelberg
Kraainem
Sterrebeek
Wezembeek
Dilbeek
Bruxelles - Brussel
Wolluwe-Saint-Lambert - Sint-Lambrechts-Woluwe
Wezembeek-Oppem
Molenbeek
Anderlecht
Ixelles - Elsene
Sint-Gertrudis-Pede
Sint-Gilles - Sint-Gillis
Tervuren
Vlezenbeek
Forest - Vorst
Uccle - Ukkel
Watermaal-Bosvoorde
Sint-Laureins-Berchem
Sint-Pieters-

We decide to assign AED's to clusters proportional to the size of the cluster. However, there were some problems caused by this assignment. This was because of two reasons:

-the number of AED's assigned per cluster is on average around 3.2380; but the minimum size of clusters is 3 (and those get 1), which results in too many AED's attributed

-there is rounding done in the assignment, so if for example 1.7 AED's get assigned to a cluster, 2 get assigned, resulting in 0.3 too many assigned

We account for this problem by both changing the rounding done, and reallocating AED's until the correct total amount of AED's was assigned. Details on how this was done are in the notebook, and depend on whether or not too many were assigned, or too few.

Now that we have the clusters and the number of AED's appointed to each cluster, we need to decide where exactly in the clusters the AED should be placed. The method is different for small clusters, for which only 1 AED was appointed, and bigger clusters, for which 2 or more AED's were appointed. For small clusters, we simply place the AED at the centroid of the cluster. For bigger clusters, we try to place the AED's in such a way that they cover as much area as possible.

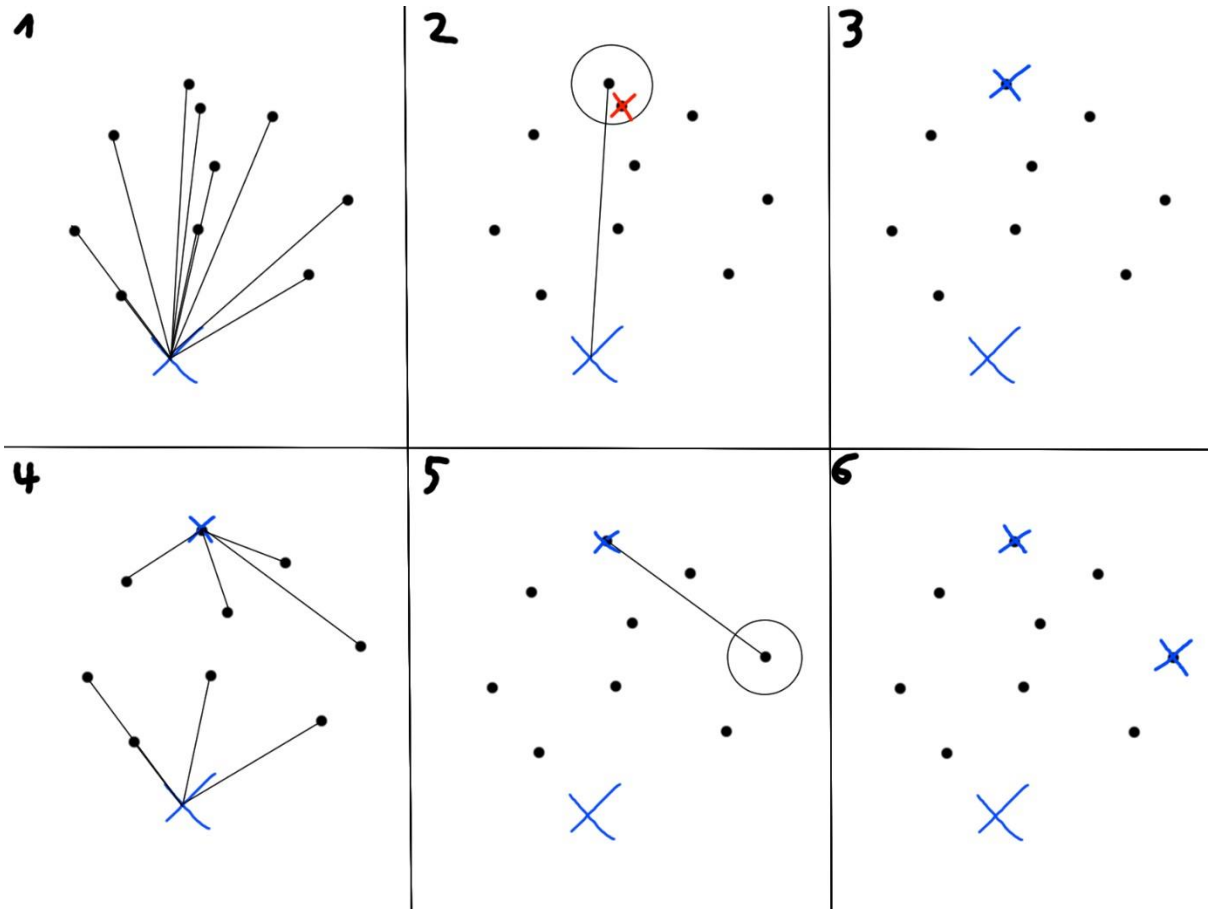
The reasoning for placing the AED's in such a way to cover as much area as possible, is the following: the exact locations within each cluster should be more or less random. We know that each cluster has a certain density in distribution, therefore it makes sense to attempt to cover as much area as possible, since it then is more likely that the future occurrence will also be covered. To do this, we selected a greedy algorithm. It is explained in detail below.

The exact algorithm for covering as much area as possible is the following (see also image below, the step in the image is indicated in the text):

Before anything, for each cluster, we generate random points, based on the points already present in the cluster. These points plus the original points in the cluster are the points contain the locations which we consider for placing the AED's assigned to the cluster. We take the bottom most point and append it to a list of centroids, this is our starting point to get the algorithm going. Then, for each point that is not in the list, we calculate the distance to this initial point (1). We take the point that is the furthest away, this point becomes a new centroid. We check the area around the centroid, to see if there are any within the range of X, with X defined earlier. The points that are within this range get removed from further analysis (2). Then the next iteration starts (3). We calculate for all remaining points the distance to the centroid it is closest to (4). Then, we pick the point that has the greatest distance to its centroid; it becomes our next centroid. We check the area around it with radius X, and remove those points (5). Then the loop repeats again(6). This process gets repeated, until the amount of centroids equals the amount of AED's we needed to place, or until there are no points left to place AED's (which means that the area is perfectly covered by AED's). Since the algorithm aims to maximize coverage at each iteration instead of looking at the bigger picture, it is a greedy algorithm. The results indicate that this is just fine.

This algorithm is applied to all clusters for which more than 1 AED was assigned. For clusters with only 1 AED assigned, the AED was simply placed at the centroid.

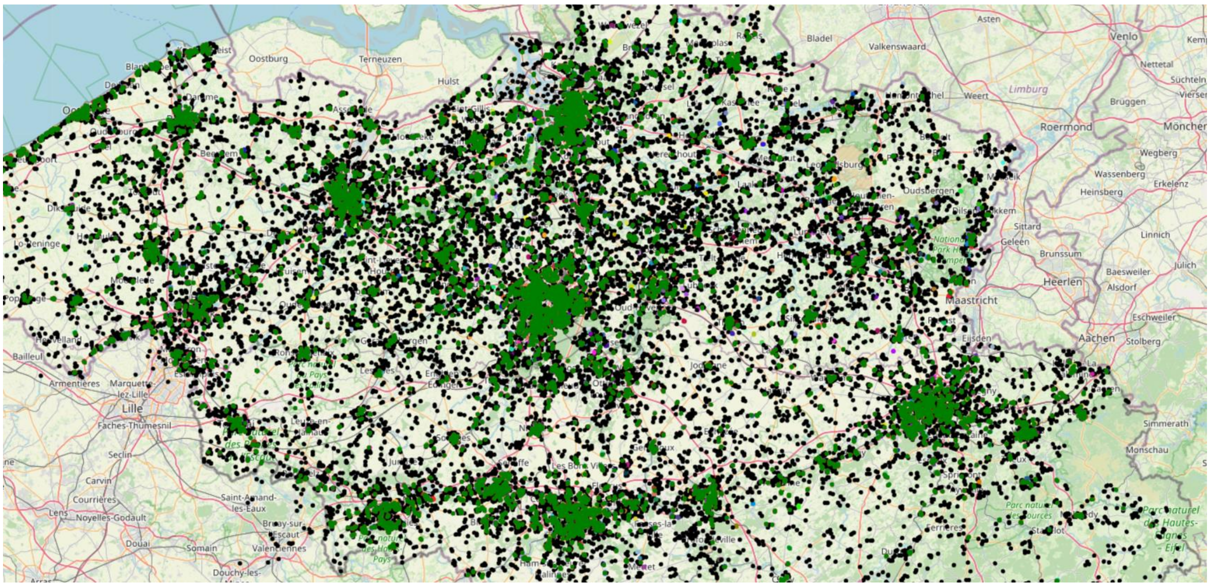
The following image explains the different steps again, and can help in understanding easily what happens:

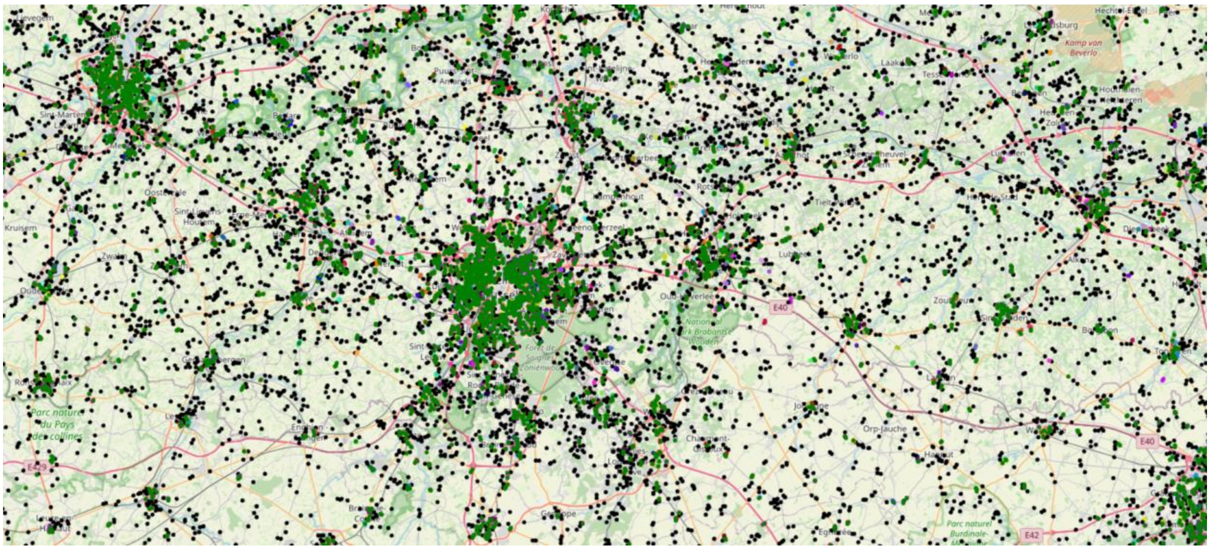


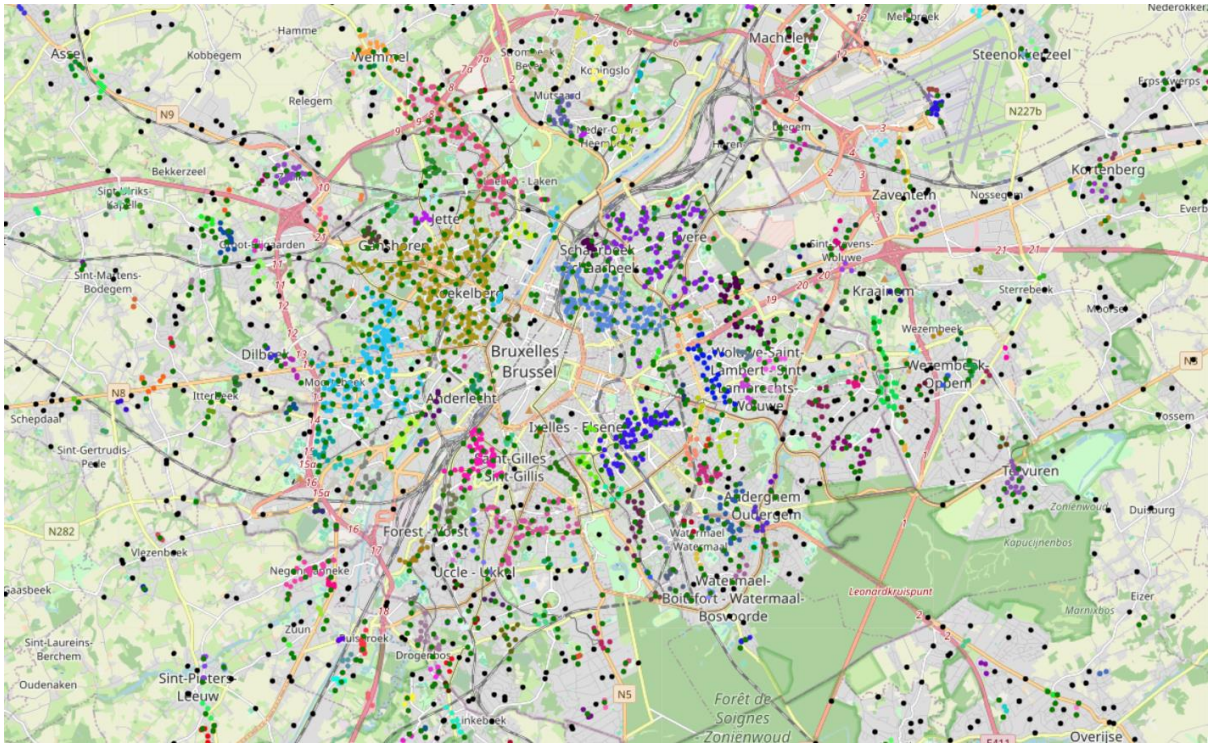
Instead of using the greedy algorithm used here, a grid method was considered, where a grid was overlaid on the cluster. However, it was not used, since it is hard to generate grids in complex shapes, while our algorithm easily deals with this. Since we are using DBSCAN, the shapes of the clusters will actually be complex, instead of simple circles or squares.

Using this method, some amount of AED's do not get placed. This is because it is possible for the algorithm to stop when perfect coverage is achieved. It is actually not so clear what to do with them; If we place them on the map, their locations will not be optimal according to our solution. We can change some parameters in order to place more AED's on the map but this does not actually guarantee better overall performance. But if we keep them, they are not used anyway. Randomly selecting the amount of unplaced AED's and not moving them in the first place is also a solution, as this saves some costs. The AED's could also be used as backups, and stored somewhere, or perhaps even sold. There are many possibilities to deal with these unassigned AED's.

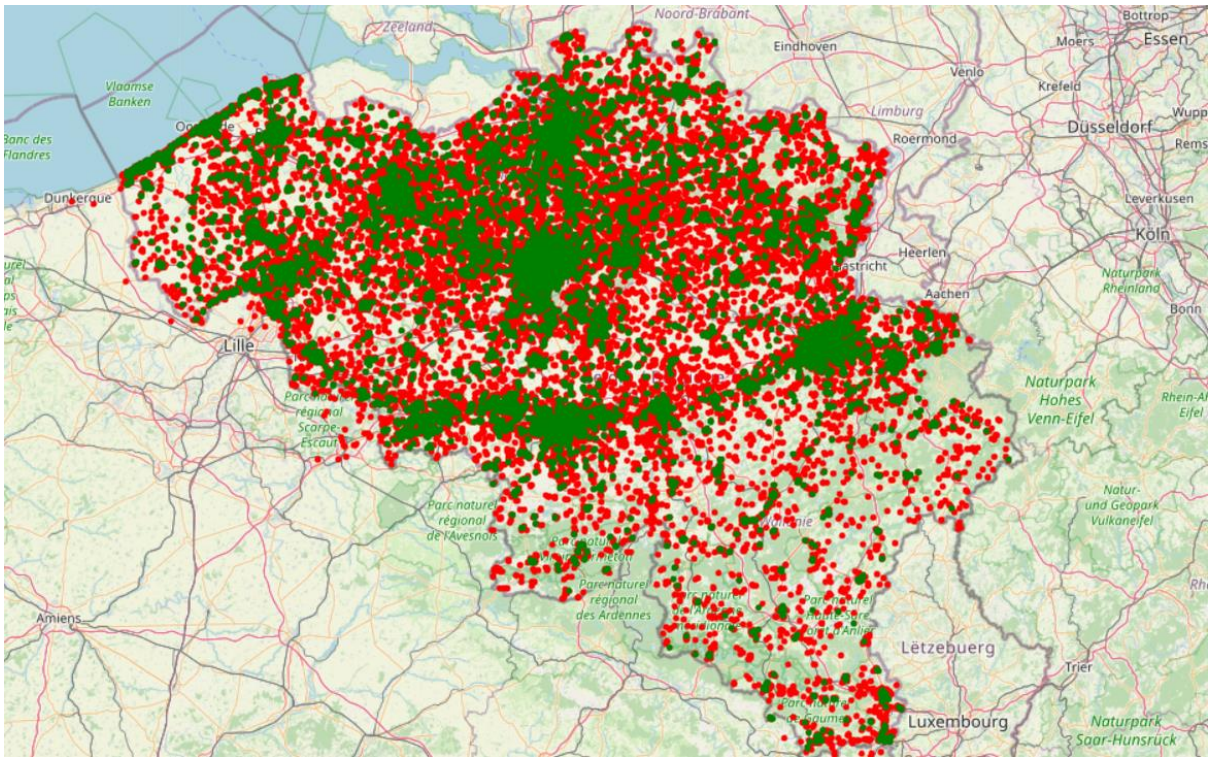
We create a plot of the solution. As expected using this method, the AED's are placed at locations with dense cardiac arrest occurrences, for larger clusters in a grid-like fashion. (colored dots are points belonging to a cluster, green dots are the placed AED’s (so excluding the AED’s that were already valid)) (plot available as AED_placed.html)


Oostburg
Terneuzen
Hulst
Bladel
Valkenswaard
Asten
Venlo
Nettetal
Weert
Limburg
Roermond
Brüggen
Wegberg
Wassenberg
Heinsberg
Erkelenz
Oudsbergen
Sittard
Geleen
Brunssum
Linnich
Baesweiler
Jülich
Heerlen
Alsdorf
Maastricht
Eijsden
Aachen
Eschweiler
Stolberg
Simmerath
Monschau
Bailleul
Armentières
Marquette-lez-Lille
Lille
Faches-Thumesnil
Seclin
Carvin
Courrières
Lens
Noyelles-Godault
Douai
Somain
Saint-Amand-les-Eaux
Valenciennes
Douchy-les-

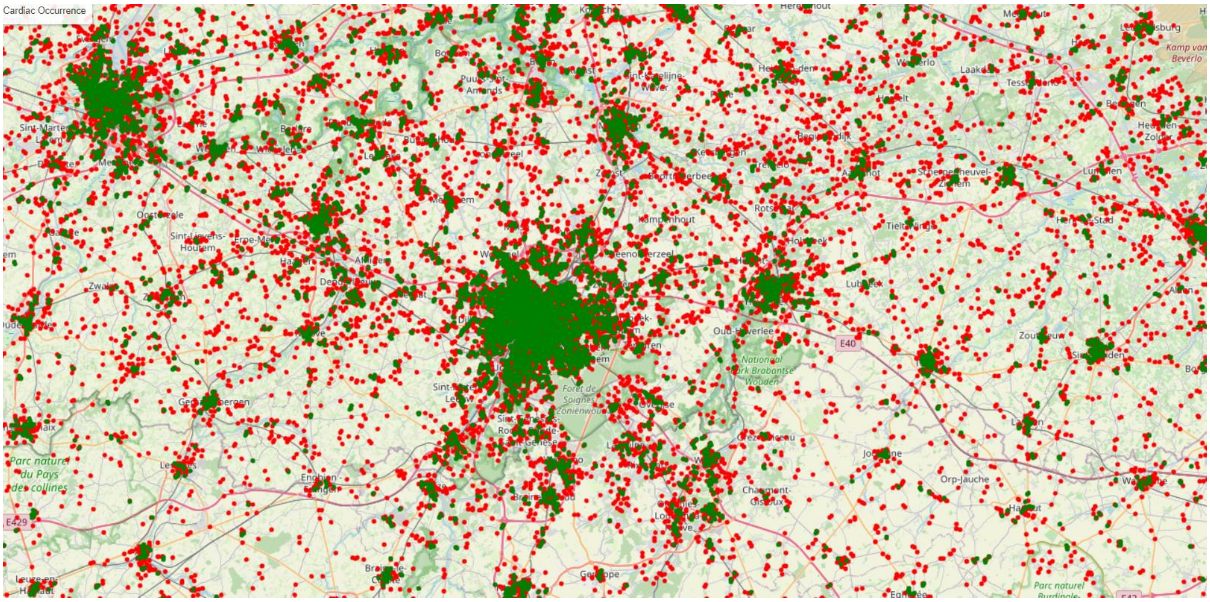

Kamp van Beverlo
E40
Orp-Jauche
E42
Parc naturel du Pays des collines

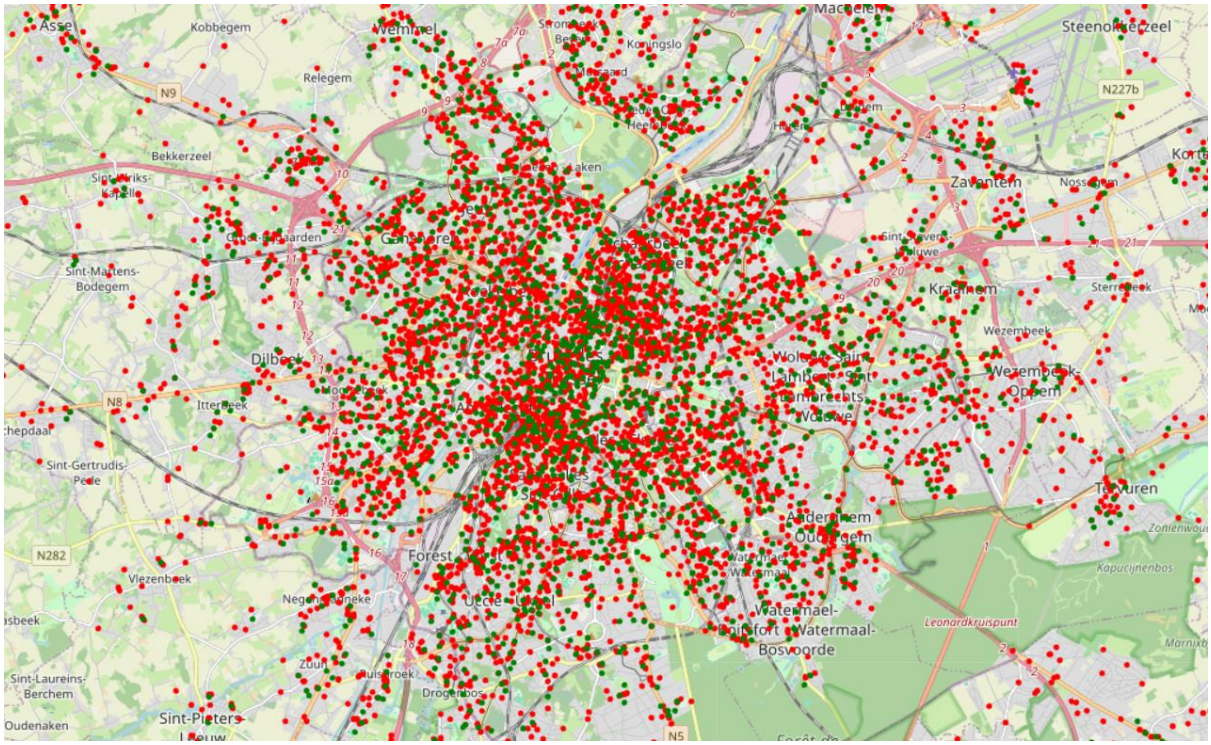


We also show the full setup, now including the AED's that were already considered valid, along with all the datapoints for the cardiac arrests. (green: AED, red: cardiac instance) (available under cardiac_arrest_AED_solution.html)




Oostburg
Terneuzen
Hulst
Bladel
Valkenswaard
Venlo
Roermond
Limburg
Weert
Sittard
Geleen
Heerlen
Maastricht
Aachen
Lille
Armentières
Seclin
Carvin
Courrières
Lens
Douai
Somain
Valenciennes
Arras
Monschau
Simmerath


Cardiac Occurrence
Kampenhout
Oud-Heverlee
E40
Orp-Jauche
Jodoigne
E429
Geraardsbergen
Parc naturel du Pays des collines



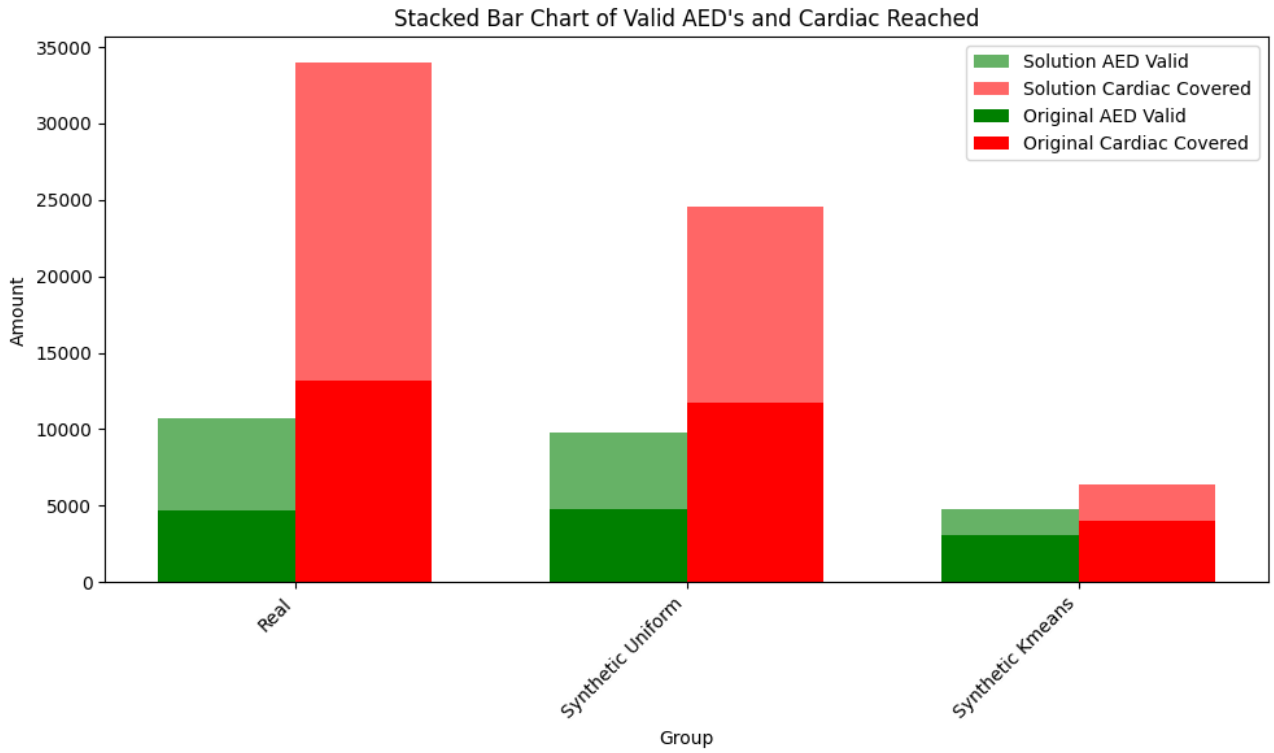
Notice the very large amount of AED’s in Brussels and other major cities now. The solution draws away AED’s from countryside and small towns, towards larger towns and cities.

Results

We also test the performance by again looking at how many AED's are now valid and how many cardiac arrests are now covered. We need to merge our solution back with the AED's that were deemed valid, since that is actually also part of our solution.

We test performance on our original data, and on our two synthetic datasets. Keep in mind that the K-means dataset was not that realistic, so performance on this one is less relevant. Using our solution, 10749 AED's (out of 12242) are now considered valid, and 33 959 cardiac arrests (out of 56 284) are now considered covered. This is a massive improvement over before (4668 valid and 13219 covered originally). The performance on the uniformly generated synthetic dataset shows a more moderate improvement in performance: 9272 are now valid, 24253 are now covered (compared to the 4821 and 11735 of the original setup). The solution also shows improvement on the K-means synthetic dataset, but not so much: 4617 AED's valid and 6232 cardiac cases covered (compared to originally 3077 original valid and 4023 covered cardiac).

A simple stacked bar chart is created in order to display these improvements in a clear fashion.


Stacked Bar Chart of Valid AED's and Cardiac Reached
Solution AED Valid
Solution Cardiac Covered
Original AED Valid
Original Cardiac Covered
Amount
35000
30000
25000
20000
15000
10000
5000
0
Real
Synthetic Uniform
Synthetic Kmeans
Group

Remarks:

When running the notebooks, there might be very small differences (a couple of instances can be different). This is caused by the geocoding, which can time out. This was accounted for, but some small amount of instances can still be different.

Although the plural “we” is often used in the report and in the notebooks, the whole project, from start to finish, including every single detail, was created by myself.