

**X MODELLING WEEK**

---

**DESIGNING ALGORITHMS FOR  
SEARCHING FOR OPTIMAL/TWIN  
POINTS OF SALE IN EXPANSION  
STRATEGIES FOR GEOMARKETING  
TOOLS**

---

**axesor**



FACULTY OF MATHEMATICS

**PARTICIPANTS:**

AMANDA CABANILLAS (UCM)  
MIRIAM FERNÁNDEZ (UCM)  
HUMAIRA HUSSEIN (University of Leicester)  
ARMIN KEKIC (University of Oxford)  
MANUEL PULIDO (UCM)  
VERÓNICA SÁNCHEZ (UCM)

**INSTRUCTORS:**

JAVIER LEÓN  
BEGOÑA VITORIANO

MÁSTER EN INGENIERÍA MATEMÁTICA

05 de julio de 2016

## Index

|   |    |
|---|----|
| 1. Introduction .....                         | 3  |
| 2. Description of the problem and work.....   | 4  |
| 3. Data .....                                 | 5  |
| 4. Finding weights .....                      | 6  |
| 5. Similarity measure and twin selection..... | 7  |
| 6. Alternative variable selection.....        | 8  |
| 7. Algorithm .....                            | 9  |
| 8. Computational results.....                 | 11 |
| 9. Conclusion and further work.....           | 15 |

## 1. Introduction

This project is about how to find new strategies for the expansion of the sales networks of businesses, mainly retail, commercial chains or franchises. The idea is to search new points of sale where they can place a new store that will be successful.

This work requires a lot of precision and sure results because companies invest large amounts of money in the creation of new shops, so a bad decision in this kind of strategies could cause big economic losses for businesses. That is why a very important part of this process is to collect the necessary information and create a database which will be the base of all the methods and tools that will be developed.

The next step is to decide the criterion for determining the new best points of sales. In this case, we consider that these points should be situated in areas with similar characteristics to areas where the companies have their current best points of sales.

In particular, in our problem we want to find the most similar point to a determined original point of sale, in order to create a new store that would work as good as the original one. We call twin to this new found point.

Then, in this work we develop a mathematical algorithm that will be a tool for the companies to make these strategic resolutions in the most efficient and sure way.

The model will be developed in the areas of optimization and methodological classification and its final aim is to be integrated as another valuable tool into the geographical expansion strategies that companies already have.

## 2. Description of the problem and work

The objective of this problem is to design an algorithm that will determine new sales locations by analyzing the characteristics of the best sales locations where the company has a successful point of sale. This aspect is very important: we work with locations, which will be defined as the census sections in a city, and not with the features of the point of sales.

We are given the location of a specific shop of a chain store which has the best sales in the city of Seville and we want to find where to locate a new similar point of sale in the city of Malaga.

A data set is provided with specific socio-demographic information at census section level for each city. The data set will be formed by variables related with population who habit in the census and with the economic properties of the census.

Moreover, a database with distances between sections and walking and driving times is also provided. This data will let us define the area of influence of a census section, what we call its neighbourhood.

The first task we have to do is to identify the key features of the given sale location in Seville and we will consider them as those which determine the high sales of the store. We will develop a model that compares a group of features of a census section with the rest of them and it returns a weight to each variable in order to classify them by their importance. The most important variables obtained by this method of weighting will be considered as the features that make special the location in Seville.

Secondly, with the characteristics found before, we want to identify a similar point of sale to the best point but in the city of Malaga. Then, we establish a similarity criterion between two different sale locations. This criterion will consider that two locations are similar if they have very approximated values in the key features.

This will let us create an optimization algorithm to locate the most similar point of sale to the reference store, in terms of the variables that score highly on that point of sale. This model searches the locations in Malaga that maximize the similarity between potential locations and our best location in Seville.

In general, the company gives a list of variables that we have to evaluate to determine the key features of the best shop. We propose an alternative way for choosing the important variables that, somehow replaces the role of an expert. We will study the database of the original city for finding out the most representative variables.

### 3. Data

The information that we are provided to solve this problem are two databases with two cities (Malaga and Seville), more or less 500 census sections and approximately 150 features per city. The first database has the socio-demographic information for each census section such as the population, the age distribution, the nationality or the economic status. And the second one has the distances between different census sections. We divided this database in 3 different matrices:

One matrix with the geographical distance in kilometers (that is symmetrical).

The second matrix with the time that it takes to you to go from each section to the others by car in minutes. This matrix is nonsymmetrical because maybe it can takes to you more time in one way than the other one because of the traffic, for example.

The last matrix with the walking time in minutes that is nonsymmetrical for the same reason than the previous matrix.

With these matrices, we previously made a study about the data and we found that there are some census sections without socio-demographic information, so we decided to delete them because they do not give us any information.

When we finished checking the databases we started to think about new variables that maybe could be significant in our study, like the average age of the people in each census section or the amount of foreign people, because there are some nationalities that have a small sample so maybe they could have less weight than if we put all the foreigners together.

Moreover, we thought that what happens on a census sections not only depends on that section but also depends on what happens on his neighbourhood, because if it takes to you only 5 minutes walking to go to another section it is possible that you go there, so you have an influence on your neighbourhood.

So, we create a  $t$ -neighbour definition:

“A section  $A$  is said to be a  $t$ -neighbour of section  $B$  if the maximum time it takes to travel from one to the other is less than or equal to  $t$  minutes”

This is a symmetric definition, so if  $A$  is a  $t$ -neighbour of  $B$  then  $B$  is a  $t$ -neighbour of  $A$  in spite of the time that it takes to you to go from  $A$  to  $B$  is not necessarily equal to the time from  $B$  to  $A$ . So you can create different definitions of neighbours like 5 minutes walking neighbour or 10 minutes by car neighbour and we replicate the original and the new variables to each definition of neighbour.

## 4. Finding weights

At this point, in a realistic scenario, the client provides us a set of variables which are supposed to be the most influent according to him. So our aim is to construct an algorithm that ranks the variables in some way such that the higher values are given to those which make special the best section.

The idea is to define a relative measure for every variable by calculating how far the best section is with respect to the others. For a variable  $v$ , the formula is defined as follows:

$$\rho_v = \frac{|v_{best} - \bar{v}|}{\sigma_v}$$

That is, the distance from the mean of the variable and then normalized by the standard deviation. With this formula, a variable is going to take a high value whenever it differs sufficiently from the mean, which will mean that this variable makes our section unique.

Note that since our algorithm must be defined for a generic problem, which means that it should work with any database and not only for the example we are given, we assume that variables are nicely distributed, what may be that the distributions are unimodal or something similar to a normal distribution. This is important for the way we define  $\rho_v$ .

We then define *weights* by

$$w_v = \frac{\rho_v}{\sum \rho}$$

Just divide each one by the sum so that they sum 1.

Now, what about highly correlated variables? For example, for a sport shop the number of young people may be correlated with the number of people who practice sport. This may cause an overweight since two correlated variables basically provide redundant information. To deal with this problem, we compute the correlation matrix, which is a symmetric square matrix whose  $ij$  entry is just the correlation between the  $i$ -th and the  $j$ -th variable. We consider as high correlation a value of 0.8. With that, we take a row (or a column) and see if that variable is correlated with others. If so, we short the variables by the weights values and keep only the one with the highest weight.

## 5. Similarity measure and twin selection

Once we have established the weights, our aim is to determine a similarity measure in order to find a similar point in the destination city which we will call twin point of the best given one from the origin city.

To do so, we define similarity in terms of error, meaning that the higher the error is, the more different the section is considered with respect to the original one.

The formula we use to compute the error of the  $i$ -th section is

$$\varepsilon_i = \sqrt{\sum_j w_j (v_j^i - v_j^*)^2}$$

This is just a weighted sum of squares of the difference between the variables in the  $i$ -th section of the destination city and the same variables from the given best section.

The weights are just to give more importance to those variable with higher weights. In this way, for a low weighted variable we do not really care if the error is high as we do for high weighted ones.

A problem that can occur here is that same variables in both cities do not move in the same ranges of values. This may cause wrong results, so to avoid this we standardize the variables subtracting the mean and dividing by the standard deviation.

With the errors we can now determine the most similar section in the destination city. But what happens if we wish to locate more than one twin point? The procedure would be the same but a problem appears here. That problem is that close sections are probably expected to have almost the same similarity measure since they may share some properties. But then when we find several twins, they may appear quite close, something which is not desirable at all.

To avoid that, we add a penalization term to the error formula in the following way:

$$\varepsilon'_i = \varepsilon_i - K d_{ij}$$

Here, the distance is multiplied by a constant  $K$  which can be calibrated depending on how much we want the distance to be penalized. The larger the  $K$  is, the more we penalize the distance.

This formula is cumulative. When we locate the first twin we do not care about distances. But then, before selecting the following one, we subtract the distance from the already selected to the others. In the next iteration, we subtract the distance from the previously selected to the rest that have not been selected, and so on. So we do not only take into account the distances to the one selected in the previous iteration, but the distances to all the ones already selected.

Again, a problem could arise here because errors and distances do not move in the same scale. To solve that, we just divide each magnitude by the maximum of each one to re-scale them.

## 6. Alternative variable selection

As it has been described before, our algorithm needs the variables or features we want to consider for finding the twin as an input. In general, an expert in the company's sector of activity suggests these variables, based on economic decisions. Now, we want to replace the expert and we use the data of the original city of Seville for finding useful variables.

To do this task, we use principal component analysis, PCA. This procedure returns the variables that describe the variance of the data and distinguishes the different sections the best.

First, we do data treatment. The variables have different magnitudes, so we need to standardize the data. In this case, due to the distributions of the variables, it is better to rescale by the median than by the mean, in order to have all the variables in the same order of magnitude.

We also filter out the variables that have most of the values equal to 0, to avoid distorting the data.

After PCA, we obtain a new variable, the first principal component which is a linear combination of the original variables and it is the one that collects the most quantity of information by itself. Then, the original variables with largest coefficient in absolute value are those that contribute the most for the variance of the data set. Therefore, we should select few of those ones as the variables that make the difference among all the locations of the city.

However, PCA returns the more significant variables for the data set, that is, the variables that collect more information of the data. But an important variable for the data may not necessarily be important for the best section. That is why we use this technique only as a previous variable selection and not for assigning weights.

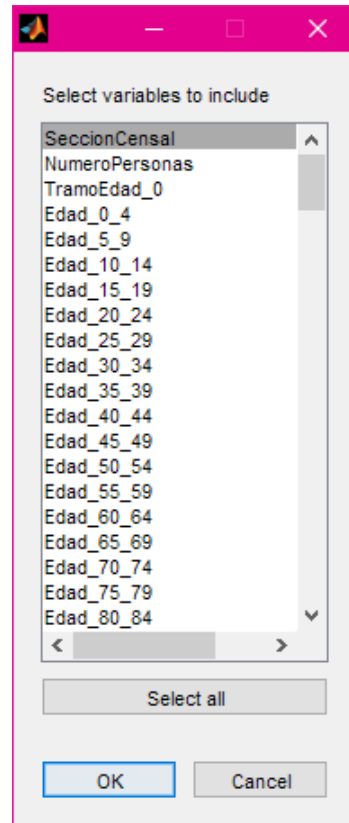
An interesting result is that at least the first ten variables obtained by PCA are economic topics.



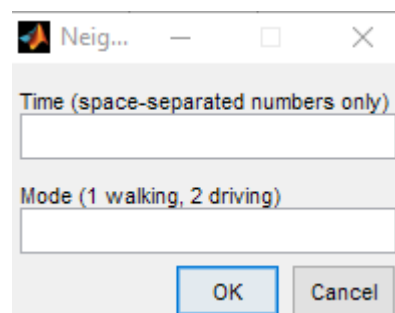
## 7. Algorithm

The main program is a function that calls input arguments by pop-up windows and returns the twins of the census section that you choose.

In the first window you can select all the variables that you want to include in the algorithm for searching the twin:

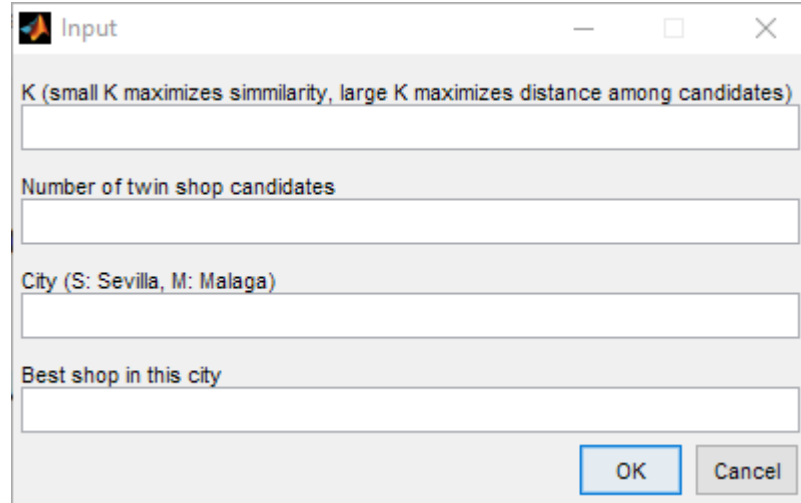


The second window is for defining the neighbourhood.



So if you put, for example, Time = 5 (minutes) and Mode = 2 it means that your neighbourhood is all the census sections that you can go there in less than 5 minutes driving.

And in the last window you can choose the  $K$  parameter in order to maximize similarity or the distance among candidates, the number of twin shop candidates, the city with the original shop (Seville or Malaga), and the census section of the shop that we want to calculate the twins.



This main function first of all loads the databases, and also calls another functions for solving the problem step by step:

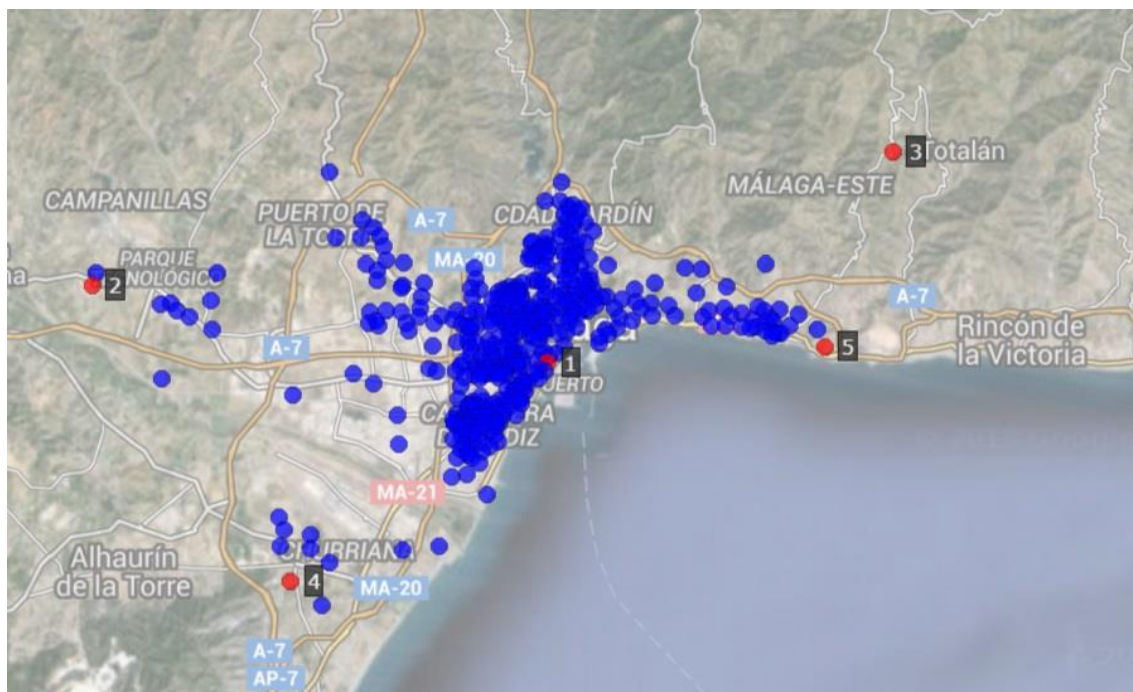
- neighbour\_features: This function calculates all the neighbouring sections (using the neighbourhood function) and replicates all the selected variables for the neighbourhood.
- find\_twins: This function calculates the  $n$  twins of the original city:
  - First of all it is necessary to find the weights of all the variables that we want to use (with the find\_weights function).
  - After that we normalize the data to mean zero and standard deviation 1 with the normalize\_data function.
  - Afterward we calculate the error (a measure of dissimilarity) between all the census sections and the original section with the calc\_errors function.
  - Finally we choose the best twins, minimizing the error between the original city and the twin and maximizing the distance between all the twins (using the  $K$  parameter).

## 8. Computational results

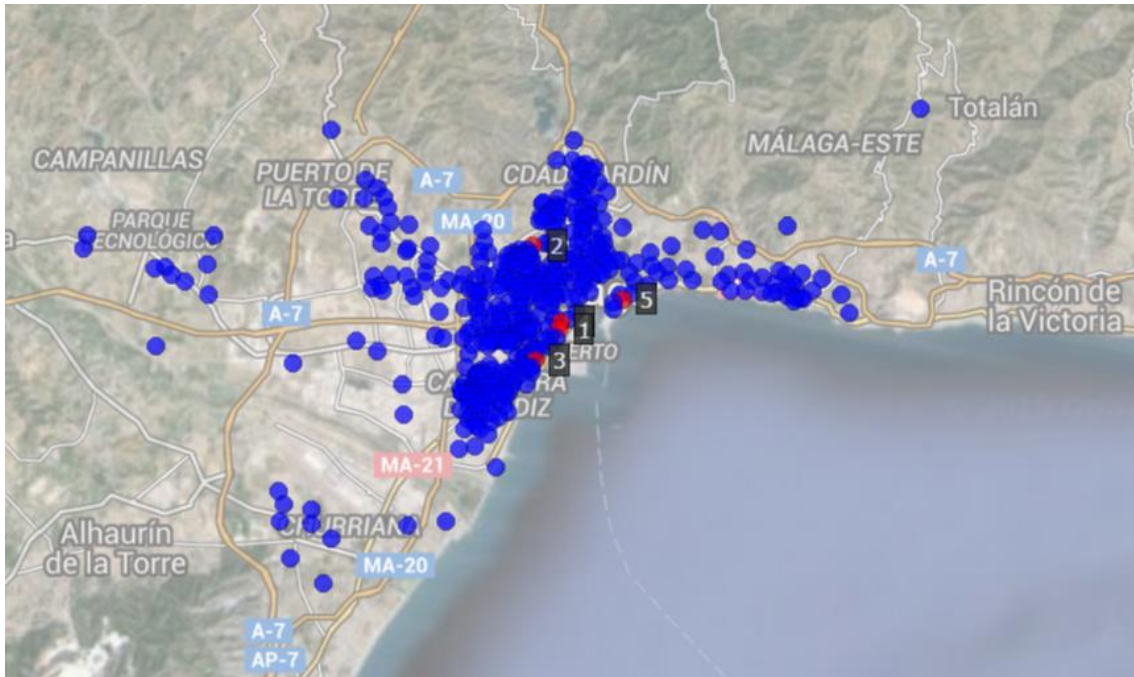
We have performed several tests with the program, changing the value of the  $K$  parameter.

In Picture 1 and Picture 2 the city of Malaga is shown. The blue points are all the locations that we consider in Malaga and the red ones are the locations that the program returns as the most similar places to the location in Seville. The number indicates the order of similarity.

If we take a large value of  $K$ , we obtain the locations more separated than with a small value. For example, we take the census sections 4109103001 in Seville, 5 minutes walking neighbourhood, all variables and we want to find the 5 most similar locations in Malaga. In Picture1 we take  $K = 1$  and in Picture 2 we take  $K = 0$ .

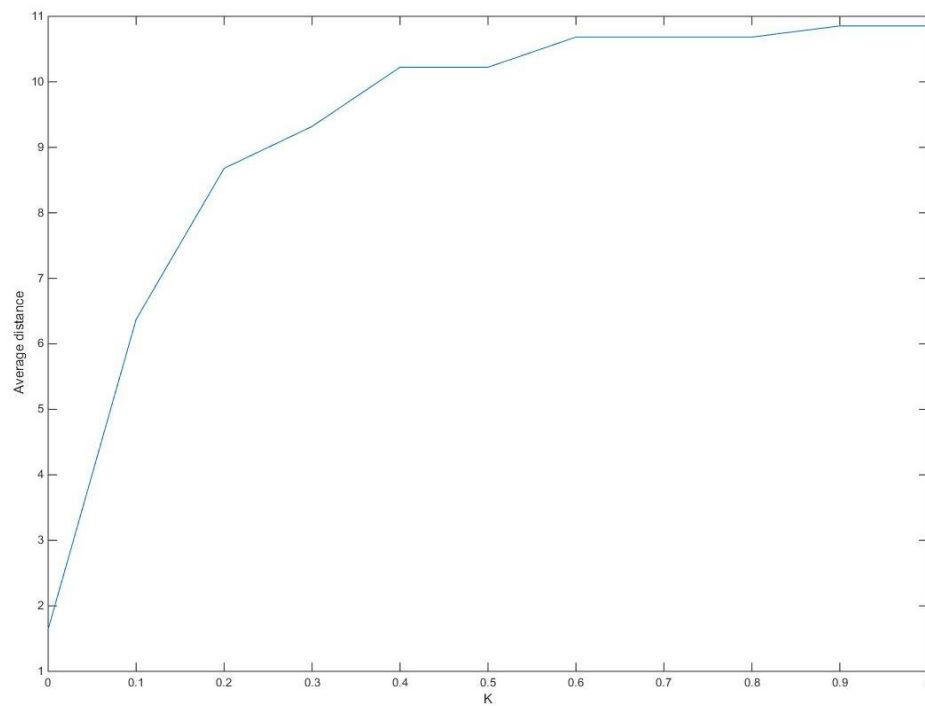


Picture 1



Picture 2

So, for large values of  $K$ , the average distance between the locations is larger too, as shown in Picture 3.

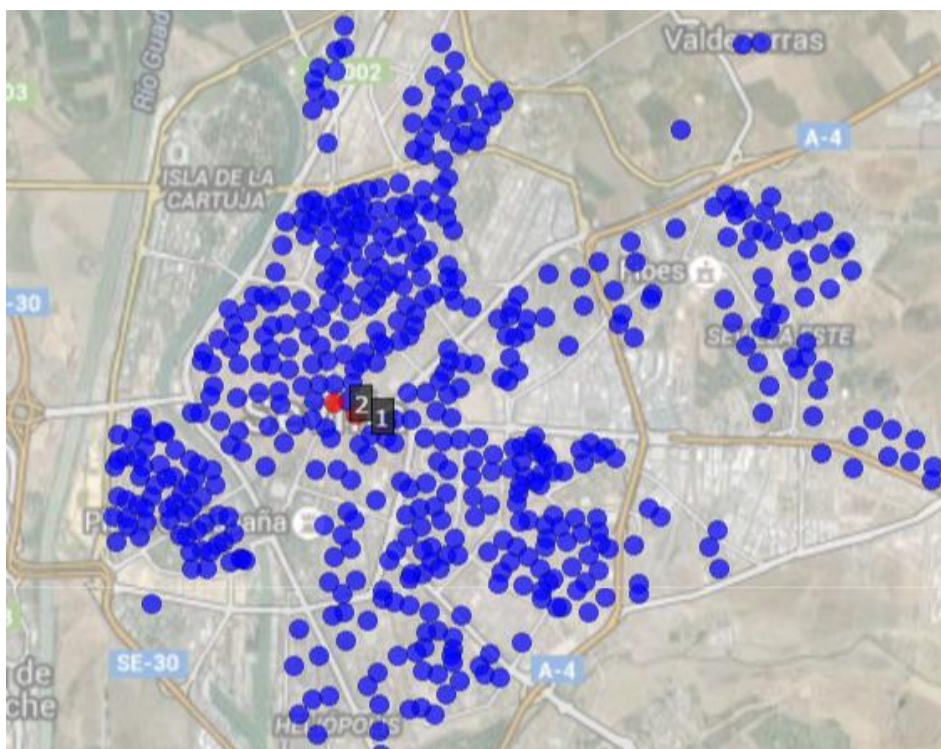


Picture 3

We tested the effectiveness of the algorithm obtaining the twin of a twin. For this we first found the twin in Malaga of a section from Seville. After that, we used the given twin in Malaga as an original section to find a twin in Seville. A desirable result would be that the final twin obtained in Seville is quite close to the first one. Mathematically we can express that as

$$Twin(Twin(Section)) = Section$$

We choose one location in Seville, the district point 4109103001 (the number two point in Picture 4). We consider all the variables with a neighbourhood of 5 minutes walking and  $K = 0$  (we do not penalize the distances, because we only obtain one location). We obtained as the most similar place in Malaga the district point 2906702034. Introducing the same parameters, we obtained the district point 4109108008 in Seville as the most similar to the point 2906702034 in Malaga (the number one point in Picture 4). The two points are so close, so we can say that the algorithm runs well.



Picture 4

In the next table you can see some execution times that we obtained with different parameter values. The time execution of the program is instantaneous, and the time is longer only when we take big neighbourhoods. That occurs because with big neighbourhoods we consider so much variables and calculating all the weights requires more time.

| Data (var, neighborhood, k, num locations) | Execution time (seconds) |
|--|--------------------------|
| (all variables, -, 0, 5)                   | 0.601511                 |
| (all variables, -, 0.2, 5)                 | 0.693765                 |
| (all variables, -, 0.45, 5)                | 0.573208                 |
| (all variables, -, 0.75, 5)                | 0.578169                 |
| (all variables, -, 1, 5)                   | 0.584181                 |
| (1 variable, -, 0, 5)                      | 0.594635                 |
| (20 variable, -, 0, 5)                     | 0.56209                  |
| (50 variable, -, 0, 5)                     | 0.556663                 |
| (70 variable, -, 0, 5)                     | 0.559931                 |
| (100 variable, -, 0, 5)                    | 0.574150                 |
| (all variables, -, 0, 1)                   | 0.605542                 |
| (all variables, -, 0, 20)                  | 0.576359                 |
| (all variables, -, 0, 50)                  | 0.576405                 |
| (all variables, -, 0, 100)                 | 0.715284                 |
| (all variables, -, 0, 200)                 | 0.953198                 |
| (all variables, 5 min walking, 0, 5)       | 1.815886                 |
| (all variables, 10 min walking, 0, 5)      | 1.982400                 |
| (all variables, 5 min by car, 0, 5)        | 2.870806                 |
| (all variables, 10 min by car, 0, 5)       | 4.005033                 |

## 9. Conclusion and further work

This is a general algorithm and we are only taking into account socio-demographic information. We could also include successful shops or more information related to the shops given, such as sales indicators, in order to obtain better. For example, if we had information about the kind of shop we could find better features of the shop and the location, because it is not the same to find the better location for a supermarket than a toy shop.

Another possibility could be using data from not successful shop in order to obtain bad features and penalize it with negative weights.

Other future work could be determining the distribution of each variable. We supposed a nice distribution but it is not necessarily true. If we obtain the distribution we can calculate better weights for the algorithm and we can obtain better results.