Practical application 3 : Unsupervised Classification

María Ayuso Luengo

November 2021

1 Introduction

A cluster is a collection of data items wich have high similarity between them and on the other hand low similarity with data items from other clusters. The general goal of clustering is finding the optimal division of the data to obtain low variance within clusters and high variance between them. This methods are part of unsupervised learning since there is not a class variable in which we want to divide the data.
In order to define these clusters is important to decide which "similarity" measure we will use and take into account the data types.

2 Problem description

Nowadays, before booking any trips or touristic attractions it's usual to look beforehand for recommendations and online ratings. In order to build a travel recommendation system we will apply clustering methods to a dataset with different user average ratings on churches, resorts, beaches, etc. This will be useful because while some people prefer going away to look for nature or relaxing others look for a more cultural approach. A good user clustering will help to look for ratings from people with similar preferences with the searcher.

The dataset consists on average ratings from 5456 users on different touristic sites shown in table 1. The ratings range from 1 to 5.

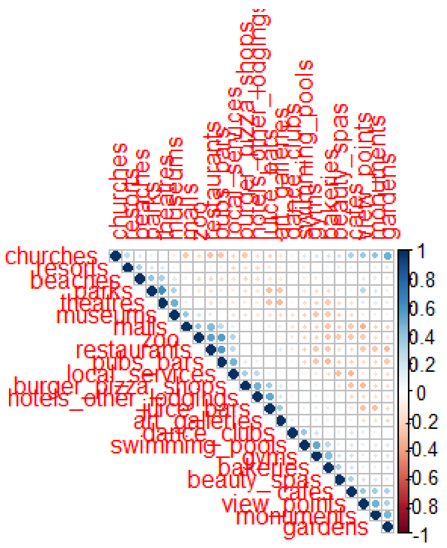
Category 1	churches	Category 9	restaurants	Category 17	swimming pools
Category 2	resorts	Category 10	pubs and bars	Category 18	gyms
Category 3	beaches	Category 11	local services	Category 19	bakeries
Category 4	parks	Category 12	burguer and pizza shops	Category 20	beauty spas
Category 5	theatres	Category 13	hotels or other lodgings	Category 21	cafes
Category 6	museums	Category 14	juice bars	Category 22	view points
Category 7	malls	Category 15	art galleries	Category 23	monuments
Category 8	zoo	Category 16	dance clubs	Category 24	gardens

Table 1: Variable description

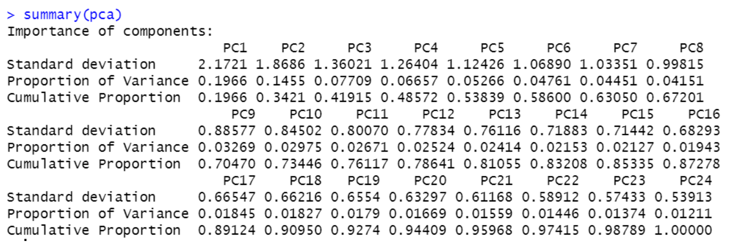
3 Data preprocess

Before applying any machine leraning algorithms is important to start with a data preprocess phase. Firstly, we can see that there are no missing values. Then we will proceed with a transformation of the variables names to have a better understanding of them.
Analyzing our dataset, we can see that it presents many outliers. These observations can be misleading for the clustering algorithms since they're based on distances. For this reason instead of using the euclidean distance, we will apply the manhattan distance for building the different clusters. This approach it's chosen because since euclidean distance aggregates the square of the differences it's more likely to be influenced by outliers, whereas manhattan distance will give more robust results to the presence of this data types.

Aditionally, to avoid any redundant variables we will study the correlation matrix of the variables. In Figure 1a we notice that in general all variables have a low correlation, therefore we decide to mantain all of them for the clustering. Moreover, when trying a PCA the first principal component only acumulates 19.66% of the observed variance. This result only consolidates the decision to keep all variables in the study.



(a) Correlation matrix graph



```
> summary(pca)
Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6     PC7     PC8
Standard deviation     2.1721 1.8686 1.36021 1.26404 1.12426 1.06890 1.03351 0.99815
Proportion of Variance 0.1966 0.1455 0.07709 0.06657 0.05266 0.04761 0.04451 0.04151
Cumulative Proportion  0.1966 0.3421 0.41915 0.48572 0.53839 0.58600 0.63050 0.67201
                           PC9    PC10    PC11    PC12    PC13    PC14    PC15    PC16
Standard deviation     0.88577 0.84502 0.80070 0.77834 0.76116 0.71883 0.71442 0.68293
Proportion of Variance 0.03269 0.02975 0.02671 0.02524 0.02414 0.02153 0.02127 0.01943
Cumulative Proportion  0.70470 0.73446 0.76117 0.78641 0.81055 0.83208 0.85335 0.87278
                          PC17    PC18   PC19    PC20    PC21    PC22    PC23    PC24
Standard deviation     0.66547 0.66216 0.6554 0.63297 0.61168 0.58912 0.57433 0.53913
Proportion of Variance 0.01845 0.01827 0.0179 0.01669 0.01559 0.01446 0.01374 0.01211
Cumulative Proportion  0.89124 0.90950 0.9274 0.94409 0.95968 0.97415 0.98789 1.00000
```

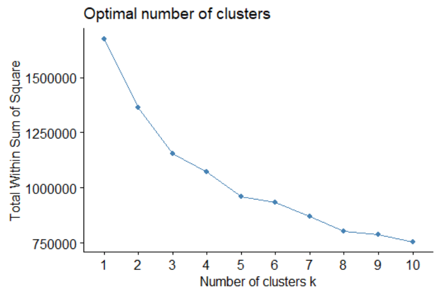
(b) Results from PCA

4 Methodology

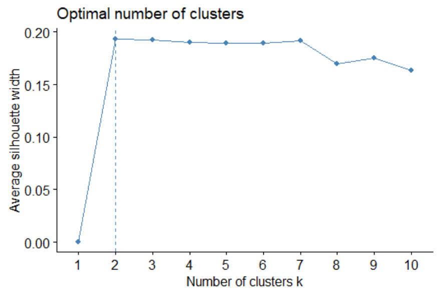
4.1 Partitional clustering

4.1.1 K-means

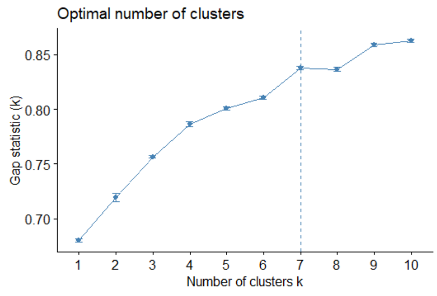
K-means algorithm is well known for clustering. As it name explains it builds k clusters where the centroids are calculated as the means of the different observations included in said cluster. Since the value of k must be an input for the algorithm we will take a look at three different measures to determine this number.
The basic idea of clustering methods is to minimize the total intra-cluster variation. The elbow method focuses on the total within-cluster sum of squares for different values of k and looks for an "elbow" (bend) on the graph. The value of k where the bend falls is considered as an appropiate number of clusters. We can see the graph obtained in figure 2a.
Another method for choosing the optimal k is the silhouette coefficient. Once the clustering has been done the silhouette of an observation represents the distance to neighboring clusters, therefore the higher the average silhouette is the better the cluster performs.
Finally, the gap statistic will be used to estimate the optimal value of k. The technique can be used for any clustering algorithm, in this case K-means, and compares the total intracluster variation for different values of k with their expected values under null reference distribution of the data. For more information on the gap statistic refer to [1] .
Inspecting these three approaches we decide to study K-Means with both $k = 2, 7$.
Once we have chosen both the distance measure and the number of clusters to look for, we are ready to proceed with the K-means algorithm and analyze the results.
It's important to note that the results from K-means depend on the centroids randomly assigned at the beginning of the algorithm. For this reason we will initialize it 50 times and select the one with the least total intra-variance.



(a) Elbow method



(b) Silhouette method

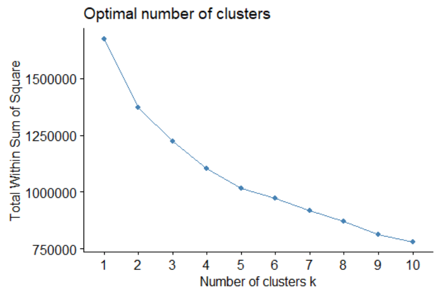


(c) Gap statistic

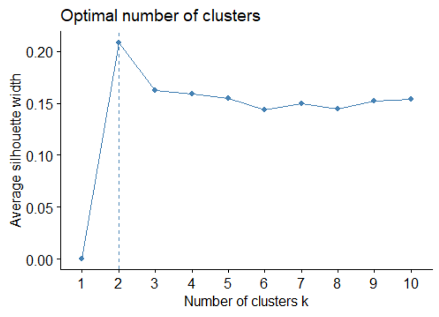
Figure 2: Results from different measures to find optimal k in K-Means

4.1.2 PAM

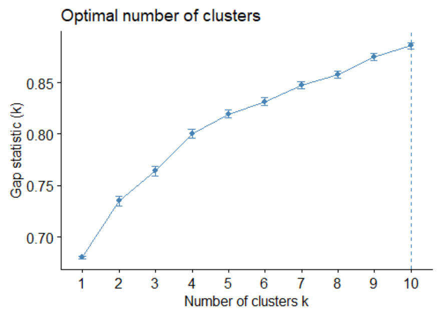
PAM(Partitioning Around Medoids) is a K-medoids algorithm. This method is a robust alternative to K-means in presence of outliers, however takes a higher computational cost. In this case, instead of calculating the means of the instances of each cluster, a specific observation is considered as the "centroid" of the cluster, and it's referred to as the medoid. This medoid is chosen as the cluster observation with the most average similarity within the cluster.
Analogously to K-Means we will look at the Within Cluster Sum Of Square (WSS), the average silhouette and Gap statistic to choose the optimal number of clusters. The results are presented in figure 3. Looking at the average silhouette coefficient we can see that $k = 2$ seems to give good results. On the other hand we can't see the 'knee' in Figure 3a and $k = 10$ as the gap statistic graph recomends will be difficult to interpret. On that account we will also study $k = 7$, since it does not look in the graphs as too bad, and will allow us to compare it with the clusters resulted by K-Means.



(a) Elbow method



(b) Silhouette method



(c) Gap statistic

Figure 3: Results from different measures to find optimal k in PAM

4.1.3 CLARA

The main disadvantage of the PAM algorithm is its computational cost. CLARA(Clustering Large Applications) is an improvement of this algorithm that implements a version of K-Medoids with resampling. After applying PAM to each data subset and assigning the remaining instances to the clusters built, CLARA selects the clusters with the least WSS.
For this algorithm we got really similar results to PAM, as expected, for the 3 measures used to select k. Consequently we will build two models with $k = 2, 7$. The graphs for this algorithm can be examined in Appendix A

4.2 Hierarchical clustering

On the contrary to K-means, an advantage of hierarchical clustering is that it is not necessary to choose the number of clusters before applying the algorithm.

4.2.1 Agglomerative

Agglomerative hierarchical clustering is built "bottom-up". When initializing, each observation represents its own cluster and these start merging in an iterative process that we will be able to visualize on the dendogram presented in the results section.
Concerning the approach to define cluster similiratity we will stdudy different types of linkages : complete, single, average, centroid and ward. Afterwards we will look at their cophenetic coefficient. This index is a measure of the correlation between the distance of points in feature space and the distance on the dendogram ([2]).
As we can see on Figure 4 the linkage that provides better results is average.

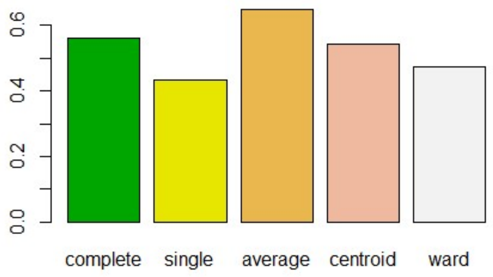


Figure 4: Barplot for correlation between feature distance and cophenetic for different linkages.

4.2.2 Divisive

Divisive hierarchical clustering, also known as DIANA (DIvisive ANAlysis Clustering), works inversely to agglomerative clustering and is built "top-down". At the beginning all observations start belonging to the same cluster and then this algorithm splits clusters recursively. An advantage of DIANA is that is not necessary to choose a linkage, just a similarity measure which makes it easier to employ.

4.3 EM

Model based approaches assume a variety of data models and apply maximum likelihood estimation to identify the most likely model and number of clusters. The mclust package selects the optimal model according to BIC for EM initialized by aglomerative hierarchical clustering for parameterized Gaussian mixture models. In this case we can see in figure 5 that the optimal model was VEV with 10 clusters, however the BIC value remains close between VEV and EEV. In this case we will study the 3 cluster types that provide the best BIC.

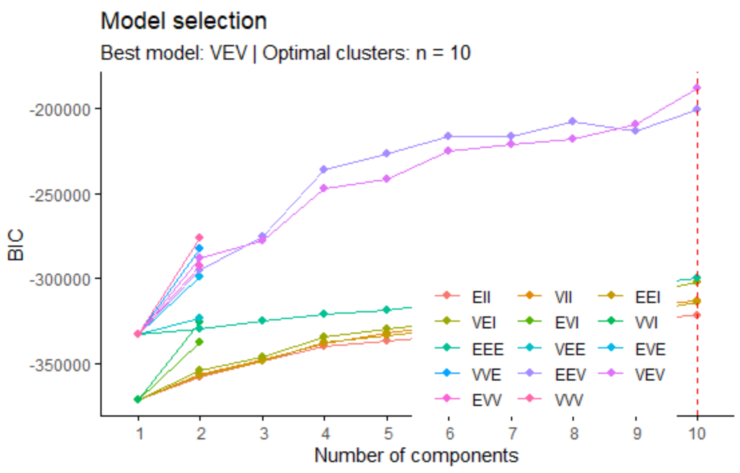


Figure 5: BIC for different number of clusters and models in `MClust` package.

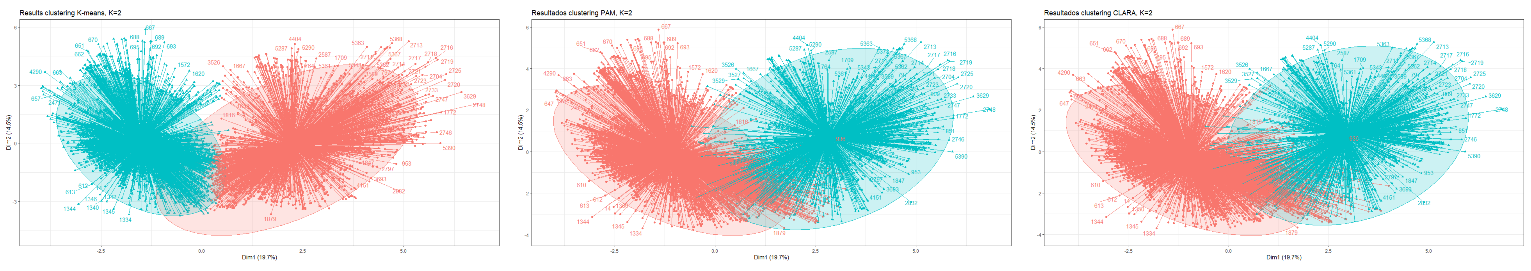
Model	Distribution	Volume	Shape	Orientation
EII	Spherical	Equal	Equal	-
VII	Spherical	Variable	Equal	-
EEI	Diagonal	Equal	Equal	Coordinate axes
VEI	Diagonal	Variable	Equal	Coordinate axes
EVI	Diagonal	Equal	Variable	Coordinate axes
VVI	Diagonal	Variable	Variable	Coordinate axes
EEE	Ellipsoidal	Equal	Equal	Equal
EVE	Ellipsoidal	Equal	Variable	Equal
VEE	Ellipsoidal	Variable	Equal	Equal
VVE	Ellipsoidal	Variable	Variable	Equal
EEV	**Ellipsoidal**	**Equal**	**Equal**	**Variable**
VEV	Ellipsoidal	Variable	Equal	Variable
EVV	Ellipsoidal	Equal	Variable	Variable
VVV	Ellipsoidal	Variable	Variable	Variable

Table 2: Geometric characteristics of the different models considered by the `MClust` package.

5 Results

5.1 Partitional clustering

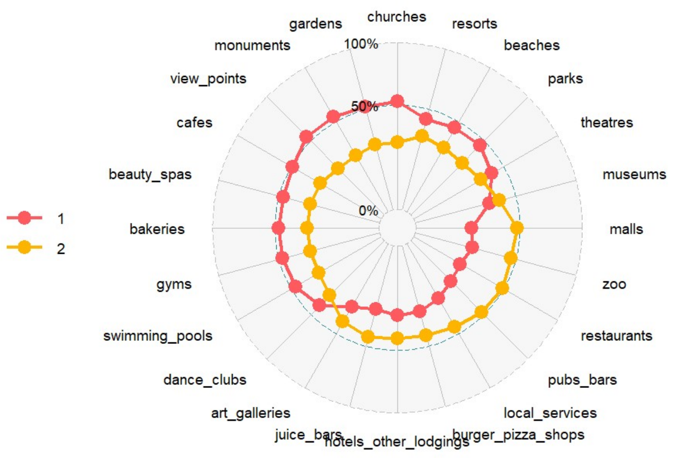
- K = 2
  For this section we will focus on the results obtained for partitional clustering (K-Means, PAM and CLARA) for K=2. First, we can see on Figure 6 an approximate visualization of the clusters obtained. The dataset used to build this clusters had 24 variables, therefore we applied PCA to facilitate a 2-D visualization. We can see that the clusters overlap but we can also see that the firs two principal components only acumulate 34.2% of variance, therefore this representation is not too accurate of the real distance of compactness of the clusters.



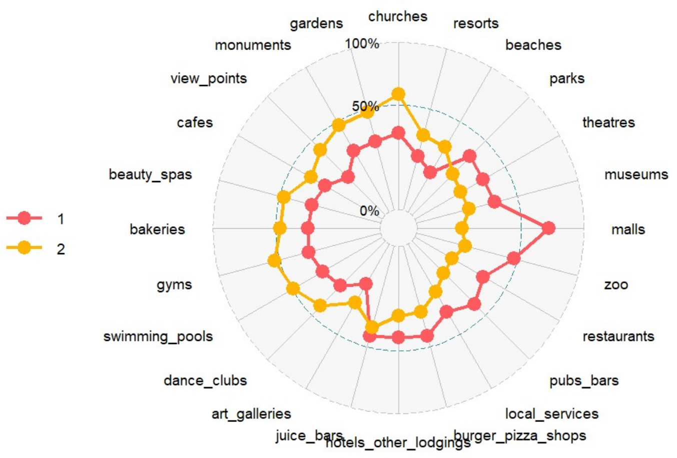
(a) K-Means cluster visualization with first two principal components (b) PAM cluster visualization with first two principal components (c) CLARA cluster visualization with first two principal components

Figure 6: K=2 partitional clustering representation

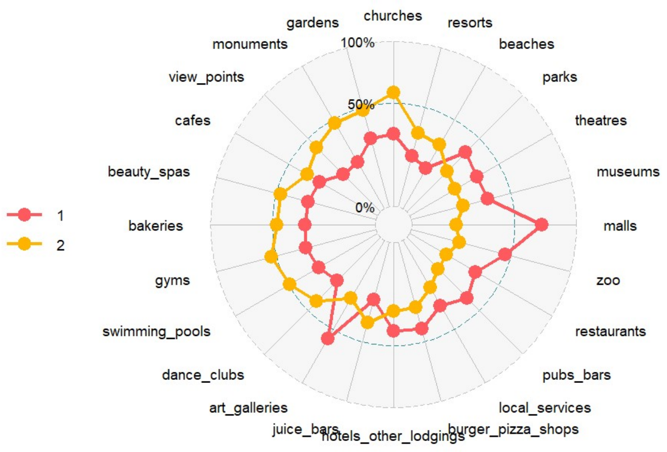
Now, in order to understand what these clusters represent we will look at Figure 7, where 3 radar charts are presented. In general we see that in all 3 algorithms the clusters are closer, on the one the one hand (right side of the charts) in variables 'parks' and 'theatres' and also looking at the left side in variables 'dance clubs' and 'art galleries'. Generally we see that the cluster dominant on the left side of the charts values more self-care activities like 'gyms' or 'beauty spas', and cultural/traveling activities like 'monuments', 'resorts' or 'beaches'. Meanwhile, the second cluster which is more dominant on the right side of the charts focuses more on gastronomical activities 'restaurants', 'pubs and bars', etc and recreatonial activities like 'malls' and 'zoo'.



(a) K-Means cluster centers visualization



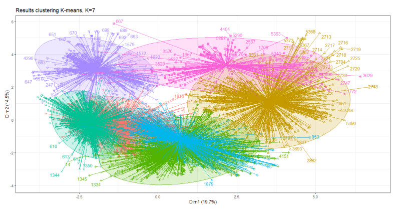
(b) PAM cluster medoids visualization



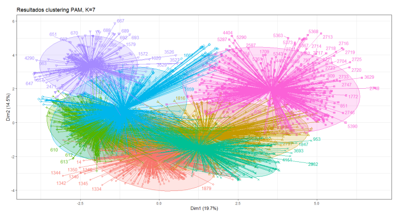
(c) CLARA cluster medoids visualization

Figure 7: K=2 partitional clustering centers and medoids

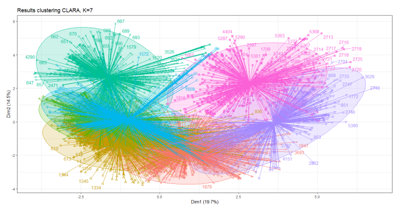
- K = 7
  Again, anagously to what we just saw with K=2 we start by visualizing the clusters with a principal component analysis. In this case, again, we see much overlaping between clusters but we can't consider this as a performance evaluation because of the low variance explained by PC1 and PC2.



(a) K-Means cluster visualization with first two principal components



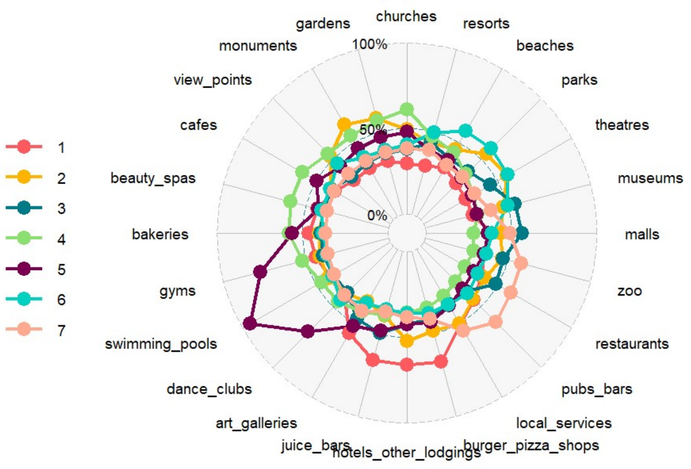
(b) PAM cluster visualization with first two principal components



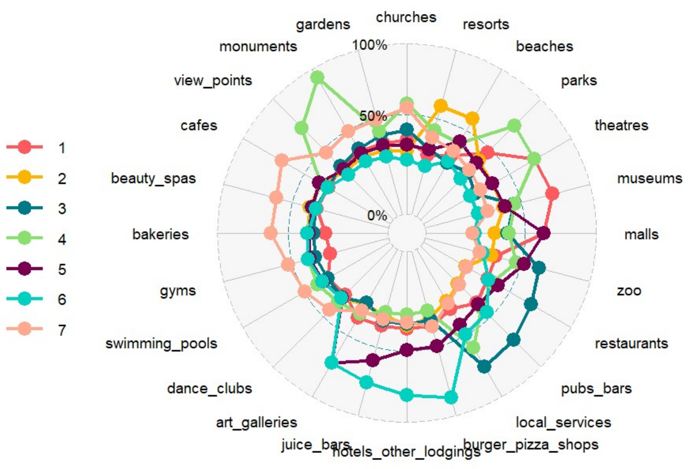
(c) CLARA cluster visualization with first two principal components

Figure 8: K=7 partitional clustering representation

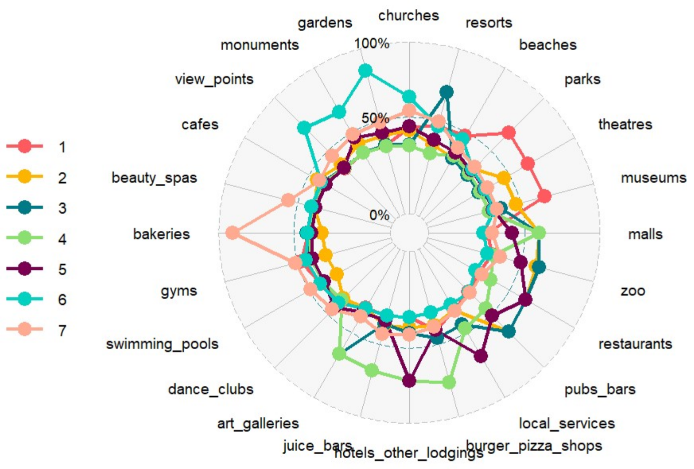
Nextly we will try to categorize the 7 clusters built by partitional methods. It's difficult to have a clear interpretation with this many models but we will try to get a generalization of some clusters. Usually we can see that the same cluster represents 'view points', 'monuments' and 'gardens. Also 'restaurants', 'pubs and bars' and 'local services' tend to go hand-in-hand. Finally, self-care activities are predominant in clusters 5,7 and 7 for K-Means, PAM and CLARA respectively. Depending on the algorithm used the predominance of this categorization is more or less solid.



(a) K-Means cluster centers visualization



(b) PAM cluster medoids visualization



(c) CLARA cluster medoids visualization

Figure 9: K=7 partitional clustering centers and medoids

5.2 Hierarchical

An advantage of hierarchical clustering, as we mentioned, is that the number of clusters is not necessary for the input. However once we have the dendograms for both agglomerative and divisive clustering, which can be found in Appendix B, a key decision is where to cut the trees.

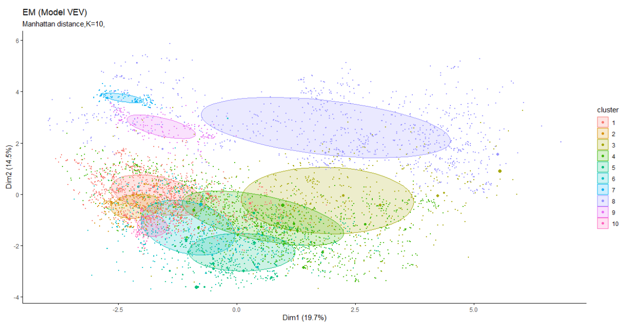
We will start cutting both trees through the height that provides 2 clusters. In this case, on the contrary to partitional clustering, we find that the clusters built are really imbalanced. Although divisive clustering seems somehow better than agglomerative, in table 3 we notice that this might not be the best clustering method for the users.

	Agglomerative	Divisive
Cluster 1	5449	4040
Cluster 2	5	1414

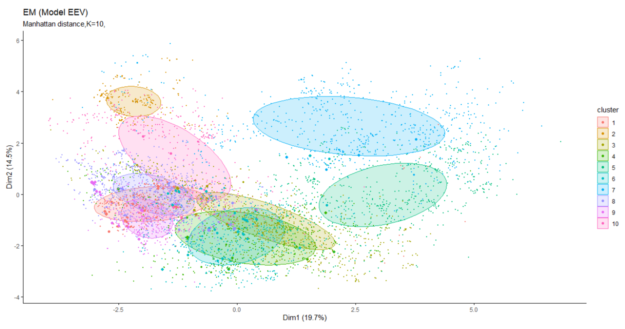
Table 3: Number of observations in each cluster

5.3 EM

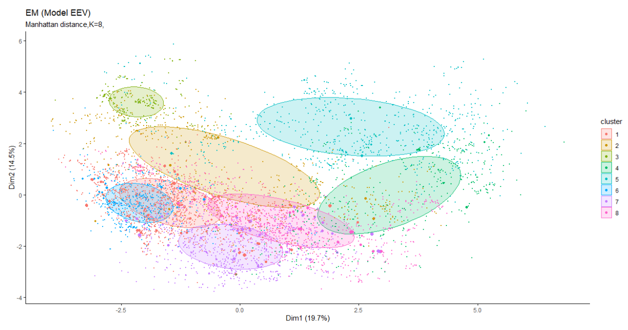
Based on the BIC results for the different models and number of clusters, we will study the 10 clusters built by VEV and EEV and 8 clusters built by EEV. Firstly, let's remember that this algorithm gives each observation a probability of belonging to each cluster. In Figure 10 we can visually look at this uncertainty. The bigger the observation is on the graph the more uncertain is its assignation.



(a) Cluster visualization based on uncertainty + PCA with model VEV



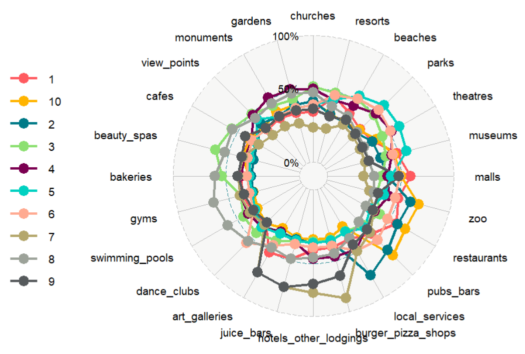
(b) Cluster visualization based on uncertainty + PCA with model EEV



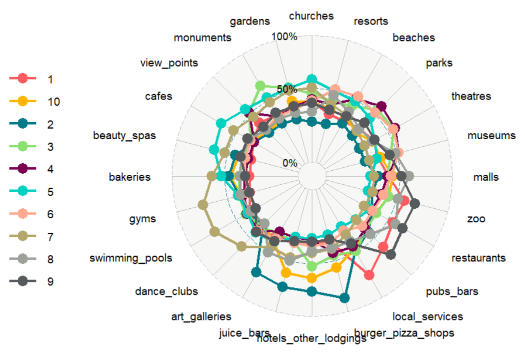
(c) Cluster visualization based on uncertainty + PCA with model EEV

Figure 10: Model based clusters

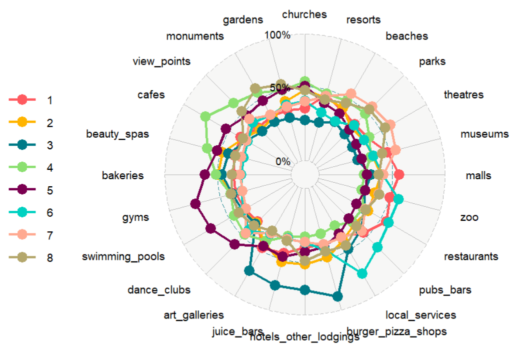
In general, observations belong to the clusters with a fairly good certainty. Maybe we can see an increment in clusters 4,5 and 6 but it's not significant. On the other hand let's look what these clusters represent (Figure 11).
It's hard to find a clear pattern with so many clusters however there are some overall repetitive dispositions. Firstly, in all three models we can see that in general users that rate positively 'gyms' also do it for 'smimmig pools' and 'bakeries'. Looking more at the gastronomical side we see a difference in clusters between users that enjoy more 'restaurants' and 'pubs and bars' and users that rate higher 'juice bars' and 'burgue and pizza shops'. Moreover focusing on these last variables we can also see a difference between fairly high ratings and another cluster that has averagely more ratings on these places than the rest but are more critic. This example can be seen in clusters 7 and 9, 2 and 10, and 2 and 3 for subfigures a,b anc c respectively in Figure 11.



(a) VEV model average cluster visualization (10 clusters)



(b) EEV model average cluster visualization

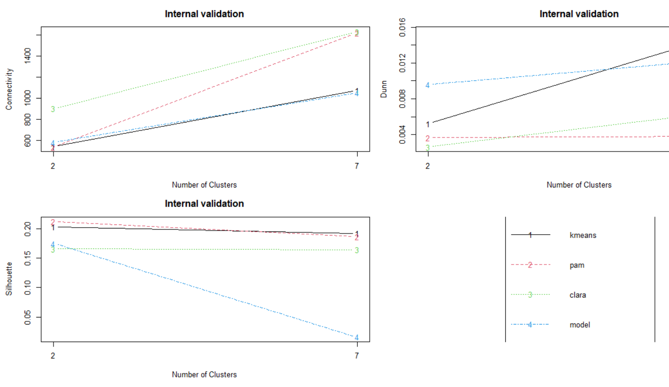


(c) VEV model average cluster visualization (8 clusters)

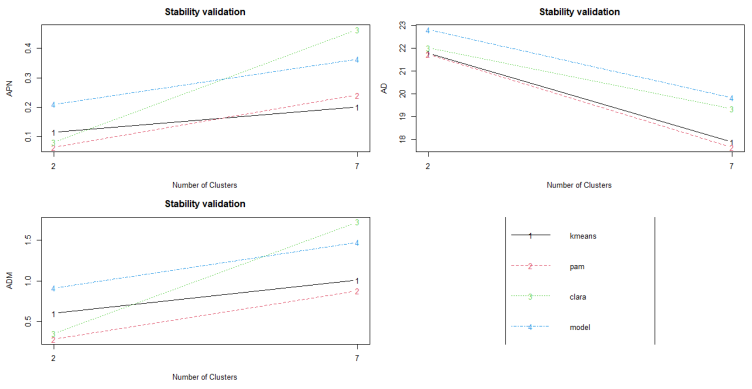
Figure 11: Model based average cluster representations

6 Discussion

Since there is not a target variable in which we want to categorize our dataset is not easy to compare between the different results obtained. We are going to fundament our criteria in the results obtained using the package `clValid` ([3]). Here we are going to focus on two different types of validation measures: internal and stability.
The internal measures include the connectivity, Silhouette Width, and Dunn Index. The conectivity as can be extracted by its name indicates the degree of connectedness of the clusters, and both the Silhouette Width and the Dunn Index combine measures of compactness and separation of the clusters. These should be minimized, maximized and maximized respectively.
The stability measures are a special version of internal measures which evaluate the stability of a clustering result by comparing it with the clusters obtained by removing one column at a time. In this case all measures should be minimized
The annex to the report contains the graphs for these values considering all algorithms mentiones in the study. Nevertheless, we will analyze the graphs without the hierarchical algorithms since these invalid the other results. This is due to the fact that as we saw, the clusters built by agglomerative and DIANA are really imbalanced and don't give much information. This disparity also causes higher compactness and stability, therefore we can't take these measures as performance evaluations.
Considering 2 clusters we can see that in general both internal and stability measures are optimized by the algorithms 1 and 2, i.e K-Means and PAM. Meanwhile when K=7 although K-Means and PAM are still the ones that perform best, it's notable that algorithm 4, EM, is also predominant in the internal measures.



(a) Internal validity



(b) Stability validity

Figure 12: Cluster algorithm comparisons

7 Conclusion

In short, although comparing different clustering results can be rather hard since there is not a response variable we were able to use some validity measurements to find the "optimal" number of clusters and algorithm for our dataset.
We chose both K-Means and PAM as the best results. To sumarize all the results, in the next four tables we can see a cluster naming suggestion that has been obtained by analyzing the radar graphs. These group names could be useful for a future recommendation system in order to look for ratings from people with similar tastes to the user.

Cluster	Name
1	Travel, adventure and sports
2	Social, food and art

(a) K-Means

Cluster	Name
1	Self-care, tourism
2	Entertainment and food

(b) PAM

Table 4: Naming suggestion for 2 clusters

Cluster	Name
1	Low-cost getaway
2	Travel critic
3	Haters
4	Influencers
5	Sporties
6	Relax
7	Foodies

(a) K-Means

Cluster	Name
1	Culture
2	Relax
3	Foodies
4	Nature lovers
5	Food critic
6	Low-cost getaway
7	Self-care

(b) PAM

Table 5: Naming suggestion for 7 clusters

References

[1] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society Series B*, 63:411–423, 02 2001.

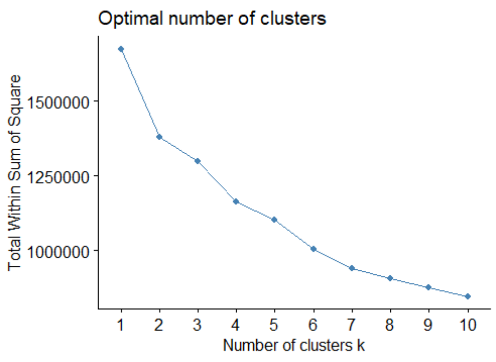
[2] Sinan Saraçli, Nurhan Doğan, and İsmet Doğan. Comparison of hierarchical cluster analysis methods by cophenetic correlation. *Journal of inequalities and Applications*, 2013(1):1–8, 2013.

[3] Guy Brock, Vasyl Pihur, Susmita Datta, and Somnath Datta. clValid: An R package for cluster validation. *Journal of Statistical Software*, 25(4):1–22, 2008.

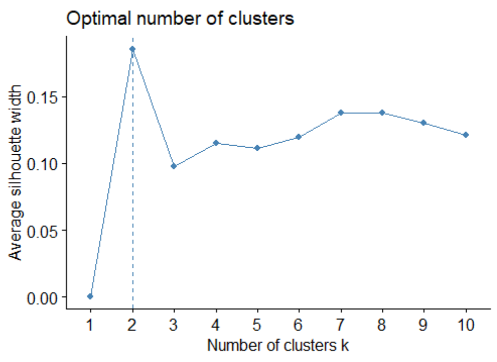
[4] Peter Langfelder, Bin Zhang, and Steve Horvath. Defining clusters from a hierarchical cluster tree: the dynamic tree cut package for r. *Bioinformatics*, 24(5):719–720, 2008.

[5] Chris Fraley and Adrian E Raftery. How many clusters? which clustering method? answers via model-based cluster analysis. *The computer journal*, 41(8):578–588, 1998.

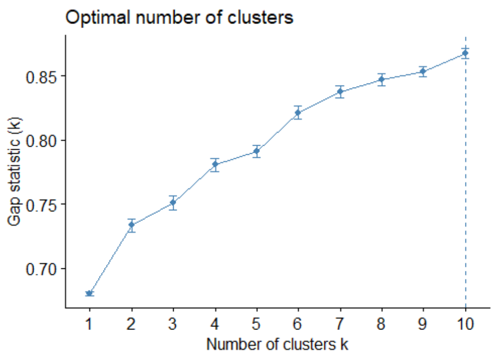
A CLARA



(a) Elbow method



(b) Silhouette method



(c) Gap statistic

Figure 13: Results from different measures to find optimal k in CLARA

B Hierarchical dendograms

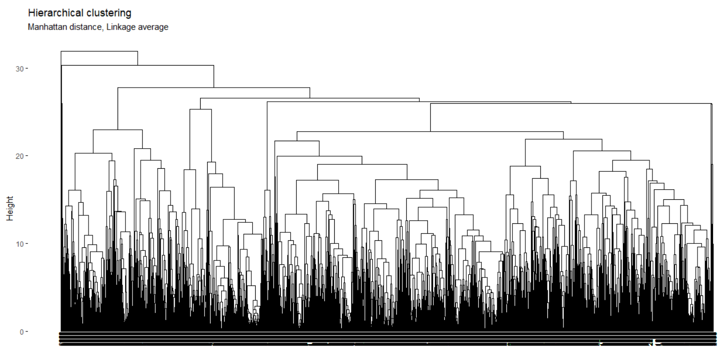


Figure 14: Agglomerative clustering dendogram

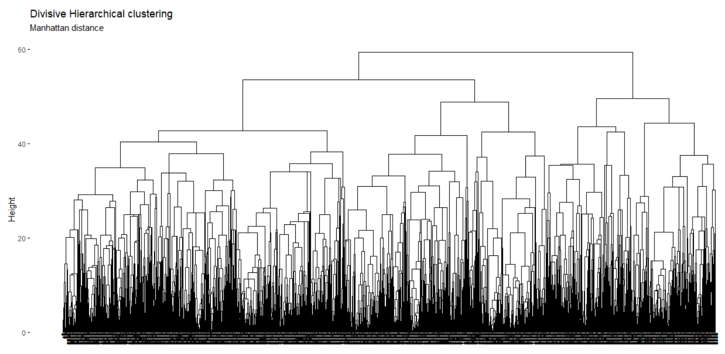


Figure 15: DIANA dendogram

C Cluster validation including hierarchical algorithms

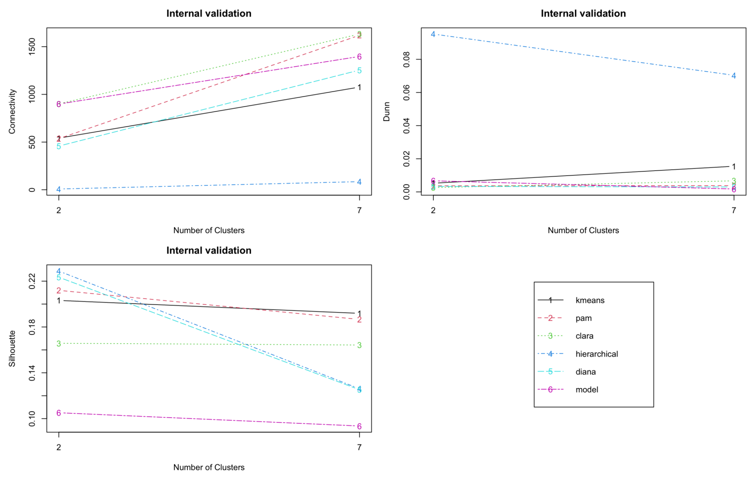


Figure 16: Internal validity

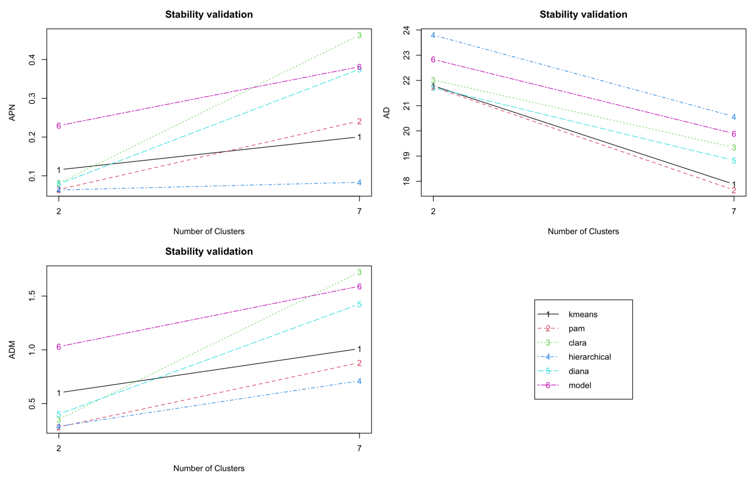


Figure 17: Stability validity

D R

```r
library(readr)
library(cluster)
library(dplyr)
library(ggplot2)
library(readr)
library(Rtsne)
library(ISLR)
library(factoextra)
library(NbClust)
library(mclust)
library(EMCluster)
library(corrplot)
library(cclust)
library(ggradar)
library(ggpubr)
library(clValid)

travel = read_csv("C:/Users/maria/OneDrive/Escritorio/Master/Machine Learning/Practical Application 3/google_review_ratings.csv",
                  col_types = c("-",rep("n",23),"-"))

#DATA PREPROCESS
#See if there are missing values
missing = sum(is.na(travel))
#Column name transformation
colnames(travel) = c('churches', 'resorts', 'beaches', 'parks', 'theatres',
                     'museums', 'malls', 'zoo', 'restaurants', 'pubs_bars',
                     'local_services', 'burger_pizza_shops', 'hotels_other_lodgings',
                     'juice_bars', 'art_galleries', 'dance_clubs', 'swimming_pools',
                     'gyms', 'bakeries', 'beauty_spas', 'cafes', 'view_points',
                     'monuments', 'gardens')
#Low correlation between variables
summary(travel)
corrplot(cor(travel), method = "circle", type = "upper")
#0.62 max correlacion

#Scale the data
datos = scale(travel)
datos =as.data.frame(datos)


boxplot(datos)

#PCA
pca = prcomp(datos)

#####PARTITIONAL CLUSTERING#####
#K-MEANS
####################ELBOW METHOD######################
fviz_nbclust(x = datos, FUNcluster = kmeans, method = "wss", k.max = 10,
             diss = dist(datos, method = "manhattan"))
####################SILHOUETTE METHOD######################
fviz_nbclust(x = datos, FUNcluster = kmeans, method = "silhouette", k.max = 10,
             diss = dist(datos, method = "manhattan"))
####################GAP STATISTIC######################
fviz_nbclust(x = datos, FUNcluster = kmeans, method = "gap_stat", k.max = 10,
             diss = dist(datos, method = "manhattan"))
#K = 2
km2_clusters <- kmeans(x = datos, centers = 2, nstart = 50)
km2_clusters
#Cluster visualization
fviz_cluster(object = km2_clusters, data = datos, show.clust.cent = TRUE,
             ellipse.type = "t", star.plot = TRUE, repel = TRUE) +
  labs(title = "Results clustering K-means, K=2") +
  theme_bw() +
  theme(legend.position = "none")

```

```r
#Radarchart
km2_centers = km2_clusters$centers
df = cbind(cluster =c(1,2), km2_centers)
ggradar(df,grid.min = -2,grid.max = 2)

#K=7
km7_clusters <- kmeans(x = datos, centers = 7, nstart = 50)
km7_clusters
#Cluster visualization
fviz_cluster(object = km7_clusters, data = datos, show.clust.cent = TRUE,
             ellipse.type = "t", star.plot = TRUE, repel = TRUE) +
  labs(title = "Results clustering K-means, K=7") +
  theme_bw() +
  theme(legend.position = "none")

#Radarchart
km7_centers = km7_clusters$centers
df = cbind(cluster =c(1,2,3,4,5,6,7), km7_centers)
ggradar(df,grid.min = -3,grid.max = 4)

#PAM
####################ELBOW METHOD######################
fviz_nbclust(x = datos, FUNcluster = pam, method = "wss", k.max = 10,
             diss = dist(datos, method = "manhattan"))
####################SILHOUETTE METHOD######################
fviz_nbclust(x = datos, FUNcluster = pam, method = "silhouette", k.max = 10,
             diss = dist(datos, method = "manhattan"))
####################GAP STATISTIC######################
fviz_nbclust(x = datos, FUNcluster = pam, method = "gap_stat", k.max = 10,
             diss = dist(datos, method = "manhattan"))

#K=2
pam2_clusters <- pam(x = datos, k = 2, nstart = 50, metric = "manhattan")
pam2_clusters
#Cluster visualization
fviz_cluster(object = pam2_clusters, data = datos, show.clust.cent = TRUE,
             ellipse.type = "t", star.plot = TRUE, repel = TRUE) +
  labs(title = "Resultados clustering PAM, K=2") +
  theme_bw() +
  theme(legend.position = "none")

#Radarchart
pam2_centers = pam2_clusters$medoids
df = cbind(cluster =c(1,2), pam2_centers)
ggradar(df,grid.min = -2,grid.max = 2)

#K=7
pam7_clusters <- pam(x = datos, k = 7, nstart = 50, metric = "manhattan")
pam7_clusters
#Cluster visualization
fviz_cluster(object = pam7_clusters, data = datos, show.clust.cent = TRUE,
             ellipse.type = "t", star.plot = TRUE, repel = TRUE) +
  labs(title = "Resultados clustering PAM, K=7") +
  theme_bw() +
  theme(legend.position = "none")

#Radarchart
pam7_centers = pam7_clusters$medoids
df = cbind(cluster =c(1,2,3,4,5,6,7), pam7_centers)
ggradar(df,grid.min = -3,grid.max = 3)

#CLARA
####################ELBOW METHOD######################
fviz_nbclust(x = datos, FUNcluster = clara, method = "wss", k.max = 10,
             diss = dist(datos, method = "manhattan"))
####################SILHOUETTE METHOD######################
fviz_nbclust(x = datos, FUNcluster = clara, method = "silhouette", k.max = 10,
             diss = dist(datos, method = "manhattan"))
####################GAP STATISTIC######################
fviz_nbclust(x = datos, FUNcluster = clara, method = "gap_stat", k.max = 10,
              diss = dist(datos, method = "manhattan"))

#K=2
clara2_clusters <- clara(x = datos, k = 2, metric = "manhattan", stand = TRUE,
                         samples = 50, pamLike = TRUE)
clara2_clusters
#Cluster visualization
fviz_cluster(object = clara2_clusters, data = datos, show.clust.cent = TRUE,
              ellipse.type = "t", star.plot = TRUE, repel = TRUE) +
  labs(title = "Resultados clustering CLARA, K=2") +
  theme_bw() +
  theme(legend.position = "none")
#Radarchart
clara2_centers = clara2_clusters$medoids
df = cbind(cluster =c(1,2), clara2_centers)
ggradar(df,grid.min = -2,grid.max = 2)

#K=7
clara7_clusters <- clara(x = datos, k = 7, metric = "manhattan", stand = TRUE,
                         samples = 50, pamLike = TRUE)
clara7_clusters
#Cluster visualization
fviz_cluster(object = clara7_clusters, data = datos, show.clust.cent = TRUE,
              ellipse.type = "t", star.plot = TRUE, repel = TRUE) +
  labs(title = "Results clustering CLARA, K=7") +
  theme_bw() +
  theme(legend.position = "none")

#Radarchart
clara7_centers = clara7_clusters$medoids
df = cbind(cluster =c(1,2,3,4,5,6,7), clara7_centers)
ggradar(df,grid.min = -4,grid.max = 4)

#HIERARCHICAL
######################AGGLOMERATIVE##############################
dist <- daisy(datos, metric = "manhattan")
mat <- as.matrix(dist)

hc_manhattan_completo <- hclust(d = dist, method = "complete")
hc_manhattan_single   <- hclust(d = dist, method = "single")
hc_manhattan_average  <- hclust(d = dist, method = "average")
hc_manhattan_centroid  <- hclust(d = dist, method = "centroid")
hc_manhattan_ward  <- hclust(d = dist, method = "ward.D")
cor = c()
cor[1] = cor(x = dist, cophenetic(hc_manhattan_completo))
cor[2] = cor(x = dist, cophenetic(hc_manhattan_single))
cor[3] = cor(x = dist, cophenetic(hc_manhattan_average))
cor[4] = cor(x = dist, cophenetic(hc_manhattan_centroid))
cor[5] = cor(x = dist, cophenetic(hc_manhattan_ward))

barplot(cor,
        names.arg = c("complete","single","average","centroid","ward"),
        col = terrain.colors(5))

dend_1 = as.dendrogram(hc_manhattan_average)

fviz_dend(x = hc_manhattan_average, cex = 0.6)+
  labs(title = "Hierarchical clustering",
       subtitle = "Manhattan distance, Linkage average")

clusters_agglo <- cutree(tree = hc_manhattan_average, k = 2)


######################DIVISIVE##############################
hc_diana <- diana(x = dist, diss = TRUE, stand = FALSE)

dend_2 = as.dendrogram(hc_diana)

fviz_dend(x = hc_diana, cex = 0.5) +
  labs(title = "Divisive Hierarchical clustering",
```

```r
       subtitle = "Manhattan distance")

clusters_diana <- cutree(tree = hc_diana, k = 2)
tab_diana = table(clusters_diana, dnn = list("clusters", "tipo de cliente"))

#Comparing dendograms
tanglegram(dend1 = dend_1, dend2 = dend_2, highlight_distinct_edges = TRUE,
           common_subtrees_color_branches = TRUE)
cor_cophenetic(dend1 = dend_1, dend2 = dend_2)

#EM
set.seed(1)
model_clustering1 <- Mclust(data = datos, G = 1:10)
summary(model_clustering1)
summary(model_clustering1$BIC)

fviz_mclust(object = model_clustering1, what = "BIC", pallete = "jco") +
  scale_x_discrete(limits = c(1:10))

model_clustering2 <- Mclust(data = datos, G = 10, modelNames = "EEV")
model_clustering3 <- Mclust(data = datos, G = 8, modelNames = "EEV")

fviz_mclust(model_clustering1, what = "uncertainty", pallete = "jco")+
  labs(title = "EM (Model VEV)",
       subtitle = "Manhattan distance,K=10, ")
fviz_mclust(model_clustering2, what = "uncertainty", pallete = "jco")+
  labs(title = "EM (Model EEV)",
       subtitle = "Manhattan distance,K=10, ")
fviz_mclust(model_clustering3, what = "uncertainty", pallete = "jco")+
  labs(title = "EM (Model EEV)",
       subtitle = "Manhattan distance,K=8, ")

#Radarchart 1
clust_model1 = list()
for (i in 1:10) {
  clust_model1[[i]] = datos[model_clustering1$classification==i,]
}

mat = matrix(0,10,24)
for (i in 1:10){
  for (j in 1:24){
    mat[i,j] = mean(clust_model1[[i]][,j])
  }
}


df = cbind(cluster =c(1,2,3,4,5,6,7,8,9,10), mat)
colnames(df) = c('cluster','churches', 'resorts', 'beaches', 'parks', 'theatres',
                 'museums', 'malls', 'zoo', 'restaurants', 'pubs_bars',
                 'local_services', 'burger_pizza_shops', 'hotels_other_lodgings',
                 'juice_bars', 'art_galleries', 'dance_clubs', 'swimming_pools',
                 'gyms', 'bakeries', 'beauty_spas', 'cafes', 'view_points',
                 'monuments', 'gardens')
ggradar(df,grid.min = -3,grid.max = 3)

#Radarchart 2
clust_model2 = list()
for (i in 1:10) {
  clust_model2[[i]] = datos[model_clustering2$classification==i,]
}

mat2 = matrix(0,10,24)
for (i in 1:10){
  for (j in 1:24){
    mat2[i,j] = mean(clust_model2[[i]][,j])
  }
}


df2 = cbind(cluster =c(1,2,3,4,5,6,7,8,9,10), mat2)
```

```
colnames(df2) = c('cluster','churches', 'resorts', 'beaches', 'parks', 'theatres',
                  'museums', 'malls', 'zoo', 'restaurants', 'pubs_bars',
                  'local_services', 'burger_pizza_shops', 'hotels_other_lodgings',
                  'juice_bars', 'art_galleries', 'dance_clubs', 'swimming_pools',
                  'gyms', 'bakeries', 'beauty_spas', 'cafes', 'view_points',
                  'monuments', 'gardens')
ggradar(df2,grid.min = -3,grid.max = 3)

#Radarchart 3
clust_model3 = list()
for (i in 1:8) {
  clust_model3[[i]] = datos[model_clustering3$classification==i,]
}

mat3 = matrix(0,8,24)
for (i in 1:8){
  for (j in 1:24){
    mat3[i,j] = mean(clust_model3[[i]][,j])
  }
}


df3 = cbind(cluster =c(1,2,3,4,5,6,7,8), mat3)
colnames(df3) = c('cluster','churches', 'resorts', 'beaches', 'parks', 'theatres',
                  'museums', 'malls', 'zoo', 'restaurants', 'pubs_bars',
                  'local_services', 'burger_pizza_shops', 'hotels_other_lodgings',
                  'juice_bars', 'art_galleries', 'dance_clubs', 'swimming_pools',
                  'gyms', 'bakeries', 'beauty_spas', 'cafes', 'view_points',
                  'monuments', 'gardens')
ggradar(df3,grid.min = -3,grid.max = 3)

#VALIDATION
intern = clValid(datos, nClust = c(2,7),
                   clMethods = c("kmeans", "pam", "clara","model"),
                   validation = "internal", metric = "manhattan",
                   method = "average", maxitems = nrow(datos))
op <- par(no.readonly=TRUE)
par(mfrow=c(2,2),mar=c(4,4,3,1))
plot(intern, legend=FALSE)
plot(NULL)
legend("center", clusterMethods(intern), col=1:9, lty=1:9, pch=paste(1:9))
par(op)

stab = clValid(datos, nClust = c(2,7),
                  clMethods = c("kmeans", "pam", "clara","model"),
                  validation = "stability", metric = "manhattan",
                  method = "average", maxitems = nrow(datos))
par(mfrow=c(2,2),mar=c(4,4,3,1))
plot(stab, measure=c("APN","AD","ADM"),legend=FALSE)
plot(NULL)
legend("center", clusterMethods(stab), col=1:9, lty=1:9, pch=paste(1:9))
par(op)
```