Ecole Nationale Superieure d'Informatique et Analyse de Systemes

Unsupervised learning Project

Breast Cancer clustering using Chameleon method

Submitted by:
Ait Hammadi Abdellatif

Supervisor:
Pr. M. LAZAAR

Semester IV: 2020 / 2021

Contents

List of Figures

List of Tables

Introduction

Clustering means dividing the data into groups (known as clusters) in such a way that objects belonging to a group are similar to each other but they are dissimilar to objects belonging to other groups. It can be used in market research, business, land use, biology, atmospheric research, astrology, web based application plant observation and load analysis in power system [1]. In data mining, It is useful technique for discovering interesting data distributions and pattern in the underlying data.

Chameleon is a hierarchical agglomerative clustering algorithm that use dynamic modelling to determine the similarity between the peers of clusters. In Chameleon, cluster similarity is assessed based on how well-connected objects are within cluster and on the proximity clusters. That is, Two clusters are merged if there interconnectivity is high and they are close together [2].

Generally hierarchical clustering methods are conceptually simple, theoretical properties are well understood and when clusters are merged/split, the decision is permanent, in other word the numbers of different alternatives that need to be examined is reduced. As well as, this methods have some limitations in terms of inability to scale well, divisive methods can be computational hard and there is no back-tracking capacity. In particular, Chameleon cannot applied to high dimensions [3].

We applied our Chameleon clustering method to the Wisconsin breast cancer dataset. To increase the quality of the clustering performance, we apply the autoencoder as a prepossessing process to reduce feature and remove the noise.

1 Chameleon Method

The Chameleon algorithms for clustering work in 3 phases :

(a) Getting sparse graph for the data using K-Nearest neighbor algorithm.

(b) Partition the graph using multi graph partitioning algorithms to get smaller clusters depending upon their similarity.

(c) In the third phase the algorithm merge the smaller clusters to remove the noisy data and make the cluster with significant size.

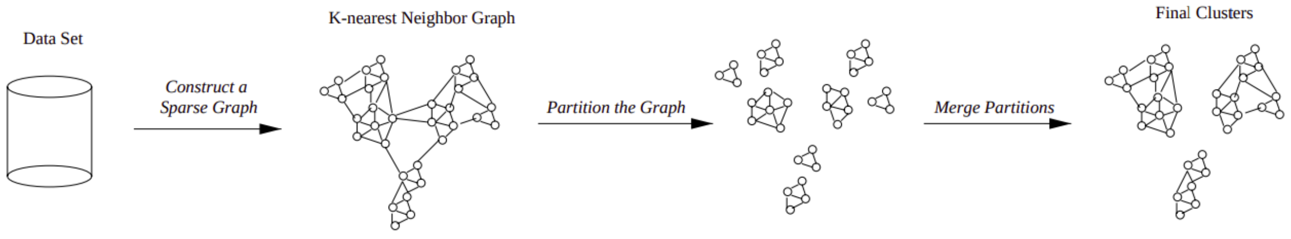


Figure 1: Overall framework CHAMLEON

K-nearest neighbor is a supervised algorithm used to find similarity between data. It assigns a class-label to data values and then finds the one that belong to the same class, and connect them with an edge.

The third phase of merging of the clusters is done on basis of their similarity to get final clusters. Similarity is measured by two parameters

- **Inter-connectivity** refers to the connection between two nodes
- **Closeness** is defined as the similarity between two nodes in two different clusters

The relative Inter-connectivity is the connection between two nodes of two different clusters. For two clusters the relative inter-connectivity and relative closeness are calculated using the formulas as follows for the two clusters C_i and C_j

$$RI(C_i, C_j) = \frac{|EC(C_i, C_j)|}{\frac{|EC_{Ci}|+|EC_{Cj}|}{2}}$$

where,

- $|EC(C_i, C_j)|$ is the sum of weighted of edges that connect C_i and C_j.
- $|EC_{Ci}|$ is the weighted sum of edges partition the cluster into roughly equal parts.

While, internal closeness of the cluster is the average weight of the edge in the particular cluster.
Relative closeness of two clusters is defined as :

$$RC(C_i, C_j) = \frac{\bar{S}_{EC(C_i,C_j)}}{\frac{|C_i|}{|C_i|+|C_j|}\bar{S}_E C_{Ci} + \frac{|C_j|}{|C_i|+|C_j|}\bar{S}_E C_{Cj}}$$

where,

- $\bar{S}_{EC(C_i,C_j)}$ is the average weight of edge that belong to a cut-set of the cluster C_i and C_j.
- $\bar{S}_E C_{Ci}$ is the average weight of edge that belong to a minimum cut of cluster C_i
- $|C_i|$ is the number of node in the cluster C_i

Two clusters are merged if they have high value for the product of relative inter-connectivity and relative closeness. Sometimes, a user can assign a higher priority to a particular parameter using α.

$$RI(C_i, C_j)^\alpha * RC(C_i, C_j)$$

For the two clusters if the above function holds value superior than or equal to the threshold value for similarity then clusters are merged.
if $\alpha > 1$ then more priority given to relative inter-connectivity else if $\alpha < 1$ relative closeness gets more priority. Essentially, in case $\alpha = 1$ both of them have the same priority.

Algorithm 1: Pseudo code for merging algorithms for final clusters

```
RI relative inter-connectivity;
RC relative closness;
$\alpha$ user defined parameter
$\beta$ RI x RC;
th threshold value to take merging decision;
n be the number of the clusters to be merged
for i = 0...n do
    for j = i+1...n do
        Calculate RI for $C_i$ and $C_j$
        Calculate RC for $C_i$ and $C_j$
        Calculate $\beta = RI^\alpha \times RC$
        if $\beta >= th$ then
            merge the two clusters $C_i$ and $C_j$
        else
        end
    end
end
```

2 Dataset Exploration

2.1 Introducing the data

The ”Diagnostic Wisconsin Breast Cancer Database” is a publicly available data set from the UCI machine learning repository [4]. The dataset gives information about tumor features, that are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. For each observation there are 10 features, which describe tumor size, density, texture, symmetry, and other characteristics of the cell nuclei present in the image. The mean, standard error and ”worst” mean (mean of the three largest values) of these features were computed for each image, resulting in 30 features. The categorical target feature indicates the type of the tumor.

The area on the aspirate slides to be analyzed was visually selected for minimal nuclear overlap. The image for digital analysis was generated by a JVC TK-1070U color video camera mounted above an Olympus microscope and the image was projected into the camera with a 63 x objective and a 2.5 x ocular. The image was captured as a 512 x 480 resolution, 8 bit/pixel (Black and White) file. The aspirated material was expressed onto a silane-coated glass slide, which was placed under a similar slide. A typical image contains approximately from 10 to 40 nuclei. After computing 10 features for each nucleus, the mean, standard error and extreme value was computed, as it mentioned above. These features are modeled such that higher values are typically associated with malignancy.

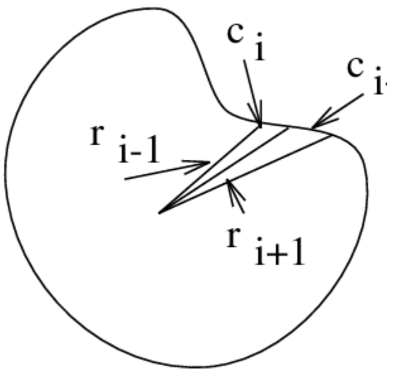
This data Set is a table of shape 569 x 33 (rows x columns) which have no null values. The values take different scales and there are ten real-valued features computed for each cell nucleus :

- **Radius** (mean of distances from center to points on the perimeter)
- **Texture** standard deviation of gray-scale values
- **Perimeter**
- **Area**
- **Smoothness** local variation in radius lengths
- **Compactness** ($\frac{perimeter^2}{area} - 1.0$)
- **Concavity** severity of concave portions of the contour
- **Concave points** number of concave portions of the contour
- **Symmetry**
- **Fractal dimension** [1](”coastline approximation” - 1)

Texture is a standard deviation of gray-scale values. Each pixel of an image is represented by the 8-bit integer, or a byte, from 0 to 255 providing the amount of light, where 0 is clear black and 255 is clear white. The darker the image is the lower is the mean of intensity level of a pixel, i.e. byte. So, the SD of gray-scale values means how intense levels are spread for particular individual cells. The higher SD the more contrasting the image is.

the smoothness is quantified by measuring the difference between length of radial line and the mean length of two radial lines surrounding it.

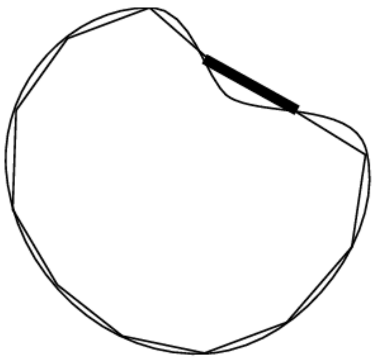
[1]a **fractal dimension** is a ratio providing a statistical index of complexity comparing how detail in a pattern (strictly speaking, a fractal pattern) changes with the scale at which it is measured.



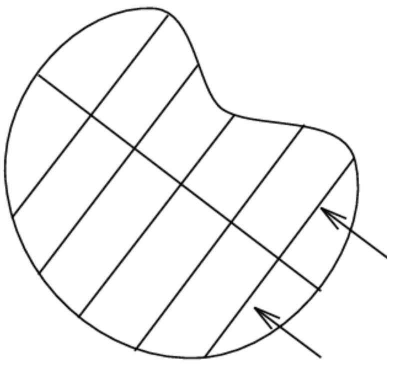
If the number is small, then the contour is smooth in that region:

$$smoothness = \frac{\sum_i \left| r_i - \frac{r_{i-1}+r_{i+1}}{2} \right|}{perimeter}$$

The concavity is captured by drawing chords between two boundary points, which lie outside the nuclear. For the concavity_mean the mean value of these lengths is calculated.



In order to measure symmetry, the major axis, or longest chord through the center, is found. We then measure the length difference between lines perpendicular to the major axis and the nuclear boundary in both directions.



	count	mean	std	min	25%	50%	75%	max
radius_mean	569.0	14.1	3.5	6.9	11.7	13.3	15.7	28.1
texture_mean	569.0	19.2	4.3	9.7	16.1	18.8	21.8	39.2
perimeter_mean	569.0	91.9	24.2	43.7	75.1	86.2	104.1	188.5
area_mean	569.0	654.8	351.9	143.5	420.3	551.1	782.7	2501.0
smoothness_mean	569.0	0.0	0.0	0.0	0.0	0.0	0.1	0.1
compactness_mean	569.0	0.1	0.0	0.0	0.0	0.0	0.1	0.3
concavity_mean	569.0	0.0	0.0	0.0	0.0	0.0	0.1	0.4
concave points_mean	569.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2
symmetry_mean	569.0	0.1	0.0	0.1	0.1	0.1	0.1	0.3
fractal_dimension_mean	569.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
radius_se	569.0	0.4	0.2	0.1	0.2	0.3	0.4	2.8
texture_se	569.0	1.2	0.5	0.3	0.8	1.1	1.4	4.8
perimeter_se	569.0	2.8	2.0	0.7	1.6	2.2	3.3	21.9
area_se	569.0	40.3	45.4	6.8	17.8	24.5	45.1	542.2
smoothness_se	569.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
compactness_se	569.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1
concavity_se	569.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3
concave points_se	569.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
symmetry_se	569.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
fractal_dimension_se	569.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
radius_worst	569.0	16.2	4.8	7.9	13.0	14.9	18.7	36.0
texture_worst	569.0	25.6	6.1	12.0	21.0	25.4	29.7	49.5
perimeter_worst	569.0	107.2	33.6	50.4	84.1	97.6	125.4	251.2
area_worst	569.0	880.5	569.3	185.2	515.3	686.5	1084.0	4254.0
smoothness_worst	569.0	0.1	0.0	0.0	0.1	0.1	0.1	0.2
compactness_worst	569.0	0.2	0.1	0.0	0.1	0.2	0.3	1.0
concavity_worst	569.0	0.2	0.2	0.0	0.1	0.2	0.3	1.2
concave points_worst	569.0	0.1	0.0	0.0	0.0	0.0	0.1	0.2
symmetry_worst	569.0	0.2	0.0	0.1	0.2	0.2	0.3	0.6
fractal_dimension_worst	569.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2

Table 1: The general statistics. including mean, std, median, percentiles, and range

2.2 Visualisation

2.2.1 Data Distributions

We can see in 2 that some features are pretty skewed. We can measure its skewness using pandas skew method and we can try comparing it to a log transformation of the same values to see if we can reduce the skewness.

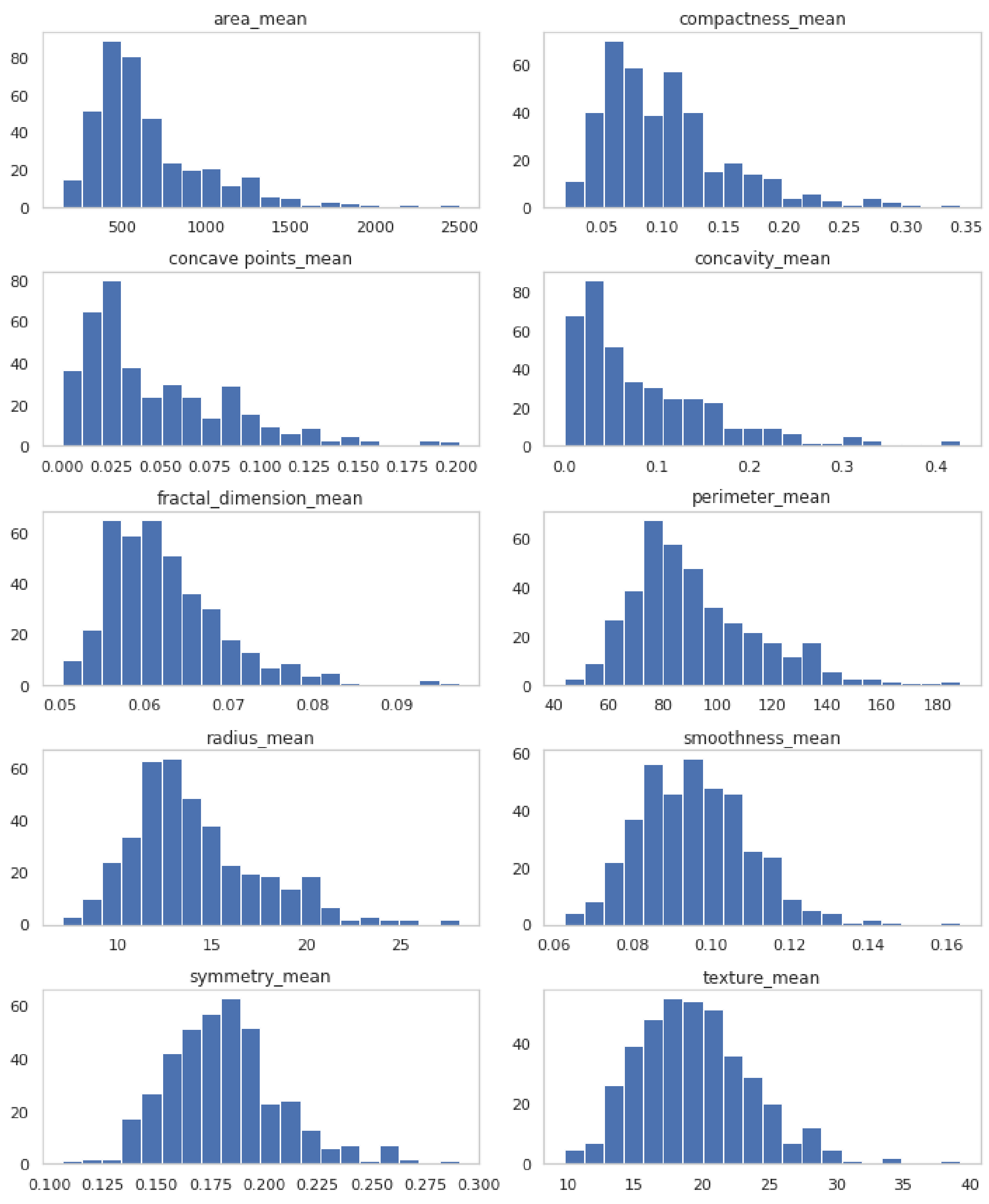


Figure 2

There are four features with skewness higher than one after the log transformation: Compactness, Concavity, Concave Points and Fractal Dimension. Perhaps the measure error on them is higher or maybe it is somehow biased. Let's explore how are the standard errors for each measure.

	Original Skewness	Log Transformed	Skewness Reduction
radius_mean	0.957984	0.341400	0.616583
texture_mean	0.646057	-0.026770	0.672828
perimeter_mean	1.003896	0.315615	0.688281
area_mean	1.637987	0.268267	1.369719
smoothness_mean	0.603007	0.550194	0.052813
compactness_mean	1.314372	1.151288	0.163084
concavity_mean	1.481724	1.274931	0.206793
concave points_mean	1.223607	1.129755	0.093852
symmetry_mean	0.688502	0.602852	0.085650
fractal_dimension_mean	1.315506	1.284973	0.030532

Table 2: Original Skewness, Log Transformed, Log Transformed

	Mean	Error	Error pct
radius	14.126503	0.406775	2.879519
texture	19.438241	1.217377	6.262793
perimeter	91.904422	2.867628	3.120229
area	655.325377	40.296108	6.149023
smoothness	0.095857	0.006924	7.223479
compactness	0.102648	0.025142	24.493824
concavity	0.088497	0.032268	36.462107
concave points	0.048535	0.011583	23.864573
symmetry	0.181364	0.020421	11.259927
fractal_dimension	0.062620	0.003761	6.006799

This might explain part of our high skewness: from our four highly skewed features, three of them have standard errors of more than 20%. Many things can cause that (e.g. uncallibrated measuring instruments).

2.2.2 Correlations

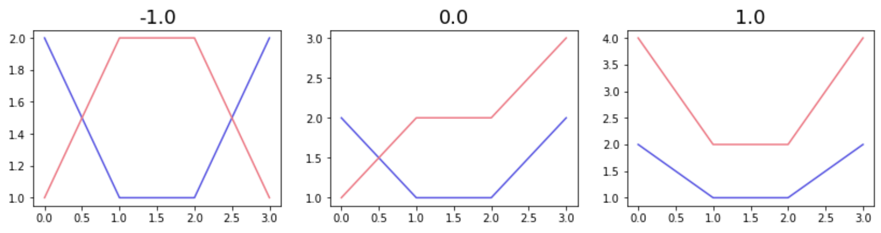


Figure 3: Negative correlation, No correlation, and Positive correlation

Linear correlation is helpful in gaining quick intuition about the relation between two signals. However, it has some drawbacks. For instance, it won't account for sequential displacements (we would need to use lags) or non-linearities.

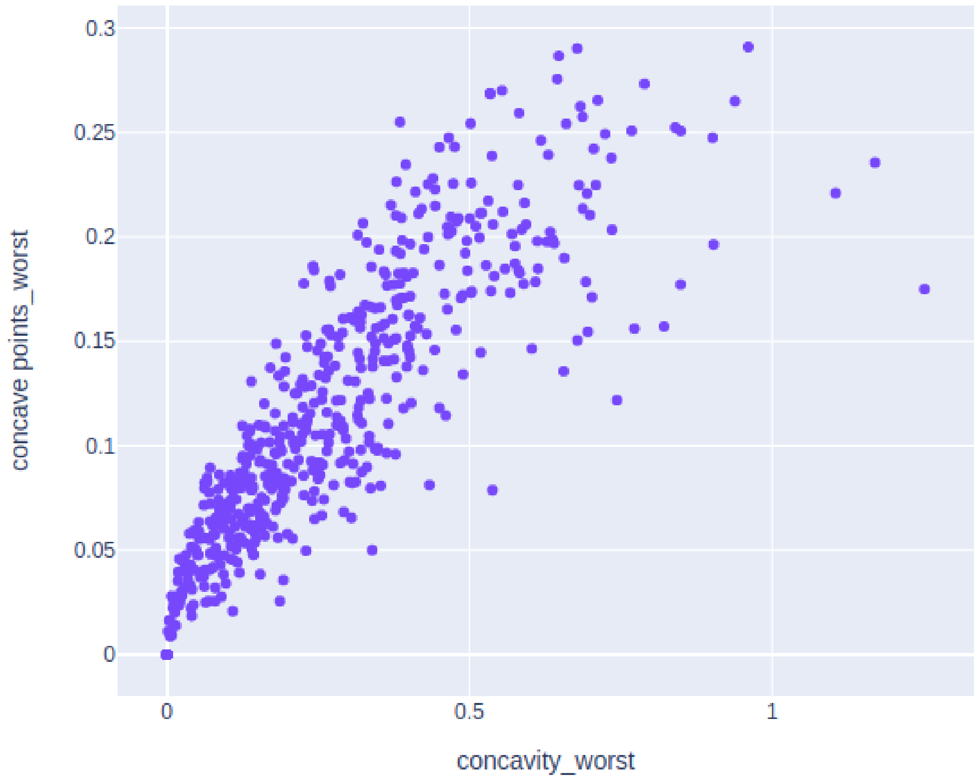


Figure 4: Check Correlation

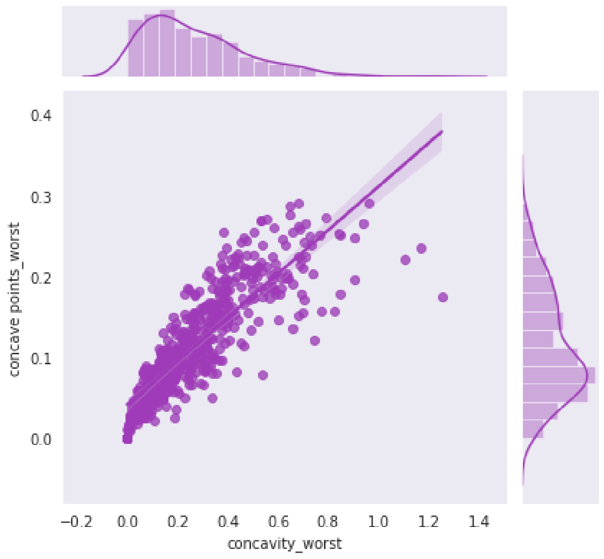
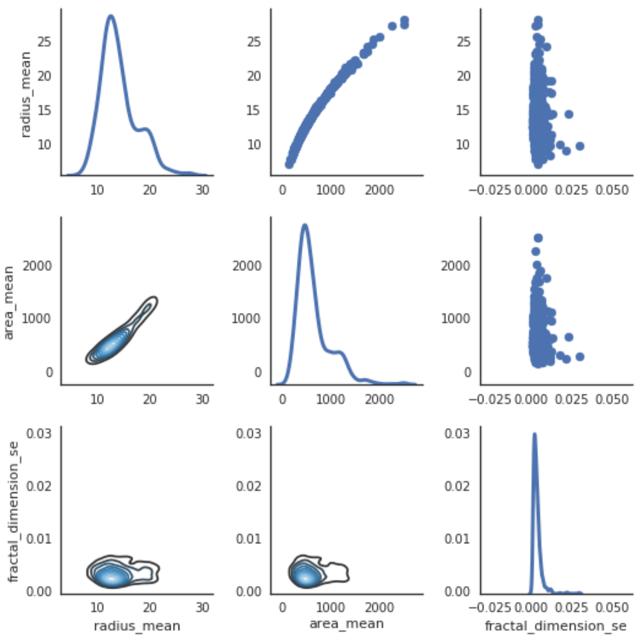


Figure 5: Joinplot for check correlation



The above graphs 4 5 show that concave worst and concave points worst are correlated.

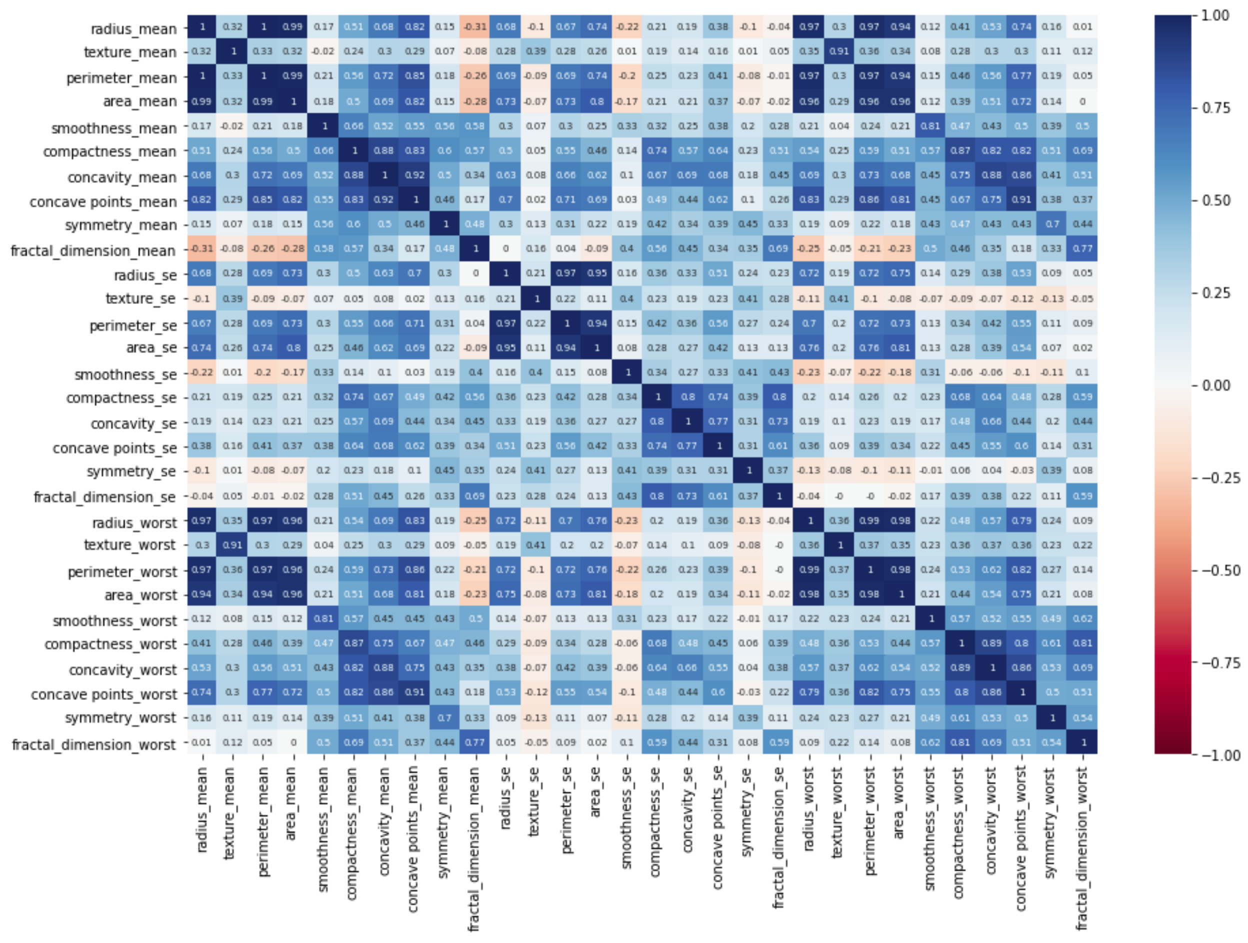


Figure 6: the heatmap with all the correlations between features

We see many positive correlations (blue). However, this heatmap is messy. For visualization purposes, it would be better to group features that are highly correlated together. To do so, we will do hierarchical clustering.

2.3 Preprocessing : Autoencoder Neural Network

Autoencoder is an unique Neural Network that learns how to represent the data with less features. Put in simple words, the autoencoder:

- Reads the X_train data (without the target)
- Funnels the data through a hidden neuron layer with less neurons than features
- Expands the data again to the same number of neurons as the first input layer (i.e. n_neurons = n_features)

We train the NN to replicate the input data and constrain it by having fewer neurons in the middle layers. That way, the network has to 'choose' which information to 'hold' in order to best represent the essence of the data. The image illustrates an autoencoder:

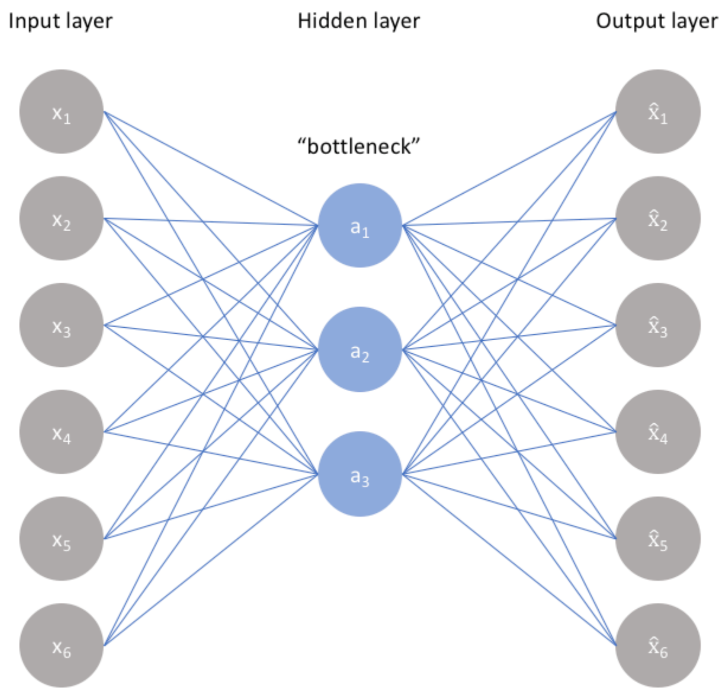


Figure 7: Autoencoder

Basically, the Autoencoder contains encoder and decoder as components. We train the model by both of the components, but at the end of the training we use the trained encoder to generate the new reduced data set.

3 Experimental Results

3.1 Preprocessing : Autoencoder

The model architecture is made on a Tensorflow platform to implement the encoder and decoder sequentials, and train the model for a different number of hidden neurons 1,3,5,10 and 13. At the end, we save the new generated data set in the form of a csv file in order to apply the chameleon algorithm.

According to the experimentation 8 9 10 11 12, the closer number of the hidden neurons get to the actual number of features, the smaller value of the error. however with less hidden neurons give a good result proven by the fact that the model can reproduce the original data from less features.

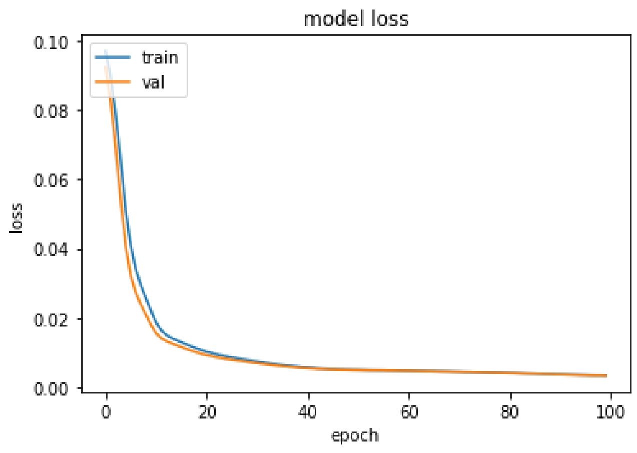


Figure 8: Autoencoder training loss with 13 hidden neurons

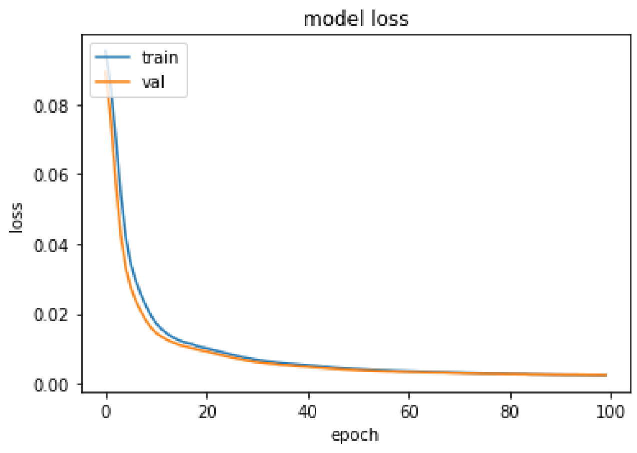


Figure 9: Autoencoder training loss with 10 hidden neurons

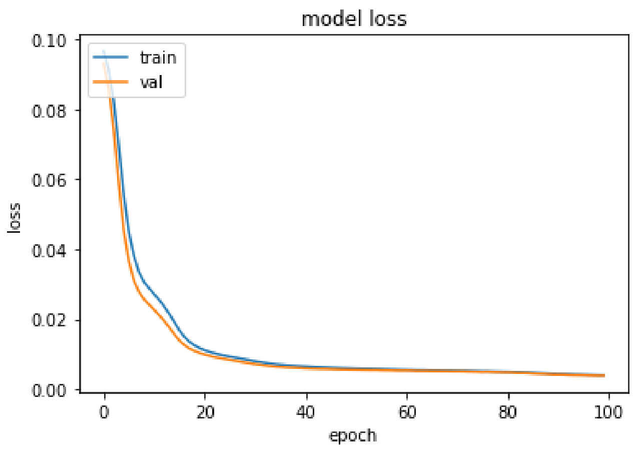


Figure 10: Autoencoder training loss with 5 hidden neurons

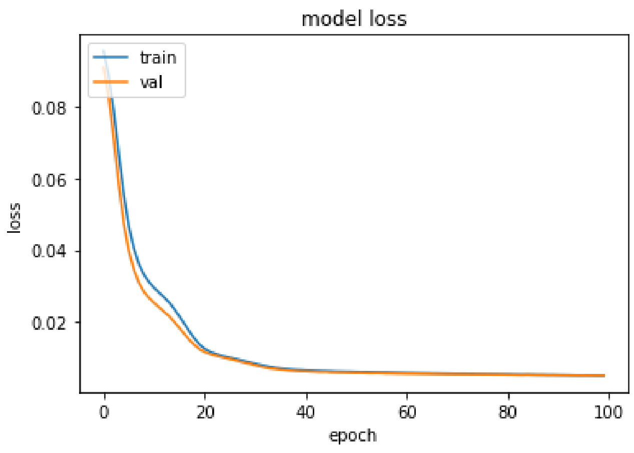


Figure 11: Autoencoder training loss with 3 hidden neurons

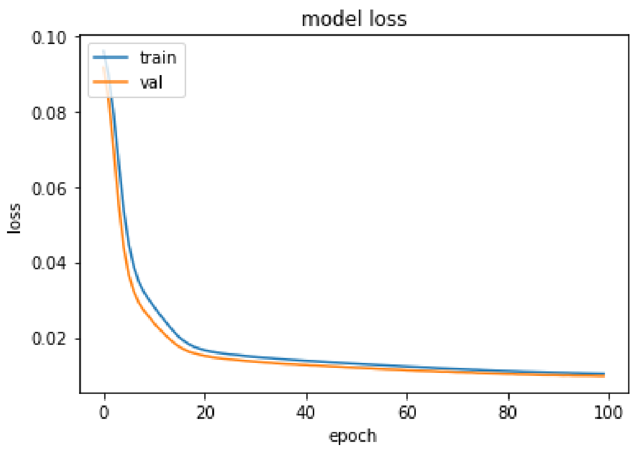


Figure 12: Autoencoder training loss with one hidden neuron

All of the training models performed have proven that the architecture is very efficient in getting fewer errors that represent the original dataset. For example, they all have significant learning even though the difference is very small in the last value error.

3.2 Chameleon

We use in our experimentation chameleon cluster function implementated by clustviz library on the schema where we aggregate the two criteria in one function. This function take as parametres:

df	input dataframe
k	desired number of clusters (the algorithm is susceptible for early stopping if the best score is 0)
knn	parameter k of K-nearest_neighbors
m	number of clusters to reach in the initial clustering phase
alpha	exponent of relative closeness; the larger, the more important relative closeness is than relative interconnectivity
verbose0	if True, print general infos
verbose1	if True, print infos about the prepartitioning phase
verbose2	if True, print labels of merging clusters and their scores in the merging phase
plot	if True, show plots
return	dataframe with cluster labels and dictionary of merging scores (similarities)

3.3 Evaluation algorithms

We use two evaluation algorithm implementated in sklearn library :

3.3.1 Davies bouldin score

The score is defined as the average similarity measure of each cluster with its most similar cluster, where similarity is the ratio of within-cluster distances to between-cluster distances. Thus, clusters which are farther apart and less dispersed will result in a better score. The minimum score is zero, with lower values indicating better clustering.[5]

3.3.2 Silhouette score

The Silhouette Coefficient is calculated using the mean intra-cluster distance (a) and the mean nearest-cluster distance (b) for each sample. The Silhouette Coefficient for a sample is $\frac{b-a}{max(a,b)}$. To clarify, b is the distance between a sample and the nearest cluster that the sample is not a part of. Note that Silhouette Coefficient is only defined if number of labels is $2 \leq n\_labels \leq n\_samples - 1$. [6]

3.4 Investigation

To investigate the best parameters for our problem we evaluate the algorithm for different values of k of K-nearest_neighbors 13. We can conclude that the value of k that give the high accuracy is 2. Concerning the alpha parameter doesn't affect in the score. .

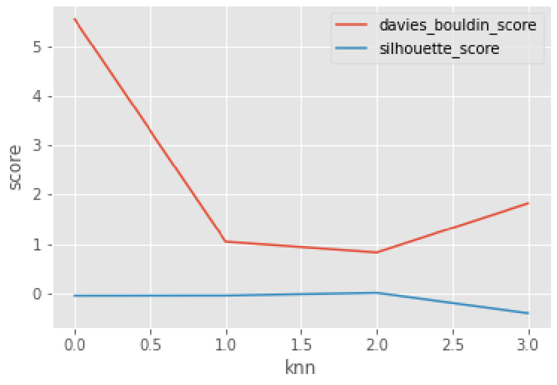


Figure 13: Evaluate many value of k for K-nearest_neighbors

we evaluate the clustering on different hyperparameterization autoencoders and we compute the autoecoder vall error to evaluate the quality of the reduction of the characteristics and the value of the Davies Bouldin and Silhouette scores to evaluate the quality of the clustering.

	AutoEncoder	AutoEncoder val loss	Features	Davies bouldin score	Silhouette score
1	-	-	30	1.04	-0.047
2	x	0.0034	13	0.500	0.564
3	x	0.0025	10	0.500	0.320
4	x	0.0039	5	0.500	0.589
5	x	0.0051	3	9.188	0.227
6	x	0.0098	1	0.500	0.558

Table 3: Parameters investigation and models evaluation

As we can see from the table, overall, the closer the chosen hidden neuron number was to the original number, the feature reduction has good quality, with example models 2 and 3 having 0.0034 and 0.0025 respectively as the minimum value error compared to the other scenario.

According to the clustering score, Davies Bouldin did not discriminate between models 2, 3, 4 and 6 because they have the save score of 0.5 as the best score, otherwise Silhouette was able to distinguish and give a good evaluation in providing different score, the best of them according to the last rater is model 4 with a score of 0.589.

We can conclude that the best model among our scenario is model 4 take as in the 5 hidden neurons autoencoder, in word order there are only 5 components in the reduced data set.

Conclusion

Hierarchical clustering can be helpful in understanding our data better. It also improves the visual representation of correlation heatmaps, making it easier to find groups of correlated features. As future insight, to reduce the complexity of time of the chameleon algorithm, there is a tool to perform a dynamic modeling of parallel hierarchical clustering. The algorithm will be thread independent, so it will be viable to run through Platform. The algorithm will be done in such a way that works on both parallel and distributed systems.

Bibliography

[1] R. Dashora, H. Bajaj, A. Dube, and G. A, "Parallel algorithm for the chameleon clustering algorithm using dynamic modeling," International Journal of Computer Applications, vol. 79, pp. 11–17, Oct. 2013. doi: 10.5120/13760-1600.

[2] G. Karypis, E.-H. Han, and V. Kumar, "Chameleon a hierarchical clustering algorithm using dynamic modeling," Computer, vol. 32, pp. 68 –75, Sep. 1999. doi: 10.1109/2.781637.

[3] R. Chauhan, Clustering techniques: A comprehensive study of various clustering techniques. [Online]. Available: http://ijarcs.info/index.php/Ijarcs/article/viewFile/2179/2167.

[4] O. L. M. Dr. William H. Wolberg W. Nick Street, Breast cancer wisconsin (diagnostic) data set. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29.

[5] D. L. Davies and D. W. Bouldin, "A cluster separation measure," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-1, no. 2, pp. 224–227, 1979. doi: 10.1109/TPAMI.1979.4766909.

[6] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," Journal of Computational and Applied Mathematics, vol. 20, pp. 53–65, 1987, issn: 0377-0427. doi: https://doi.org/10.1016/0377-0427(87)90125-7. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0377042787901257.