

Applied Machine Learning Report on Practice Assignment

WILSON COLIN and YAQIAO DENG

Winter Semester 2016

I. Dataset

We chose our dataset not according to its ability to be easily differentiable for our expert eyes, but according to what the plot looks like after they have been added on the simulation. Indeed, we added different types of object, and we decided empirically if the dataset was good or not.

As we can see, for humans it is easy to cluster each data due to their obvious differences. According to the plot, we can see that it's possible to separate the data in 3 clusters, but the borders are blurry and they might be some outliers.

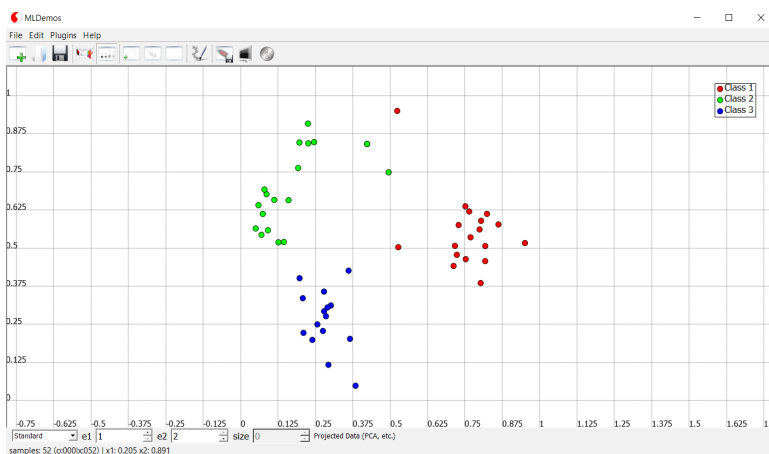


Figure I.1 : Our dataset



II. PCA

The PCA method allows users to decrease the computational cost of high-dimensionnal dataset in order to stock data (Image Compression) or to decrease the time necessary to extract features with algorithms. But PCA can also project data in order to make them more separable and improve the efficiency of algorithms used to extract features.

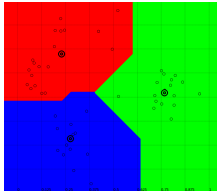
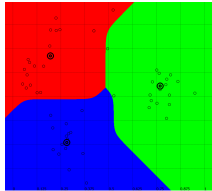
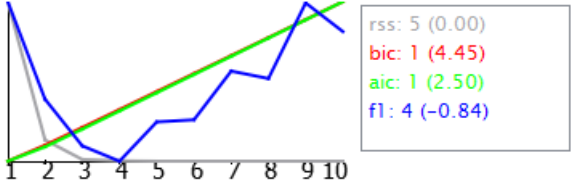
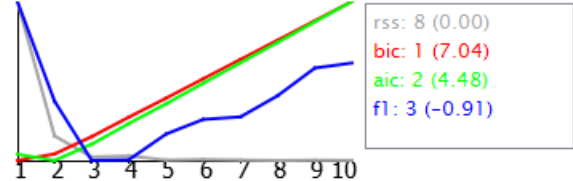
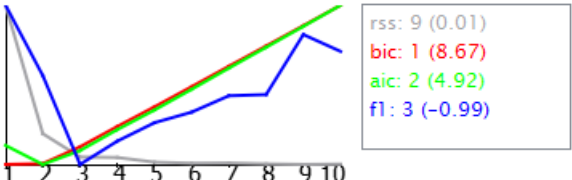
Accoding to our dataset, if we project on the first 7 eigenvectors we can represent 70,1 % of our dataset (cf Figure I.2 in annex). So this projection could be interessting for extracting features, as we keep enough amount of data and we decrease significantly the number of dimensions.

As it has been said, PCA is also used to make data more separable. Indeed, depending on the eigenvectors chosen, we will be able to separate data according to different criteria. Each eigenvectors will highlight a specific criteria (cf Figure I.3 in annex) which shows projection of all our images onto the corresponding eigenvectors. If for example we choose the eigenvectors 1 – 17 we have the red class easily separable from other. That means the eigenvectors 1 and 17 highlight a parameter which is different between the classes (for instance the background between the red and the two others classes). If we study the eigenvectors 3 - 5 we have a really bad projection. And if we study the eigenvectors 1 - 2 we have a dataset well separated even if there are some outliers.

Thus, in the rest of the project we will use a projection onto the eigenvectors 1 – 2 in order to have a good clustering, classification and so on.

III. Clustering

In order to visualize the impact of hyperparameters depending on the method used, results are shown in the following table.

Hyperparameters	K-means	Soft K-means
Type of metrics	<p>L1 metrics produces a stiff boundary while L2 metrics produces a smoother boundary.</p> <div style="display: flex; justify-content: space-around; align-items: center;"> <div style="text-align: center;">  <p>Figure III.1 L1-norm</p> </div> <div style="text-align: center;">  <p>Figure III.2 L2-norm</p> </div> </div>	
Initialization	<p>Depending on the initial location of the centroid of the cluster, they will not converge to the same position. The worst initialization is when centroids are equidistant to a local axis of symmetry of the dataset, or when they are initialized very close to each other. If we choose the PCA projection on the eigenvectors 1 - 2, the final locations of the centroids is approximatively almost the same. But if we take a bad PCA projection, for example on the eigenvectors 1 - 17, the initialization becomes bad.</p>	
Number of Clusters	<p>When we do a PCA projection we focus on some characteristics which differentiate classes. If classes are easily separable like in the projection on the eigenvectors 1-2, that means there is a criteria which differentiate the 3 classes, and by using a clustering algorithm, we can separate the data according to this criteria. Then by changing the number of cluster we will focus on the variation of this criteria. The more cluster, the more we will take into account little variations.</p> <p>The optimal number of clusters used for K-means and Soft K-means can be found by using the F1-measure. But in order to use correctly the f-measure, we need to set the training/testing ratio. We can see on the figures below that depending on this ratio, we do not have the same optimal number of cluster. In fact, this ratio imposed the number of datapoint used to do the clustering. We will see on the part IV, what does it mean to have a low ratio or a high one and its impact on the confidence that one can give to our model.</p> <p>Here, according to our knowledge of the dataset and the f-measure with a training/testing ratio of at least 55% , the optimal number of cluster appears to be three.</p> <div style="display: flex; justify-content: space-around; align-items: flex-end;"> <div style="text-align: center;">  <p>Figure III.3 F1-measure with a Train ratio of 10%</p> </div> <div style="text-align: center;">  <p>Figure III.4 F1-measure with a Train ratio of 55%</p> </div> </div> <div style="text-align: center; margin-top: 20px;">  <p>Figure III.5 F1-measure with a train ratio of 100%</p> </div>	

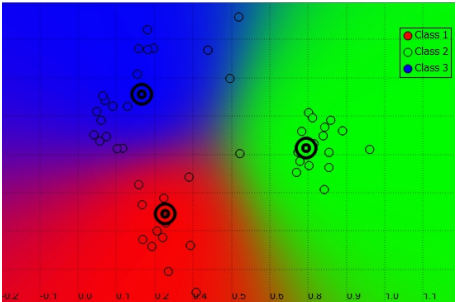
Bêta	/	<p>If we take a huge β, the radius of the inclusion circle will be small. Thus, the separation will be sharp.</p> <p>If β is small, the radius will be huge, and each cluster will take into account too many data. Then clusters will converge to the same location.</p> <p>The value of β allow us to modify the probability of each points to be in one cluster or another. So we choose β according to what we want. A huge β will sharply separate cluster, but a simple K-mean can do it. Or a small β in order to have a blurry boundary. Thus, we can see if the membership of a point to a cluster or another is well defined or not.</p> <p>For example, the plot below has been set with a small β. We can see that the membership of some points in a class or another is blurry.</p>	
------	---	--	---

Figure III.6 Clustering with $\beta = 10$

The problem with K-means and Soft K-means is that we do not consider some isolated points as outliers. That means if we have few isolated points very far from the other data, the clusters will try to fit as much as possible the dataset and the clustering will probably be less representative. So we could prefer to use the DBSCAN algorithm.

DBSCAN has two hyperparameters which are β (as Soft K-means) and the minimum number of points to consider a group of points as a cluster. Thus, as shown in the Figure III.4 in annex, we can have very localized clusters, less sensitive to isolated points which are not representative of a class in our dataset. With this algorithm, we do not know the number of clusters, we just set parameters and see how many clusters are plotted.

In conclusion, if we want to clearly separate the data without take into account the probability for a point to be in a cluster or another, we will privilege a simple K-mean. *A contrario* if we want to consider the probability of a point to belong to a cluster or another, we will prefer Soft K-means. But, if our dataset is composed by some isolated points which are far from the other points, we will opt for DBSCAN. Moreover, it is important to consider the shape of the clusters. DBSCAN is better to fit globular shape unlike K-means. According to our dataset, we have some outliers but they are not so far from the other points; they are localized close to the boundary. So in our particular case, we will prefer to use Soft-K-means with 3 clusters and $\beta = 17$, to consider that some points are unsettled.

IV. Classification

The classification in MLDemos is not feasible for dataset with more than two dimensions. Unfortunately, two dimensions represent only 54,1% of our dataset (Figure I.2 in annex). So if the classification does not work well, it could be necessary to increase the number of dimension. The goal of this part is to compare the sensitivity of each algorithm performances according to their hyperparameters.

Train/Test ratio

If we use a small training set and we got a good f-measure, that means our classification is robust. Indeed, a small training set means a huge Test set. So in other words, we made a strong test on revelants points of the dataset.

But, generally the error grows and the f-measure decreases when we decrease this ratio. With a small ratio, the simulation will be more trustworthy, so we need to find a compromise.

Higher the ratio, stronger the classification, but less trustworthy. *Vice versa* Lower the ratio, lower the classification, but more trustworthy (cf figures IV.1 and IV.2 on the next page).

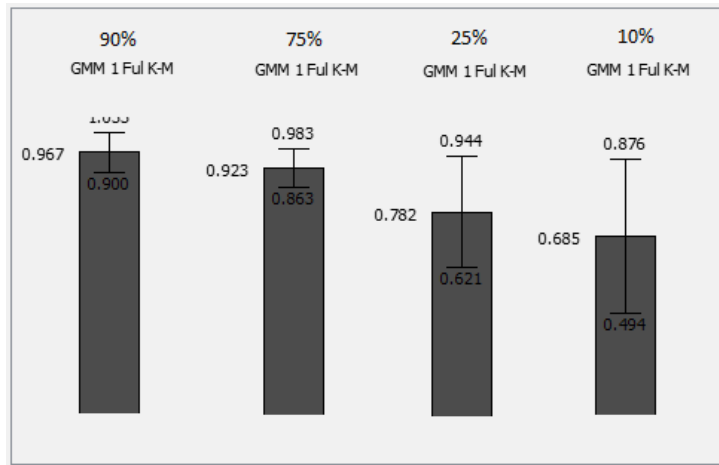


Figure IV.1 : Evolution of the f-measure depends on the train/test ratio

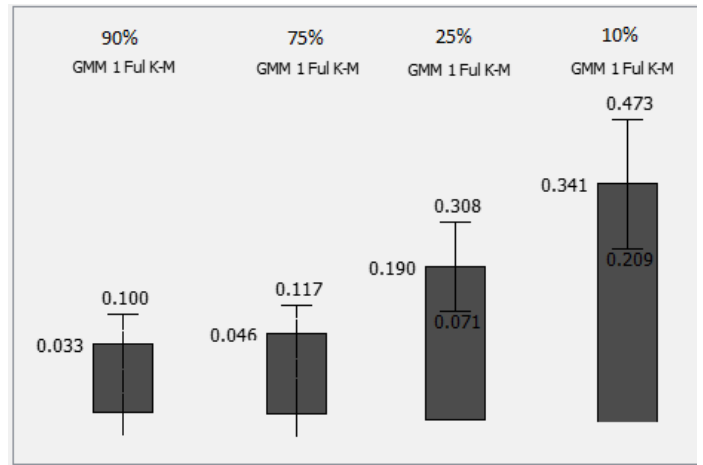
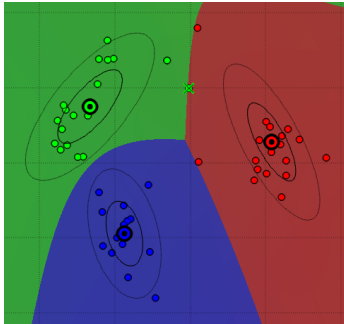
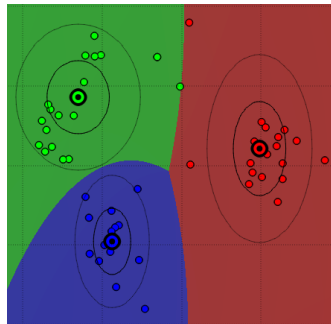


Figure IV.2 : Evolution of the Error depends on the train/test ratio

Cross validation

The cross validation split our dataset into the number of folds according to our ratio. So if we have three datapoint with 66% ratio. There is no sense to put more than three folds because else it will be redundant.

Hyperparameters	Gaussian Mixture Model (GMM) + Bayes
Type of Covariance <ul style="list-style-type: none"> Full Diagonal Spherical 	<p>The single driver that lies behind the choice of covariance type is the computational cost. Indeed the full matrix will be more computationally costly than the diagonal or the spherical one. But changing the type will also change the shape of the gaussian. Basically we should use the full covariance matrix to fit the data as well as possible. But if fortunately, our cluster are spread in a way that a diagonal or a spherical matrix could be used to fit the data more precisely, it is better in a computational sense to use them.</p> <p>For example, we can see on the plot below that a diagonal matrix fit perfectly the data. So it's better to use it than a full covariance matrix.</p> <div style="display: flex; justify-content: space-around; align-items: flex-end;"> <div style="text-align: center;">  <p>Figure IV.3 Full Covariance Matrix</p> </div> <div style="text-align: center;">  <p>Figure IV.4 Diagonal Covariance Matrix</p> </div> </div>
Initialization	The initialization will impact the mean of the Gaussian which will fit the cluster (cf box in Initialization in the K-mean table above).
Number of Mixture Gauss	<p>The number of gaussian will allow to fit more correctly the data. Each Gaussian will fit a local set of datapoint. So by playing between the type of gaussian and the number we can decrease the computational cost while maintaining a good clustering.</p> <p>An optimisation could be to use different type and number of gaussian for each cluster. Thus, we could decrease the computational cost while keeping gaussian which fit correctly the cluster. For instance, on the plots below we can see that one gaussian per class is enough to have a good classification. But there is two points very close to the border between the class green and red when we use only one gaussian per class, this could be solved by using two</p>

gaussians. As we can see, with two gaussian this problem of points close to the border is solved, but we lost the connexity of the red class.

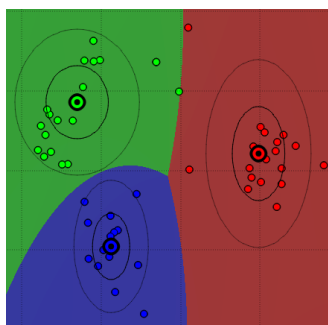


Figure IV.5 One Gaussian per Class

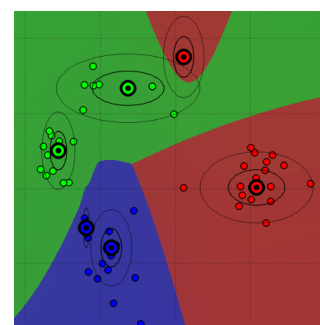


Figure IV.6 Two Gaussian per Class

Hyperparameters	Support Vector Machine (SVM)
Kernel <ul style="list-style-type: none"> Linear RBF 	<p>This changes the shape and the types of measure. If we have data which can be easily be separable by a line, a linear kernel would be appropriate. If the shape is more complicated, a RBF kernel will be suitable to follow the shape of the classes.</p> <div style="display: flex; justify-content: space-around; align-items: flex-end;"> <div style="text-align: center;"> <p>Figure IV.7 Linear</p> </div> <div style="text-align: center;"> <p>Figure IV.8 RBF</p> </div> <div style="text-align: center;"> <p>Figure IV.9 More complicated dataset</p> </div> </div> <p>According to our dataset, whether linear or RBF, both fit perfectly classes. But on the example called “More complicated one”, we can see that it is hard to separate each class with lines. So a RBF kernel would be better.</p>
RBF Width	<p>This parameter impact the thickness of the margin. When the width is big, the model is too constrained and cannot capture the complexity or “shape” of the data. The region of influence of any selected support vector would include the whole training set. The resulting model will behave similarly to a linear model. If the width is too small, the radius of the area of influence of the support vectors only includes the support vector itself and no amount of regularization with the penalized factor C will be able to prevent overfitting.</p> <div style="display: flex; justify-content: space-around; align-items: flex-end;"> <div style="text-align: center;"> <p>Figure IV.10 Width = 0,001</p> </div> <div style="text-align: center;"> <p>Figure IV.11 Width = 1</p> </div> </div>
Penalty Factor C	<p>This factor will penalize the points which are misclassified or in the margin. By increasing this factor, we will give more weight to those misclassified points when computing the margin. So the</p>

higher C , the smaller the margin. Reciprocally, the smaller C , the higher the margin. The white points are the points located in the margin. Some points are into the margin but they are not white because there are not in the training set but in the testing one (cf Figure IV.12 and Figure IV.13).

Note : It is important to say that the number of mixture gaussian is not only limited by the computational cost, but also by the number of points in the dataset. Indeed, if the amount of points per class is too low, adding more gaussian will not fit better the dataset. This effect is shown on the Figure IV.14 below.

In conclusion if we want to classify our dataset, we will prefer to use Support Vectors Machine. As we have some points very close to the border, Gaussian Mixture Model tends to misclassified them. Then by using SVM with a RBF kernel and the following parameters : width = 0,9 – $C = 200$, we can penalize those points

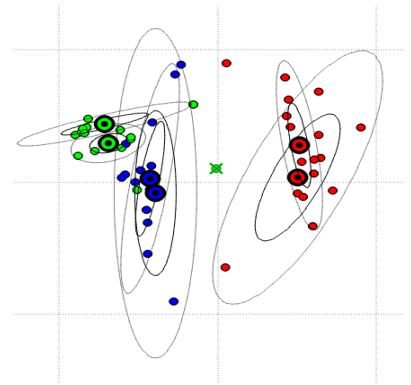


Figure IV.14 : Two Gaussians per class

V. Regression

In this section, we will study two algorithms which are Gaussian Mixture Regression (GMR) and Support Vector Regression (SVR) in a qualitative and quantitative way. Thus we will highlight impacts of the hyperparameters. This survey will be conducted through a second dataset. The first one is not suitable for this experiment because of its non bijectivity, there is multiples output y for a specific input x and because it does not represent a continuous function.

V.1. Dataset

This new dataset correspond to a zoom on the face of someone. Below each image, we assign a value which quantify the amount of zoom operated. Thus, thanks to a regression we can evaluate this output “ y ” according to a new image.

Enter the output metric for each of your images.

0.0	0.1	0.2	0.3	0.4
0.5	0.6	0.7	0.8	0.9
1.0	1.1	1.2	1.3	1.4
1.5	1.6	1.7	1.8	1.9
2.0	2.1	2.2		

Figure V.1.1. New dataset with the new dimension which corresponds to the output y

V.2. Qualitative approach

In order to see the impact of hyperparameters, we will study the two dimensional dataset, with the output “y” on the y-axis and the dimension which is the most suited for regression on the x-axis. According to the plot below (Figure V.2.1) the best dimension is the first one; the function is almost bijective. This survey is qualitative because it consider only one dimension of the dataset.

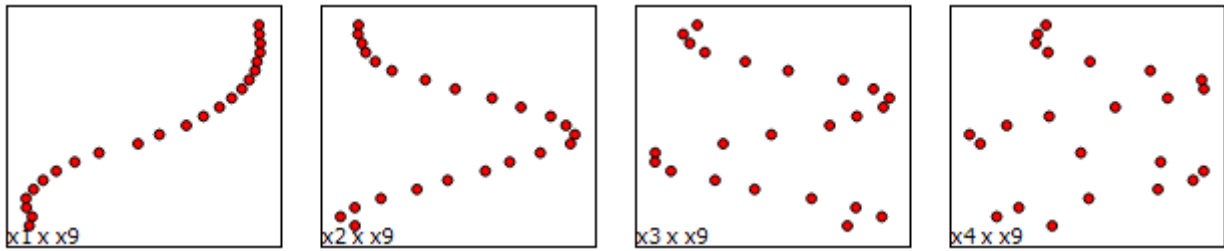


Figure V.2.1. Plot the output y in function of the first four dimensions

GMR

With this algorithm, there are only three hyperparameters which can influence the regression, the number of mixture components, the second type of covariance matrix and the initialization.

- The number of gaussians will allow to fit more or less the dataset. But it is important to note that if the number of gaussian is too large compare to the amount of points, we will not have a better fit and the variance which is the relative dispersion of the predictive model will increase. Moreover, more gaussians means more computational cost. So it is important to carefully choose the number of mixture components. According to our dataset a number of five seems adequate.

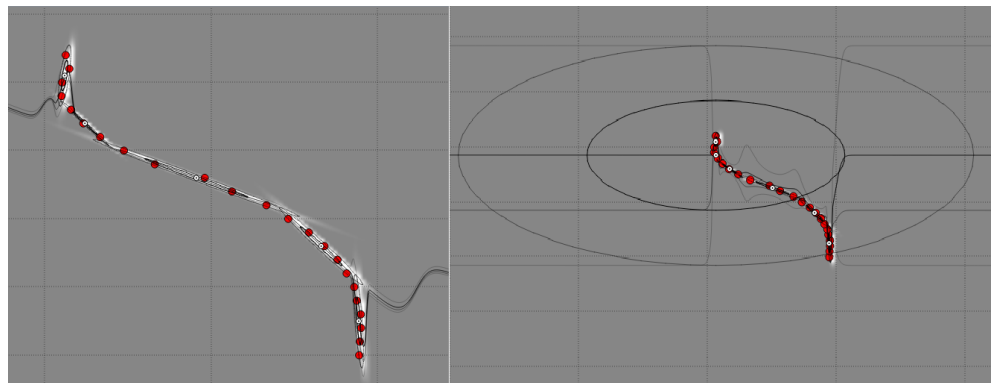


Figure V.2.2. 5 Gaussians (Left) and 5 Gaussians (Right)

- In regards to the type of covariance matrix, it will impact not only the variance but also the likelihood represented in a gray scale as shown in the figure below.

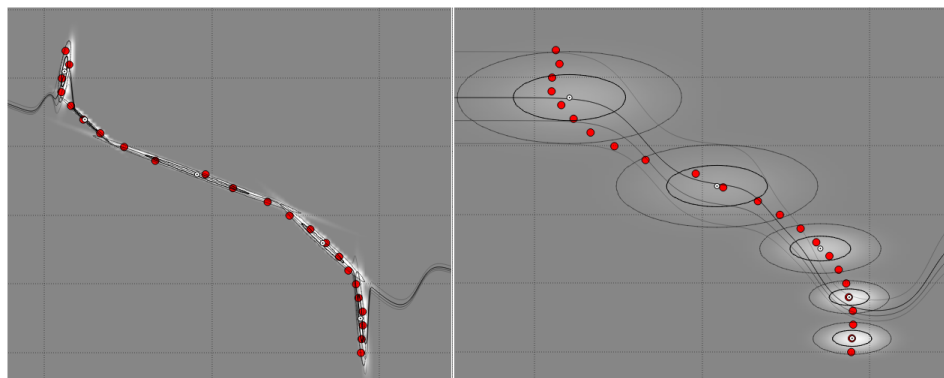


Figure V.2.3. Full Covariance Matrix (Left) and Diagonal Covariance Matrix (Right)

- And finally, the type of initialization. The impact of this parameter is evident, with K-means the position of the centroids will almost be the same and they will allow to fit the dataset fairly well, unlike with a random initialization which is unpredictable and not optimized. But K-means is computationally costly and sometimes it does not be very optimized for particular dataset (cf part III).

SVR

This algorithm uses four parameters to compute the regression. The first one is the choice of the kernel, but here we will only use the RBF one. The linear kernel is not adapted to fit with our dataset.

Secondly, we need to set the kernel width. This hyperparameter will be set depending on if we want a smooth regression or a very sharp one. In other words, a huge kernel width will take into account more points to estimate the output “y” than a small kernel width which will consider less points to estimate the output. Thus a huge kernel width will be less sensitive to noise but also less accurate and *vice versa* for a small one.

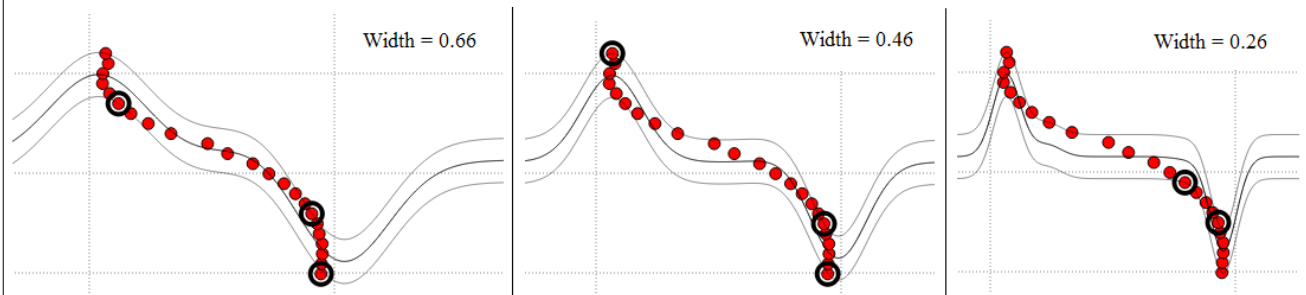


Figure V.2.4. Impact of the kernel width on the regression

Thirdly, the penalty factor. Contrary to SVM it penalizes only the points which does not satisfy the constraint, in other words it penalize the points which are out of the ϵ -tube, the support vectors. So higher the penalty factor, higher will be the fit on the support vectors.

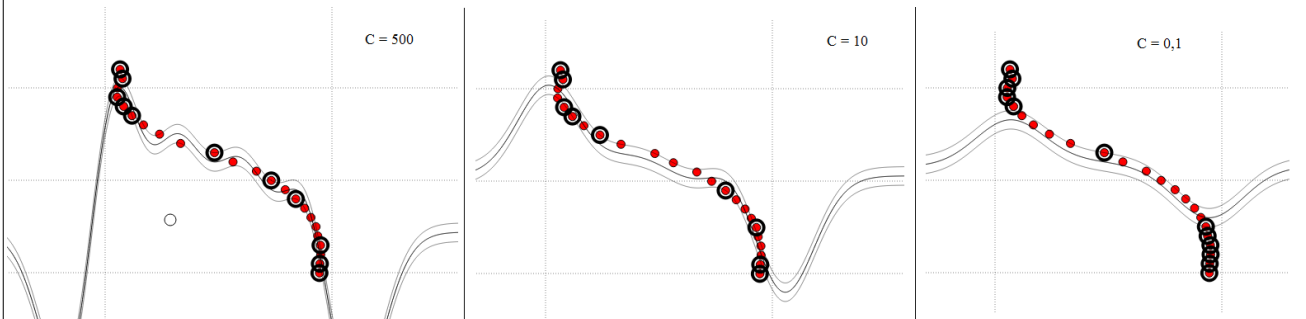


Figure V.2.5. Impact of the penalty factor

Finally, we can change ϵ -tube which corresponds to the uncertainty of the output. This parameter will define the number of support vectors. Indeed, higher the ϵ -tube is, fewer the number of support vectors will be. So the penalty factor and the ϵ -tube are highly correlated.

V.3. Quantitative approach

Now that we saw the impact of the hyperparameters in a qualitative way, it is interesting to study the impact of those hyperparameters on the multi-dimensional dataset. Our first approach focused on a unique dimension, but if we study the impact of the hyperparameters on several dimensions it is not easy to clearly see which are the optimal hyperparameters. We will then consider all the dataset and visualize the regression error in terms of Mean Square Error (MSE). We could use as many dimensions as we want, but we will focus on the first eight dimensions in this survey.

GMR

As we can see on the plot below, the optimum number of gaussians is not five anymore, but obviously one in terms of MSE.

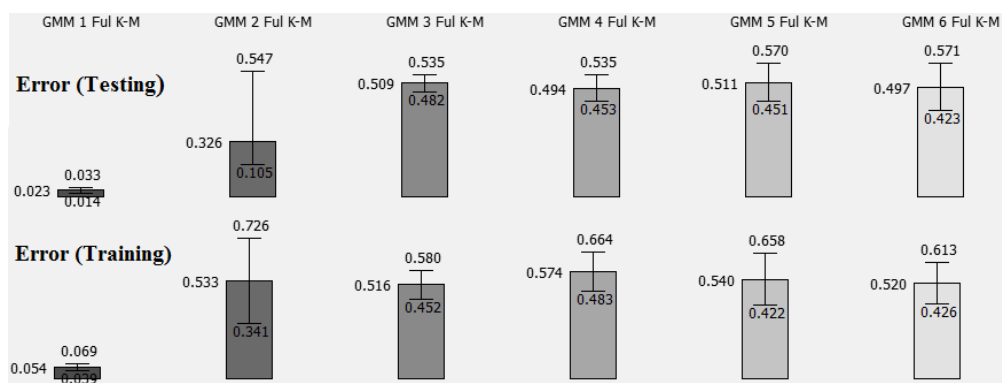


Figure V.3.1 Impact of the number of gaussians on the MSE

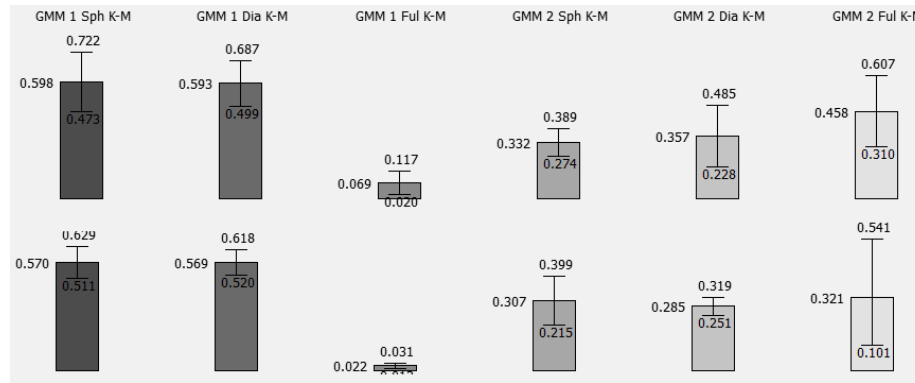


Figure V.3.2 Impact of the type of covariance matrix on the MSE (Testing Error on top and Training Error below)

Concerning the type of matrix, the best one is a full covariance matrix if we use only one gaussian, and the spherical covariance matrix if we use two gaussians. So the type of covariance matrix is intrinsically linked to the number of gaussians used to fit the dataset.

SVR

First of all, let's try to fix the kernel width. As we can see on the adjacent graph, a kernel width of 8 is optimized to regress our dataset.

With this parameter fixed, we can try to find the best ϵ -tube in our regression. According to the graph Figure V.3.4 the optimum ϵ -tube seems to be 0,07. We could have chosen an ϵ -tube of 0,05 in order to have a lower training error, but we can see that the testing error is bigger. So the regression would be more accurate but less trustworthy.

And finally, with those two parameters fixed, we can set the penalty factor to minimize the margin. As show on the Figure V.3.5 we can choose a penalty factor of 10 as well as 100. But as it has been said before, if the penalty factor is too big, we will overfit our dataset. So we prefer to choose the penalty factor of 10.



Figure V.3.3 Impact of the Kernel width on the MSE

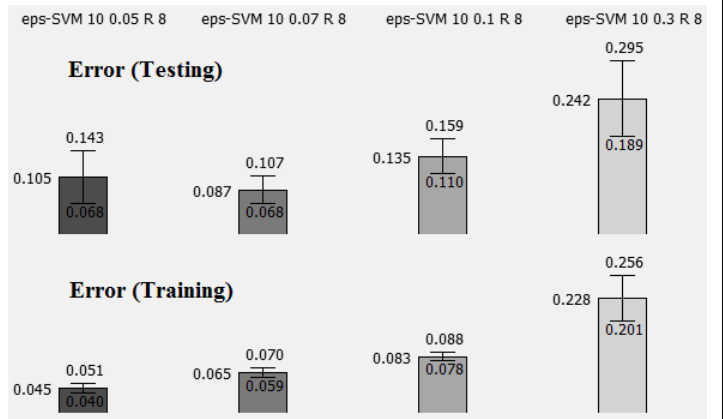


Figure V.3.4 Impact of the ϵ -tube on the MSE

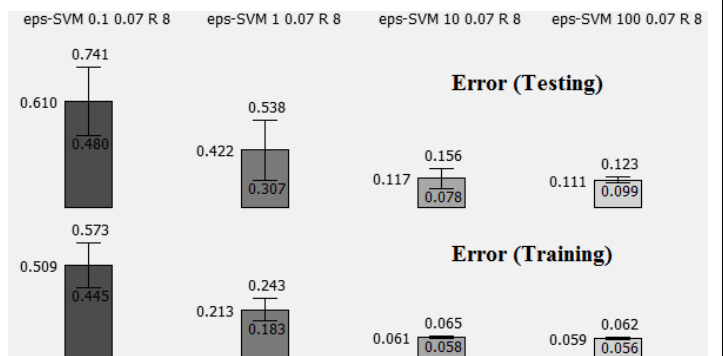


Figure V.3.5 Impact of the penalty factor on the MSE

Note : All the graph have been plotted with a train/test ratio of 66% and a 7-cross validation.

VI. Conclusion

Depending on what we are looking for and according the features of our dataset we can make some conclusions about what we have studied. First of all, we can separate data by projecting onto the eigenvectors $e_1 \times e_2$. This projection separates the three classes according to their differences (Humans, fruits, Costume). Then, if we want to cluster our dataset to clearly separate them, we can use a soft K-means with the following hyperparameters : clusters $K = 3$ and $\beta = 17$.

In addition, our purposes could also be to find the boundary between each class in order to classify future points we would add. A good way could be to use SVM with the following hyperparameters : a RBF Kernel, a width of 0,1, a penalty factor about 50, a 66% train/test ratio with 10-cross validations.

And finally, if we want to do a regression on our second dataset, we can use SVR with the following hyperparameters : A RBF kernel with a width of 8, an ϵ -tube of 0,07 and a penalty factor about 10.

Nevertheless, the most important thing is not the algorithm or the models used, but the dataset. We see through this report that a good dataset can be clustered, classified or regressed easily. Withal, a bad dataset, even if we carefully design the model to fit it, we will have a bad clustering, classification or regression.

Annex

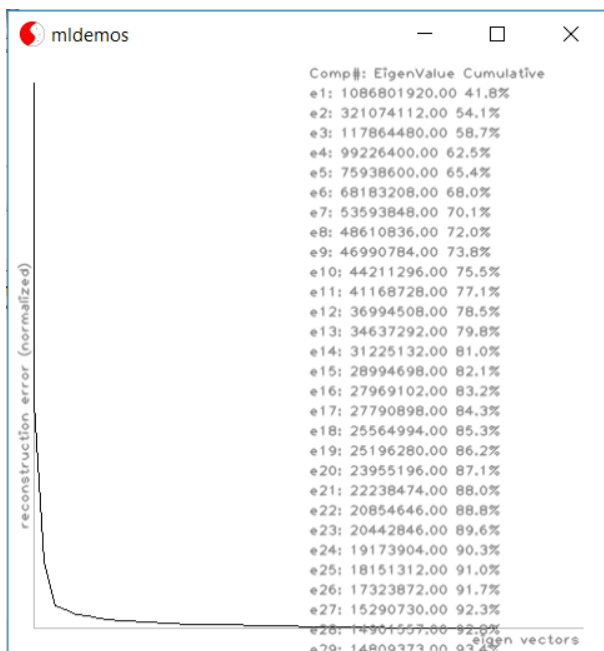


Figure I.2 Reconstruction Error of the first dataset

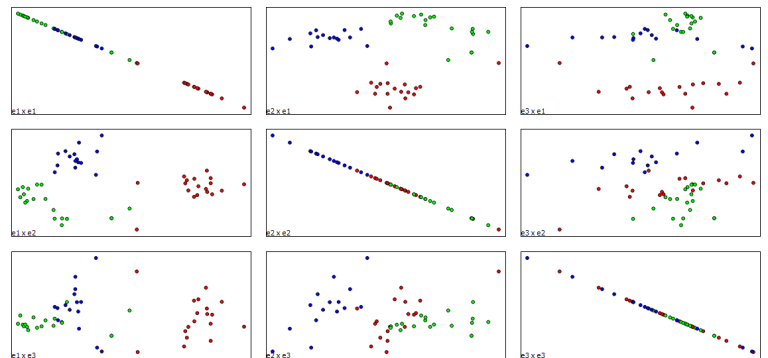


Figure I.3 Projection of the first dataset on different eigen vectors

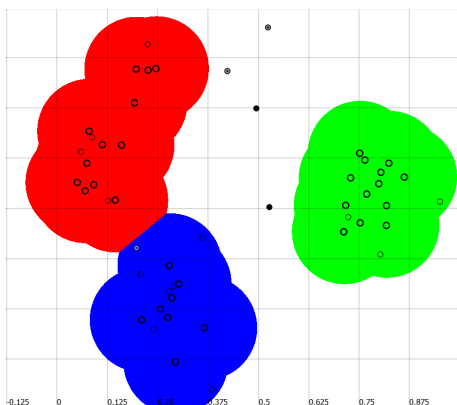


Figure III.4 DBSCAN Minpts = 3 and $\epsilon = 0,13$

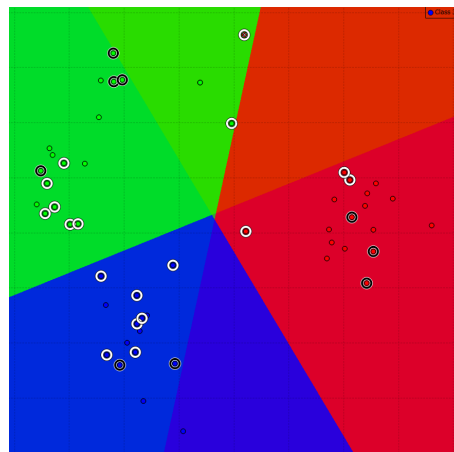


Figure IV.12 SVM width = 50 $C = 2$

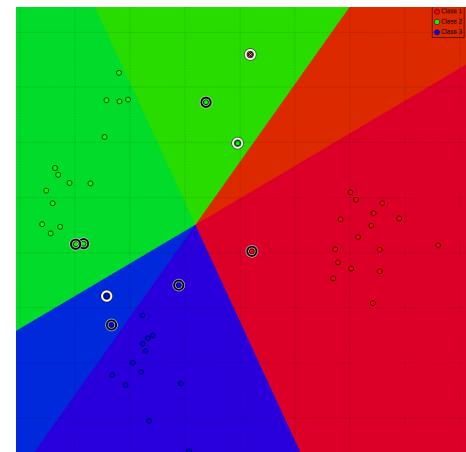


Figure IV.13 SVM width = 50 $C = 20$