Introduction to Machine Learning
Work 1
Clustering and factor analysis exercise

Marc Oliu Simón
Joan T. Matamalas Llodrà

October 28, 2012

1 Parsing the files

The first step in the development of both K-Means and PCA algorithms has been the development of a parsing function for the ARFF files. For this purpose, the documents have been read line by line, and the comment lines, empty lines and types definition lines have been discarded.

The signature for the function is the following one:

$$function[matrix] = parser\_ariff(path)$$

As it can be seen, the input value is the path of the data set to load, while the output value is an array of integers representing the matrix parsed from the input file.

The lines containing the matrix data have been parsed by converting the comma-separated rows (which represent individuals of the population) into their fields, and the column representing the classification of those individuals have been discarded. That is so because the goal of the practice is to implement an unsupervised classifier, and thus there is no need to maintain the classifier for each element.

For the fields containing unknown values, a NaN (Not a Number) value has been assigned in order to transform the rest of the fields from strings to numeric values.

2 Standardizing the data and assigning missing values

Once the matrix is obtained, the data is standardized to give an equal importance to all the fields of the data, and in the process the value for the outliers is assigned.

A standardization process has been chosen over normalization because of the continuous nature of the variables. Since the variables of the data set represent a continuous value over the feature, and since the features follow a normal distribution, it's better to standardize to avoid features containing elements at various standard deviations from the mean from gaining importance over the features not containing them.

For example, if a feature had an individual past the second standard deviation ($P < 0.95$) this feature data would be more compressed if normalized than another without any individual past the second standard deviation, since this individual would have a value of 0 or 1 for this feature.

On the other hand, when standardizing, 67% of the population will fall within one standard deviation, independently of the existence of outliers or low probability individuals that could fall past the second standard deviation.

The function developed to perform the standardization and assignment of missing values has the following signature:

$$function[matrixZ] = standarizer(matrix)$$

The process followed to standardize the data is the following one for each feature:

1. Calculate the mean of the feature ignoring the unknown values (NaNs)
2. Assign the obtained mean to the unknown values
3. Standardize the feature

This process is done simultaneously for all features of the data set by working with all of the columns (features) of the matrix at the same time.

3 Implementation of the K-Means

3.1 The K-Means Function

The K-Means algorithm has been implemented inside a function named k_means, which has the following signature:

$$function[best\_output, best\_centroids, best\_inertia] = k\_means(dataMatrix, k, seed, repetitions)$$

As it can be seen, the function accepts three input parameters, and gives three parameters as output. The input parameters represent the following:

- **dataMatrix:** This parameter is the matrix over which the k-means algorithm will perform the clustering. Each row of the matrix is an individual of the population and each column one of the features.
- **k:** Is the number of clusters the data will be classified in.

- **seed:** It's an integer number used as seed for a random number generator. The method to select the initial seeds of the K-Means algorithm consists on assigning each individual to a cluster and then calculating the centroids of each cluster. *Default 1*

- **repetitions:** Number of repetitions of the algorithm, returns the best clusters so far. *Default 1*

The three output parameters of this function are the following ones:

- **best_output:** Vector with the best cluster for each individual obtained after n repetitions.

- **best_centroids:** Centroids of the clusters (1 row per cluster)

- **best_inertia:** Vector with the inertia of each cluster

Our K-Means algorithm implementation starts out by assigning each individual to one of the K clusters randomly, and then calculating it's centroids. This way, the initial seeds are obtained for the algorithm to run. Then it clusters the individuals to the centroids by using the euclidean distance between each individual and each centroid, assigning the individual to the nearest centroid.

This process is repeated iteratively, Recalculating the centroids for the new clusters and then creating the clusters again using the newly calculated centroids to assign the individuals to each cluster. The process stops when there is convergence, that is, the centroids don't change anymore and each individual is re-assigned to the same cluster to which it was assigned the previous iteration.

3.2 Generating and selecting the clusters

To generate the clusters for the K-Means classifier, an iterative process has been implemented, in which the data is clustered for values of K ranging from 2 to 5, and for each value of K different seeds are tried.

Since the seed in our implementation is an integer value that is used as the seed of a random number generator which in turn creates an initial random clustering of the data, a range of numbers from 1 to 25 is tried to generate the initial seeds for the clusters.

From the different seeds used, the best clustering is selected by using the clusters internal inertia as evaluation method. For the clusters with the same value of K, the one that has the lowest inner cluster inertia are considered the best, since the distance between the cluster members and their centroids is minimized, meaning the clusters are more compact.

Unfortunately, this same procedure can't be used to evaluate which is the best value of K. This is because a higher value of K always minimizes the inner cluster inertia, and the best value would turn out to be a K value equal to the number of individuals in the

data set (cluster inertia = 0). For that reason, the K-Means algorithm is run for the different K values, and a reduced dimensionality 3D plot is shown for each value of K, by applying Multi Dimensional Scaling (MDS), allowing for a manual selection of the best K value.

4 Implementation of the PCA

The PCA is implemented in the function *pca*. The signature of that function is:

$$function[outData, transformedData, eVectors, eValues, informativeFeatures] = \\ = pca(dataMatrix, eValueThreshold)$$

The inputs parameters are:

- **dataMatrix:** This parameter is the matrix over which the k-means algorithm will perform the clustering. Each row of the matrix is an individual of the population and each column one of the features.
- **eValueThreshold:** The minimum value of an eigenvalue to accept the associate eigenvector as featured vector.

As output the function provides:

- **outData:** The data in the original coordinates with the reduction of dimensions applied.
- **trasnfomedData:** The transformed data with dimensions reduced
- **eVectors:** eigenVectors of the data
- **eValues:** eValues of the data
- **informativeFeatures:** the index of the most informative attributes

The principal component analysis (PCA) is a procedure used to reduce the dimensionality of the original data. The steps in what the algorithm is decomposed are:

1. **Subtract the mean from data:** This step was omitted. The algorithm works with standardized data, so we've already subtracted the mean in the standardization step.
2. **Calculate the convariance matrix:** This is an important step that gives us information about the variance of the several attributes. We have used the *cov* function to obtain this details.
3. **Calculate the eigenvectors and eigenvalues of covariance matrix:** Matlab has a function that does that task (*eig*). The function returns a matrix with the eigenvectors and other with the eigenvalues on the diagonal. Both are ascendent ordered so we need to revert that in order to keep them in descendent order. The eigenvalues are put in a column with the command *diag* in order to be more useful.

4. **Choose components and construct a new featured vectors set:** We have to choose the most informative attributes to represent the data. We have created a matrix containing the vectors that have eigenvalues above a determined threshold, by default 1.

5. **Transform the original data based on the set of featured vectors:** Now, with the featured eigenvectors in our hands, we can proceed to transform our original data. To do that, we multiplay the transposed version of the featured vectors matrix by the transposed data matrix. That result will have n number of dimensions, where n is the number of eigenvectors selected.

6. **Reconstruct the old data back:** If we wanna reconstruct the data in our originals coordinates we would multiply the matrix of featured vectors by the matrix obtained in the previous step.

Additionally to this steps, we need to know which attributes have more information. To do that, we choose from the absolute values of each featured vector the maximum value, the index of this maximum is the index the attribute that we need.

5 Analysis of the data sets

5.1 Data set: Pima Indians Diabetes Database

File: pima_diabetes.arff

This dataset corresponds to a study of the diabetes disease over a population of Pima indians. For the analysis, a group of 8 variables were taken into account, which are:

1. Number of times pregnant
2. Plasma glucose concentration a 2 hours in an oral glucose tolerance test
3. Diastolic blood pressure (mm Hg)
4. Triceps skin fold thickness (mm)
5. 2-Hour serum insulin (mu U/ml)
6. Body mass index (weight in kg/(height in m)^2)
7. Diabetes pedigree function
8. Age (years)

Each of the individuals has been classified as having diabetes or not, but for the purpose of this work, the classification isn't required. The first step in the analysis of the data will be to try a classification of the individuals by using the K-Means classifier with different values of K. The figure 1 show the result of this classification.

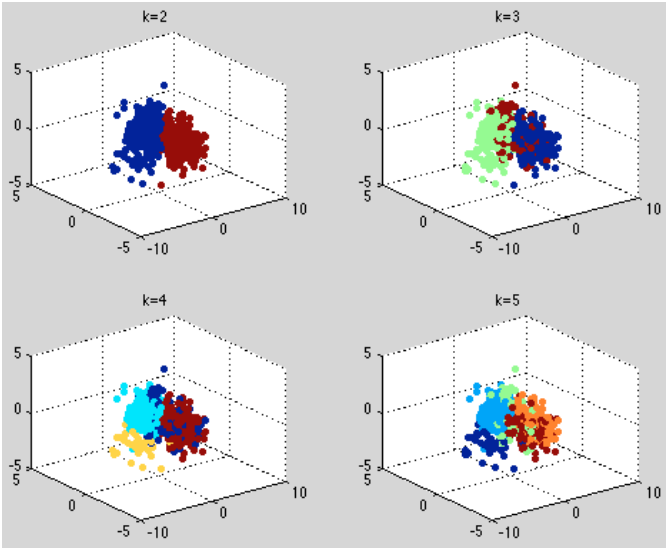

Figure 1: k-means clustering for diabetes data without use PCA

For each consecutive K value, the inertia of the clusters is further reduced, as expected. Since the above representation is a 3D representation of an 8-dimensional space, it's hard to tell which K value gives a better separation between clusters, although it seems that the K=2 and K=3 clusters have a better separation. What can be done, although, is to check for big leaps in the inertia of clusters when increasing the value of K. The following table shows the inertia of the clusters for the different K values:

K=2	**K=3**	**K=4**	**K=5**
5122.05	4354.151	3913.8495	3621.0224

Table 1: Inertia of the clusters without use the PCA

As it can be seen, the inner cluster inertia for K=3 is 85% of that of K=2, which means it has not been reduced more than what would be expected when increasing K in a random clustering. For that reason, it will be considered that K=2 is the best number of clusters for the data.

When applying a Principal Component Analysis to the data, it can be found that the most relevant features for the data analysis are the 6th, 8th and 3rd features, in that order. Those are:

- 6. Body mass index (weight in kg/(height in m)^2)
- 8. Age (years)
- 3. Diastolic blood pressure (mm Hg)

Knowing the disease, those results make sense, since diabetes normally comes as a result of a bad diet. High blood pressure and high mass index represent a bad died, while

an advanced age represent a sustained bad diet over time.

Those results have been obtained by discarding the eigenvalues lower than the threshold 1, and then selecting the most relevant components associated with the remaining eigenvectors.

Even though, the power of the PCA algorithm doesn't lie only in the selection of features, but in the reduction of dimensionality in the transformed space. A way to check the validity of this reduction of dimensionality is to perform a K-Means clustering over the transformed data with reduced dimensionality, plotting the results at the same original coordinates system, figure 2.

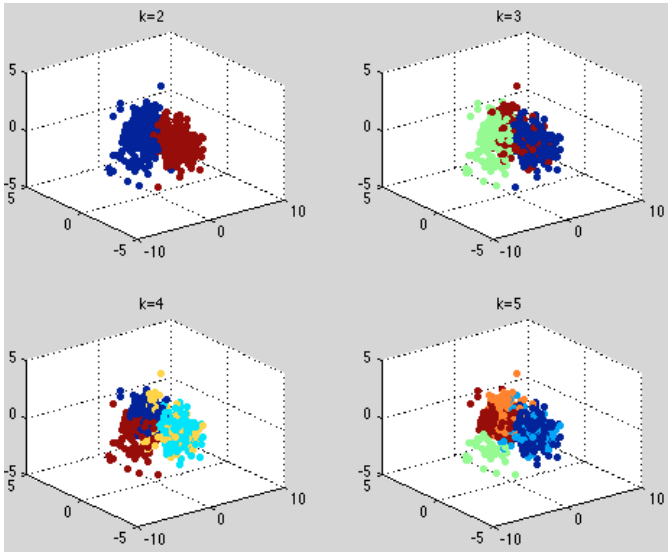

Figure 2: k-means clustering for diabetes data using PCA

As it can be seen, the result of the classification doesn't seem to have changed a lot. That is because the three dimensions plotted in both cases are the three most relevant dimensions according to the PCA algorithm. Since the transformed data has removed the noise mostly by reducing the influence of the other dimensions of the plot, and the data was more evenly distributed among the other dimensions, the classification result is similar.

If the new inner cluster inertia values are taken into account once the PCA has cleared the data, it can be seen that the inertia is reduced a lot, even though the final clusters are almost the same:

K=2	**K=3**	**K=4**	**K=5**
2713.89	1955.84	1610.03	1408.60

Table 2: Inertia of the clusters using the PCA

This implies that the noise has been cleared. By reducing the noise, the individuals are closer together, and the distance to the centroids is shorter.

5.2 Data set: Vehicle silhouettes

File: vehicle.arff

This dataset corresponds to a study of the classification of cars into four different kinds () according o their silhouette. The data originally used for the classification is composed of 18 variables, which are:

1. COMPACTNESS (average perim)**2/area
2. CIRCULARITY (average radius)**2/area
3. DISTANCE CIRCULARITY area/(av.distance from border)**2
4. RADIUS RATIO (max.rad-min.rad)/av.radius
5. PR.AXIS ASPECT RATIO (minor axis)/(major axis)
6. MAX.LENGTH ASPECT RATIO (length perp. max length)/(max length)
7. SCATTER RATIO (inertia about minor axis)/(inertia about major axis)
8. ELONGATEDNESS area/(shrink width)**2
9. PR.AXIS RECTANGULARITY area/(pr.axis length*pr.axis width)
10. MAX.LENGTH RECTANGULARITY area/(max.length*length perp. to this)
11. SCALED VARIANCE ALONG MAJOR AXIS (2nd order moment about minor axis)/area
12. SCALED VARIANCE ALONG MINOR AXIS (2nd order moment about major axis)/area
13. SCALED RADIUS OF GYRATION (mavar+mivar)/area
14. SKEWNESS ABOUT MAJOR AXIS (3rd order moment about major axis)/sigma_min**3
15. SKEWNESS ABOUT MINOR AXIS (3rd order moment about minor axis)/sigma_maj**3
16. KURTOSIS ABOUT MINOR AXIS (4th order moment about major axis)/sigma_min**4
17. KURTOSIS ABOUT MAJOR AXIS (4th order moment about minor axis)/sigma_maj**4
18. HOLLOWS RATIO (area of hollows)/(area of bounding polygon)

The first step in the analysis of the data will be to try a classification of the individuals by using the K-Means classifer with different values of K. The plots 3 show the result of this classification.

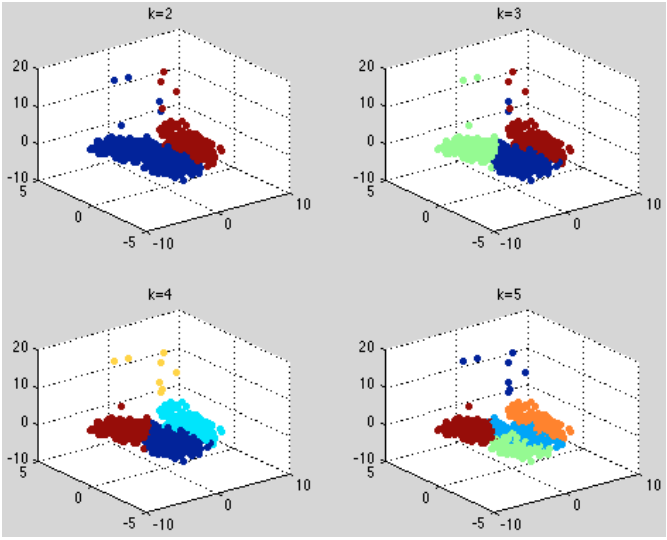

Figure 3: k-means clustering for vehicle data without use PCA

As it can be seen in the plot, it seems the three first classifications are the best ones. For a value of K=2, two very distinct groups of data are classified, and for K=3 the clustering seems to make an apparently unnatural division of the former first cluster into two distinct ones. In the case of K=4, although, it can be seen that a fourth cluster is made with those elements of the previous clusters that seemed to be outliers in the previous clusters. To check if the increase of the clusters number is good, the inner cluster inertia can be checked:

K=2	**K=3**	**K=4**	**K=5**
8958.65	7299.39	5973.53	5400.05

Table 3: Inertia of the clusters without use the PCA

As the number of clusters increases, as expected, the innertia decreases. But although the inertia doesn't decrease much when going from K=2 to K=3 (82.4% of the inertia of K=2), it does when going from K=3 to K=4 (81% of the inertia of K=3). It has to be taken into account that de pace of decrease in inertia is inversely proportional to the number of clusters. That is, as the number of clusters increases, the inertia decreases more slowly. For that reason, the 81% of the inertia when going from K=3 to K=4 is much more significative than the same decrease when going from K=2 to K=3. It can be said that the best clustering in this case is for 4 clusters (K=4).

An analysis by applying the PCA algorithm to the data set tells us that the most important components of the not discarded eigenvectors once applied an eigenvalue threshold of 1 are the 7th, 17th, 5th and 16th dimensions or attributes, in that order of relevance, which are:

- 7. Scatter ratio (inertia about minor axis)/(inertia about major axis)
- 17. Kurtosis about major axis (4th order moment about minor axis)/sigma_maj**4

- 5. Pr. Axis aspect ratio (minor axis)/(major axis)
- 16. Kurtosis about minor axis (4th order moment about major axis)/sigma_min**4

It's easy to see that attributes 7 and 5, which represent the distribution of force in the vehicle and the relative dimensions of the vehicle (width/length) are clearly influent when trying to describe a vehicle. Also, the Kurtosis attributes are important, since they describe the distribution of the forces along the two axis of the vehicle.

If the reduced dimensionality data obtained of the PCA by transforming the data set and then discarding the eigenvectors with a lower eigenvalue than 1 is used to classify the individuals, and then the individuals are plotted in the same coordinates space than before, the result is this one, figure 4.

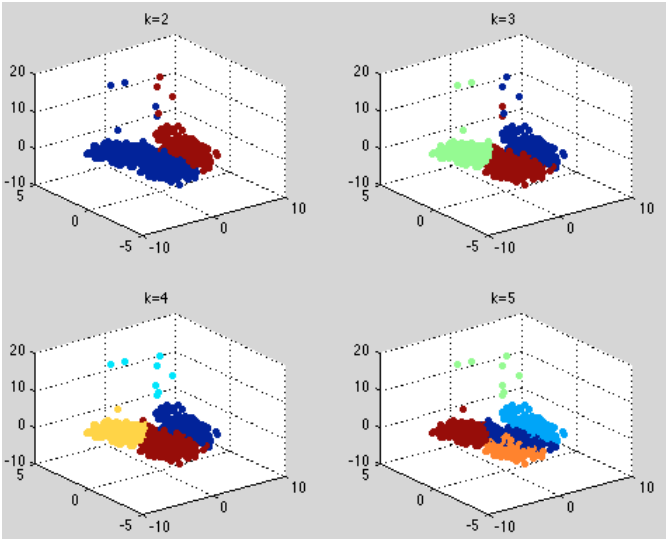

Figure 4: k-means clustering for vehicle data using PCA

It can be seen that the result is very similar to that of the first plot. This means that the reduction of dimensionality hasn't discarded any relevant dimensions. It can now be checked what has happened to the inner inertia of the clusters:

K=2	K=3	K=4	K=5
6875.77	5234.29	3957.5551	3399.9879

Table 4: Inertia of the clusters using the PCA

It can be seen that now the inertia of K=4 is 75.6% of that of k=3, meaning there is a greater reduction in inertia when increasing the number of clusters. This happens because now that the dimensionality is reduced, the individuals are closer together. It serves as a measure of the reduction of noise, but it can't be taken into account as a factor for determining a better separation between clusters, since the addition of a new cluster always has a greater impact in inertia if the space has a lower dimensionality.

5.3 Data set: Wine recognition data

File: wine.arff

This is a data set used to classify wines coming from three different cultivars of the same region of italy. The goal would be to classify the different wines into their corresponding cultivars according to 13 different variables obtained from a chemical analysis of the wine.

In this case, there is no description to the 13 available attributes that are being analyzed, so they will be referred by their order in the data set with an integer value between 1 and 13.

The first step in the analysis of the data will be to try a classification of the individuals by using the K-Means classifier with different values of K. The plots 5show the result of this classification.

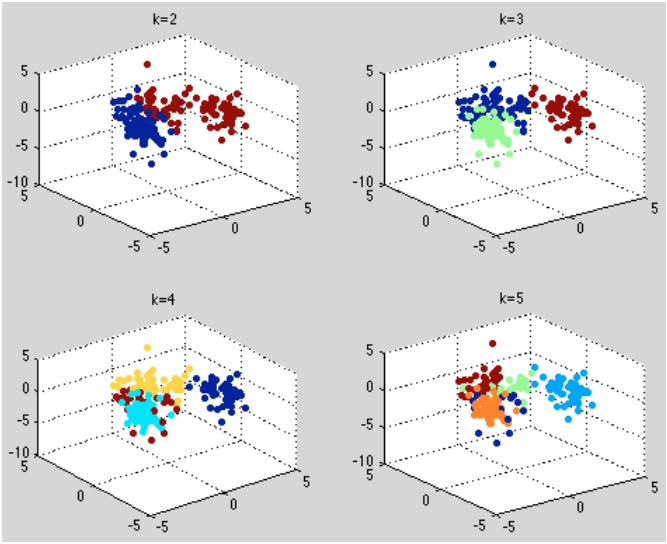

Figure 5: k-means clustering for wines data without use PCA

In this case, at plain sight, there appears to be a good clustering only in the case of K=3. For the case K=2 the red cluster is taking two groups of individuals which are clearly different, and for the case of K=4, there is a superposition between clusters, where the red and clear blue clusters are touching one with the other and there's no clear separation between them.

Even though, a good manner to check if it's really the case, is to check the reduction of the inner clusters inertia when the value of K increases.

- Inertia of K=3 with respect to K=2: 77%
- Inertia of K=4 with respect to K=3: 92%
- Inertia of K=5 with respect to K=4: 94%

K=2	K=3	K=4	K=5
1649.44	1270.74	1168.77	1097.37

Table 5: Inertia of the clusters without use the PCA

In this case, the inertia decreases when increasing the K value from 2 to 3, and has a very small decrease when further increasing the value of K.

For that reason, it can be said that the optimal K value is for K=3.

An analysis by applying the PCA algorithm to the data set tells us that the most important components of the not discarded eigenvectors once applied an eigenvalue threshold of 1 are the 7th, 10th and 3rd dimensions or attributes, in that order of relevance.

If the reduced dimensionality data obtained of the PCA by transforming the data set and then discarding the eigenvectors with a lower eigenvalue than 1 is used to classify the individuals, and then the individuals are plotted in the same coordinates space than before, the result is this one, figure 6.

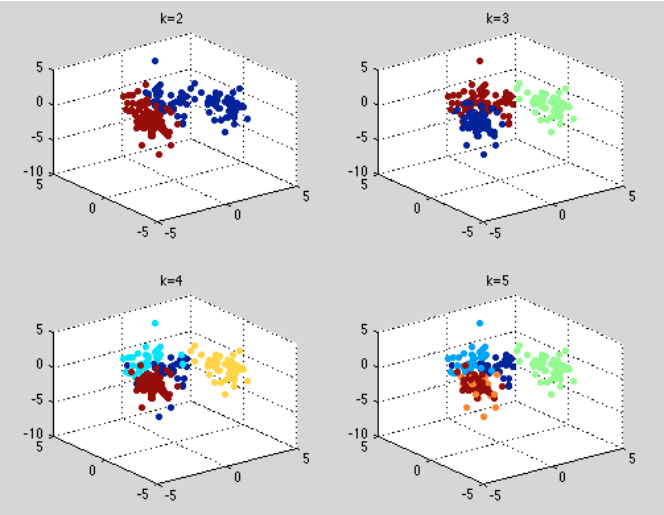

Figure 6: k-means clustering for wines data using PCA

The classification of the K-Means over the reduced dimensionality transformed data is, again, similar to that of the K-Means over the original data. As in the previous cases, this means that the reduced dimensionality has only discarded not relevant eigenvectors, and as such reduced the noise. As in the previous cases, this noise reduction can be seen by using the inner cluster inertia, which is lower than in the original data:

K=2	K=3	K=4	K=5
881.18	510.11	427.05	368.32

Table 6: Inertia of the clusters using the PCA

5.4 Data set: Ionosphere

File: inosphere.arff

This data set is a collection of information obtained from a ground array of antennas that analyzed data from the ionosphere and classified the individuals according to the existence of some structure in the ionosphere (good) and no structure at all (bad), which means the bombarded ions passed through the ionosphere and reached the ground.

The sampled data consists on 34 predictor attributes, all of them continuous and real.

The first step in the analysis of the data will be to try a classification of the individuals by using the K-Means classifier with different values of K. The following plots, figure 7 show the result of this classification.

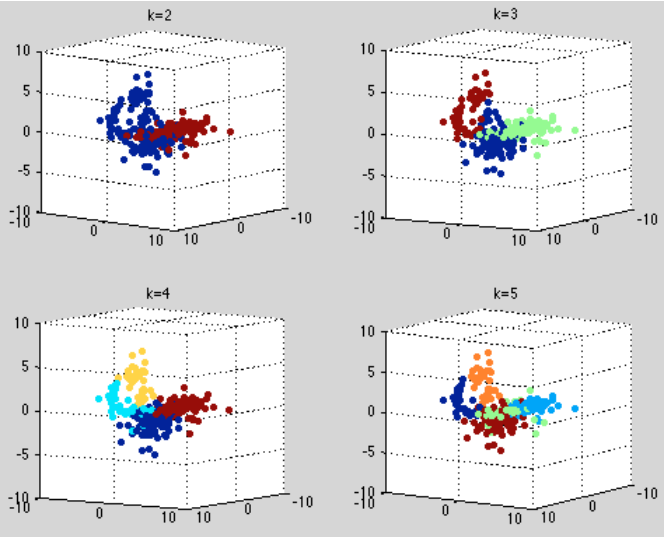

Figure 7: k-means clustering for ionosphere data without use PCA

Apparently, there seems to be a good separation between clusters for all values of K, specially for K=3 and K=4. This can be checked by comparing the differences inertia when K increases:

K=2	K=3	K=4	K=5
9060.10	8238.18	7503.47	7101.62

Table 7: Inertia of the clusters without use the PCA

- Inertia of K=3 with respect to K=2: 91.0%
- Inertia of K=4 with respect to K=3: 91.4%
- Inertia of K=5 with respect to K=4: 94.6%

In this case, the small decrease in inertia in all cases show us that the apparent good classification of clusters with a value of K greater than 2 is just a result of the plot for the three most important dimensions, and thus, there must be some other important dimensions.

The best classification is for a value of K=2.

An analysis by applying the PCA algorithm to the data set tells us that the most important components of the not discarded eigenvectors once applied an eigenvalue threshold of 1 are the 15th, 20th, 3rd, 34th, 1st, 8th and 24th dimensions or attributes, in that order of relevance. As proposed when analyzing the inertia with the K-Means, in this case there are many more important dimensions, a total of 7 compared to the 3 plotted.

If the reduced dimensionality data obtained of the PCA by transforming the data set and then discarding the eigenvectors with a lower eigenvalue than 1 is used to classify the individuals, and then the individuals are plotted in the same coordinates space than before, the result is this one, figure 8

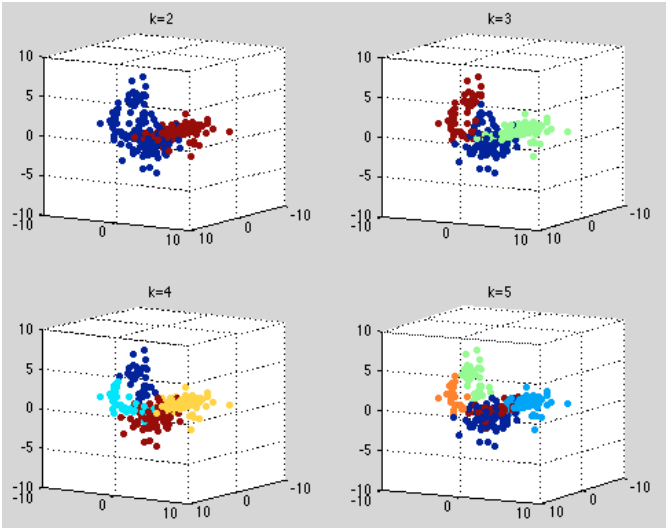

Figure 8: k-means clustering for ionosphere data using PCA

In that case, the K-Means classified the individuals in the same group as before in most of the cases, meaning that the noise has been discarded without affecting much the outcome. It can also be seen, though, that the red cluster for a K value of 5 is smaller, with some of its former individuals moved to the two blue clusters. That is, probably, because once the noise has been deleted, some elements have been moved closer together, and that fifth cluster was taking a range of noisy individuals to itself.

If the new inertia for the K-Means algorithm is calculated, it can be seen that it is reduced, as the noise is lower and the elements are closer together:

K=2	**K=3**	**K=4**	**K=5**
5702.83	4887.81	4174.16	3777.99

Table 8: Inertia of the clusters using the PCA