UNIVERSITY OF IOANNINA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Assignment 2 - Classification and Clustering

Terizi Chrysoula, Student ID: 430

May 23, 2020

Task 2.1. Classification using Bagging and Random Forest

The simulations applied to the previous normalized data set which consists of 1K records. The class labels can receive the value "1" or the value "2". In this task, two additional classification methods, Bagging and Random Forest, must be applied to the dataset for different values of the number of classifiers, where $n\_classifiers = [25, 50, 75, 100]$. The goal of this task is to measure the generalization error of the methods. It is about a measure of how accurately an algorithm is able to predict outcome values for infinite unseen data. Nevertheless, it cannot be computed for an unknown probability distribution. However, an approach to this metric can be calculated for a finite set of data.

2.1.1 Bagging

The Bagging method is a widely used an ensemble learning algorithm in machine learning. The algorithm builds multiple models from randomly taken subsets of train dataset and aggregates learners to build overall stronger learner. To estimate the error of this Bagging classifier, the cross-validation method has been used. In the previous assignment, the value of *k*-fold cross validation was fixed to ten. In this current assignment, the values of cross-validation do not specified, and as a consequence, ten different values of *k*-fold cross validation have chosen to be checked. These are [5, 10, ..., 50] by step 5.

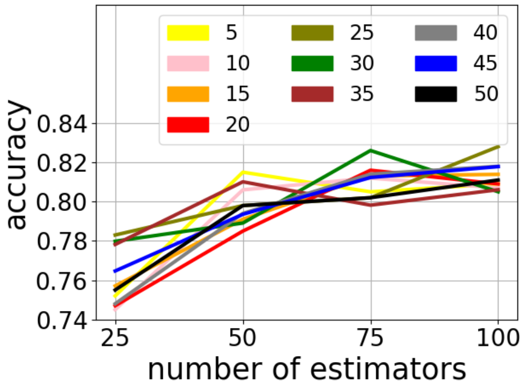


Figure 0.1: Accuracy score of Bagging classifier for different number of classifiers, 25, 50, 75, and 100 and for different *k*-fold cross validation thresholds, [5, 10, ..., 50] by step 5.

Looking at the Figure 0.1, we notice that the worst accuracy score achieved while the number of classifiers is 25, regardless of the *k*-fold cross validation values. The performance of the Bag-

Table 0.1: The best combinations for Bagging classifier, for the hyperparameters *k*-fold cross validation and number of classifiers, which achive high accuracy score.

k-fold cross validation	*#classifiers*	**Accuracy**
25	100	0.828
30	75	0.826

Table 0.2: Accuracy score for the Bagging classifier using *10*-fold cross validation for different values of number of estimators.

k-fold cross validation	*#classifiers*	**Accuracy**
10	25	0.744
10	50	0.805
10	75	**0.812**
10	100	0.807

ging classifier improves as the number of estimators increases. If the number of the classifiers is equal to 100, the accuracy score ranges from 0.80 to 0.82. Furthermore, the higher correctness succeeds in 2 cases (see Table 0.1), the first one is for *25*-fold cross validation in conjunction with 100 classifiers and the second one is for *30*-fold cross validation and 75 classifiers. In addition, focusing on the *10*-fold cross validation (see Table 0.2), as has been applied in the previous assignment, the best score is observed for 75 estimators.

2.1.2 Random Forest

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. To measure the generalization (sampling) error for the Random Forest method, the OOB[1] error is used. The OOB error is the mean prediction error on each training sample X_i, using only the trees that did not have X_i in their bootstrap sample. The simulation estimates the OOB error working on Random Forest algorithm for different values of number of classifiers, $[25, 50, 75, 100]$ and the minimum number of examples in a leaf is equal to 5. We notice that the best OOB error comes through 0.699 with 100 estimators (see Table 0.3).

Table 0.3: OOB score for the Random Forest classifier for different values of number of estimators.

#classifiers	**OOB error**
25	0.642
50	0.663
75	0.652
100	**0.699**

Comparison with the results of the previous classifiers. In the previous assignment, five classifiers were tested, which are as follows, K-Nearest Neighbor, Naïve Bayes, Decision Tree, Linear SVM and SVM using RBF kernel function. In the Table 0.4 is presented the previous accuracy scores and these from this current assignment for the Bagging and Random Forest classifiers. Observing the Table 0.4, we remark that SVM method using RBF kernel function remains at the first place with 0.815 successful correct predictions. In the second place rises the ensemble Bagging

[1] Abbreviation of out-of-bag error.

method with 0.812 score using 100 classifiers. The Decision Tree and Random Forest methods follows with 0.708 and 0.699 accuracy score respectively.

Table 0.4: Accuracy score for each tested classifier. The methods are sorted in descending order based on the accuracy score.

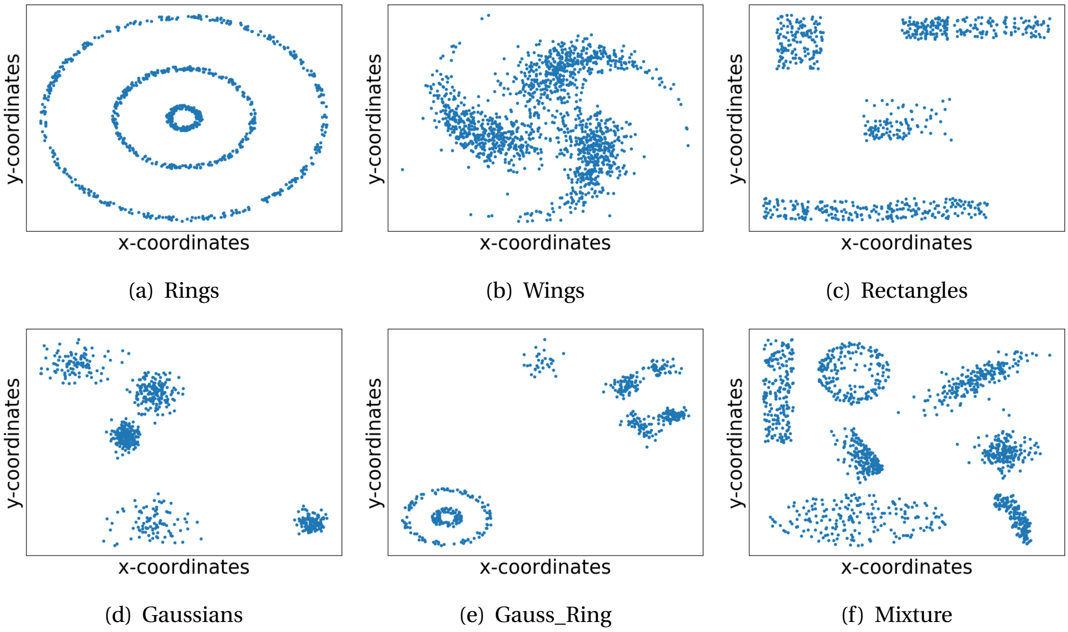
Order	Method	*Hyperparameters*	**Accuracy**
1	SVM RBF	Gamma = 0.01, C = 510k	**0.815**
2	Bagging	10-fold cross validation, Estimators = 100	**0.812**
3	Decision Tree	Max depth = 517, Leaf size = 4	0.708
4	Random Forest	Leaf size (min) = 5, Estimators = 100	0.699
5	KNN	K = 9	0.602
6	SVM Linear	-	0.512
7	Naïve Bayes	-	0.50

Task 2.2. Clustering

A collection of six complex synthetic datasets was given. This task is organized into four basic stages. Initially, a visualization of the datasets is presented and which aims to identify the exact number of the groups (insights into *2.2.1*). We continue by applying three clustering algorithms to the datasets, these are k-means, agglomerative clustering and spectral clustering using RBF kernel (insights into *2.2.2*). The thrird part of the task focuses on finding the best value of a hyperparameter of the spectral clustering algorithm, so that, the best solution is achieved (insights into *2.2.3*). And the last part of the task (insights into *2.2.4*), study the estimation of the best number of clusters for the dataset without ring pattern.

2.2.1 Visualization of the datasets

The initial step of this task is the visualization of the data, so as to recognize the actual number of groups per data set. Looking at the Figure 0.2, we can easily observe the number of groups with naked eye. The Table 0.5 provides information about the actual number of clusters for each one of the given dataset. A difficulty was observed for the *Gauss_Ring* dataset (see Figure 0.2 (e)), where the number of the groups was not clear. And as a consequence, three different numbers of groups are proposed (see Table 0.5), and these are 5, 6 or 7.



(a) Rings
(b) Wings
(c) Rectangles
(d) Gaussians
(e) Gauss_Ring
(f) Mixture

Figure 0.2: Vizualization of the datasets.

Table 0.5: Datasets' number of clusters.

	Rings	Wings	Rectangles	Gaussians	Gauss_Ring	Mixture
#clusters	3	3	4	5	5 or 6 or 7	7

2.2.2 Clustering algorithms simulation and results

In this second step, three different clustering algorithms were tested on the data sets. The selected clustering methods are the *k-means*, *agglomerative clustering* testing *Single* and *Average* links and the last method is the *spectral clustering* testing three different values of *sigma* hyperparameter, 0.10, 0.50 and 1.0. A crucial starting point of the clustering problem, is the initialization of the number of clusters. We assume that we initialize this hyperparameter, *n_clusters*, with the value that has been observed from the previous visualization (see Table 0.5).

About the *Rings* dataset, we observe that the best division is achieved for *Agglomerative* method using *Single* links (see Figure 0.3 (b)). We would except that the *Spectral* method would have the best clustering results, however, this behaviour does not appear (see Figures 0.3 (d)-(f)). We assume that the low *sigma* values is responsible for this bad performance. Furthermore, the *k-means* method yields as we would expect as the algorithm constructs specific linear discriminant hyperplanes (see Figure 0.3 (a)). Also, the algorithm does not work well on non-spherical shaped clusters and on different variance clusters.

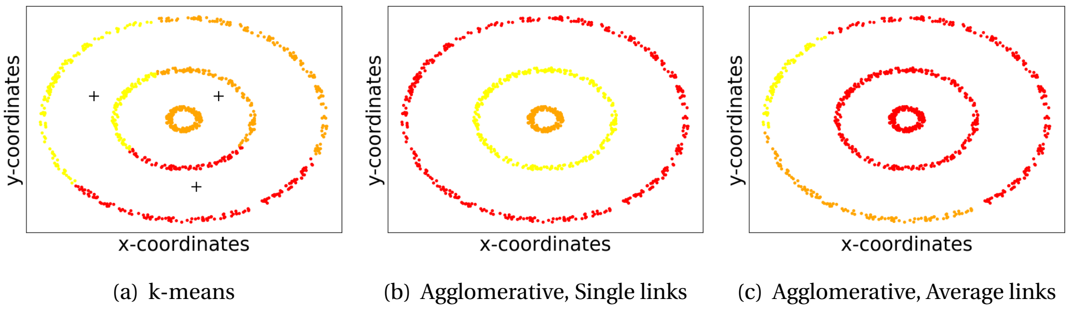
About the *Wings* dataset (see Figure 0.4), we observe that the *Spectral* clustering method (with *sigma* hyperparameter equals to 1.0) performs better that the other two clustering methods (see Figure 0.4 (f)). Moreover, the *k-means* algorithm has good clustering results, nonetheless, the tails of the wings does not cluster correct as they are not close to the center of the same cluster (see Figure 0.4 (a)). Also, the results of the *Agglomerative* clustering method with *Single* links, label the dataset as a single group (see Figure 0.4 (b)). We could explain this behavior due to the fact the this specific algorithm is hierarchical.

About the *Rectangles* dataset, we observe that all three methods work well (see Figure 0.5). More specifically, the *k-means* tends to has a good performance because the groups are not overlapping and any point belongs exclusively to a single cluster (see Figure 0.5 (a)). Further, the *Agglomerative* and *Spectral* clustering methods perform well because each group is dense and remote from the rest groups (see Figure 0.5 (b)-(f)).

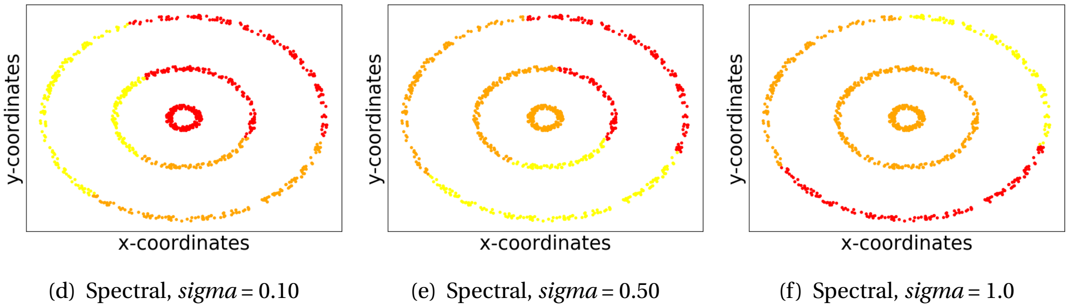
About the *Gaussians* dataset, *k-means* and *Agglomerative* with *Average* links methods perform better (see Figures 0.6 (a) and (c)). The *Spectral* clustering method operates fine, however, some points do not label correct (see Figures 0.6 (d)-(f)).

About the *Gauss_Ring* dataset (see Figures 0.7, 0.8 and 0.9), we expect that the *k-means* method will not be the best clustering method for the *Ring* pattern. With success, we notice this assumption from the visualization (see Figures 0.7 (a), 0.8 (a) and 0.9 (a)), as the *Ring* pattern is divided linearly. We could except that the *Spectral* method would recognize the *Ring* pattern, nevertheless, looking at the Figures 0.7 (d)-(f), 0.8 (d)-(f) and 0.9 (d)-(f), this does not happen. We assume that the low *sigma* values are the cause. Most likely, higher *sigma* values can label the *Ring* pattern correct (more insights into the part *2.2.3*). Furthermore, the *Agglomerative* using *Single* links, labels correct the *Ring* pattern. About the *Gaussians* shapes, we except that the *k-means* algorithm can label correct them and this occurs (see Figures 0.7 (a), 0.8 (a) and 0.9 (a)). Although, *Agglomerative* method using both *Single* and *Average* links (see Figures 0.7 (b)-(c) and 0.8 (b)-(c)), cluster them correct.

About the last *Mixture* dataset, looking at the Figure 0.10, we notice that the *k-means* clustering algorithm has the best label recognition. The shapes of the dataset are dense, well seperated and almost spherical shaped groups (see Figure 0.2 (f)).

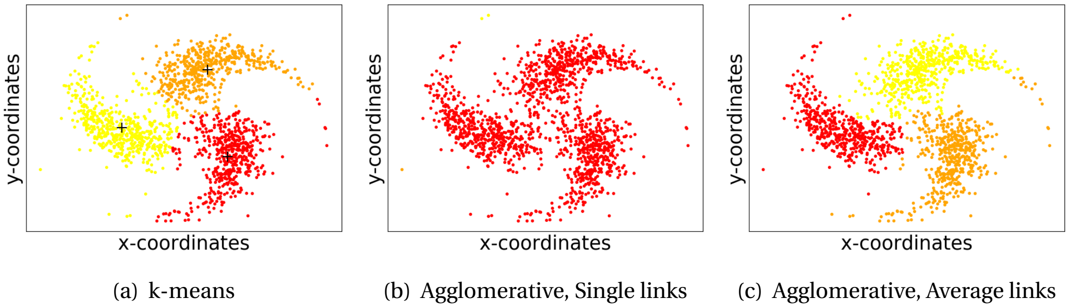


(a) k-means (b) Agglomerative, Single links (c) Agglomerative, Average links

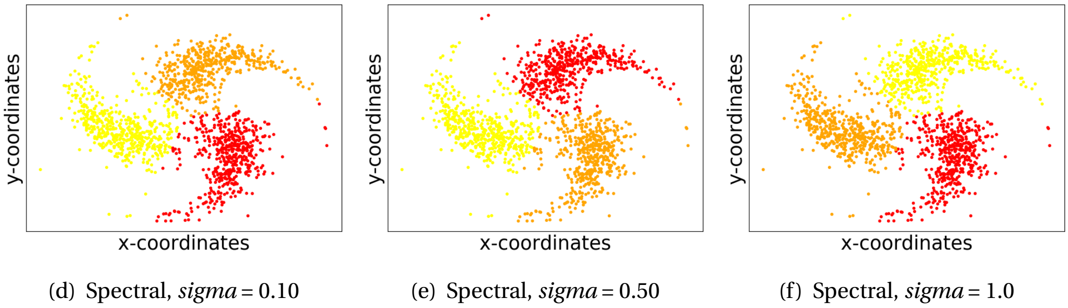


(d) Spectral, *sigma* = 0.10 (e) Spectral, *sigma* = 0.50 (f) Spectral, *sigma* = 1.0

Figure 0.3: Clustering results for *Rings* dataset, *n_clusters* = 3.

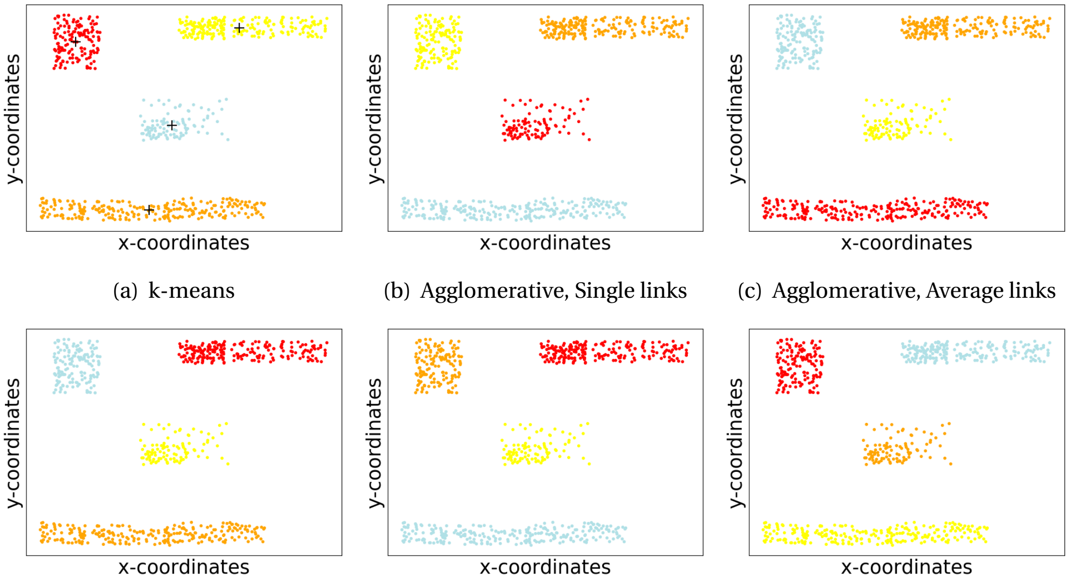


(a) k-means (b) Agglomerative, Single links (c) Agglomerative, Average links



(d) Spectral, *sigma* = 0.10 (e) Spectral, *sigma* = 0.50 (f) Spectral, *sigma* = 1.0

Figure 0.4: Clustering results for *Wings* dataset, *n_clusters* = 3.



(a) k-means

(b) Agglomerative, Single links

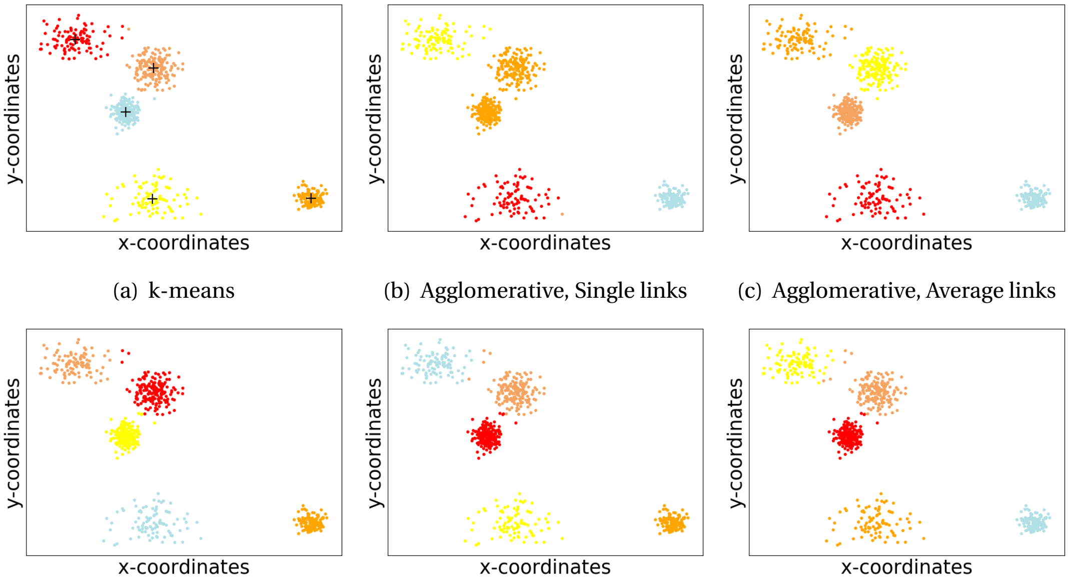
(c) Agglomerative, Average links

(d) Spectral, *sigma* = 0.10

(e) Spectral, *sigma* = 0.50

(f) Spectral, *sigma* = 1.0

Figure 0.5: Clustering results for *Rectangles* dataset, *n_clusters* = 4.



(a) k-means

(b) Agglomerative, Single links

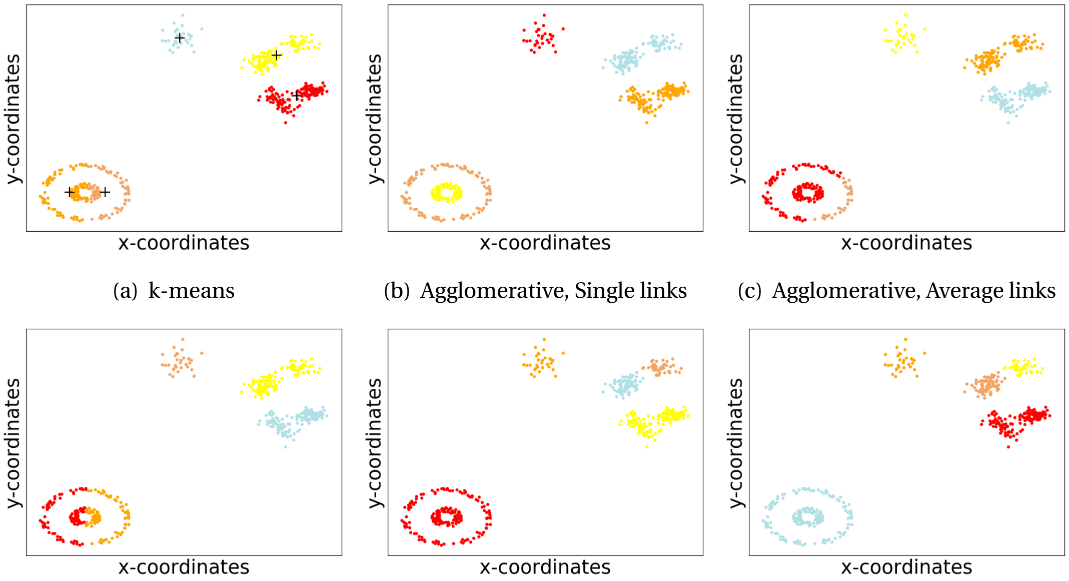
(c) Agglomerative, Average links

(d) Spectral, *sigma* = 0.10

(e) Spectral, *sigma* = 0.50

(f) Spectral, *sigma* = 1.0

Figure 0.6: Clustering results for *Gaussians* dataset, *n_clusters* = 5.



(a) k-means

(b) Agglomerative, Single links

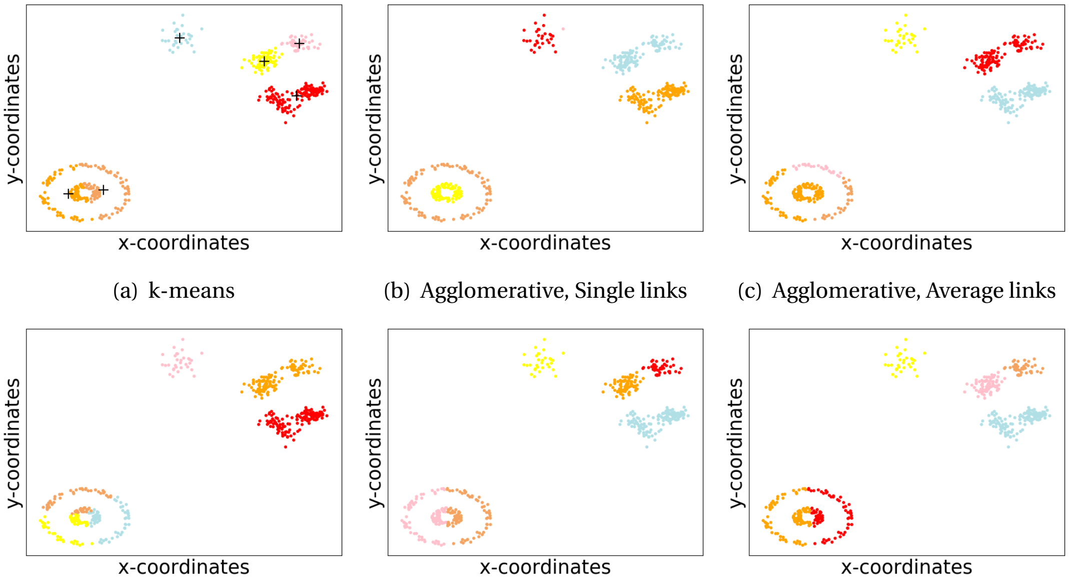
(c) Agglomerative, Average links

(d) Spectral, *sigma* = 0.10

(e) Spectral, *sigma* = 0.50

(f) Spectral, *sigma* = 1.0

Figure 0.7: Clustering results for *Gauss_Ring* dataset, *n_clusters* = 5.



(a) k-means

(b) Agglomerative, Single links

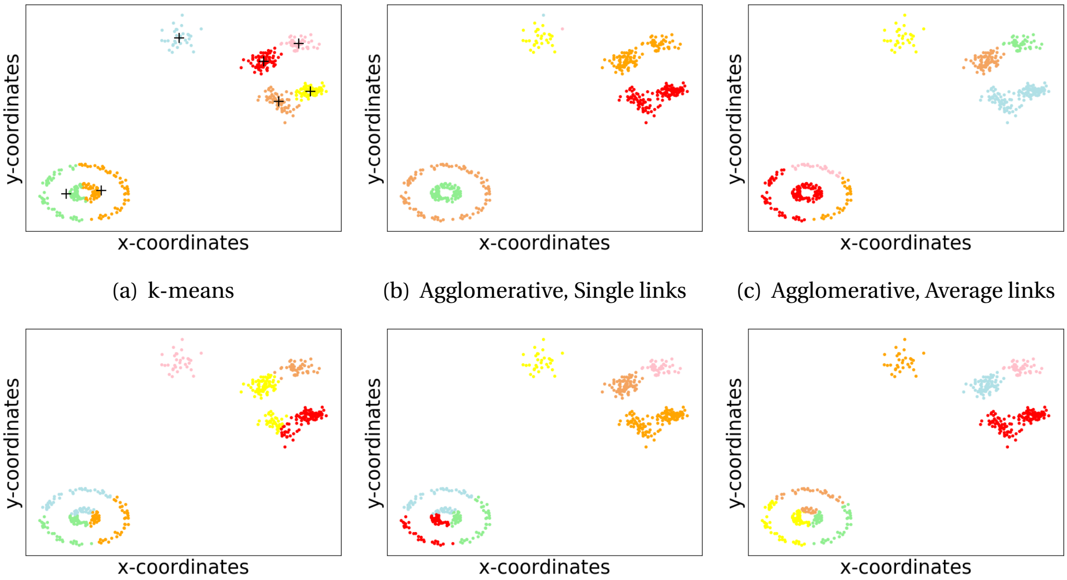
(c) Agglomerative, Average links

(d) Spectral, *sigma* = 0.10

(e) Spectral, *sigma* = 0.50

(f) Spectral, *sigma* = 1.0

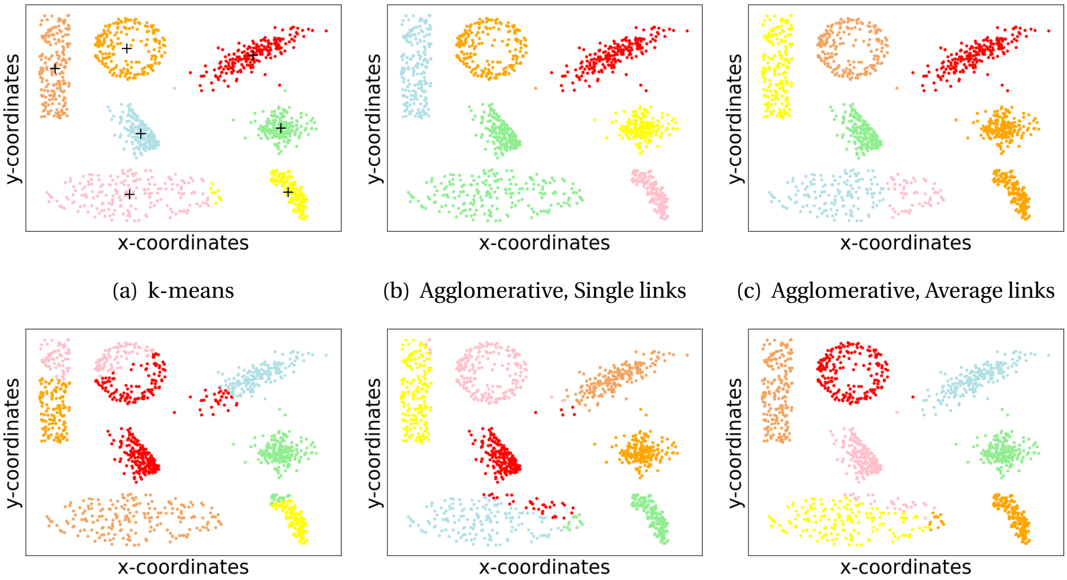
Figure 0.8: Clustering results for *Gauss_Ring* dataset, *n_clusters* = 6.



(a) k-means (b) Agglomerative, Single links (c) Agglomerative, Average links

(d) Spectral, *sigma* = 0.10 (e) Spectral, *sigma* = 0.50 (f) Spectral, *sigma* = 1.0

Figure 0.9: Clustering results for *Gauss_Ring* dataset, *n_clusters* = 7.



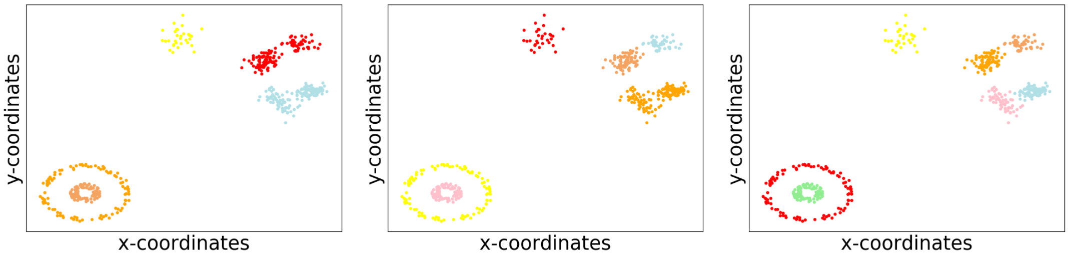
(a) k-means (b) Agglomerative, Single links (c) Agglomerative, Average links

(d) Spectral, *sigma* = 0.10 (e) Spectral, *sigma* = 0.50 (f) Spectral, *sigma* = 1.0

Figure 0.10: Clustering results for *Mixture* dataset, *n_clusters* = 7.

2.2.3 Estimate of the sigma hyperparameter for Gauss_Ring dataset

The *Gauss_Ring* dataset contains a *Ring* pattern of two rings and three or four or five *Gaussians* groups. We tested various *sigma* values from 1.5 to 20.0 by step 0.50. In the case of three *Gaussians* teams, we notice that the best *sigma* values for which they achieve proper grouping are the following intervals, $A = [8.5, 11.5]$, $B = [13, 13.5]$, $C = [14.5, 17.5]$ by step 0.50 and *sigma* = 20. In the case of four *Gaussians* teams, the best *sigma* values range between the intervals, $A = [5.5, 16]$ by step 0.50, *sigma* = 17.0 and *sigma* = 18.5 In the final case, the number of the *Gaussiana* groups is five and the most appropriate *sigma* values range between $A = [5, 6]$ by step 0.50. In the Figure 0.11, we present the clustering results for these three cases and for a selected *sigma* value.



(a) 3 *Gaussians, sigma* = 9.0 (b) 4 *Gaussians, sigma* = 6.0 (c) 5 *Gaussians, sigma* = 5.5

Figure 0.11: Clustering results for *Gauss_Ring* dataset considering the best *sigma* value.

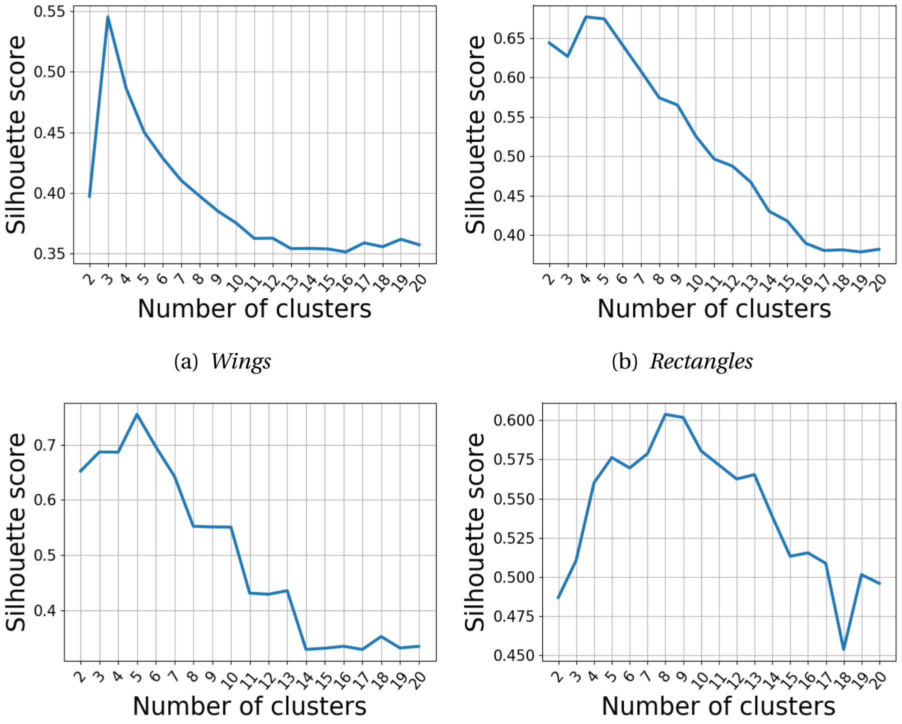
2.2.4 Estimate the n_clusters of k-means using Silhouette score

In the last part of this task, we have to estimate the actual number of the set of examples of *k-means* algorithm which do not contain *rings*. Various initial *n_clusters* values were tested during this simulation, and these range from 2 to 20 number of clusters by step 1.

Focusing on the Figure 0.12, we can find out the actual number of clusters with the highest silhouette score. Easily, we notice that the results of the hyperparameter *n_clusters* match with these from the vizualization (see Table 0.5) for all datasets except the *Mixture* set (see Table 0.6). About the *Mixture* dataset, we assumed that the number of groups is seven, however, we note that the best number of groups is eight with 0.603 silhouette score in contrast with the silhouette score of seven groups which is equal to 0.578 (more insights into Figure 0.12 (d)). Furthermore, we observe that for all data sets, while the number of the clusters increases considerably in relation to the actual number of groups, the silhouette score decreases dramatically. A vizualization of the datasets with the number of clusters which has arisen from the silhouette score presented in Figure 0.13.

Table 0.6: Number of clusters based on the vizualization and the estimate from silhouette score

Dataset	*n_clusters* (visualization)	*n_clusters* (estimate)	**Silhouette score**
Wings	3	3	0.545
Rectangles	4	4	0.676
Gaussians	5	5	0.755
Mixture	7	8	0.603



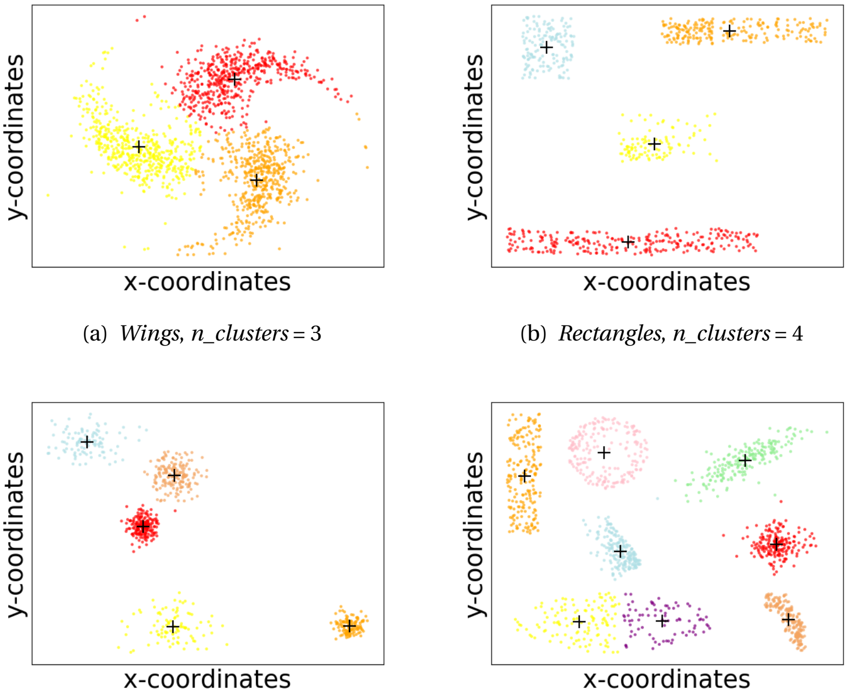
(a) *Wings*

(b) *Rectangles*

(c) *Gaussians*

(d) *Mixture*

Figure 0.12: Plots the Silhouette score for the data sets (without *rings*) while the number of clusters ranges between [2, 20] by step 1.



(a) *Wings, n_clusters* = 3

(b) *Rectangles, n_clusters* = 4

(c) *Gaussians, n_clusters* = 5

(d) *Mixture, n_clusters* = 8

Figure 0.13: Clustering results for the data sets (without *rings*) using *n_clusters* the value which has arisen from the silhouette score.