Home Assignment 2

Javier Galindos Vicente, 214373IV

The student would like to present the work offline in class.

I. Feature selection

Functions implemented:

- JG_fisherScore
- JG_fisherScoreDataset

II. Classification

In this section, the results obtaining for classification using decision trees and knn algorithm are analyzed and compared. The dataset generated for this task is presented in Fig. 1. First of all, the dataset is split in two sets: one for training and another one for testing the goodness of the model. The ratio between the training and testing set is 70%/30%.

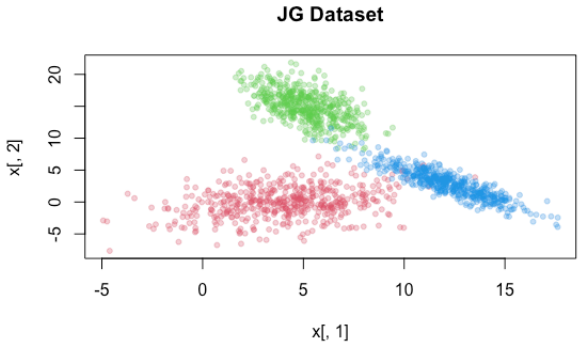

Fig. 1: Generated dataset.

A. K-nearest neighbors

The hyper-parameter k of knn algorithm is found using leave-one-out cross validation, the optimal k is 5. The classification obtained in the testing set are presented in Fig. 2. The misclassified data points are shown as black crosses. The confusion matrix of the testing set is presented in Table I.

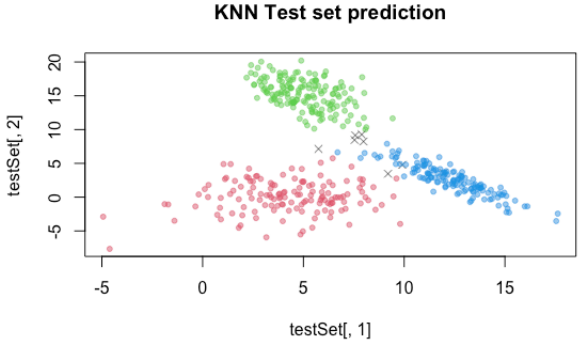

Fig. 2: Predicted output using knn algorithm.

		Ground Truth		
		1	2	3
Predicted	1	136	0	0
	2	0	168	2
	3	2	2	140

TABLE I: Confusion matrix of knn in testing dataset

B. Decision Tree

The decision tree algorithm is designed for numerical datasets. The splitting criteria is based on Gini Index. The tree stops growing after all nodes are leaf nodes (i.e. all instances of a certain node belongs to the same class). The classification obtained in the testing set are presented in Fig. 3.. The confusion matrix of the testing set is presented in Table II.

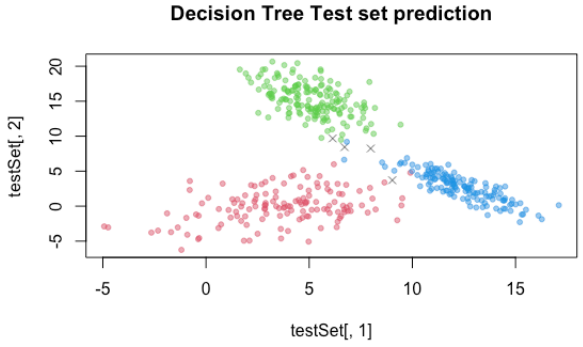

Fig. 3: Predicted output using decision tree algorithm.

		Ground Truth		
		1	2	3
Predicted	1	137	0	0
	2	0	168	1
	3	1	2	141

TABLE II: Confusion matrix of decision tree in testing dataset

The goodness parameters of both algorithm for unseen data (i.e. testing set) are presented in Table III, Table IV , Table V and Table VI. Since the dataset is generated from three Gaussian distributions and there is not complex shapes, both algorithms perform with very high accuracy. Furthermore, due to the dataset is balanced (i.e. same number of instances for each class), precision, recall and F1 score have similar and high values. Overall, both algorithms perform with very high goodness parameters (higher than 95%).

III. Kernel Trick

The first step is to generate a “two half moon” dataset. Towards archiving this task, two subsets based on sinusoidal

Accuracy

KNN	Decision Tree
99.33 %	98%

TABLE III: Accuracy on test set.

Precision

	KNN	Decision Tree
Class 1	100 %	98.6 %
Class 2	100 %	100 %
Class 3	97.9 %	95.2 %

TABLE IV: Precision on test set.

Recall

	KNN	Decision Tree
Class 1	99.2 %	98.6 %
Class 2	98.8 %	96.4 %
Class 3	100 %	99.2 %

TABLE V: Recall on test set.

F1 Score

	KNN	Decision Tree
Class 1	99.6 %	98.6 %
Class 2	99.4 %	98.2 %
Class 3	98.9 %	97.2 %

TABLE VI: F1 score on test set.

functions are created. The subsets are shift to archive the "two half moon" shape. During the generation of the dataset, some random noise have been introduced. The aforementioned dataset is presented in Fig. 4.

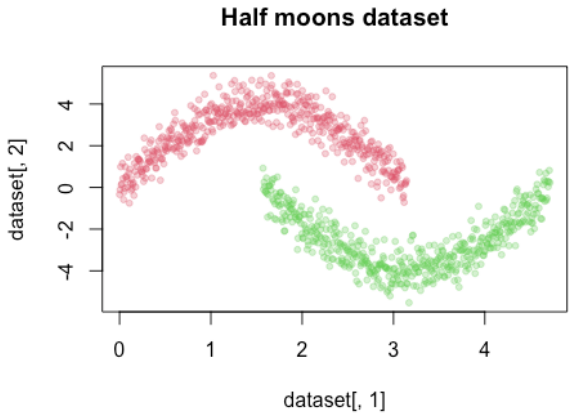

Fig. 4: "Two half moon" dataset.

Afterwards, it is needed to find a function in 2D capable of separating the two clusters ("two moons") to then be projected in 3D and make the clusters linearly separable. To perform this task, four points have been manually selected within the dataset, to find a 3-degree polynomial capable of separate the cluster (observe Fig. 5). This function have been estimated using the lagrange interpolation:

$$f(x) = 2.33x^3 - 17.5x^2 + 39.66x - 26 \quad (1)$$

And projected to 3 dimensions:

$$z(x, y) = 2.33x^3 - 17.5x^2 + 39.66x - 26 - y \quad (2)$$

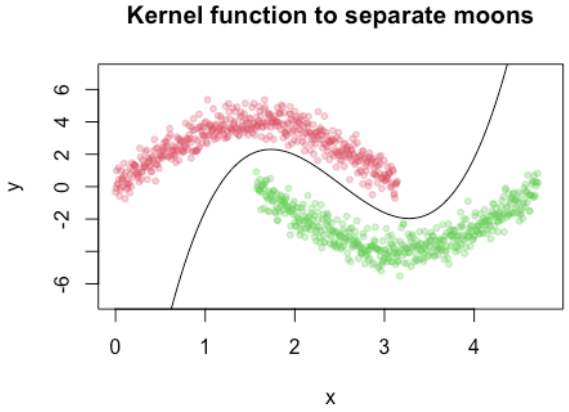

Fig. 5: Function to separate clusters.

With this transformation, the clusters are linearly separable with the hyperplane ($z = 0$) as could be seen in Fig. 6.

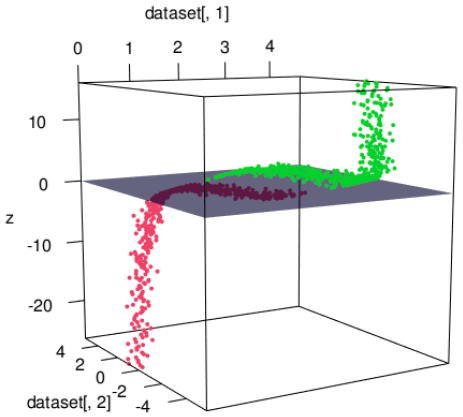

Fig. 6: "Moons" separated by the hyperplane ($z = 0$).

IV. Data preprocessing

In this section, a dataset of 7 features have been created. 4 variables are independent (generated from a random Gaussian distribution), and the other 3 variables are not linear transformations of the independent variables (i.e., dependent variables = square transformation of independent variables). As could be observed in Fig. 7, the non-linear dependencies leads to PCA components that explain similar variance. Thus, PCA does not lead to reduce dimensionality due to a high information loss. Each variable explains approximately 14 % of the variance, thus removing one components would yield to maintain less than the standard 95% of the total variance.

Fig. 7: Variance explained for each PCA component.