Classification (**in-class defense**)

Omar El Nahhas, 214316IV

I. Introduction

The goal of this assignment is to analyse and compare the application of supervised classification methods and validation thereof. A 3D data set with 2400 datapoints and 4 clusters is created to execute the experiments, see Fig. 1. All functions have been developed from scratch in R, except for functions allowed by the teaching staff. Relevant sources for code (besides course slides/code) have been mentioned in the scripts. All self-written functions have been compared and validated with R packages to confirm their correctness.

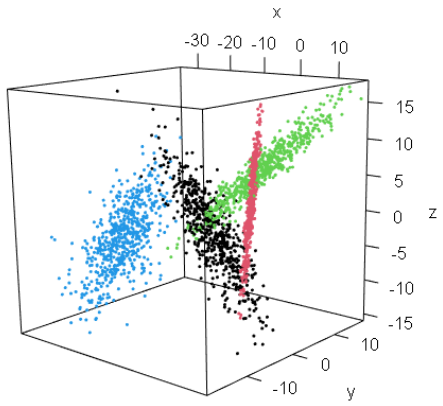


Fig. 1: 3D plot of the generated input data.

II. Feature selection

For feature selection, Fisher Score has been implemented for the 3D dataset, see Table I. The higher the relative Fisher Score, the better the feature performs in discriminating between classes.

TABLE I: Feature selection using Fisher Score.

Feature	Fisher Score
X	5.476
Y	0.946
Z	0.767

Given this dataset, the X and Y feature have the most discriminating power amongst the features.

III. Classification

In this report, two supervised classification methods have been developed: Decision Tree and K-Nearest Neighbours. In order to validate and compare the models, the proposed 3D dataset is randomly split into 90% training data and 10% validation data, resulting in 2160 training samples and 240 validation samples. The model's prediction quality on the validation sample is then measured according to their accuracy, precision, recall and F1-score.

A. Decision Tree classifier

The Decision Tree classifier was implemented and tested with stopping criteria related to the maximum tree depth and minimum information gain. A criteria regarding the minimum size of the leaf was added to 'filter' the best split based on information gain. This was done to prevent ending up with too small data splits, helping avoid overfitting of the model. The experiment has been run with a maximum depth of 20, a minimum leaf size of 5 and a minimum information gain per split of 1e-5. This implementation resulted in an overall accuracy of 92.9% on the validation data with corresponding metrics in Table II, confusion matrix in Table III and misclassification plot in Fig. 2.

TABLE II: Decision Tree metrics with 92.9% accuracy.

Metric	Class 1	Class 2	Class 3	Class 4	Overall
Precision	0.9492	0.8689	0.9153	0.9836	0.9292
Recall	1.0000	0.9138	0.8182	1.0000	0.9330
F1-score	0.9739	0.8908	0.8640	0.9917	0.9301

TABLE III: Decision Tree confusion matrix.

		Actual			
		1	**2**	**3**	**4**
Predicted	**Class 1**	56	0	3	0
	Class 2	0	53	8	0
	Class 3	0	5	54	0
	Class 4	0	0	1	60

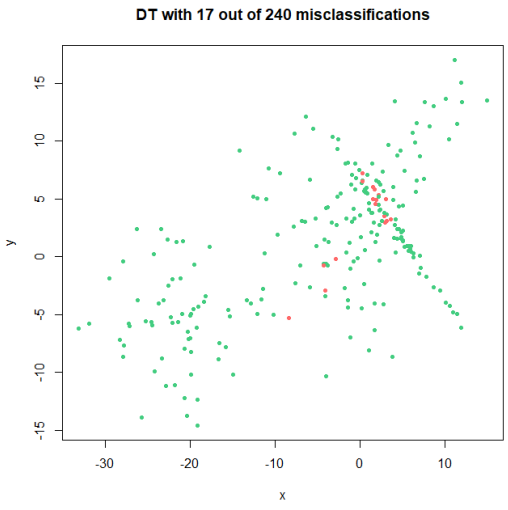


Fig. 2: Misclassification plot of the Decision Tree classifier.

B. K-Nearest Neighbours classifier

The K-NN classifier was implemented on the dataset, where the optimal K-neighbours is found by brute-force training the model with all possible K-neighbours (within a defined limit) to find the optimal K. The experiment has been run with a optimal K of 3 on the validation set. This implementation resulted in an overall accuracy of 96.7% on the validation data with corresponding metrics in Table IV, confusion matrix in Table V and misclassification plot in Fig. 3.

TABLE IV: 3-NN metrics with 96.7% accuracy.

Metric	Class 1	Class 2	Class 3	Class 4	Overall
Precision	0.9655	0.9194	0.9833	1.000	0.9671
Recall	1.0000	0.9828	0.8939	1.000	0.9692
F1-score	0.9825	0.9500	0.9365	1.000	0.9672

TABLE V: 3-NN confusion matrix.

		Actual			
		1	**2**	**3**	**4**
Predicted	**Class 1**	56	0	2	0
	Class 2	0	57	5	0
	Class 3	0	1	59	0
	Class 4	0	0	0	60

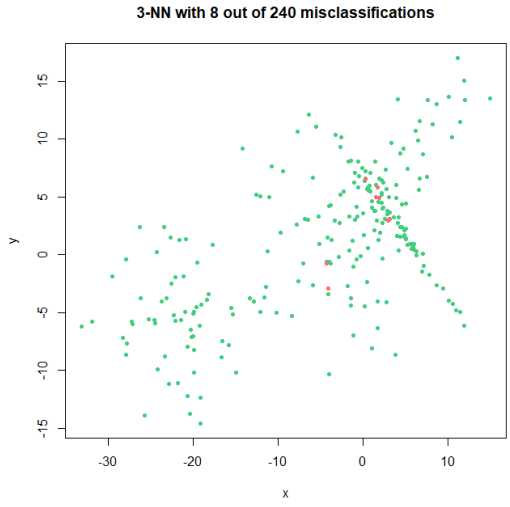


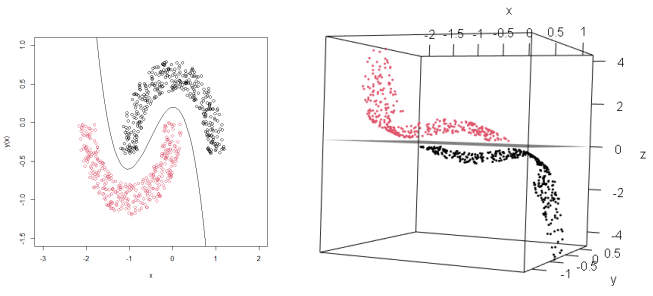
Fig. 3: Misclassification plot of the 3-NN classifier.

IV. Kernel trick

The proposed polynomial kernel to separate the 2D two moon data cluster is the 3D plane in Eq. 1 and the 2D line in Eq. 2. The separation is visualised in 2D in Fig. 4a and in 3D in Fig. 4b. The polynomial kernel is found using the Lagrange polynomial interpolation and using four points between the two moon clusters to cross through.

$$z = -1.379x^3 - 2.113x^2 + 0.066x + 0.2 - y \quad (1)$$

$$y = -1.379x^3 - 2.113x^2 + 0.066x + 0.2 \quad (2)$$



(a) 2D (b) 3D

Fig. 4: 2D and 3D representation of the polynomial kernel.

V. Data pre-processing

According to the theory, if PCA does not work correctly, the features have non-linear relationships, or no relationship at all. Therefore, a dataset is proposed with the first 5 features as independent features generated by a random normal distribution, where the other 3 features are made dependent on the random normal distributed features with a non-linear relationship. The full dataset can be reproduced using Eq. 3, 4, 5 and 6 with seed 1337. As a result of this dataset, the variance per Principal Component is nearly equal (see Fig. 5).

$$Feature_{1-5} = random\ normal\ distribution \quad (3)$$

$$Feature_6 = Feature_1 * sin(Feature_1) \quad (4)$$

$$Feature_7 = Feature_1 * cos(Feature_1) \quad (5)$$

$$Feature_8 = Feature_1 * tanh(Feature_2) \quad (6)$$

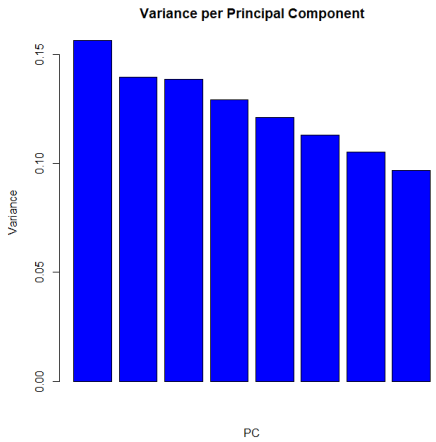


Fig. 5: Principal Components showing near-equal variance.

The five random normal distributions show as circular-shaped plots with nearly equal variance across the radius, making it nearly impossible for PCA to find an axis with the most variance which would improve the interpretability of the data. As a result, the dataset includes 5 non-related variables and 3 non-linearly related variables. Therefore, the dimension of this dataset cannot be reduced, as the explained variance in the dataset would be below the recommended 95% threshold when removing a single Principal Component.