Projet MEDICHAL

Groupe : Africa
Membres : PATAULT Paul, FERREIRA Jules, NDAO Ndieme, ISKOUNEN Feriel, ESTEVAN Benjamin et SERRE Gaëtan
URL du challenge : https://codalab.lri.fr/competitions/625
Repo Github : https://github.com/LE10EENFAIT/Africa

I. Description du problème [0]

Le challenge MEDICHAL consiste à détecter si une cellule est infectée par le parasite de la malaria grâce à une photo de la cellule en question.
Nous avons choisi ce challenge pour l'aspect médical du projet. Nous pensons que c'est une des plus belles utilisation du machine learning.
Pour cela nous disposons d'un jeu de données contenant 16 534 images en couleur. Chaque images contient 19 features tel que le nombre de pixels noir ou bien la couleur moyenne.
La métrique du challenge est l'aire sous la courbe ROC (Receiver Operating Characteristic)
Pour réaliser ce projet, nous avons séparé notre groupe en trois sous-groupes. Un binôme s'occupera du preprocessing de jeu de données, c'est à dire la transformation des données brutes en données utilisables plus efficacement par le classifier (ex : la sélection de features) ; un binôme se chargera du choix du modèle, du classifieur et l'entraînement de ce dernier avec les données de test ; enfin le dernier binôme s'occupera de l'affichage des résultats de manière compréhensible et pertinente.

II. Description des sous-groupes

A. Preprocessing (FERREIRA Jules et ESTEVAN Benjamin)

Nous avons filtré les données de l'ensemble d'entraînement grâce à la forêt d'arbres décisionnels (Random forest). Ainsi nous avons pu nous débarrasser des données aberrantes. Nous avons ensuite analysé la variance des différentes features pour savoir lesquelles allaient être conservées après avoir fait une PCA (Principal Component Analysis) [1] et nous les avons comparées aux features conservées après sélection par le modèle ExtraTreesClassifier [2] (schéma 2, 3 et 4).
Après avoir ajusté les paramètres des deux modèles de façon à ne conserver dans les deux cas que 4 features, nous avons comparé les scores de cross validation obtenus dans les 2 cas.
Ainsi le score de cross validation en ne conservant que 4 features ne dépasse pas 0,73 avec la PCA alors qu'on arrive à un score de 0,95 avec le modèle ExtraTreesClassifier de sélection de features, les features conservées par les deux modèles n'étant bien évidemment pas les mêmes. Nous avons donc décidé de n'utiliser que le modèle ExtraTreesClassiflier puisqu'il ne nécessite

que 4 features pour atteindre un score de 0,95 là où la PCA en nécessite 7 pour arriver à un score de 0,94 avec les données analysées.

B. Modèle (PATAULT Paul et SERRE Gaëtan)

Nous avons testé cinq modèles utilisant la cross-validation sur vingt ensembles de test pris aléatoirement dans l'ensemble des données fournies par le sous-groupe preprocessing. Nous remarquons que les données sont relativement simples à séparer étant donné que le score moyen pour chaque modèle est toujours supérieur à 55%. Cependant, un modèle sort du lot : la forêt d'arbres décisionnels (Random forest (schéma 1)) [3] avec un score moyen de 95% (voir schéma 1).

Nous avons donc choisi d'utiliser une random forest pour séparer nos données. Ensuite, grâce à la méthode RandomizedSearchCV [4] de la librairie sklearn, nous avons pu estimer les meilleurs hyper-paramètres pour notre random forest afin d'avoir le plus grand score possible et ainsi éviter le sur-apprentissage.

C. Visualisation (ISKOUNEN Feriel et NDAO Ndieme)

Nous avons voulu bien explorer nos données pour cela nous avons commencé par utiliser un algorithme de clustering, nous avons choisi l'algorithme k-mean pour pouvoir séparer notre ensemble de données en sous-groupes distincts et nous avons voulu visualiser le résultat en fonction de min_gray et max_gray qui sont d'après la matrice de corrélations les features les plus corrélés avec le fait qu'une cellule soit infectée. Nous avons ensuite fait une analyse en composantes principales pour réduire les dimensions des données et pouvoir mieux visualiser celles-ci et nous les avons affichées en fonction des composantes principales (schéma 5). Ajouté à cela, nous avons aussi affiché la surface de décision d'un arbre de décision formé sur des paires de features [5] (schéma 6). Comme ces deux résultats ne permettaient pas discriminer les deux classes, nous avons utilisé l'algorithme de T-SNE de sklearn [6]. Nous avons essayé cet algorithme sur l'ensemble des données avec 19 features (schéma 7) et sur nos données avec 4 features après notre preprocessing (schéma 8). On remarque que les données "parasitized" et "uninfected" forment deux groupes bien distincts uniquement sur les données avec 4 features.

III. Résultat préliminaire

Grâce aux travaux de chaques groupes, nous avons réduit le nombre de features passant de 19 à 4 tout en gardant un score au moins similaire sinon supérieur (~95%) et nous avons décidé d'utiliser une random forest pour séparer nos données.

IV. Tests & Pseudo-Code

Nous avons réalisés plusieurs tests unitaires pour chaque groupe. Toutes nos fonctions ont été appelées et nous avons réalisé plusieurs exemples en appelant sklearn en dehors de nos fonctions et vérifiant que les deux retournent le même résultat.

Pour ce qui est du pseudo-code, nous ne sommes pas en mesure d'en fournir, étant donné que la grande majorité de notre code sont des algorithmes implémentés dans sklearn, que nos connaissances actuelles ne nous permettent pas de comprendre toutes les subtilités.

V. Annexe

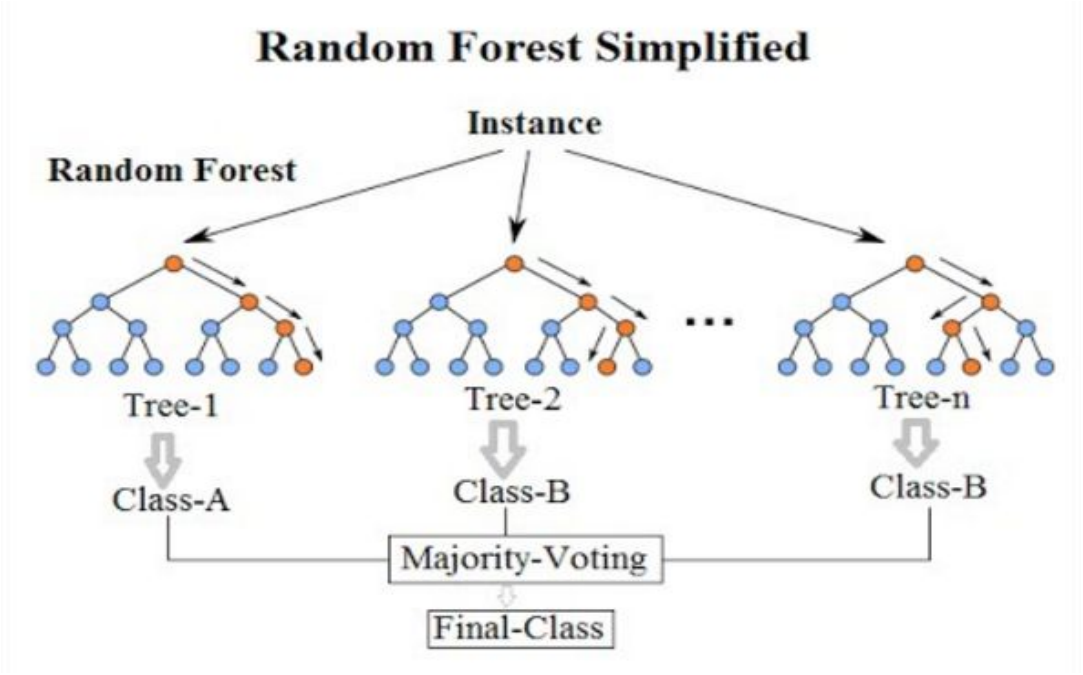

Schéma 1 : Schéma simplifié du fonctionnement d'une random forest

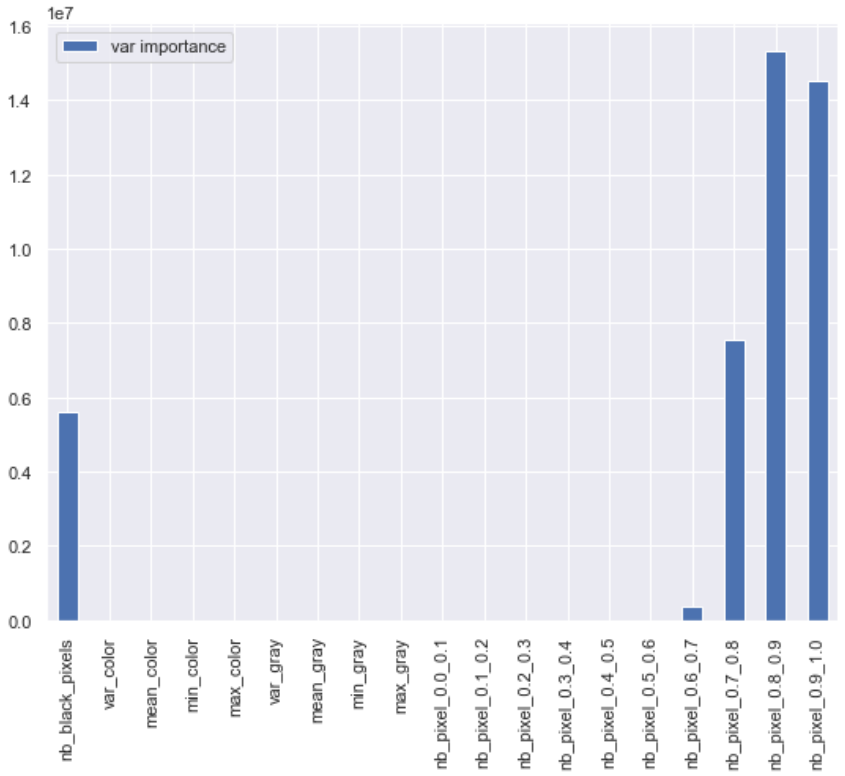

Schéma 2 : Histogramme de la variance pour chaque feature

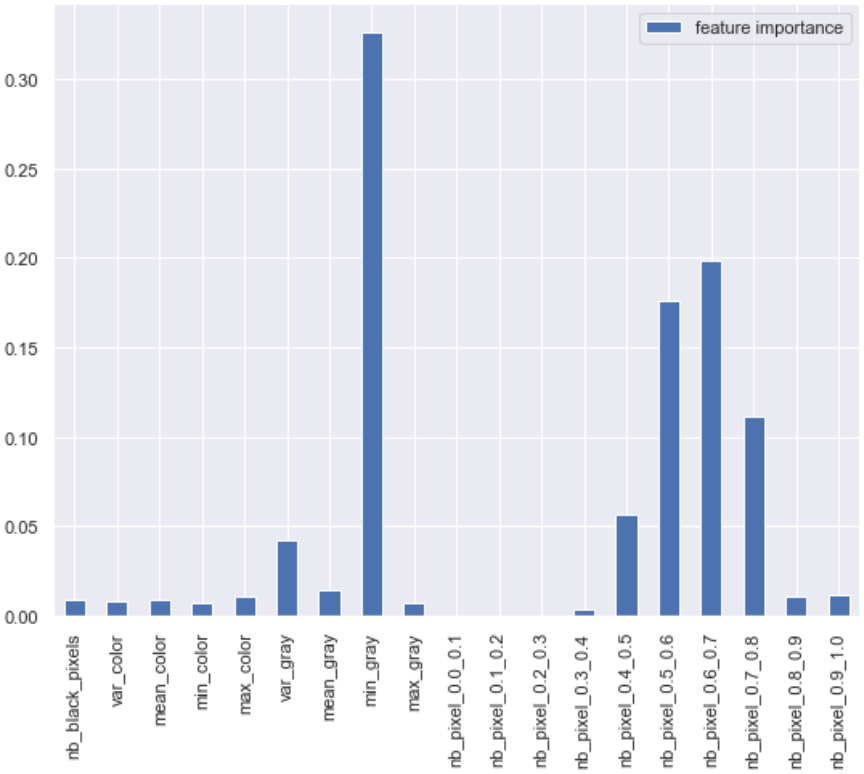

Schéma 3 : Histogramme de l'importance de chaque feature

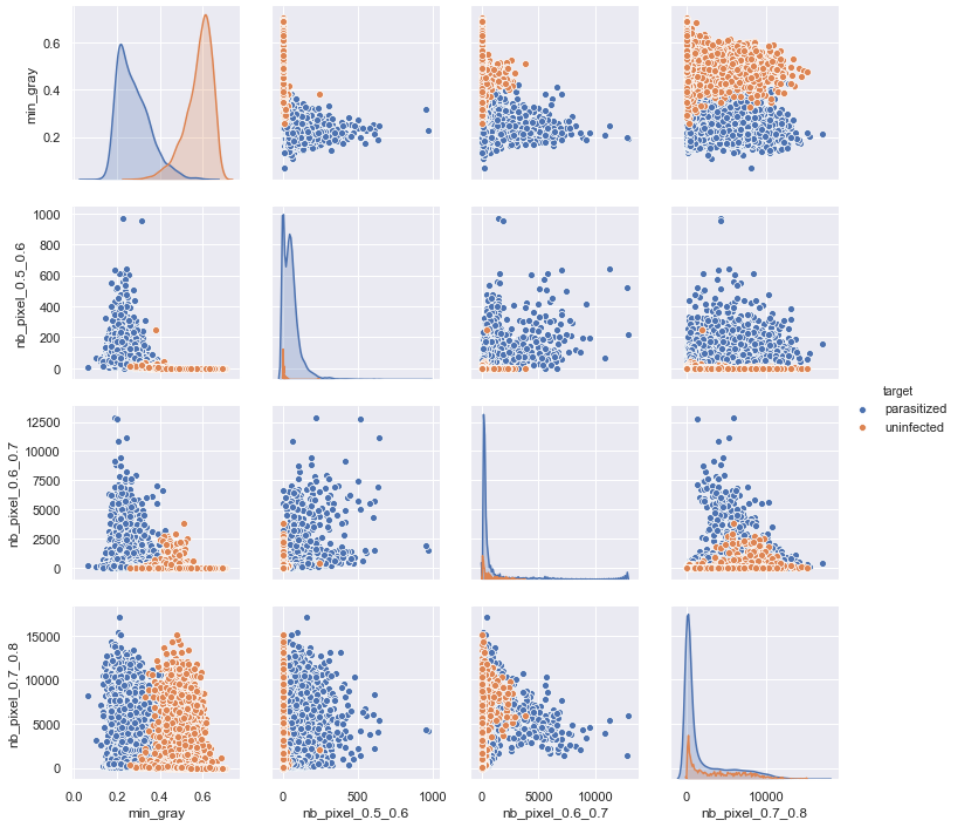

Schéma 4 : Schémas comparatifs des cellules saines et infectées par paires de features

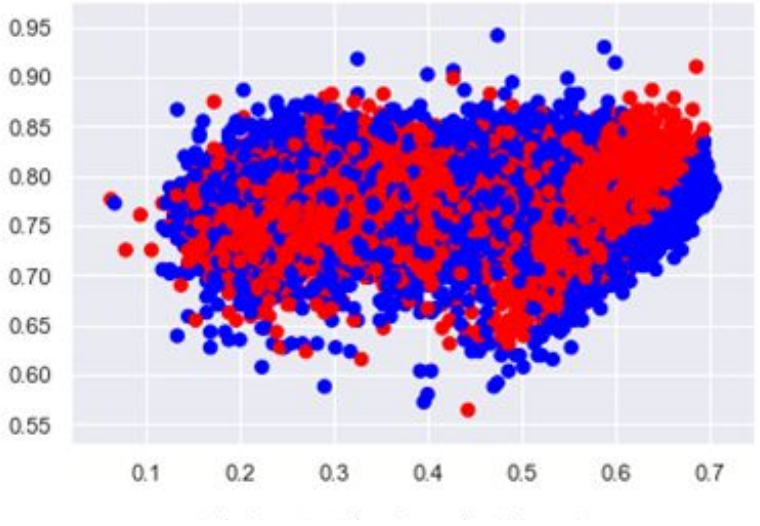

Schéma 5 : Résultat du Clustering

Rouge : parasitized

Bleu : uninfected

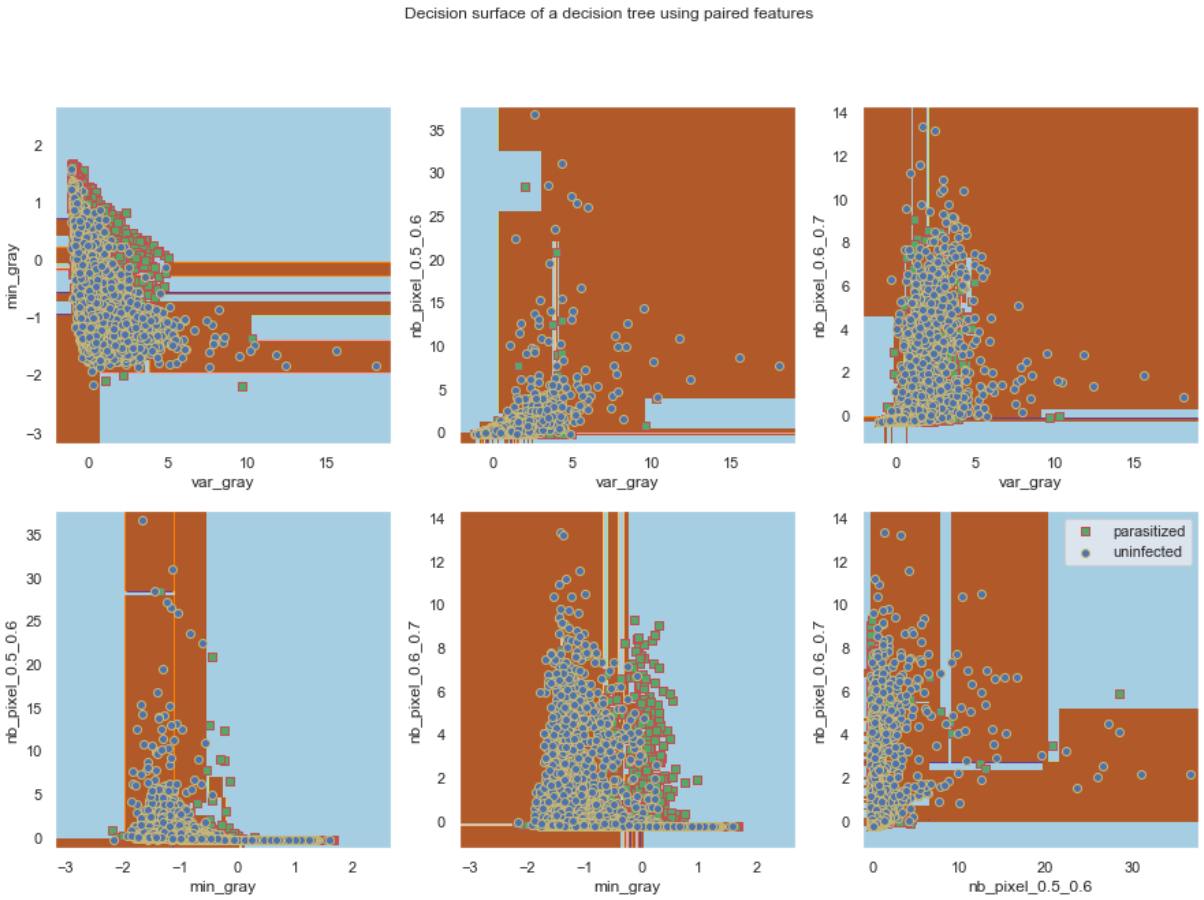

Schéma 6 : Surface de décision d'un arbre de décision formé sur des paires de features
Bleu : surface parasitized
Marron : surface uninfected

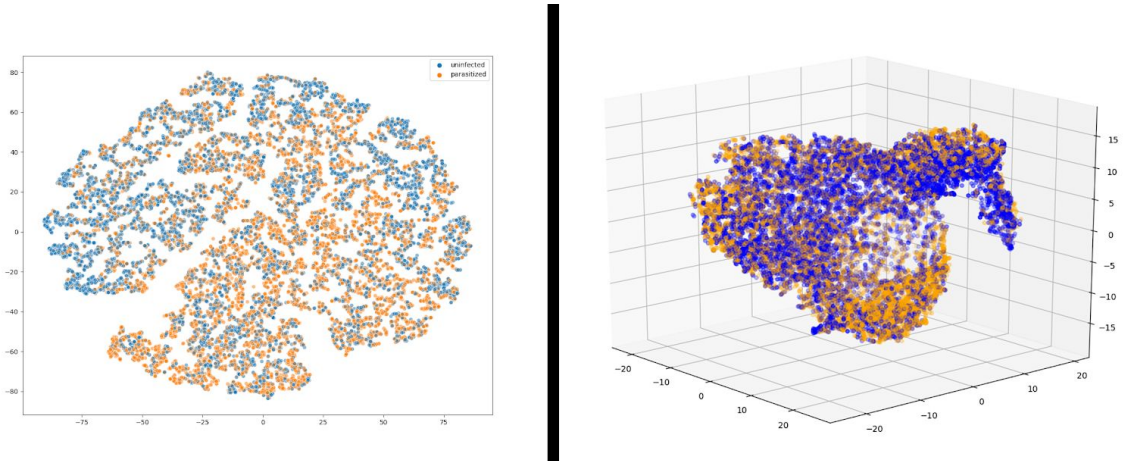

Schéma 7 : T-SNE 2D (gauche) T-SNE 3D (droite) sur données avec 19 features
Orange : parasitized
Bleu : uninfected

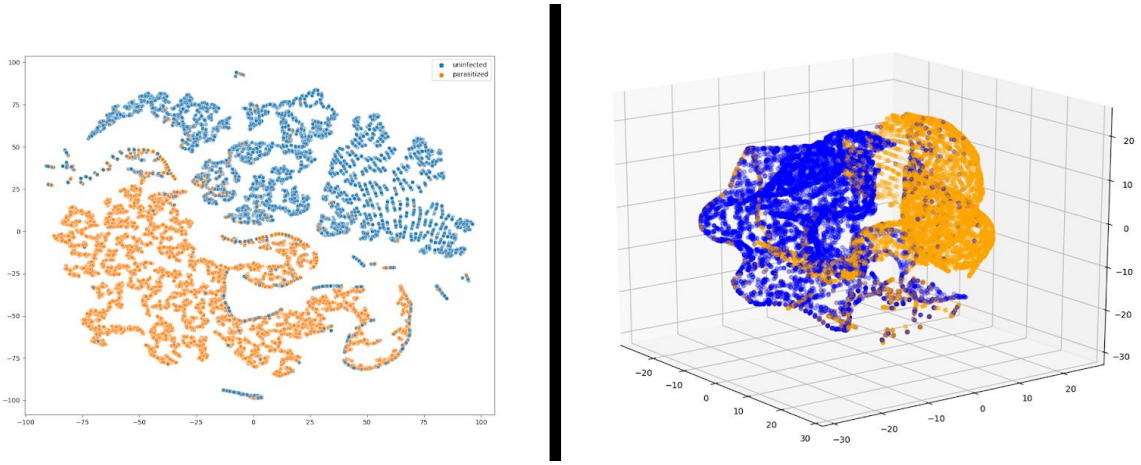

Schéma 7 : T-SNE 2D (gauche) T-SNE 3D (droite) sur données avec 4 features
Orange : parasitized
Bleu : uninfected

Tableau 1: Informations sur les données

Dataset	*Num. Examples*	*Num. Variables/Features*	*Has categorical variables?*	*Has missing Data?*	*Num. examples in each class*
Training	*16534*	*4*	*No*	*No*	*8276 : Parasitized* *8258 : Uninfected*
Valid(ation)	*5512*	*4*	*No*	*No*	*Unknown*
Test	*5512*	*4*	*No*	*No*	*Unknown*

Tableau 2: Renseignement des résultats préliminaires

Method	*RandomForest*	*Neural Network*	*KNeighbors*	*Naive Bayes*	*Perceptron*
Training	*0.006*	*0.39*	*0.12*	*0.1*	*0.05*
CV	*0.06*	*0.38*	*0.11*	*0.09*	*0.07*
Valid(ation)	0.0514	0.4339	0.0675	0.0845	0.1055

On renseigne ici le taux d'erreur (sur [0, 1], avec 0 comme valeur souhaitable) de nos modèles sur les différents ensembles de données à prédire. On remarque ici que RandomForest est le meilleurs sur tous les jeux de données. De plus, les autres modèles ne sont pas nécessairement meilleurs sur l'ensemble d'entraînement que sur les autres.

VI. Références

[0] Cours de L2 « Mini-projet ». Isabelle Guyon. 2020

[1] sklearn PCA : https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

[2] sklearn ExtraTreeClassifier :
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html

[3] sklearn RandomForest :
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[4] sklearn RandomizedSearchCV :
https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html

[5] sklearn plot decision surface of a decision tree :
https://scikit-learn.org/stable/auto_examples/tree/plot_iris_dtc.html

[6] sklearn TSNE : https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html