UFR des Sciences
CAMPUS DE VERSAILLES

Université Paris Saclay site de Versailles

Machine Learning
Rapport de projet

SUPPORT2 dataset UCI

Etudiants :
Abdeldjalil SMAHII
Oussama BOUAKAZ

Enseignant :
Mme. Marie SZAFRANSKI

6 novembre 2023

1 SUPPORT2 Dataset

1.1 Rappel général sur les données

Le dataset **SUPPORT2** est un ensemble de données medico-légales qui cobntient des informations sur les patients admis dans des unités de soins intensifs aux USA entre 1989 et 1994 dans le cadre d'un projet SUPPORT *Study to Understand Prognoses and Preferences for Outcomes and Risks of Treatments*. Le dataset SUPPORT2 contient des informations sur les patients telles que, leurs données démographiques, médicales et des informations sur les préférences de traitement. Les informations sur les traitements comprennent des données sur les traitements médicaux, les interventions chirurgicales et les traitements de soutien. Les informations sur les résultats comprennent des données sur le décès, la durée du séjour à l'USI et la qualité de vie. ces informations sont distribuées sur 45 colonnes dont 3 font un role d'une varibale target (varibale dependanteà à prédire) et 42 variables indépendantes (features). Dans le cadre de notre problématique supervisée nous avons considéré que la target **death** pour pour faire une classification binaire, si le patient mourra ou surivivra. Vous trouvez dans le notebook en détail la description de chaque variables indépendantes (age, sex, dzclass, dzgroup, etc...).

1.2 Séléction des caractéristiques (Features Selection)

La sélection des caractéristiques est un processus itératif qui vise à identifier un sous-ensemble de variables pertinentes à partir des caractéristiques du dataset. Dans notre cas nous avons procédé de plusieurs méthodes de selection des caractéristiques afin de réduire le nombre de features :

1. Lecture de descriptif du dataset
2. Suppression des colonnes fortement corrélées
3. Niveau de détail de l'information
4. Utilisation des techniques PCA, Package Feature Selection du Sklearn
   — Analyse des Composantes Principales (PCA)
   — Sélection de K-meilleures Variables
   — Sélection en se basant sur un Classifier
   A la fin de ces trois méthodes, nous faisons **une intersections** des caractéristiques afin d'avoir un ensemble de features plus restreints.

1.3 Problématique supervisé

1.3.1 Construction des modèles predéctifs (Models construction)

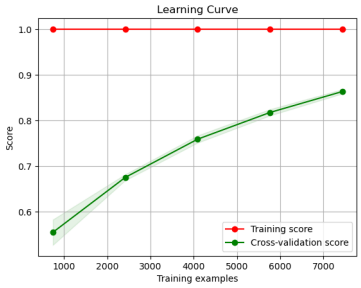
Support Vector Machine

L'objectif de l'algorithme SVM est de trouver l'hyperplan qui maximise la marge entre les classes. La marge est la distance entre l'hyperplan de séparation et le points de données les plus proches de chaque classe. Pour les problèmes non-linéairement séparables, trouver un hyperplan qui sépare les données est difficile. Afin de contourner ce problème, l'algorithme applique une transformation sur les données nommé 'Mapping to Higher Dimension' sur les données ce qui permet de passer à une dimension superieur. La méthode consiste à

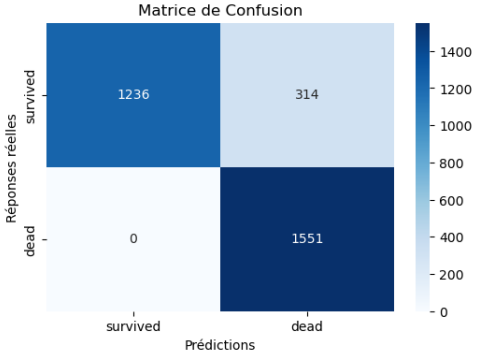
appliquer une transformation non-linéaire aux données d'entrée (les caractéristiques), afin de les projeter dans un espace de dimension plus élevée où elles seront plus facilement séparables. Cette transformation peut être effectuée en utilisant différentes fonctions de noyau (kernel functions).

les hyperparamètres optimisés du modèle :
{'C' : 1, 'coef0' : 0.0, 'decision_function_shape' : 'ovo', 'gamma' : 10, 'kernel' : 'rbf'}

Résultats de ce modèle :



(a) Courbe d'apprentissage



(b) Matrice de confusion du modèle

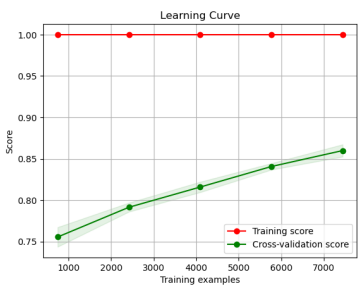
Figure 1 – Résultats du modèle SVM

XGBoost

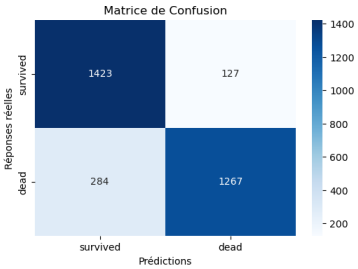
Cet algorithme est basé sur la technique de **Boosting** qui est une technique d'apprentissage ensembliste qui vise à combiner un ensemble d'algorithmes simple afin d'obtenir un algorithme puissant, chaque algorithme est subi d'être en underfitting et la séquence des algorithmes fait que chacun compense les faiblesse de son prédécesseur. Dans le cas de **XGboost** les algorithmes simples sont des **arbres de décision**. Chaque arbre de décision est entraîné sur les données d'entraînement, et ses prédictions sont ensuite combinées pour obtenir une prédiction finale. XGBoost utilise une technique de **gradient boosting** pour combiner les arbres de décision. Le gradient boosting consiste à ajuster chaque arbre de décision de manière à minimiser l'erreur de prédiction du modèle.

les hyperparamètres optimisés du modèle : {'colsample_bytree' : 0.7, 'gamma' : 0.05, 'learning_rate' : 0.1, 'max_depth' : 7, 'min_child_weight' : 1, 'n_estimators' : 500, 'reg_alpha' : 0.05, 'reg_lambda' : 0, 'subsample' : 0.8}

Résultats de ce modèle :



(a) Courbe d'apprentissage



(b) Matrice de confusion du modèle

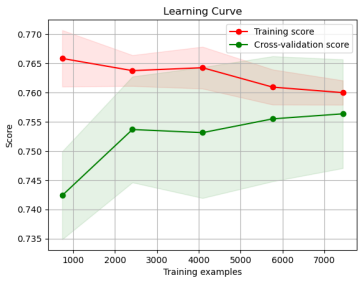
Figure 2 – Résultats du modèle XGBoost

LogisticRegression

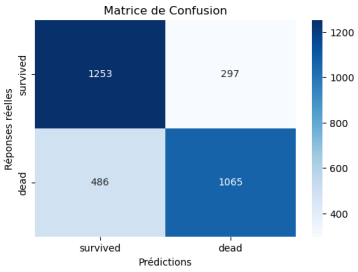
Cet algorithme est basé sur un modèle statistique qui permet de prédire des variables catégorielle en fonction des variables dependantes, cet algorithme utilise une fonction logistique **(fonction sigmoide)** pour convertir les prédictions en probabilités qui prend des valeurs entre 0 et 1.

les hyperparamètres optimisés du modèle : {'C' : 0.3593813663804626, 'penalty' : 'l2', 'solver' : 'liblinear'}

Résultats de ce modèle :



(a) Courbe d'apprentissage



(b) Matrice de confusion du modèle

FIGURE 3 – Résultats du modèle Regression Logistique

Comparaison des méthodes:

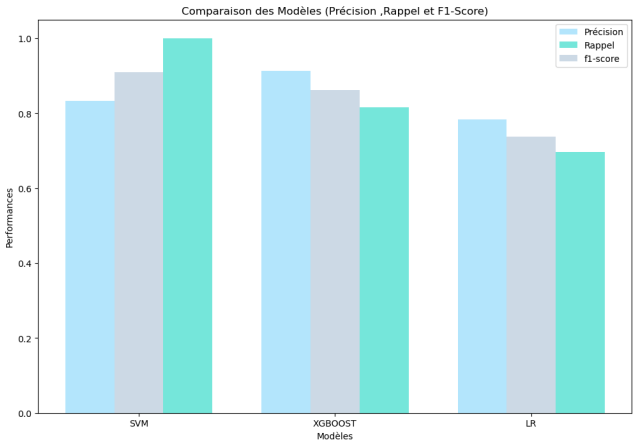


Figure 4 – Comparaison entre F1-Score, Rappel, Précision des modèles

Le modèle XGBoost a la précision la plus élevée parmi les trois modèles, et cela indique qu'il identifie les observations positives réelles tout en évitant les faux positifs (minimise le nombre de faux positifs), mais son rappel est faible par rapport au SVM qui a réussi d'avoir le meilleur rappel (100%) ce qui signifie qu'il a réussi à identifier toutes les vraies positives dans le test dataset, donc il ne manquera aucun positif. La régression logistique a des performances plus modestes en termes de précision et de rappel par rapport aux autres modèles.
En terme de moyenne harmonique, c'est le SVM qui a réussi à équilibré entre son rappel et sa précision.

1.4 Problématique non-supervisé

1.4.1 K-Means

L'objectif de cet algorithme est de regrouper un ensemble de données en K clusters différents en mesurant la variance entre les clusters.

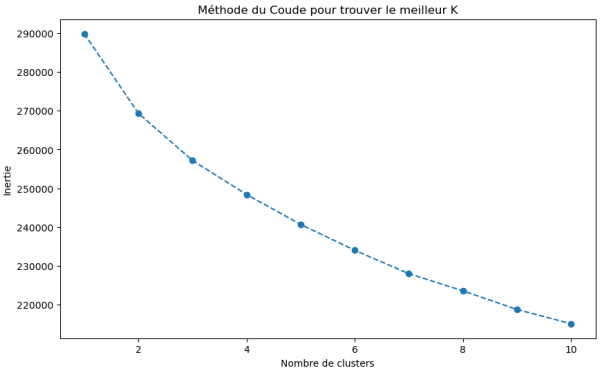
L'algorithme k-means, est basé sur un paramètre clé : (**n_clusters**) qui represente le nombre de clusters K. Le choix approprié de K est crucial, car il influence directement sur la qualité de regroupement. Il existe un autre paramètre (**init**) qui permet de spécifier la méthode d'initialisation des centroides des clusters, qui par défaut sont initialisé aléatoirement et cela peut conduire à des résultats suboptimaux, donc nous avons opté pour la méthode d'initialisation (**k-means++**) qui vise à choisir des centroides de manières qu'ils soient éloignés les uns des autres afin d'améliorer la capacité de convergence de l'algorithme.

La méthode du Coude

Consiste à tracer un graphique montrant comment la distance entre les points de données et leurs centres de cluster (Inirtie) diminue à mesure que le nombre de clusters K augmente.

Résultat :

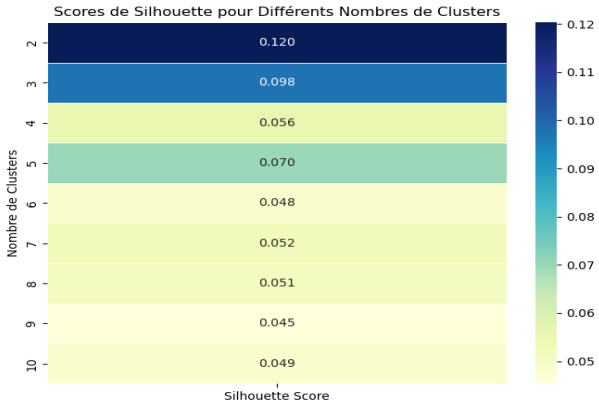
Pour interpréter ce graphe, nous cherchons le point où l'ajout d'un nouveau cluster n'entraîne pas une réduction significative de la variation entre cluster. En analysant le graphique, nous observons que la descente du graphe devient moins abrupte à partir du nombre de clusters égal à 3. Par conséquent, selon la méthode du coude, le meilleur nombre de clusters possible pour nos données est 3.



La méthode de la Silhouette

consiste à évaluer chaque point de données s'il s'ajuste bien à son propre cluster par rapport aux autres clusters. Pour chaque point, un score de silhouette est calculé en comparant la distance moyenne avec les points du même cluster à la distance moyenne avec les points d'autres clusters voisins.

Résultat:
Le meilleur nombre de clusters pour notre dataset, donné par la méthode de silhouette c'est 2.



Visualisation et comparaison

Chacune des deux méthodes a donné un nombre de clusters, nous visualisons nos données en utilisant les deux nombre (2,3) de clusters recommandés, en se servant de PCA pour réduire la dimension des données. D'après les résultats ci-dessous on peut constater que la visualisation de nos clusters en choisissant k = 2 est légèrement meilleur que celle de k = 3.

(a) K = 2 (b) K = 3

FIGURE 5 – Les cluster en variant le paramètre K

Vu que notre problème initial était un problème de classification binaire nous allons tester le score **'jaccard_similarity'** lorsque le k vaut 2 et nous comparons les labels donnés par l'algorithme k-means avec nos étiquettes principales.

Résultat :
Coefficient de Similarité de Jaccard : **0.31**. On remarque que le score de similarité est un peu faible, on déduit alors que l'algorithme K-Means ne correspondent pas très bien à la distribution naturelle des données.

1.4.2 L'algorithme hiérarchique

L'algorithme hiérarchique est une méthode de clustering qui commence avec chaque point de données comme un cluster individuel et fusionne progressivement les clusters voisins pour former des clusters plus grands.

Le paramètre initial clé pour optimiser l'algorithme hiérarchique est la méthode de liaison (linkage method). La méthode de liaison détermine comment mesurer la distance ou la similarité entre les clusters lors de leur fusion. On a 4 différentes méthodes de liaison ['ward', 'complete', 'average', 'single'].

Nous avons testé notre modèle pour obtenir la meilleure méthode de linkage

Observation :

Selon les résultats précédents, la méthode optimale pour regrouper efficacement nos données est la méthode de liaison "Ward".

Nous avons également utilisé la méthode de **silhouette** pour trouver le nombre optimal de clusters pour les données, et nous avons obtenu 2 comme nombre optimal. Cependant, le score de similarité de Jaccard était légèrement faible par rapport à l'algorithme de K-means.