27/07/2020

Projet S2 :

Classification des étudiants.

Réalisé par :

Houssam ZOUHRI

Abderahmane HAMDOUCHI

Encadré par :

Pr Ali IDRI

REMERCIEMENTS

Au terme de ce travail, je tiens à remercier :

Monsieur le Professeurs Ali IDRI, notre cher professeur en matière de machine Learning et notre chef de master qui a bien voulu que ce travail soit réalisé suite à l'excellente formation qu'on a eue dans ladite matière.

Veuillez trouver ici, messieurs, le témoignage de notre haute gratitude et nos sincères considérations.

Monsieur Ahmed RATNANI, notre chef département CSHES pour son soutien de la formation

Nos imminents Professeurs pour la qualité de formation qu'ils dispensent.

Tous les membres et le Staff de l'UM6P pour les efforts qui leur fournirent pour la réussite de la formation.

Abderahmane HAMDOUCHI

Houssam ZOUHRI

Abstract

La capacité de prédire la tendance de performance des étudiants est très importante pour améliorer leurs compétences pédagogiques. C'est devenu une connaissance précieuse qui peut être utilisée à des fins différentes ; par exemple, un plan stratégique peut être appliqué pour le développement d'une éducation de qualité. Ce projet propose l'application de techniques d'exploration de données pour prédire les notes finales des élèves en fonction de leurs données historiques. Dans les études expérimentales, trois techniques d'exploration de données bien connues (KNN, arbre de décision CART, MLP et SVM) ont été utilisées sur l'ensembles de données pédagogiques liés à la leçon de mathématiques et à la leçon de portugais. Les résultats ont montré l'efficacité des techniques d'apprentissage de l'exploration de données lors de la prédiction des performances des élèves.

Liste des tableaux

Tableau 1:Les principales caractéristiques de l'ensemble de données. 29

Tableau 2:les 27 meilleurs modèles suivant leurs hyperparamètres et accuracy pour le modèle KNN 38

Tableau 3: 27 meilleurs modèles suivant leurs hyperparamètres et accuracy pour le modèle CART 39

Tableau 4:27 meilleurs modèles suivant leurs hyperparamètres et accuracy pour le modèle MLP. 39

Tableau 5:27 meilleurs modèles suivant leurs hyperparamètres et accuracy pour le modèle SVM. 40

Tableau 6: Meilleur Accuracy KNN 42

Tableau 7:Meilleur Precision KNN 42

Tableau 8:Meilleur AUC_ROC KNN 42

Tableau 9:Meilleur Modèle KNN 43

Tableau 10:DataSet qui réalise le meilleur score en KNN 43

Tableau 11:Meilleur accuracy en CART 44

Tableau 12:Meilleur precision en CART 45

Tableau 13:Meilleur AUC_ROC en CART 45

Tableau 14: Le meilleur modèle du CART 45

Tableau 15: DataSet Correspendante au meilleur modèle CART 45

Tableau 16: Meilleur accuracy en MLP 47

Tableau 17:Meilleur precision en MLP 48

Tableau 18: Meilleur MLP en AUC_ROC 48

Tableau 19: Meilleur Modèle MLP 48

Tableau 20: Meilleur modèle SVM en accuracy 48

Tableau 21:Meilleur modèle SVM en Precision 49

Tableau 22:Meilleur modèle SVM en AUC_ROC 49

Tableau 23: Le meilleur modèle du SVM 49

Liste des figures

Figure 1: Graphe représentant AUC_ROC 31

Figure 2: technique d'imbalanced DATA 34

Figure 3: choix des méthodes de Feature Selection 35

Figure 4: Représentation graphique du meilleur modèle en KNN 44

Figure 5:Reprèsentation graphique du modèle CART 46

Figure 6: L'impureté en fonction de alpha 47

Liste des abréviations

Symbols	Signification
ANN	Artificial Neural Network
AUC	Area Under the Curve
FN	False Negative
FP	FP False Positive
TP	True Positive
KNN	K Nearest Neighbours
ML	Machine Learning
ROC	Receiver Operator Characteristic
SMOTE	Synthetic Minority Oversampling Technique
SVM	Support Vector Machine
MLP	MultiLayer Perceptron
DOF	Degree of liberty

Table des matières

REMERCIEMENTS 2

Abstract 3

Liste des abréviations 7

Introduction générale 10

Background and related work 13

I.Description de toutes les techniques utilisées : 14

1.1. CHI-SQUARD : 14

1.2. Mutual Information 15

1.3. VarianceThreshold 16

1.4. GridSearch 16

1.5. Le plus proche voisin (Nearest Neighbors) (k-NN) : 17

1.6. Arbre de décision (CART) 19

1.7. Support Vecteur Machine (SVM) 21

1.8. Multi layer Perceptron (MLP) 23

II.Représentation de la littérature : 24

2.1. Première exemple : 25

2.2. 2éme exemple : 25

Experimental design 26

I. Description du Dataset 27

2.1. Quelques aperçus des statistiques: 29

2.2. Classification des attributs selon leurs natures : 30

II. Criteria de performance 30

III.Expérimental process 31

3.1. Step 1: Prétraitement (Preprocessing)..... 32
3.2. Step 2 : Entrainement des modèles de classification : 36
3.3. Step 3 : Evaluation 38
Résultats..... 41
I.Présentation : 42
1.1 KNN 42
1.2. CART..... 44
1.3. L'utilisation de alpha pour élaguer l'arbre 46
1.4. MLP..... 47
1.5. SVM 48
II.Discussion : 49
III.Limites d'étude..... 50
CONCLUSION 51

Introduction générale

Pour contribuer au développement d'un pays, on se focalise toujours à s'investir sur leurs principaux domaines rentables qui sont l'économie, l'industrie et politique, et malgré leurs priorités et leurs impacts directs sur l'accroissement de l'état, on ne peut plus écarter le socle, la base réelle et le premier influenceur sur l'évolution d'un pays en général et d'une société en particulier et qui demeure continuellement le premier acteur de la réussite d'une organisation , qui est sans doute le système éducatif.

En outre, l'éducation œuvre dans nos sociétés à faire de la population de « bons citoyens ». Elle inculque les valeurs de nos sociétés et de nos institutions à ses bénéficiaires. Les systèmes d'éducation éduquent et introduisent des connaissances et des manières d'agir qui sont dites primordiales quant au bon développement de nos sociétés.

Ainsi que l'accompagnement des étudiant au cours de leur processus instructif, reste parmi les mesures primordiales à adopter afin de faire réussir ce développement, car ils sont les acteurs majeurs de ce système en question, alors on est appelé à leur évaluer constamment et les accompagner pour leurs corriger, ainsi de connaitre les facteurs cruciaux qui influencent leur échec ou leur réussite, afin de les corriger et intervenir avant qu'ils puissent engendrer des pertes irréversibles, car on peut jamais changer un passé confirmé. D'où vient la nécessité de la mise en place d'une tactique prédictive permettant de bien contrôler et intercaler au moment opportun. En plus du manque important des outils de prise de décision dans ce domaine par rapport aux autres.

Etant donné, notre groupe a travaillé, sur cet aspect afin de mettre en place un outil de prise de décision permettant de prévenir la réussite d'un étudiant selon leurs données, et d'analyser les facteurs importants qui influencent cette décision, en vue de les corriger ou de les faire évaluer.

En effet, on a développé un modèle de classification qui s'entoure sur 4 techniques qui font des modèles différents, qui sont évalués suivant différentes métrique qui sont *ROC* (receiver operating characteristic), *l'accuracy* (exactitude), et la *précision* afin de choisir celui qui satisfait au maximum ces mesures.

Le présent rapport comporte 3 chapitres, dont le premier comportant le background et la revue de la littérature qui comprend une description sur toutes les techniques utilisées ainsi que les travaux académiques qui sont déjà fait et leurs résultats. Ensuite le 2éme chapitre qui aborde la conception expérimentale et qui explique les différentes étapes procédées pour l'élaboration des modèles, avec leurs figures et leurs résumée, ainsi que les critères et les métriques de performance utilisé en vue d'évaluer ces modèles. Pour finir avec le dernier chapitre qui traite les résultats composés par les présentations contenant les figures et les tables d'évaluation de score, en plus des discussions pour expliquer le pourquoi de ces figures et de ces résultats, et clôturer par les limitations de l'étude menée suite aux quelques limites et restrictions.

Background and related work

I. Description de toutes les techniques utilisées :

1.1. CHI-SQUARD :

The Pearson's Chi-Squared test : est une technique de feature Selection utilisé dans les problèmes de classification dont les inputs sont tous catégoriques, qui est un test d'hypothèse statistique qui vérifier l'indépendance entre les variables catégoriques, via le table de contingence qui permet de présenter une variable en fonction d'une autre variable.

Ce test permet de vérifier l'absence de lien statistique entre deux variables X et Y. Les deux sont dites indépendantes lorsqu'il n'existe aucun lien statistique entre elles, dit autrement, la connaissance de X ne permet en aucune manière de se prononcer sur Y. L'hypothèse nulle (H0) de ce test est la suivante : les deux variables X et Y sont indépendantes.

En termes de valeur p, l'hypothèse nulle est généralement rejetée lorsque $p \leq 0,05$.

La signature de la méthode est la suivante : scipy.stats.chi2_contingency() qui est une fonction de la bibliothèque scipy qui prent comme paramètres :

- Observed : (de type array), dans notre cas la table de contingence qui exprime la fréquence d'une variable par rapport à une autre
- Correction : (de type booléen) : utilisé lorsqu'il est Ture et la valeur des dégrées de liberté est égal à 1 qui permet de d'ajuster chaque valeur observée de 0,5 vers la valeur estimée correspondante
- Lambda_(de type float or str): par défault la méthode utilise Pearson CHI-2, mais ce paramètre permet d'utiliser la famille de divergence de puissance « Cressie-Read »

Cette méthode return 4 valeurs différentes qui sont comme suit :

- chi2 : (de type float) la valeur de chi2 ou bien du test statistique
- p : (de type float) : le p-value du test
- dof : (de type int) qui est le degré de liberté

- expected : (de type ndarray) : liste des fréquences estimés, basées sur les sommes marginales du tableau.

1.2. Mutual Information

Cette méthode est aussi utilisée dans la construction des arbres de décisions qui permet de séléctionner les variables qui maximisent le gain d'information ou bien qui minimise l'entropie , car un grand gain d'information correspond à une valeur réduite de l'entropie, ainsi l'entropie représente la surprise d'avoir un évènement, si par exemple un évènement qui as une petite probabilité de réalisation, mais elle se réalise ou vice versa, on dit dans ce cas qu'on a une grande valeur d'entropie , ainsi il est exprimé par la formule ci-après:

$$H(S1) = \sum_{i=1}^{cs1} \frac{\#C_i}{N1} \log_2 \left(\frac{N1}{\#C_i}\right)$$

- cs1 : nombre de classes dans S1
- #Ci: nombre d'individus dans Ci
- H(S1) mesure l'impureté de S1

La signature de la méthode Gain d'information ou bien mutual d'information s'écrit sous la forme :

sklearn.feature_selection.mutual_info_classif(X, y, discrete_features= 'auto', n_neighbors=3, copy=True, random_state=None)

X : une liste ou bien une matrice de dimension (nombre d'instances, nombre d'attributs) Matrice des attributs.

Y : une liste, de dimension (nombre d'instances) Vecteur de Target.

- **discrete_features :** peut prendre trois types en argument qui sont :
  - Auto (la valeur par défaut), qui désigne automatiquement chaque attribut, s'il est dense (Continue), ou bien séparé (discret)
  - Booléen : pour déterminer si tous les attributs sont continus ou bien discret.
  - List : de dimension des attributs, dont chaque élément de la liste désigne l'attributs associé, s'il est discret ou bien continu

- **n_neighborsint** : de type int (prend par défaut la valeur 3) qui désigne le nombre de voisins utilisés pour déterminer le plus proche voisin.
- **Copy** : de type booléen, qui prend par default la valeur « True », et qui indique s'il faut cloner les données, s'il la valeur choisie « Faux », les anciennes valeurs seront écrasées.
- **random_state** : qui est le paramètre du générateur de nombres pseudo-aléatoire, pour ajouter un bruit aux variables continues afin de supprimer les valeurs répété il prend 3 types au choix qui sont:
- **int** : random_state utilise le nombre pour générer les nombres aléatoirs
- **Instance de RansomState** : est générateur de nombres aléatoires

1.3. VarianceThreshold

Sélecteur d'entités qui supprime toutes les entités à faible variance.

Cet algorithme de sélection de fonctionnalités ne regarde que les fonctionnalités (X), pas les sorties souhaitées (y)

Les paramètres de VarianceThreshold

- **flotteur de seuil , en option**

Les entités dont la variance de l'ensemble d'apprentissage est inférieure à ce seuil seront supprimées. La valeur par défaut est de conserver toutes les entités avec une variance non nulle, c'est-à-dire de supprimer les entités qui ont la même valeur dans tous les échantillons.

1.4. GridSearch

Recherche exhaustive sur les valeurs de paramètres spécifiées pour un estimateur.

Les membres importants sont en forme, prévoyez.

GridSearchCV implémente une méthode « fit » et « score ». Il implémente également « prédire »,«prédire_proba», «fonction_décision», «transformer» et «inverser_transform» s'ils sont implémentés dans l'estimateur utilisé.

Les paramètres de l'estimateur utilisé pour appliquer ces méthodes sont optimisés par une recherche de grille à validation croisée sur une grille de paramètres.

Les paramètres de GridSearch

- **objet estimateur estimateur.**

Ceci est supposé implémenter l'interface d'estimateur scikit-learn. L'estimateur doit fournir une scorefonction ou scoringdoit être passé.

- **param_grid dict ou liste de dictionnaires**

Dictionnaire avec des noms de paramètres (str) comme clés et des listes de réglages de paramètres à essayer comme valeurs, ou une liste de ces dictionnaires, auquel cas les grilles couvertes par chaque dictionnaire de la liste sont explorées. Cela permet d'effectuer une recherche sur n'importe quelle séquence de réglages de paramètres.

- **scoring str, appelable, liste / tuple ou dict, par défaut = Aucun**

Une seule chaîne (voir Le paramètre de notation: définir les règles d'évaluation du modèle) ou un appelable (voir Définir votre stratégie de notation à partir des fonctions métriques) pour évaluer les prédictions sur l'ensemble de test.

- **cv int, générateur de validation croisée ou un itérable, par défaut = Aucun**

Détermine la stratégie de fractionnement de validation croisée. Les entrées possibles pour cv sont:

- Aucun, pour utiliser la validation croisée 5 fois par défaut,
- entier, pour spécifier le nombre de plis dans a (Stratified)KFold,
- Répartiteur CV ,

1.5. Le plus proche voisin (Nearest Neighbors) (k-NN) :

Une méthode non paramétrique utilisée pour la classification et la régression. Dans les deux cas, il s'agit de classer l'entrée dans la catégorie à laquelle appartient les k plus proches voisins dans l'espace des

caractéristiques identifiées par apprentissage. Le résultat dépend si l'algorithme est utilisé à des fins de classification ou de régression. Mais dans notre cas, on se focalise uniquement sur le KNN associé à la classification. Dont le résultat est une classe d'appartenance. donc l'objet d'entrée est classifié selon le résultat majoritaire des statistiques de classes d'appartenance de ses k plus proches voisins, (k est un nombre entier positif généralement petit). Si k = 1, alors l'objet est affecté à la classe d'appartenance de son proche voisin.

La signature de la méthode est la suivante :

*sklearn.neighbors.KNeighborsClassifier(n_neighbors=5,weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None, **kwargs)*

Les paramètres de cette méthode sont les suivantes :

- **n_neighbors** (de type int ou bien facultatif**) :**Le nombre des voisins à utiliser pour la classification, la valeur prise par défaut est 5
- **Weights** : de type string ou méthode, il est facultatif, il prend par défaut la valeur 'uniform' :Est une fonction de poids utilisé, pour la prédiction des valeurs possibles :
    - 'uniform' : tous les voisins proches ont un poids uniforme.
    - 'distance' :le poids est pris inversement à la distance.
- **algorithm** : qui prend en paramètre 4 valeurs qui sont ('auto', 'ball_tree', 'kd_tree' et 'brute'), ce paramètre est facultatif.

  Cet algorithme est utilisé pour calculer le plus proche voisin

    - 'ball_tree' utilise l'algorithme ball tree
    - 'kd_tree' utilise l'algorithme arbre k-d
    - 'brute' utilisé l'algorithme brute force pour la recherche .

- **Metric** : Ce type string ou fonction, il prend par défaut la valeur 'minkowski' : La métrique des distances utilisé qui peut avoir plusieurs valeurs différentes, comme il se peut que l'utilisateur

- **metric_params** : il est de type dictionnary , ce paramètre est facultatif, il permet d'ajouter des argument, dans le cas de déclaration d'une fonction dans le paramètre Metric
- **n_jobs** : int ou None , il est facultatif et prend la valeur None par défaut.

C'est le nombre de taches en parallelle, fréquement, on choisit la valeur -1 pour que tous les processeurs puissent fonctionner en meme temps.

1.6. Arbre de décision (CART)

Classification and Regression Trees est un terme introduit par Leo Breiman, pour se référer à l'arbre de décision qu'on peut utiliser pour la classification ou bien la décision afin de modéléser les problèmes.

Dans ses principes il semble à l'ID3, ainsi la seule différence est qu'il utilise le fonctionnement Greedy spliting ou bien le fractionnement de Gourmande comme son nom l'indique. on essaie à faire La répartition avec le meilleur coût en utilisant l'indice du GINI au lieu de l'entropie (ou bien coût le plus bas car nous minimisons le coût).

Le Cart utilise une méthode d'élagage très sophistiquées qui s'appelle l'élagage à complexité de coût ou bien (cost complexity pruning) qui utilise un paramètre alpha et qui est intégré SKlearn à partir de la version 0.22.

Scikit-learn utilise une version optimisée de l'algorithme CART; cependant, cette version ne prend pas en charge les variables catégoriques pour l'instant.

Mais on utilise l'algorithme standard DecisionTreeClassifier en choisissant GINI dans le paramètre « **criterion** »

Les paramètres de l'algorithme :

- **critère {"gini", "entropy"}, default = "gini"**

La fonction pour mesurer la qualité d'une scission. Les critères pris en charge sont «gini» pour l'impureté de Gini et «entropie» pour le gain d'information.

- **splitter {"best", "random"}, default = "best"**

La stratégie utilisée pour choisir la division à chaque nœud. Les stratégies prises en charge sont «meilleures» pour choisir la meilleure division et «aléatoire» pour choisir la meilleure division aléatoire.

- **max_depth int, par défaut = Aucun**

La profondeur maximale de l'arbre. Si None, les nœuds sont développés jusqu'à ce que toutes les feuilles soient pures ou jusqu'à ce que toutes les feuilles contiennent moins de min_samples_split échantillons.

- **min_samples_split entier ou float, par défaut = 2**

Le nombre minimum d'échantillons requis pour fractionner un nœud interne:

- Si int, alors considérez min_samples_splitcomme le nombre minimum.
- Si float, alors min_samples_splitest une fraction et représente le nombre minimum d'échantillons pour chaque division.ceil(min_samples_split * n_samples)

- **min_samples_leaf int ou float, par défaut = 1**

Le nombre minimum d'échantillons requis pour être au niveau d'un nœud feuille. Un point de partage à n'importe quelle profondeur ne sera considéré que s'il laisse au moins min_samples_leafdes échantillons d'apprentissage dans chacune des branches gauche et droite. Cela peut avoir pour effet de lisser le modèle, notamment en régression.

- **min_weight_fraction_leaf float, par défaut = 0.0**

La fraction pondérée minimale de la somme totale des poids (de tous les échantillons d'entrée) requis pour être au niveau d'un nœud feuille. Les échantillons ont le même poids lorsque sample_weight n'est pas fourni.

- **max_features int, float ou {"auto", "sqrt", "log2"}, par défaut = Aucun**

Le nombre de fonctionnalités à prendre en compte lors de la recherche de la meilleure répartition:

- **max_leaf_nodes int, par défaut = Aucun**

Cultivez un arbre avec max_leaf_nodesde la meilleure façon. Les meilleurs nœuds sont définis comme une réduction relative des impuretés. Si aucun, alors nombre illimité de nœuds feuilles.

- **min_impurity_decrease float, par défaut = 0,0**

Un nœud sera scindé si cette scission induit une diminution de l'impureté supérieure ou égale à cette valeur.

- **min_impurity_split float, par défaut = 0**

Seuil d'arrêt précoce de la croissance des arbres. Un nœud se divisera si son impureté est au-dessus du seuil, sinon c'est une feuille.

- **class_weight dict, liste de dict ou «balancé», par défaut = Aucun**

Poids associés aux classes dans le formulaire . Si aucun, toutes les classes sont censées avoir un poids un. Pour les problèmes à sorties multiples, une liste de dictionnaires peut être fournie dans le même ordre que les colonnes de y.{class_label: weight}

- **ccp_alpha flottant non négatif, par défaut = 0,0**

Paramètre de complexité utilisé pour l'élagage à coût-complexité minimale. Le sous-arbre avec la plus grande complexité de coût qui est plus petit que ccp_alphasera choisi. Par défaut, aucun élagage n'est effectué. Voir Élagage à coût-complexité minimale pour plus de détails.

1.7. Support Vecteur Machine (SVM)

SVM Les machines à vecteurs de support (également réseaux à vecteur de support). Il présente l'une des méthodes de prédiction les plus robustes. Étant donné un ensemble d'exemples d'apprentissage, chacun marqué comme appartenant à l'une ou l'autre des deux catégories, un algorithme d'apprentissage SVM construit un modèle qui attribue de nouveaux exemples à une catégorie ou à l'autre, ce qui en fait un classificateur linéaire binaire non probabiliste. Un modèle SVM est une représentation des exemples sous forme de points dans l'espace, mappés de manière à ce que les exemples des catégories distinctes soient divisés par un espace clair aussi large que possible. De nouveaux exemples sont ensuite mappés dans ce même

espace et supposés appartenir à une catégorie basée sur le côté de l'écart sur lequel ils se situent.

Les paramètres de SVM :

- **Flottant C, par défaut = 1,0**

Paramètre de régularisation. La force de la régularisation est inversement proportionnelle à C. Doit être strictement positive. La pénalité est une pénalité de 12 au carré.

- **noyau {'linéaire', 'poly', 'rbf', 'sigmoïde', 'précalculé'}, par défaut = 'rbf'**

Spécifie le type de noyau à utiliser dans l'algorithme. Il doit s'agir de "linear", "poly", "rbf", "sigmoid", "precalculé" ou appelable. Si aucun n'est indiqué, «rbf» sera utilisé. Si un appelable est donné, il est utilisé pour précalculer la matrice du noyau à partir des matrices de données; cette matrice doit être un tableau de formes .(n_samples, n_samples)

- **degré int, par défaut = 3**

Degré de la fonction noyau polynomiale ('poly'). Ignoré par tous les autres noyaux.

- **gamma {'scale', 'auto'} ou float, par défaut = 'scale'**

Coefficient de noyau pour «rbf», «poly» et «sigmoïde».

si gamma='scale'(par défaut) est passé, il utilise 1 / (n_features * X.var ()) comme valeur de gamma,

si 'auto', utilise 1 / n_features.

- **coef0 float, par défaut = 0.0**

Terme indépendant dans la fonction du noyau. Il n'est significatif que dans «poly» et «sigmoïde».

- **tol float, par défaut = 1e-3**

Tolérance pour le critère d'arrêt.

- **cache_size float, par défaut = 200**

Spécifiez la taille du cache du noyau (en Mo).

- **max_iter int, par défaut = -1**

Limite stricte des itérations dans le solveur, ou -1 pour aucune limite.

1.8. Multi layer Perceptron (MLP)

Un MLP se compose d'au moins trois couches de nœuds : une couche d'entrée, une couche cachée et une couche de sortie. À l'exception des nœuds d'entrée, chaque nœud est un neurone qui utilise une fonction d'activation non linéaire. MLP utilise une technique d'apprentissage supervisé appelée rétropropagation pour la formation. Ses multiples couches et son activation non linéaire distinguent MLP d'un perceptron linéaire. Il peut distinguer les données qui ne sont pas linéairement séparables

Les paramètres de MLP se déclinant ainsi :

- **hidden_layer_sizes tuple, length = n_layers - 2, par défaut = (100,)**

Le ième élément représente le nombre de neurones dans la ième couche cachée.

- **activation {'identity', 'logistic', 'tanh', 'relu'}, default = 'relu'**

Fonction d'activation pour le calque caché.

- **solveur {'lbfgs', 'sgd', 'adam'}, default = 'adam'**

Le solveur pour l'optimisation du poids.

- **batch_size int, default = 'auto'**

Taille des minibatchs pour les optimiseurs stochastiques. Si le solveur est 'lbfgs', le classificateur n'utilisera pas de mini-correspondance. Lorsqu'il est réglé sur «auto»,batch_size=min(200, n_samples)

- **learning_rate {'constant', 'invscaling', 'adaptive'}, default = 'constant'**

Calendrier d'apprentissage pour les mises à jour de poids.

- **learning_rate_init double, par défaut = 0,001**

Le taux d'apprentissage initial utilisé. Il contrôle le pas de mise à jour des poids. Utilisé uniquement lorsque solveur = 'sgd' ou 'adam'.

- **tol float, par défaut = 1e-4**

Tolérance pour l'optimisation. Lorsque la perte ou le score ne s'améliore pas au moins tolpendant n_iter_no_changedes itérations consécutives, à moins qu'il ne learning_ratesoit réglé sur «adaptatif», la convergence est considérée comme atteinte et l'entraînement s'arrête.

- **momentum float, par défaut = 0,9**

Momentum pour la mise à jour de la descente de gradient. Doit être compris entre 0 et 1. Utilisé uniquement lorsque solveur = 'sgd'.

II. Représentation de la littérature :

La prévision des performances des élèves est essentielle pour que les éducateurs obtiennent une rétroaction précoce et prennent des mesures immédiates ou précoces si nécessaire pour améliorer les performances de l'élève. Cette prédiction peut être gérée en localisant la source du problème. Doit-il provenir d'activités supplémentaires auxquelles l'élève participe, de problèmes familiaux ou de problèmes de santé. Tous ces facteurs peuvent avoir un effet majeur sur les performances des élèves. Le fait d'avoir un ensemble de données sur les performances des élèves peut nous aider à étudier de tels cas.

Jusqu'à présent, les algorithmes d'exploration de données ont été appliqués à différents des domaines tels que la formation d'ingénieur, l'éducation physique et Éducation en langue anglaise]. Certaines études se sont concentrées sur les élèves du secondaire, alors que certains d'entre eux se sont intéressés à l'enseignement supérieur. Alors que certains les études d'exploration de données se sont concentrées sur la prédiction des performances des élèves, certaines études ont étudié les performances de l'instructeur.

2.1. Première exemple :

Ce travail a présenté deux modèles de prédiction pour l'estimation de la performance de l'élève à l'examen final. Le travail a utilisé l'ensemble de données populaire fourni par l'Université de Minho au Portugal, qui se rapporte à la performance en matière de mathématiques et qui comprend 395 échantillons de données. Ils ont appliqué à la fois l'algorithme Support Vector Machine et l'algorithme K-Nearest Neighbour sur l'ensemble de données pour prédire la note de l'élève, puis avons comparé leur précision. Les résultats des études empiriques ont indiqué que Support Vector Machine a obtenu des résultats légèrement meilleurs avec un coefficient de corrélation de 0,96, tandis que le voisin le plus proche K a obtenu un coefficient de corrélation de 0,95.

Published in: 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)

2.2. 2éme exemple :

Cette étude est faite par Ferda Ünal. Dans cette expérience, trois algorithmes ont été comparés sur la version à cinq niveaux et la version binaire, qui sont l'arbre de décision, forêt aléatoire et Bayes naïfs. Pour la version de classement à cinq niveaux pour cet ensemble de données ont été obtenues avec un taux de précision de 73,50% avec l'algorithme de forêt aléatoire. Cependant, ce taux de précision a été augmenté avec la version de classement binaire de cet ensemble de données. Dans l'ensemble de données, où la note finale est classée sous forme binaire (réussite ou échec), le taux de précision a été augmenté à 93,07%.

- Soumis: 25 novembre 2019
- Révisé: 31 janvier 2020
- Publié: 28 mars 2020

NB : dans ces deux projets ils ont laissé les deux variables G1 et G2 les notes du 1 et 2éme semestre.

Experimental design

I. Description du Dataset

Dans cette étude, deux ensembles de données accessibles au public ont été utilisés pour prédire les performances des élèves. Les deux ensembles de données ont été collectés dans l'enseignement secondaire de deux écoles portugaises. Les attributs de l'ensemble de données concernent les notes des élèves et les fonctionnalités sociales, démographiques et liées à l'école. Toutes les données ont été obtenues à partir des rapports scolaires et des questionnaires. Le premier ensemble de données contient des informations sur les performances des élèves en leçon de mathématiques, et l'autre contient des données sur les élèves tirées de la leçon de portugais. Les deux jeux de données ont 33 attributs comme indiqué dans le tableau 1.

Feature	Description	Type	Values	Statistique descriptive
sex	Sexe de l'étudiant	Binaire	Féminin ou masculin	F=53% M=47%
age	Age de l'étudiant	Numérique	(De 15 à 22 ans)	Médiane =15ans
school	Ecole de l'élève	Binaire	GP (Gabriel Pereira) MS (Mousinho da Silveira)	GP=88% MS=12%
address	L'adresse de résidence de l'élève	Binaire	Urbain ou rural	U=78% R=22%
Pstatus	Statut de cohabitation des parents	Binaire	Vivre ensemble=T Ou séparément=A	T=90% A=10%
Medu	Éducation de la mère	Numérique	De 0 à 4	-
Mjob	Emploi de la mère (	Nominal	Enseignant, santé, services, à domicile, autres	Other=36% Services=26%
Fedu	Éducation du père	Numérique	De 0 à 4	-
Fjob	Travail du père	Nominal	Enseignant, santé, services, à	Other=55% Services=28%

			domicile, autres	
guardian	Tuteur de l'élève	Nominal	Mère, Père ou autres	Mother=69% Father=23%
famsize	Taille de la famille	Binaire	"LE3" (inférieur ou égal à 3) ou "GT3" (supérieur à 3)	GT3=71% LE3=29%
Famrel	Qualité des relations familiales	Nominal	De 1 très mauvais à 5 excellents	1=08 2=18 3=68 4=195 5=106
reason	Raison pour laquelle vous avez choisi cette école	Nominal	Près de chez soi, réputation de l'école, préférence de cours ou autres	Course=37% Home=28%
traveltime	Temps de trajet du domicile à l'école	Nominal	1– <15 min., 2–15 à 30 min., 3–30 min. à 1 heure, ou 4–> 1 heure	1=256 2=107 3=23 4=08
studytime	Durée d'étude hebdomadaire	Nominal	- <2 heures, 2–2 à 5 heures, 3–5 à 10 heures ou 4–> 10 heures	-
failures	Nombre d'échecs de classe passés	Nominal	n si 1 <= n <3, sinon 4	0=312 1=50 2=17 3=16
schoolsup	Soutien scolaire	Binaire	Oui ou non	-
famsup	Soutien éducatif familial	Binaire	Oui ou non	-
activities	Activités parascolaires	Binaire	Oui ou non	-
paidclass	Classes	Binaire	Oui ou non	-
internet	Accès Internet à domicile	Binaire	Oui ou non	-

nursery	Garderie fréquentée : école maternelle	Binaire	Oui ou non	-
higher	Suivre des études supérieures	Binaire	Oui ou non	-
romantic	Une relation amoureuse	Binaire	Oui ou non	-
freetime	Temps libre après l'école	Nominal	De 1 (très mauvais) à 5 (excellents)	-
goout	Sortir avec des amis	Nominal	De 1 (très mauvais) à 5 (excellents)	-
Walc	Consommation d'alcool du week-end	Nominal	De 1 (très mauvais) à 5 (excellents)	-
Dalc	Consommation d'alcool en journée de travail	Nominal	De 1 (très mauvais) à 5 (excellents)	-
health	Etat de santé actuel	Nominal	De 1 (très mauvais) à 5 (excellents)	-
absences	Nombre d'absences scolaires	Numérique	De 0 à 93	-
G1	Grade G1 première période	Numérique	De 0 à 20	-
G2	Grade de deuxième période	Numérique	De 0 à 20	-
G3	Note finale G3	Numérique	De 0 à 20	-

Tableau 1: Les principales caractéristiques de l'ensemble de données.

2.1. Quelques aperçus des statistiques:

Age : L'âge moyen est de 16 ans.

Absences : le moyen est de 6 jours d'absence, et nous repérons des valeurs aberrantes avec 75 jours d'absence

Dalc : La consommation quotidienne d'alcool chez les enfants est très faible (ce qui est bien)

2.2. Classification des attributs selon leurs natures :

Variables Catégoriques Ordinales	Variables Catégoriques Non Ordinales	Variables Catégoriques Non Ordinales à 2 Valeurs	Variables numériques
• Medu - mother's education • Fedu - father's education • traveltime- home to school travel time • studytime - weekly study time • Failures • famrel - quality of family relationships • freetime - free time after school • goout - going out with friends • Dalc - workday alcohol consumption • Walc - weekend alcohol consumption • health - current health status	• Mjob - mother's job • Fjob - father's job • reason - reason to choose this school • guardian - student's guardian	• school • sex • address • famsize • Pstatus • schoolsup • famsup • paid • activities • nursery • higher • internet • Romantic	• age * • absences * • G1 - first period grade • G2 - second period grade • G3 - final grade

Tableau 1:Classification du DATA

II. Criteria de performance

Nous prédirons les étiquettes de classe pour savoir si un étudiant va réussir ou non. Par conséquent, nous avons besoin d'une mesure appropriée pour évaluer les étiquettes de classe prédites.

La tâche se concentre sur la classe positive (FAIL). La précision et le rappel sont un bon point de départ. Maximiser la précision minimisera les faux positifs, et maximiser le rappel minimisera les faux négatifs dans les prédictions faites par un modèle.

- **Précision** = TruePositives / (TruePositives + FalsePositives)
- **Accuracy** = (TP + TN)/(TP + TN + FP + FN)
- **Balaced Accuracy** :$= \frac{1}{2}\left(\frac{TP}{P} + \frac{TN}{N}\right)$

- **AUC** (Area Under The Curve)
- **ROC** (Receiver Operating Characteristics)

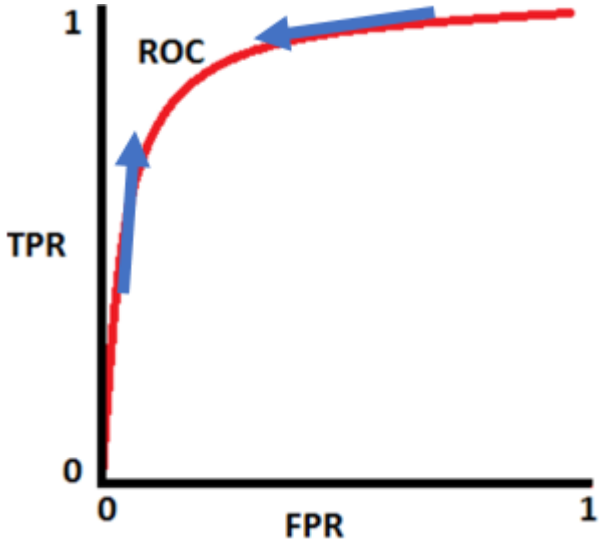

Figure 1: Graphe représentant AUC_ROC

- Avec :
- $\text{TPR /Recall / Sensitivity} = \frac{TP}{TP + FN}$
- $\text{FPR} = 1 - \text{Specificity}$
- $\text{Specificity} = \frac{TN}{TN + FP} = \frac{FP}{TN + FP}$

Dans l AUC on cherche à maximiser les True Positives et minimiser les False négative, autrement dit augmenter sensitivity et specificty en parallèle

III. Expérimental process

Dans l'ensemble de données brutes, la note finale se situe entre 0 et 20, comme dans de nombreux pays, où 0 est la pire note et 20 le meilleur score. Étant donné que la note finale des élèves est sous la forme d'un entier,

la classe prévue devrait être sous la forme de valeurs catégorielles, les données devaient être transformées en catégories selon une politique de notation. Dans notre étude, nous avons utilisé la moyenne des 3 semestres G1, G2 et G3 pour faire une classification selon la moyenne de ces 3 semestres. Pour comparer les résultats, nous avons également classé la note finale comme « réussir» et «échouée». Comme le montre le tableau :

Moins de 10	***Plus de 10***
FAIL	SUCESS

3.1. Step 1: Prétraitement (Preprocessing)

a) Encodage de variables catégorielles

Un modèle d'apprentissage automatique ne peut pas traiter des variables catégorielles (sauf pour certains modèles). Par conséquent, nous devons trouver un moyen de les coder (représenter sous forme de nombres) avant de les remettre au modèle. Donc nous devons d'abord convertir ces valeurs catégorielles en valeurs numériques.

Certains des attributs sont ordinaux, certains sont binaires, certains sont numériques et certains sont nominaux. Nous devons effectuer un prétraitement des données. Pour les attributs binaires oui-non, nous avons utilisé des 0 et des 1.

Nous l'avons fait pour « schoolsup, famsup, paid, activities, nursery, higher, internet, et romantic ». Les attributs « famrel, freetime, goout, Dalc, Walc et health » sont ordinaux; les valeurs de ces valeurs vont de 1 à 5. Les attributs « Medu, Fedu, traveltime, studytime, failures » sont également ordinaux; les valeurs vont de 0 à 4 ou de 1 à 4. L'attribut « absences » est un attribut count; les valeurs vont de 0 à 93. Les attributs « sex, school, adress, Pstatus, Mjob, Fjob, guardian, famsize, reason » sont nominaux. Pour les attributs nominaux, nous avons utilisé l'encodage one hot encoding.

Pour les variables numériques absences et age on les a discrétisées selon un intervalle de classement.

NB : Toutes les variables d'entrées sont des variables catégorique.

b) Descritisation des variables numériques

- Pour uniformer les types des attributs, une discrétisation des variables numériques age et Absences a été effectué, afin de permettre un apprentissage unifié et d'appliquer les différents modèles uniformément sur tous les attributs du DataSet, ainsi pour éliminer le problème d'overfiting (retenir les abstractions au lieu des détails)

c) Imbalanced data

La formation d'un modèle d'apprentissage automatique sur un ensemble de données déséquilibré peut introduire des défis uniques au problème d'apprentissage. Les données déséquilibrées se réfèrent généralement à un problème de classification où le nombre d'observations par classe n'est pas réparti également ; nous aurons souvent une grande quantité de données / observations pour une classe (dénommée *classe majoritaire*) et beaucoup moins d'observations pour une ou plusieurs autres classes (dénommées *classes minoritaires*). Pour notre dataset la classe majoritaire est la classe SUCESS avec 58% et la classe minoritaire est FAIL avec 42%. Pour résoudre ce problème on a choisi deux techniques :

Suréchantillonnage (SMOTE) ;

La technique de suréchantillonnage des minorités synthétiques (SMOTE) est une technique qui génère de nouvelles observations en interpolant les observations dans l'ensemble de données d'origine.

Sous-échantillonnage (Nermiss)

Plutôt que de suréchantillonner les classes minoritaires, il est également possible d'atteindre l'équilibre des classes en *sous-échantillonnage* la classe majoritaire - essentiellement jeter des données pour faciliter l'apprentissage des caractéristiques des classes minoritaires.

Voici une schéma résume cette étape :

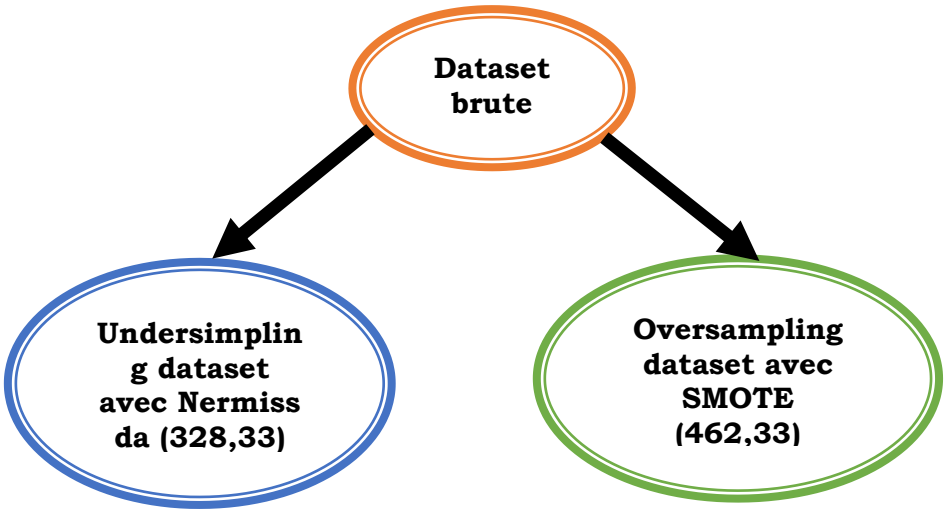

Figure 2: technique d'imbalanced DATA

A la fin de cette étape nous avons obtenu 3 datasets :

- Dataset brute ;
- Oversampling dataset ;
- Undersampling dataset ;

Et à partir de ces données on va appliquer la sélection des variables.

d) La sélection des variables (features selection)

La sélection des variables est l'un des concepts fondamentaux de l'apprentissage automatique qui a un impact énorme sur les performances de votre modèle, et qui consiste à sélectionner automatiquement ou manuellement les variables qui contribuent le plus à notre variable cible.

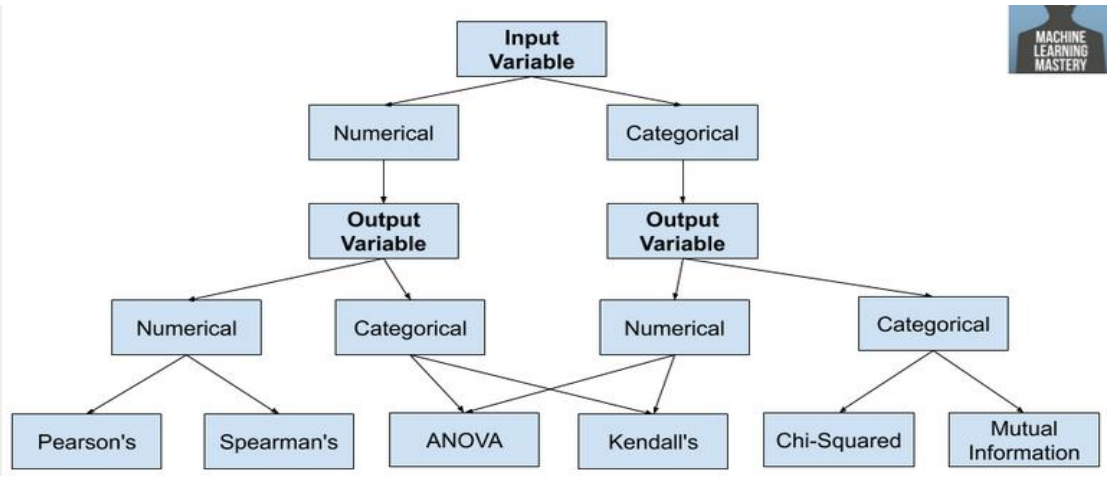

Figure 3: choix des méthodes de Feature Selection

Entrée catégorique, sortie catégorique :

Il s'agit d'un problème de classification avec des variables d'entrée catégorielles. Les filtres les plus courants pour les données catégorielles est le test du chi carré , et les informations mutuelles (gain d'informations) du domaine de la théorie de l'information.

- **Information mutuelle** : est une méthode puissante qui peut s'avérer utile à la fois pour les données catégorielles et numériques.
    - **Test du chi carré** (tableaux de contingence) : est un test d'hypothèse statistique qui suppose (l'hypothèse nulle) que les fréquences observées pour une variable catégorielle correspondent aux fréquences attendues pour la variable catégorielle. Le test calcule une statistique qui a une distribution chi-carré, nommée d'après la lettre majuscule grecque Chi (X) prononcée «ki» comme en cerf-volant.

Le test du chi carré le fait pour une table de contingence, calculant d'abord les fréquences attendues pour les groupes, puis déterminant si la division des groupes, appelée fréquences observées, correspond aux fréquences attendues.

Le résultat du test est une statistique de test qui a une distribution chi-carré et qui peut être interprétée comme rejetant ou échouant à rejeter l'hypothèse ou l'hypothèse nulle que les fréquences observées et attendues sont les mêmes.

On a ajouté également une 3éme méthode qui s'appelle **Variance-Treshold** qui est valable pour les variables catégoriques et numériques.

NB : Les 3 méthodes de sélection des variables ont été appliquées sur les trois datasets (brute, undersamling dataset et Oversampling dataset), en laissant seulement 20% ,40% ,60% des variables.

Filtre du Test du chi carré	Filtre de l'Information mutuelle	Filtre du Variancethreshold
brutChi2_20	brutMI_20	brutVAR_20
brutChi2_30	brutMI_30	brutVAR_30
brutChi2_40	brutMI_40	brutVAR_40
overChi2_20	overMI_20	overVAR_20
overChi2_30	overMI_30	overVAR_30
overChi2_40	overMI_40	overVAR_40
underChi2_20	underMI_20	underVAR_20
underChi2_30	underMI_30	underVAR_30

Tableau 2: les DataSets obtenues à l'issue

Après ces trois étapes de prétraitement on a obtenu 27 datasets.

3.2. Step 2 : Entrainement des modèles de classification :

Nous évaluerons les modèles d'apprentissage automatique suivants sur l'ensemble de données, selon quatre algorithmes :

- Knn ;
- Arbre De Decision(CART) ;
- Multi-Layer Perceptron (MLP) ;
- Support Vector Machine (SVM) ;

Pour l'évaluation des différents modèless, on a utilisé l'algorithme GridSearch qui permet de combiner avec les différents paramètres possibles afin de les évaluer et les triller en Accuracy, balanced accuracy Précision et

AUC_ROC en insérant l'algorithme de cross validation pour éviter le problème d'overfiting

Quant aux paramètres utilsés pour chaques algorithmes sont ainsi :

a) Algorithme KNN :

- n_neighbors = de 2 jusqu'à 11
- weights = 'uniform' et 'distance]
- metric = 'hamming' (car tous les vaiables sont catégoriques)

b) Algorithme de Arbre de décision (CART) :

- criterion = ['gini']
- splitter= ["best", "random"]
- max_depth=de 2 à 5
- min_samples_leaf=de 5 à 15
- max_features=de 4 à 8

c) Algorithme MLP

- activation=['logistic']
- solver=['sgd', 'adam'] #
- learning_rate=['constant', 'invscaling', 'adaptive']
- batch_size=de 200 à 400 avec pas de 50
- early_stopping=[True,False]
- learning_rate_init = avec pas de 0.001
- momentumfloat= de 0.5 à 0.9 avec pas de 0.01

d) Algorithme SVM :

- C= de 1 à 3 avec pas de 0.3
- kernel=['linear', 'poly', 'rbf', 'sigmoid']
- degree=de 1 à 4 avec pas de 1
- gamma=['scale','auto']
- tol=de 1e-5 à 1e-2 à chaque on multiplie par 10

3.3. Step 3 : Evaluation

a) Méthodes d'évaluation : Coss validation

Dans cette étude, la sélection des variables et les opérations de classification ont été menées à l'aide du jupyter de python.

Dans chaque expérience au total 27 expériences, une validation croisée de 10 fois a été effectuée pour évaluer nos modèles de classification. La précision de classification de l'algorithme pour l'ensemble de données de test a été mesurée par l'accuracy, ainsi la meilleure précision de chaque modèle selon les différents hyper-paramètres de chaque algorithme.

b) Critère d'évaluation : bourda count

Afin de sélectionner le meilleur modèle, on a obtenu a faire un bourda count, Dans cette méthode, des points sont attribués aux modèle en fonction de leur classement ; 1 point pour le dernier choix, 2 points pour l'avant-dernier choix, et ainsi de suite. Les valeurs en points pour tous les bulletins de vote sont totalisées et le modèle avec le plus grand total de points est le gagnant comme le montre les tableaux suivants :

Dataset	***N_Neighbors'***	***Weights***	***Accuracy %***
BRUTCHI2_20	*3*	*UNIFORM*	*61,1987*
BRUTCHI2_30	*10*	*DISTANCE*	*60,3885*
BRUTCHI2_40	*7*	*DISTANCE*	*59,6248*
OVERCHI2_20	*6*	*DISTANCE*	*68,173*
OVERCHI2_30	*7*	*UNIFORM*	*63,6309*
OVERCHI2_40	*8*	*DISTANCE*	*68,617*
UNDERCHI2_20	*9*	*UNIFORM*	*62,7936*
UNDERCHI2_30	*9*	*UNIFORM*	*62,2254*
UNDERCHI2_40	*4*	*DISTANCE*	*60,0758*
BRUTMI_20	*3*	*DISTANCE*	*61,1987*
BRUTMI_30	*10*	*DISTANCE*	*60,3885*
BRUTMI_40	*3*	*DISTANCE*	*58,934*
OVERMI_20	*6*	*DISTANCE*	*68,173*
OVERMI_30	*7*	*UNIFORM*	*63,6309*
OVERMI_40	*6*	*DISTANCE*	*67,7567*
UNDERMI_20	*9*	*UNIFORM*	*62,7936*

Tableau 3:les 27 meilleurs modèles suivant leurs hyperparamètres et accuracy pour le modèle KNN

DATASET	MAX_ DEPTH	MAX_ FEATURES	MIN_S AMPLES_LEAF	SPLITT	ACCURACY %
BRUTCHI2_20	3	5	5	random	67,9711
BRUTCHI2_30	3	5	5	random	67,9711
BRUTCHI2_40	3	7	5	best	68,3214
OVERCHI2_20	4	4	5	best	66,6466
OVERCHI2_30	3	4	5	best	66,6605
OVERCHI2_40	4	6	9	best	66,6512
UNDERCHI2_20	2	5	5	best	68,9015
UNDERCHI2_30	3	5	5	random	68,9015
UNDERCHI2_40	2	5	7	best	68,9015
BRUTMI_20	3	5	5	random	67,9711
BRUTMI_30	3	5	5	random	67,9711
BRUTMI_40	3	7	9	best	68,3214
OVERMI_20	4	4	5	best	66,6466
OVERMI_30	3	4	5	best	66,6605

Tableau 4: 27 meilleurs modèles suivant leurs hyperparamètres et accuracy pour le modèle CART

DATASET	BATCH_ SIZE	LEARNING_ RATE	LEARNING_ RATE_INIT	MOMENTUM	SCORE %
OVERMI_20_DUMMY_CODDING	200	constant	0,005	0,9	69,6762
OVERMI_30_DUMMY_CODDING	350	adaptive	0,003	0,9	68,1684
OVERMI_40_DUMMY_CODDING	200	constant	0,005	0,9	70,7632
UNDERMI_20_DUMMY_CODDING	250	invscaling	0,004	0,766	69,4886
UNDERMI_30_DUMMY_CODDING	200	adaptive	0,004	0,9	67,358
UNDERMI_40_DUMMY_CODDING	250	adaptive	0,004	0,5	67,0549
BRUTMI_20_DUMMY_CODDING	350	invscaling	0,005	0,633	71,1282
BRUTMI_30_DUMMY_CODDING	300	invscaling	0,003	0,9	69,6026
BRUTMI_40_DUMMY_CODDING	250	adaptive	0,002	0,5	70,1026
OVERCHI2_20_DUMMY_CODDING	200	adaptive	0,004	0,5	68,1684
OVERCHI2_30_DUMMY_CODDING	350	constant	0,005	0,9	68,3811
OVERCHI2_40_DUMMY_CODDING	250	adaptive	0,005	0,5	69,2507
UNDERCHI2_20_DUMMY_CODDING	300	invscaling	0,003	0,76	68,5795
UNDERCHI2_30_DUMMY_CODDING	300	adaptive	0,004	0,76	67,6705
UNDERCHI2_30_DUMMY_CODDING	250	constant	0,005	0,76	67,3201
BRUTCHI2_20_DUMMY_CODDING	350	invscaling	0,005	0,5	71,1282
BRUTCHI2_30_DUMMY_CODDING	350	invscaling	0,003	0,63	70,8654

Tableau 5:27 meilleurs modèles suivant leurs hyperparamètres et accuracy pour le modèle MLP.

DATASET	GAMMA	'C'	KERNEL	ACCURACY %
OVERMI_20_DUMMY_CODDING	scale	2,55555	poly	70,7678
OVERMI_30_DUMMY_CODDING	scale	3	poly	68,1776
OVERMI_40_DUMMY_CODDING	scale	1,4444	linear	69,4773
UNDERMI_20_DUMMY_CODDING	scale	1,444	poly	70,1136
UNDERMI_30_DUMMY_CODDING	auto	2,111	poly	67,3864
UNDERMI_40_DUMMY_CODDING	auto	2,1111	rbf	66,7614
BRUTMI_20_DUMMY_CODDING	scale	1,888	poly	68,9377
BRUTMI_30_DUMMY_CODDING	scale	2,5555	poly	68,6154
BRUTMI_40_DUMMY_CODDING	auto	1	poly	67,5897
OVERCHI2_20_DUMMY_CODDING	scale	3	poly	69,2553
OVERCHI2_30_DUMMY_CODDING	scale	3	poly	68,1776
OVERCHI2_40_DUMMY_CODDING	scale	1,2222	poly	67,7197
UNDERCHI2_20_DUMMY_CODDING	scale	1	poly	67,6799
UNDERCHI2_30_DUMMY_CODDING	auto	1	poly	66,7519
UNDERCHI2_30_DUMMY_CODDING	auto	1,2222	poly	65,8239
BRUTCHI2_20_DUMMY_CODDING	scale	1,2222	poly	67,1816
BRUTCHI2_30_DUMMY_CODDING	scale	2,555	poly	65,7789

Tableau 6:27 meilleurs modèles suivant leurs hyperparamètres et accuracy pour le modèle SVM.

Résultats

I. Présentation :

Dans cette partie nous allons présenter les différents résultats obtenus avec les différents modèles.

Pour le premier modèle (KNN), nous avons choisi les 5 meilleures datasets selon les 3 métriques (accuracy, précision ,sensitivité),et fonction de leurs hyperparamètres on utilisant la méthode de borda count, comme le montre les tableaux suivant :

1.1 KNN

Dataset	*n_neighbors*	*weights*	*Accuracy*	*Accuracy %*	*Rank*
overVAR_40	2	distance	0,699167	69,9167	1
overChi2_40	8	distance	0,68617	68,617	2
overMI_20	6	distance	0,68173	68,173	3
overChi2_20	6	distance	0,68173	68,173	3
overMI_40	6	distance	0,677567	67,7567	5

Tableau 7: Meilleur Accuracy KNN

Dataset	*N_Neighbors*	*Weights*	*Précision*	*Précision %*	*Rank*
overVAR_40	2	uniform	0,718281	71,8281	1
underChi2_20	4	uniform	0,697497	69,7497	2
underMI_20	4	uniform	0,697497	69,7497	2
overChi2_20	6	distance	0,68173	68,173	3
overChi2_40	8	distance	0,68617	68,617	4

Tableau 8:Meilleur Precision KNN

dataset	*n_neighbors'*	*weights*	*sensitivité*	*AUC_ROC%*	*Rank*
overVAR_40	2	distance	0,723981	72,3981	1
overChi2_40	8	distance	0,68617	68,617	2
overChi2_20	6	distance	0,68173	68,173	3
overMI_20	6	distance	0,68173	68,173	3
overVAR_30	4	distance	0,679579	67,9579	5

Tableau 9:Meilleur AUC_ROC KNN

On suite, nous avons procéder à faire le vote entre ces modèles pour choisir celui qui réalise le meilleur score.

Meilleur modèle :

Dataset	***N_Neighbors***	***Accuracy %***	***Précision %***	***Sensitivité%***	***Rank***
OverVAR_40	2	69,9167	71,8281	72,3981	1

Tableau 10:Meilleur Modèle KNN

Le tableau ce dessous représente le sous ensemble qui a réalisé le meilleur résultat, et qui montre les variables/ attributs, qui ont augmenté la précision du modèle.

	Medu	Fedu	famrel	freetime	goout	Walc	health	Age	Absences	Mjob	Fjob	reason
0	4	4	4	3	4	1	3	4	2	3	0	2
1	1	1	5	3	3	1	3	3	1	3	4	2
2	1	1	4	3	2	3	3	1	2	3	4	3
3	4	2	3	2	2	1	5	1	1	1	2	0
4	3	3	4	3	2	2	5	2	1	4	4	0
...	...	...	...	...	...	...	...	...	...	...	...	...
457	2	2	4	3	3	2	4	2	2	4	3	1
458	2	1	4	4	4	4	5	2	1	2	4	2
459	3	4	4	3	3	1	4	1	1	2	0	2
460	2	1	4	3	4	1	3	4	1	4	4	1
461	2	4	4	2	2	1	5	1	1	2	0	2

462 rows × 12 columns

Tableau 11:DataSet qui réalise le meilleur score en KNN

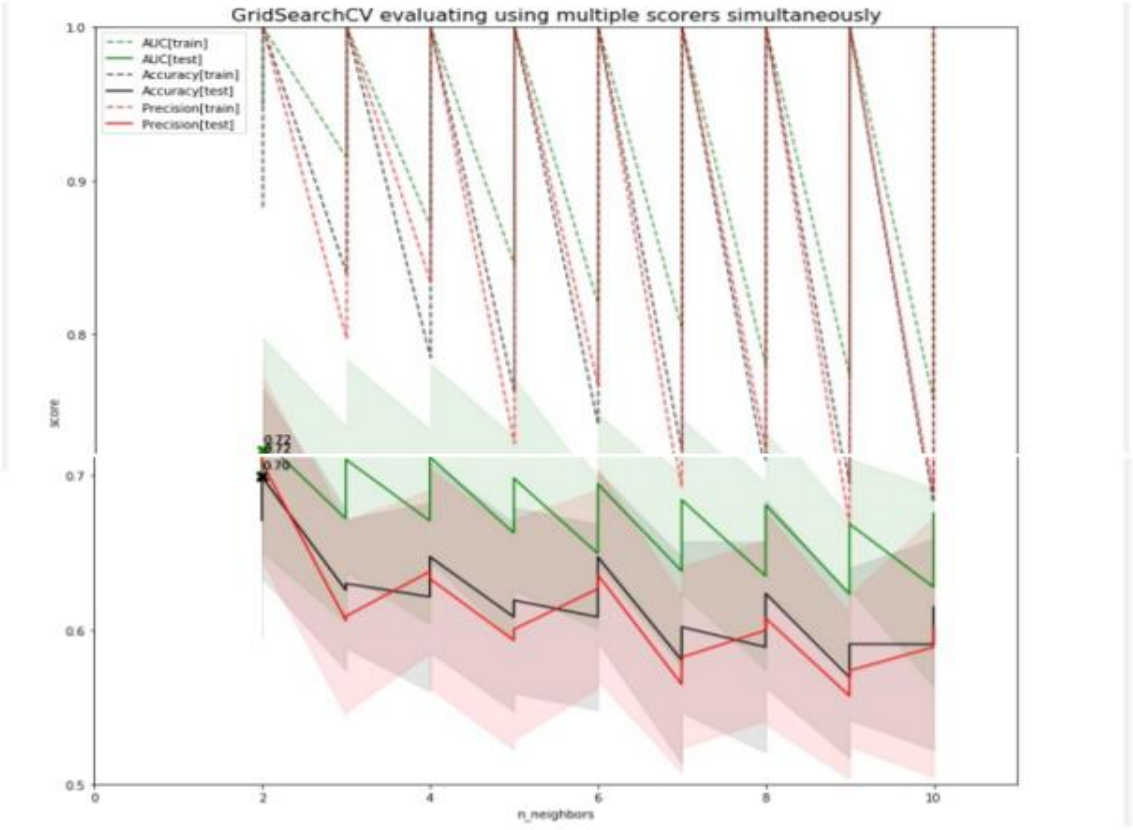

Figure 4: Représentation graphique du meilleur modèle en KNN

on remarque que tant k augmente les differents métrique déminues.

NB : on avons utiliser la même procédure avec les autres modèles apprentissage

1.2. CART

Dataset	Max_Depth	Max_Features	Min_Samples_Leaf	Accuracy%	Rank
underChi2_20	2	5	5	68,9015	1
underMI_20	2	5	5	68,9015	1
underChi2_40	2	5	7	68,9015	1
underMI_40	2	5	7	68,9015	1

Tableau 12:Meilleur accuracy en CART

Dataset	Max_Depth	Max_Features	Min_Samples_Leaf	Precison%	Rank
underChi2_20	2	4	5	83,6061	1
underMI_20	2	4	5	83,6061	2
underChi2_30	2	5	5	82,6667	3
underMI_30	2	5	5	82,6667	4
brutChi2_30	3	5	5	81,2846	5

Tableau 13:Meilleur precision en CART

Dataset	Max_Dept Merth	Max_Features	Min_Samples_Leaf	Auc_Roc%	Rank
underMI_30	4	5	9	72,9121	1
underMI_20	4	6	5	72,7426	2
underChi2_20	4	6	5	72,7426	2
underChi2_30	4	5	10	72,6753	3
brutMI_30	4	5	5	72,3193	4

Tableau 14:Meilleur AUC_ROC en CART

Les 3 tableaux montrent le score du trois métriques en fonction des hyperparamètres, en choisissant les cinq meilleurs modèles qui réalisent le score le plus élevé.

Meilleur modèle :

Dataset	Max_Depth	AUC_ROC %	Precison%	Accuracy%	Rank
underChi2_20	3	72,7426	83,6061	68,9015	1

Tableau 15: Le meilleur modèle du CART

	failures	Fedu	schoolsup	Medu	goout	freetime
0	0	2	0	1	3	2
1	0	1	0	2	1	4
2	0	3	0	2	2	2
3	0	4	0	4	4	2
4	2	2	0	4	3	4
...	...	...	...	...	...	...
323	0	1	0	3	4	3
324	1	1	0	1	1	1
325	2	2	0	2	4	5
326	3	1	0	1	3	5
327	0	1	0	1	3	2

328 rows × 6 columns

Tableau 16: DataSet Correspendante au meilleur modèle CART

Le tableau ce dessus représente le sous ensemble qui a réalisé le meilleur résultat, et qui montre les variables/ attributs, qui ont augmenté la précision du modèle.

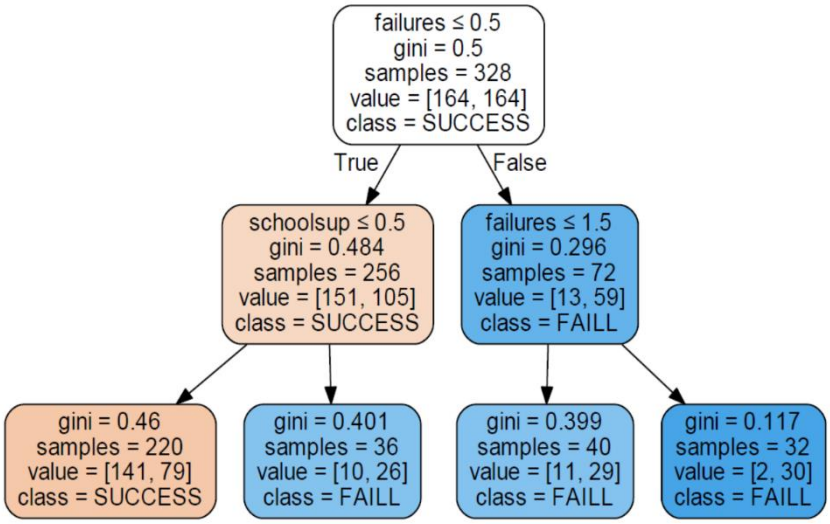

Figure 5:Reprèsentation graphique du modèle CART

1.3. L'utilisation de alpha pour élaguer l'arbre

Dataset	**Ccp_Alpha**	**Score %**	**Rank/Score**	**Rank/Ccp Alpha**
UNDERMI_20	0,014497	67,964	3	2
BRUTCHI2_30	0,0100115	67,2245	5	9
BRUTCHI2_40	0,0100115	67,2245	5	9
UNDERMI_40	0,022528	67,0833	9	1

Meilleur modèle :

Dataset	Ccp_Alpha	Score %	Rank/Score	Rank/Ccp Alpha
Undermi_20	0,014497	67,964	3	2

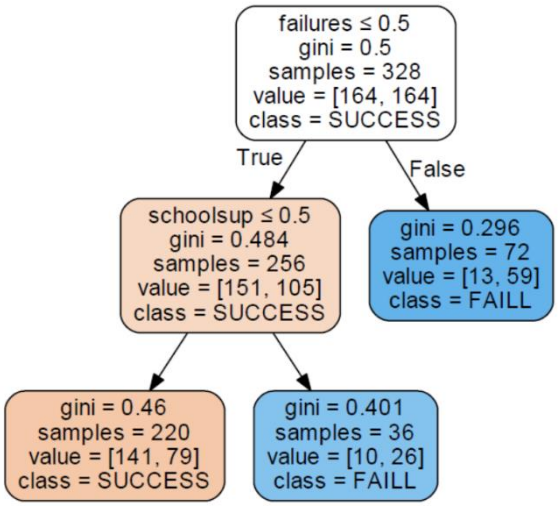

Figure 7: Modèle CART après l'élagage

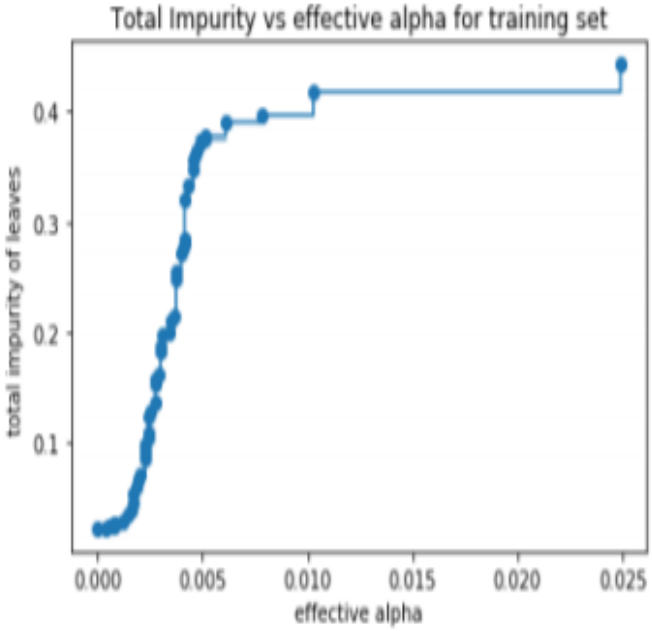

Figure 6: L'impureté en fonction de alpha

On remarque que on a obtenue presque les mêmes graphes que ça soit avec les paramètres standards ou le Ccp_alpha même si avec deux modèles différents.

1.4. MLP

Dataset	Batch_Size	Learning_Rate	Momentum	Accuracy%	**RANK**
Brutmi_20_Dummy	350	Invscaling	0,633	71,1282	1
Brutchi2_20_Dummy	350	Invscaling	0,5	71,1282	1
Brutchi2_30_Dummy	350	Invscaling	0,63	70,8654	3
Overmi_40_Dummy	200	Constant	0,9	70,7632	4
Brutmi_40_Dummy	250	Adaptive	0,5	70,1026	5

Tableau 17: Meilleur accuracy en MLP

Dataset	Batch_Size	Learning_Rate	Momentum	Précision %	**RANK**
brutChi2_30_dummy	350	invscaling	0,63	85	1
brutChi2_40_dummy	350	constant	0,63	82,1587	2
brutMI_20_dummy	350	invscaling	0,633	79,3929	3
brutMI_30_dummy	300	invscaling	0,9	78,5195	4
brutMI_40_dummy	250	adaptive	0,5	0,76316	5

Tableau 18:Meilleur precision en MLP

Dataset	Batch_Size	Learning_Rate	Momentum	AUC_ROC %	Rank
overMI_40_dummy	200	constant	0,9	77,596	1
overChi2_40_dummy	250	invscaling	0,9	76,0172	2
brutMI_40_dummy	250	adaptive	0,5	74,9339	3
brutMI_20_dummy	350	invscaling	0,633	74,322	4
overMI_30_dummy	350	adaptive	0,9	74,3028	5

Tableau 19: Meilleur MLP en AUC_ROC

Les 3 tableaux montrent le score du trois métriques en fonction des hyperparamètres, en choisissant les cinq meilleurs modèles qui réalisent le score le plus élevé.

Meilleur modèle :

Dataset	precision %	AUC_ROC %	Accuracy%	Rank
BrutMi_20_Dummy	79,3929	74,322	71,1282	1

Tableau 20: Meilleur Modèle MLP

1.5. SVM

Dataset	Gamma	'C'	Kernel	Accuracy %	Rank
overMI_20_dummy	scale	2,55555	poly	70,7678	1
underMI_20_dummy	scale	1,444	poly	70,1136	2
overMI_40_dummy	scale	1,4444	linear	69,4773	3
overChi2_20_dummy	scale	3	poly	69,2553	4
brutMI_20_dummy	scale	1,888	poly	68,9377	5

Tableau 21: Meilleur modèle SVM en accuracy

Dataset	Gamma	'C'	Kernel	Precision%	Rank
underMI_20_dummy	scale	1,666	poly	76,2417	1
underMI_30_dummy	auto	1	poly	75,4952	2
overChi2_40_dummy	scale	1,666	poly	74,9355	3
brutChi2_20_dummy	scale	2,333	poly	74,4053	4
brutChi2_30_dummy	scale	1,444	poly	74,0874	5

Tableau 22:Meilleur modèle SVM en Precision

Dataset	Gamma	'C'	Kernel	AUC_ROC%	Rank
underMI_20_dummy	scale	1,6666	poly	76,2417	1
underChi2_20_dummy	scale	1	poly	76,2417	2
underMI_30_dummy	auto	1	poly	75,4952	3
underChi2_30_dummy	auto	1,666	poly	75,4952	4
underChi2_30_dummy	auto	1	poly	73,6503	5

Tableau 23:Meilleur modèle SVM en AUC_ROC

Les 3 tableaux montrent le score du trois métriques en fonction des hyperparamètres, en choisissant les cinq meilleurs modèles qui réalisent le score le plus élevé.

Meilleur modèle :

dataset	**'C'**	**AUC_ROC%**	**Precision%**	**accuracy %**	**RANK**
underMI_20_dummy	1,444	76,2417	76,2417	70,1136	2

Tableau 24: Le meilleur modèle du SVM

II. Discussion :

Algorithmes	***Dataset***	***Accuracy %***	***Précision %***	**AUC_ROC%**	**Rank**
KNN	OverVAR_40	69,9167	71,8281	72,3981	4
CART	underChi2_20	72,7426	83,6061	68,9015	1
MLP	BrutMi_20_Dummy	79,3929	74,322	71,1282	2
SVM	underMI_20_dummy	76,2417	76,2417	70,1136	3

Le tableau ce dessus résume les quatre meilleurs modèles en utiliser la méthode du borda count.

Le meilleur modèle de précision globale est le CART, qui identifie correctement 83,6% des élèves (échoués et réussis combinés). Ce modèle est également le meilleur pour prédire les étudiants qui ont réussi le cours.

Le deuxième meilleur modèle global et pour identifier les étudiants qui ont réussi le cours est le réseau de neurones à perceptron multi-couches (MLP) avec une précision globale de 74,3%, ce qui est un peu proche du KNN et SVM mais avec une accuracy plus élevée 79%. Le support vecteur machine (SVM) est le troisième meilleur modèle pour identifier les étudiants qui ont réussi le cours correctement, ce qui est très similaire au MLP. Cependant, la précision du KNN arrive en dernier classement.

En conséquence, il peut être possible de dire que les taux de précision ont changé positivement dans tous les essais utilisant la méthode de sélection d'attributs.

III. Limites d'étude

Au cours de notre analyse, nous avons constaté qu'il était difficile de prédire avec une grande précision la classe finale, quel que soit le type de modèle que nous construisions, contrairement aux études effectués par d'autre auteurs, cela dû a ce que la plupart des études n'ont pas supprimer les deux attributs G1 et G2 qui sont corrélés avec la note finale.

CONCLUSION

Ce projet propose l'application de techniques d'exploration de données pour prédire la classe finale (FAIL/SUCESS) des élèves en fonction de leurs données historiques. Quatre techniques de classification bien connues (KNN, arbre de décision (CART), MLP et SVM) ont été comparées en termes de taux d'exactitude. Les opérations de prétraitement sur l'ensemble de données, en classant le champ de note finale deux classes, ont augmenté le pourcentage d'estimations précises dans la classification. La méthode de sélection des attributs dans tous les algorithmes a conduit à une augmentation notable du taux de précision. Dans l'ensemble, de meilleurs taux d'exactitude ont été obtenus avec les deux algorithmes arbre de décision (CART), et MLP.

Cette étude aidera les étudiants et les enseignants à améliorer les performances des étudiants. Qui présente aux étudiants des approches de développement de modèles de prédiction basées sur des sources de données historique de l'étudiant.

A l'avenir, différentes méthodes de sélection des attributs pourront être utilisées. En outre, différents algorithmes de classification peuvent également être utilisés sur les ensembles de données, ou faire appel au deep learning pour une meilleure précision.

Bibliographie :

[1] www.wikipedia.org

[2] https://machinelearningmastery.com/

[3] https://scikit-learn.org/stable/

[4] https://towardsdatascience.com/

[5] The Hundred-Page Machine Learning Book,Author – Andriy Burkov

[6] The Elements of Statistical Learning: Data Mining, Inference, and Prediction