Rapport de projet

KOSTOV Miléna, JOUDET Sami

1 Introduction

Le problème consiste à étudier l'impact des études en distanciel sur les résultats des élèves. Il s'agit d'un problème important car les études en distanciel ont concerné des millions d'élèves et étudiants lors de la pandémie. De plus, la tendance d'avoir des cours en distanciel pourrait se généraliser et il serait intéressant pour les élèves de pouvoir accéder sur Internet à des études déjà existantes. Nous nous sommes intéressés à ce problème car nous sommes étudiants et il s'agit d'une situation qui nous a touché personnellement lors de la pandémie. Par conséquent nous avons vécu les difficultés à étudier en ligne voire à passer des examens en distanciel.

L'entrée de notre algorithme est issue d'une base de données. Dans celle-ci on peut trouver des informations personnelles de chaque élève (âge, sexe, notes avant et après le passage au distanciel etc...) ainsi que les réponses d'un questionnaire portant sur les habitudes (sommeil, temps de travail etc...) et le ressenti sur le distanciel. Nous allons utiliser un SVM (Support Vector Machine) linéaire (classification binaire) pour prédire la performance des élèves en distanciel et essayer d'identifier les élèves ayant le plus de risque d'être en échec.

D'autres méthodes seront utilisées afin d'évaluer la pertinence du modèle choisi : clustering k-means, SVM avec noyau RBF (Radial Basis Function) et SVM linéaire multi-classe.

2 Ensemble de données et fonctionnalités

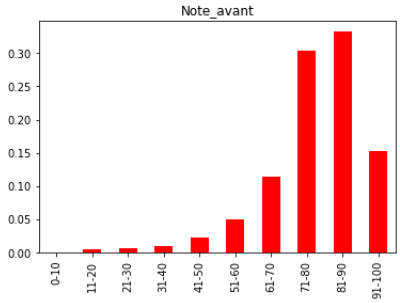
La base de données utilisée pour ce projet est originaire du site Kaggle qui est un site hébergeant de nombreuses bases de données sur des sujets divers et variés : online-education-system-review. Cette base de données comporte 23 variables au total. Aucune donnée n'est manquante. La valeur que l'on essaie de prédire est la note obtenue par les différents élèves dans un examen en ligne (en distanciel).

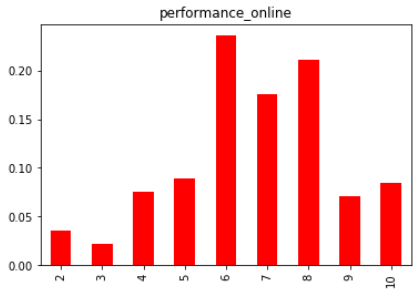
Cette base de données étant non travaillée et non normalisée, de nombreux ajustements ont dû être réalisés avant qu'elle soit exploitable :

— Normalisation des variables catégorielles et des variables numériques. Cas particulier : la variable "Average marks scored before pandemic in traditional classroom" est passée d'une variable catégorielle composée d'intervalles des notes moyennes des élèves à une variable numérique discrète décrivant un gradient de notes, ce qui n'entraîne pas de perte d'information.

— Élimination des variables bi-catégorielles doublons.

Dans le cas de la valeur à prédire, celle-ci étant une valeur numérique discrète, nous avons dû choisir la valeur de "réussite" ou "d'échec" de l'examen en ligne.





Les constats faits à partir des graphiques ci-dessus sont les suivants :

— Les notes avant le distanciel sont majoritairement au-dessus de l'intervalle 61-70.

— Les notes obtenues lors de l'examen en distanciel sont bien inférieures à celles obtenues avant : environ 50% des élèves ont obtenu 6 ou moins.

Ainsi, la colonne avec les notes obtenues en distanciel jouera le rôle de variable y dans cette étude.

3 Méthodes

Un tableau de corrélation a été utilisé afin d'avoir une idée sur les critères qui ont à priori une plus grande corrélation par rapport à la note obtenue en distanciel. Cette étape a permis de sélectionner des sous-groupes afin de comparer la précision moyenne générale et les précisions moyennes de chaque sous-groupe choisi.

SVM (Support Vector Machine)

C'est un algorithme d'apprentissage supervisé appelé Machines à Vecteurs de Support. Il permet de séparer des données selon une classe y définie. Le SVM vise à trouver la frontière qui maximise la marge entre les classes. La marge est la distance entre l'hyperplan et les données les plus proches. Ces dernières sont appelés vecteurs supports. La méthode SVM cherche à maximiser, en fonction de ω, b, la quantité :

$$\arg\max_{\omega,b} \left\{ \min_i d(x_i, H) = \frac{1}{\|\omega\|} \min_i y_i(\omega^T x + b) \right\} \tag{1}$$

Il existe plusieurs méthodes SVM :

1. *Classification binaire.* Il s'agit d'un problème de discrimination à deux classes, c'est-à-dire $y \in \{-1, +1\}$, le vecteur d'entrée x étant dans un espace X muni d'un produit scalaire.
   L'objectif est de trouver un hyperplan $\omega^T x + b = 0$ qui vise à séparer, si possible, les deux classes. Les objets futurs seront classés de la manière suivante :

$$y = +1 \quad \text{si} \quad \omega^T x + b > 0 \tag{2}$$

$$y = 0 \quad \text{si} \quad \omega^T x + b < 0 \tag{3}$$

   Au vu des constats faits précédemment à partir du graphique, on a pris la décision de considérer 6 comme étant la note pivot avec :
   — Si $y \in [2, 6]$, la nouvelle valeur de y sera 0 (classe négative).
   — Si $y \in]6, 10]$, la nouvelle valeur de y sera 1 (classe positive).
   On analyse aléatoirement plusieurs fois la base de données divisée de la façon suivante : 50% entraînement, 25% validation, 25% test. Puis on regarde la précision moyenne et la variance de précision.

2. *Noyau RBF (Radial Basis Function).* Les noyaux RBF sont la forme la plus généralisée de noyautage et l'un des noyaux les plus largement utilisés en raison de sa similitude avec la distribution gaussienne. La fonction noyau RBF pour deux points X_1 et X_2 calcule la similitude ou la proximité entre eux. Ce noyau peut être représenté mathématiquement comme suit :

$$K(X_1, X_2) = \exp\left(-\frac{\|X_1 - X_2\|^2}{2\sigma^2}\right) \tag{4}$$

   La pertinence dépend des deux paramètres suivants :
   — Gamma : inverse de l'écart-type du noyau RBF (fonction gaussienne), qui est utilisé comme mesure de similitude entre deux points.
   — C : paramètre de régularisation.
   Grâce à la fonction RandomizedSearchCV on utilise Gamma et C optimaux dans un intervalle choisi (param_grid). Ceci permet de ne pas avoir un groupe test et la base de donnée est cette fois-ci découpée de la manière suivante : 70% entraînement, 30% validation.

3. *Classification multi-classe.* On utilise l'approche One-versus-rest (OVR)
   Supposons que nos classifieurs soient des régressions logistiques. Alors $f_k(x) = p(Y = k|x)$, il est logique de prédire que x appartient à la classe pour laquelle cette prédiction est la plus élevée. Dans le cas des SVM, $f_k(x)$ indique la distance entre x et l'hyperplan qui sépare la classe

k des autres. Plus cette valeur est élevée, plus nous sommes convaincus que x appartient à la classe k. Nous allons donc simplement prédire la classe pour laquelle la fonction de décision retourne la valeur la plus élevée : $f(x) = \arg\max_k f_k(x)$.

Dans cette étude et avec cette méthode on a $y \in \{2, 3, 4, 5, 6, 7, 8, 9, 10\}$. On analyse aléatoirement plusieurs fois la base de données divisée de la façon suivante : 50% entraînement, 25% validation, 25% test. Puis on regarde la précision moyenne et la variance de précision.

Clustering k-means

C'est un algorithme d'apprentissage non-supervisé mettant en évidence les groupes "naturels" c'est à dire ceux qui se démarquent significativement les uns des autres. K-means est un algorithme de regroupement itératif dont le principe est le suivant :

— Initialisation : choix des K points au hasard comme centres des clusters.
— Réitération :

1. Affectation des points de données au centre du cluster le plus proche.
2. Changer le centre de chaque cluster pour la moyenne de ses points affectés.

— Condition d'arrêt : pour chaque centre plus aucun changement d'affectation.

L'inertia (aussi appelé Within-Cluster Sum of Squares) est la somme des carrés des distances des données à leur centre de cluster le plus proche. Elle permet de trouver le nombre de clusters optimal K.

$$\text{Inertia} = \sum_{i=1}^{k} \sum_{x_j \in C_i} \|x_j - \mu_i\|^2 \tag{5}$$

La méthode du coude permet de trouver le K optimal. Grâce au k trouvé avec le tracé du coude on sait quel est le nombre de clusters à partir duquel ajouter au moins un cluster supplémentaire n'est plus suffisament intéressant.

4 Expériences/Résultats/Discussion

Clustering

Les deux courbes du milieu comparent les mêmes paramètres avec un nombre de clusters différents. La localisation des centres reste sensiblement la même. Sur les courbes aucun cluster n'est clairement délimité. Ce résultat tend à montrer qu'utiliser un apprentissage non supervisé pour cette base de données ne permet pas d'obtenir des résultats concluants.

Donc l'idée d'utiliser un apprentissage supervisé est renforcée.

Observations faites à partir du tableau de corrélation

Cette étape permet d'avoir une idée quels critères auraient une plus grande corrélation par rapport à la note obtenue en ligne. En raison de la grande taille du tableau celui-ci sera disponible en annexe dans le code fourni.

Les critères ayant la plus grande corrélation sont les suivants :

— Le temps d'étude (un peu moins important).
— L'âge (un peu moins important).
— La qualité de la connexion Internet.
— Lever les doutes en ligne.
— La satisfaction du mode en ligne.

SVM *classification binaire*

```
matrice = np1.array(reapeated_random(10,df_1))
print('')
final_accuracy(matrice)
```

```
0123456789
La précision moyenne est  68.34
La variance de précision est 4.59
```

Base de données entière

```
matrice = np1.array(reapeated_random(10,df_1))
print('')
final_accuracy(matrice)
```

```
0123456789
La précision moyenne est  69.73
La variance de précision est 7.94
```

```
df_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1033 entries, 0 to 1032
Data columns (total 8 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   Age                           1033 non-null   float64
 1   Internet                      1033 non-null   float64
 2   Study                         1033 non-null   float64
 3   Online_mode                   1033 non-null   float64
 4   doubts_online                 1033 non-null   float64
 5   Satisfaction_online_Average   1033 non-null   uint8
 6   Satisfaction_online_Bad       1033 non-null   uint8
 7   Satisfaction_online_Good      1033 non-null   uint8
dtypes: float64(5), uint8(3)
memory usage: 43.5 KB
```

```
df_2 = df_1[["Study","Sleep","Media","Note_avant"]].copy()
matrice = np1.array(reapeated_random(10,df_2))
print('')
final_accuracy(matrice)
```

```
0123456789
La précision moyenne est  55.98
La variance de précision est 6.83
```

Base de données avec les critères ayant le moins d'influence

```
df_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1033 entries, 0 to 1032
Data columns (total 5 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Internet                  1033 non-null   float64
 1   Online_mode               1033 non-null   float64
 2   doubts_online             1033 non-null   float64
 3   Satisfaction_online_Bad   1033 non-null   uint8
 4   Satisfaction_online_Good  1033 non-null   uint8
dtypes: float64(3), uint8(2)
memory usage: 26.4 KB
```

```
matrice = np1.array(reapeated_random(10,df_1))
print('')
final_accuracy(matrice)
```

```
0123456789
La précision moyenne est  69.88
La variance de précision est 6.74
```

Après avoir isolé les critères repérés grâce au tableau de corrélation on peut comparer la précision moyenne entre la base de données dans sa totalité et les critères ayant la plus grande corrélation avec la note obtenue à l'examen en ligne. L'observation faite précédemment avec le tableau des corrélations semble être confirmée.

De plus, en cherchant à visualiser les différentes répartitions par couple de critères, on a eu l'idée de faire de même pour des triplets, et ainsi de nouveaux critères sont apparus systématiquement. D'où l'idée de créer un nouveau groupe afin de voir l'impact de ces critères sur la précision moyenne. Le résultat est qu'il s'agit de critères ayant le moins d'impact sur le résultat que les élèves obtiennent dans un examen en ligne. Il est intéressant de noter que la moyenne des notes obtenues avant les cours en distanciel fait partie de ce groupe.

SVM *classification multi-classe*

```
matrice = np1.array(reapeated_random(10,df_1))
print('')
final_accuracy(matrice)
```

```
0123456789
La précision moyenne est  66.91
La variance de précision est 10.94
```

```
df_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1033 entries, 0 to 1032
Data columns (total 8 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   Age                           1033 non-null   float64
 1   Internet                      1033 non-null   float64
 2   Study                         1033 non-null   float64
 3   Online_mode                   1033 non-null   float64
 4   doubts_online                 1033 non-null   float64
 5   Satisfaction_online_Average   1033 non-null   int64
 6   Satisfaction_online_Bad       1033 non-null   int64
 7   Satisfaction_online_Good      1033 non-null   int64
dtypes: float64(5), int64(3)
memory usage: 72.6 KB
```

```
df_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1033 entries, 0 to 1032
Data columns (total 5 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Internet                  1033 non-null   float64
 1   Online_mode               1033 non-null   float64
 2   doubts_online             1033 non-null   float64
 3   Satisfaction_online_Bad   1033 non-null   int64
 4   Satisfaction_online_Good  1033 non-null   int64
dtypes: float64(3), int64(2)
memory usage: 48.4 KB
```

```
matrice = np1.array(reapeated_random(10,df_1))
print('')
final_accuracy(matrice)
```

```
0123456789
La précision moyenne est  68.53
La variance de précision est 15.60
```

On remarque que le groupe des critères améliorant la précision reste le même. C'est également le cas pour les critères ayant la pire précision. Ce résultat tend à confirmer les observations faites avec le modèle SVM linéaire avec deux classes.

Les précisions des différents groupes repérés précédemment sont un peu moins élevées et les variances de précisions sont plus importantes avec ce modèle. Cela donne comme impression que le modèle SVM linéaire avec deux classes est pour le moment le meilleur choix.

```
matrice = np1.array(reapeated_random(10,df_1))
print('')
final_accuracy(matrice)
```

```
0123456789
La précision moyenne est  65.56
La variance de précision est 1.66
```

Base de données entière

```
df_2 = df_1[["Study","Sleep","Media","Note_avant"]].copy()
matrice = np1.array(reapeated_random(10,df_2))
print('')
final_accuracy(matrice)
```

```
0123456789
La précision moyenne est  48.03
La variance de précision est 15.69
```

Base de données avec les critères ayant le moins d'influence

SVM *Noyau RBF*

```
matrice = np1.array(reapeated_random(10,df))
print('')
final_accuracy(matrice)
```

```
0123456789
La précision moyenne est  79.49
La variance de précision est 5.76
```

Base de données entière

```
df_2 = df[["Study","Sleep","Media","Note_avant"]].copy()
matrice = np1.array(reapeated_random(10,df_2))
print('')
final_accuracy(matrice)
```

```
0123456789
La précision moyenne est  56.49
La variance de précision est 3.20
```

Base de données avec les critères ayant le moins d'influence

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1033 entries, 0 to 1032
Data columns (total 8 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Age                         1033 non-null   float64
 1   Internet                    1033 non-null   float64
 2   Study                       1033 non-null   float64
 3   Online_mode                 1033 non-null   float64
 4   doubts_online               1033 non-null   float64
 5   Satisfaction_online_Average 1033 non-null   int64
 6   Satisfaction_online_Bad     1033 non-null   int64
 7   Satisfaction_online_Good    1033 non-null   int64
dtypes: float64(5), int64(3)
memory usage: 72.6 KB
```

```
matrice = np1.array(reapeated_random(10,df))
print('')
final_accuracy(matrice)
```

```
0123456789
La précision moyenne est  70.18
La variance de précision est 1.74
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1033 entries, 0 to 1032
Data columns (total 5 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Internet                 1033 non-null   float64
 1   Online_mode              1033 non-null   float64
 2   doubts_online            1033 non-null   float64
 3   Satisfaction_online_Bad  1033 non-null   int64
 4   Satisfaction_online_Good 1033 non-null   int64
dtypes: float64(3), int64(2)
memory usage: 48.4 KB
```

```
matrice = np1.array(reapeated_random(10,df))
print('')
final_accuracy(matrice)
```

```
0123456789
La précision moyenne est  70.33
La variance de précision est 0.97
```

Les observations faites plus tôt sur l'influence des précisions moyennes par les différents groupes de critères sont de manière générale à nouveau confirmées. L'impact des critères permettant d'obtenir une précision moyenne plus élevée est moins évident à voir. Pour tous les groupes les précisions moyennes sont meilleures que celles obtenues avec les modèles précédents. Par conséquent, ce modèle est le plus efficace pour traiter cette base de données.

5 Conclusion/Futur travail

A travers 3 méthodes différentes SVM on a pu mettre en évidence que les critères suivants ont une forte influence sur la note obtenue à l'examen en ligne :

— La qualité de la connexion Internet.
— Lever les doutes en ligne.
— La satisfaction du mode en ligne.

De manière inattendue la moyenne des notes avant le distanciel n'a que peu d'influence sur la note en ligne. L'algorithme le plus efficace est le SVM à noyau RBF.

La raison pour la plus grand fiabilité du noyau RBF comparé au noyau linéaire pourrait s'expliquer par l'enchevêtrement complexe des données qui est alors difficile de séparer par un simple hyperplan. La raison des mauvaises notes est multi-factorielle et peut être due à des causes différentes, potentiellement décrites par des valeurs différentes dans la base de données. La prochaine piste possible à explorer serait d'utiliser le noyau RBF et de trouver les différentes variables qui influencent le plus l'échec ou la réussite des étudiants dans un contexte de cours en distanciel pour pouvoir au mieux les aider.

6 Contribution

Sami :
— Normalisation (normaliser.ipynb) de la base de données (résultat : educnorma.csv)
— Écriture du code dans les fichiers : SVM_linear_simple.ipynb, svm_rbf.ipynb, SVM_linear_multi-categorie.ipynb, svm_linear_parametre_c.ipynb, svm_linear_parametre_c2.ipynb et README.md.
— Organisation au propre des fichiers contenant le code.
— Rédaction avant correction des parties du rapport :
  2. Ensemble de données et fonctionnalités.
  3. Méthodes, SVM intro et 1.

Miléna :
— Choix de la base education.csv.
— Réécriture noms des colonnes (normaliser.ipynb)
— Écriture du code dans les fichiers : SVM_linear_simple.ipynb, visu.ipynb, K_means.ipynb, visu_2.ipynb.
— Rédaction des parties du rapport :
  1. Introduction.
  2. Ensemble de données et fonctionnalité.
  3. Méthodes.
  4. Expériences/Résultats/Discussion.
  5. Conclusion.
— Correction des parties et mise en page.

7 Références/Bibliographie

Cours de cette année : https://github.com/mesin-cours/methodes-simulation-informatique

https://www.kaggle.com/datasets/sujaradha/online-education-system-review

https://scikit-learn.org/stable/auto_examples/applications...

openclassrooms

https://chrisalbon.com/code/machine_learning/support_vector_machines/svc_parameters_using_rbf_kernel/

Librairies utilisées :
— pandas.
— numpy.
— sklearn.svm.
— matplotlib.pyplot.
— sklearn.datasets.
— sklearn.metrics.
— sklearn.model_selection.RandomizedSearchCV.
— sklearn.utils.fixes.loguniform.
— mpl_toolkits.mplot3d.Axes3D.
— sklearn.cluster.KMeans.