Devoir à rendre au plus tard le 4/1/2026 – Explainable AI (XAI)

vous devez compléter les exercices portant sur l'explicabilité des modèles d'apprentissage automatique. Pour chaque exercice :

- Fournissez **du code fonctionnel** utilisant des méthodes XAI (par ex. SHAP, LIME, ALE) pour expliquer les prédictions d'un modèle.
- Présentez **les résultats obtenus** avec des visualisations ou sorties pertinentes.
- Fournissez des **interprétations détaillées** pour chaque résultat, expliquant l'influence des variables sur les prédictions et la logique derrière vos observations.
- Commentez clairement votre code et structurez votre devoir pour qu'il soit compréhensible et complet,

Exercice 1 : Analyse des effets locaux et globaux sur le dataset German Credit

Objectif

Interpréter un modèle Random Forest appliqué au dataset German Credit et approfondir l'analyse à l'aide des méthodes **PDP** et **ALE**. L'accent sera mis sur :

- Analyse des effets de **variables continues et catégorielles**.
- Détection d'**interactions** entre variables.
- Identification de biais potentiels et effets non linéaires.

1. Analyse des variables continues avec PDP et ALE

Variables à considérer : age, amount, duration.

1. Tracez le PDP pour amount. Identifiez toute **non-linéarité**.
2. Tracez l'ALE pour amount. Comparez avec le PDP et expliquez les différences.
3. Calculez et comparez les **effets moyens** par tranche (quartiles) pour amount et duration.
4. Comment les **interactions potentielles** avec credit_history pourraient-elles biaiser le PDP ?
5. Proposez une méthode pour **quantifier l'importance des interactions** avec ALE.

2. Analyse des variables catégorielles avec PDP et ALE

Variables à considérer : personal_status_sex, employment_duration, purpose.

6. Tracez le PDP pour personal_status_sex. Quelles limites observez-vous pour les variables catégorielles avec plusieurs niveaux ?
7. Tracez l'ALE pour personal_status_sex. Comparez les résultats avec le PDP et justifiez les différences.

8. Pour purpose, identifiez les catégories ayant un **effet positif ou négatif** sur la probabilité de crédit good.
9. Comment l'ALE permet-il de **corriger les biais du PDP** en présence de corrélations entre variables catégorielles et continues ?

3. Analyse des interactions

10. Sélectionnez deux variables (amount et duration) et calculez un **ALE 2D**.
11. Interprétez les zones où les effets **ne sont pas additifs**.
12. Comment l'identification des interactions aide-t-elle à expliquer des décisions **locales vs globales** dans le modèle ?
13. Pour credit_history et personal_status_sex, analysez si certaines combinaisons augmentent fortement le risque bad.
14. Proposez une stratégie pour visualiser simultanément plusieurs ALE afin de **détecter des interactions complexes**.

4. Analyse des biais et robustesse

15. Examinez si l'ALE pour personal_status_sex et employment_duration révèle un **biais potentiel de genre ou d'emploi**.
16. Comparez PDP et ALE pour ces variables et discutez de la **robustesse des conclusions**.
17. Déterminez quelles variables pourraient **masquer un biais** si on se limite aux PDP.
18. Comment pourriez-vous **valider les interprétations ALE** pour un contrôleur ou un décideur bancaire ?

5. Analyse locale

19. Pour trois individus ayant credit_risk = bad, calculez les effets ALE et PDP **locaux** pour toutes les variables.
20. Comparez les effets locaux et globaux. Quels insights sur la **décision individuelle** peuvent être obtenus ?
21. Comment combiner PDP, ALE et Shapley values pour obtenir une **interprétation complète et hiérarchisée** ?

Exercice 2 : SHAP

Nous appliquons les **valeurs de Shapley** comme un outil de **quantification locale de la pertinence des caractéristiques**. Pour cela, nous implémentons la **fonction de gain SHAP marginale** (*marginal SHAP payoff function*).

(a) Chargez le jeu de données FIFA **et prédisez la** probabilité d'obtenir le "Man of the Match" **à l'aide d'une** forêt aléatoire **(*random forest*).**

Indication:
Notez que de nombreuses variables de type *float* dans le jeu de données contiennent des valeurs

manquantes. Utilisez une instance de votre **jeu de test** afin de générer une prédiction illustrative de la **probabilité qu'une équipe ait le "Man of the Match" parmi ses joueurs**.

(b) Implémentez la fonction de valeur SHAP basée sur l'échantillonnage marginal **m_vfunc(). Calculez les** fonctions de valeur marginales basées sur l'échantillonnage **v(j) pour l'instance considérée.**

1. Que signifie une **valeur SHAP positive** pour une caractéristique donnée ?
2. Que signifie une **valeur SHAP négative** ?
3. Comment interpréter une valeur SHAP proche de zéro ?
4. Classez les caractéristiques selon leur **importance locale** pour cette prédiction.

(c) Implémentation de KernelSHAP :

La méthode **KernelSHAP** est une approche efficace pour approximer les **valeurs SHAP** en ajustant **localement un modèle linéaire pondéré** autour de l'instance à expliquer. Cette méthode repose sur l'échantillonnage de **coalitions de caractéristiques**, représentées par des **masques binaires**, et sur l'apprentissage d'un modèle linéaire qui approxime le comportement du modèle complexe à proximité de l'observation étudiée.

Votre objectif est d'implémenter **KernelSHAP** et de calculer les **valeurs SHAP** pour une instance donnée.

Pour cela, implémentez les fonctions suivantes :

- **shap_weights(mask)** :
  Écrivez une fonction qui calcule le **poids KernelSHAP** associé à un masque binaire donné.
  Le masque représente une coalition de caractéristiques (1 = caractéristique présente, 0 = absente).
  Les poids doivent favoriser les coalitions de petite et de grande taille, conformément à la formulation de KernelSHAP.

- **replace_dataset(obs, X, nr_samples)** :
  Écrivez une fonction qui génère un jeu de données contenant **nr_samples observations artificielles** à partir de l'ensemble de données **X**.
  Chaque observation est construite à l'aide d'un **masque binaire** :

  - si la valeur du masque est 1, la valeur de la caractéristique correspondante est prise depuis l'observation **obs** à expliquer ;

  - si la valeur du masque est 0, la valeur de la caractéristique est prise depuis les données de référence **X**.
    Cette fonction doit également assurer la **correspondance entre l'espace binaire des masques et l'espace original des caractéristiques**.

- **shap_data(obs, X, nr_samples, predict)** :
  Écrivez une fonction qui :

  1. génère les masques binaires,

2. construit les observations correspondantes dans l'espace original des caractéristiques,
3. calcule les prédictions du modèle à l'aide de la fonction **predict**,
4. calcule les poids KernelSHAP associés à chaque masque.

La fonction doit retourner un jeu de données contenant **les masques, les prédictions et les poids**.

- **kernel_shap(obs, X, nr_samples, predict)** :
  Implémentez la méthode **KernelSHAP complète** :

  1. utilisez les données générées précédemment,
  2. ajustez un **modèle linéaire pondéré** (par exemple une régression linéaire pondérée),
  3. interprétez les coefficients du modèle linéaire comme les **valeurs SHAP** des caractéristiques pour l'observation **obs**.

(d) Calculez kernel_shap(X_test[1,], X_train, 1000, classifier_RF) en utilisant la **forêt aléatoire** définie en (a).

Questions d'interprétation:

1. Quelle caractéristique contribue **le plus à augmenter** la probabilité d'obtenir le « Man of the Match » ?
2. Quelle caractéristique contribue **le plus à diminuer** cette probabilité ?
3. Expliquez comment la somme des valeurs SHAP permet de passer de la **prédiction moyenne du modèle** à la **prédiction finale**.
4. En quoi les valeurs SHAP fournissent-elles une **explication locale et additive** ?
5. En quoi l'importance locale obtenue avec SHAP peut-elle être différente de l'importance globale des caractéristiques ?
6. Quel est l'impact du **nombre d'échantillons (nr_samples)** sur la stabilité et le temps de calcul de KernelSHAP ?
7. Citez deux **limites** de l'utilisation de SHAP pour interpréter un modèle de type Random Forest.
8. Donnez une **interprétation métier** des deux caractéristiques les plus influentes dans ce contexte footballistique.

Exercice 3: LIME

Dans ce qui suit, vous êtes guidé(e)s pour implémenter **LIME** afin d'interpréter un **Support Vector Machine (SVM)**. Nous utilisons **deux caractéristiques (numériques)** et explorons LIME sur un **problème de classification multiclasse** avec seulement deux caractéristiques numériques.

Les fichiers associés à cet exercice sont **lime.py**. Dans ce fichiers, des **fonctions d'aide à la visualisation** (*helper functions*) ont déjà été implémentées : get_grid(), plot_grid() et plot_points_in_grid().

a) Inspection des fonctions implémentées

Tout d'abord, familiarisez-vous avec les fonctions déjà implémentées dans les fichiers modèles.

- La fonction **get_grid()** prépare les données permettant de visualiser l'espace des caractéristiques.
  Elle crée une grille de taille **N × N**, et chaque point de cette grille est associé à une valeur.
  Cette valeur est obtenue à partir de la méthode **predict** du modèle.

- La fonction **plot_grid()** permet de visualiser la surface de prédiction.

- Le graphique créé est utilisé comme entrée pour la fonction **plot_points_in_grid()**, qui ajoute les points de données fournis au graphique.

Après la visualisation de la surface de prédiction (plot_grid())

1. Que représente la **surface de prédiction** affichée sur le graphique ?

2. Que signifient les **différentes couleurs** observées dans l'espace des caractéristiques ?

3. Comment interprétez-vous les **frontières de décision** visibles sur le graphique ?

4. Que peut-on dire du **comportement global du SVM** à partir de cette visualisation ?

b) Échantillonnage des points

Votre première tâche d'implémentation consiste à **échantillonner des points**, qui seront ensuite utilisés pour entraîner le **modèle explicatif local** (*local surrogate model*).

Complétez la fonction **sample_points()** en échantillonnant aléatoirement à partir d'une **distribution uniforme**.
Prenez en compte les **bornes inférieure et supérieure** issues des points de données d'entrée.

Indication :
vous pouvez utiliser la méthode dataset.get_configspace().get_hyperparameters_dict() implémentée dans le fichier **dataset.py** afin de récupérer les valeurs minimales et maximales. Pour un exemple, consultez la fonction **get_grid()** déjà implémentée.

Questions d'interprétation (b)

5. Pourquoi est-il nécessaire d'échantillonner des points autour de l'observation à expliquer ?

6. Comment la taille de la zone échantillonnée influence-t-elle l'explication produite par LIME ?

7. Que se passe-t-il si les points sont échantillonnés trop loin du point à expliquer ?

c) Pondération des points

Étant donné un point sélectionné **x** et les points échantillonnés **Z** issus de la tâche précédente, nous souhaitons maintenant **pondérer les points**.

Utilisez l'équation suivante, où **d** correspond à la **distance euclidienne**, afin de calculer le poids d'un point individuel z∈Z :

$$\phi_x(z) = \exp\left(-\frac{d(x,z)^2}{\sigma^2}\right)$$

Afin de faciliter la visualisation ultérieure, les poids doivent être **normalisés entre zéro et un**.
Enfin, retournez les poids normalisés dans la fonction **weight_points()**.

Questions d'interprétation (c)

8. Comment la distance au point x influence-t-elle le poids attribué à un point échantillonné ?

9. Pourquoi les points proches de x ont-ils un poids plus élevé ?

10. Comment cette pondération est-elle reflétée visuellement sur le graphique ?

d) Ajustement du modèle explicatif local

Enfin, ajustez un **arbre de décision** en utilisant les données d'entraînement et les poids. Retournez l'arbre entraîné dans la fonction **fit_explainer_model()**. Qu'est-ce qui pourrait poser problème ?

Questions d'interprétation (d)

12. Comment l'arbre de décision approxime-t-il localement la frontière de décision du SVM ?

13. Dans quelles régions du graphique cette approximation est-elle la plus fiable ?

14. Pourquoi l'approximation locale peut-elle devenir incorrecte loin du point x ?

15. Qu'est-ce que cela révèle sur la différence entre un modèle global et une explication locale ?

Analyse critique basée sur les visualisations

16. En quoi les graphiques générés par LIME facilitent-ils la compréhension du comportement du modèle ?

17. Dans quels cas les explications visuelles produites par LIME peuvent-elles être trompeuses ?

18. Quel est l'impact du nombre de points échantillonnés sur la qualité de l'explication ?

19. Comment la position du point x par rapport aux frontières de décision influence-t-elle l'interprétation ?

20. Citez deux limites de la méthode LIME observables à partir des graphiques.

Exercice 2 : Contre-factuels

Les **explications contre-factuelles** constituent un outil précieux pour expliquer les prédictions des modèles d'apprentissage automatique. Elles indiquent à l'utilisateur comment les caractéristiques doivent être modifiées afin de prédire un **résultat souhaité**. L'une des approches les plus simples pour générer des contre-factuels consiste à déterminer, pour une observation donnée **x** (*x_interest*), le **point de données le plus proche** dont la prédiction est égale au résultat souhaité[1](*Wexler et al. (2019) : « The What-If Tool: Interactive Probing of Machine Learning Models »).*

Dans l'exercice suivant, vous devez implémenter cette approche dite **WhatIf** pour un **classifieur binaire**. Les fichiers associés à cet exercice sont **whatif.py**.

a) Implémentez les étapes suivantes dans generate_whatif() :

(i) Extraire un sous-ensemble des données contenant les observations ayant une prédiction différente de celle de *x_interest* (celle-ci correspond à notre prédiction souhaitée).

(ii) Calculer les **distances de Gower par paires** entre *x_interest* et les points de données restants.

Indication:
La fonction **gower** en Python proposent des implémentations de la distance de Gower.

(iii) Retourner le point de données le plus proche comme **contre-factuel** pour *x_interest*.

Testez votre fonction à l'aide du code d'exemple fourni.

b) Quels attributs vus en cours (**validité, parcimonie, …**) cette approche satisfait-elle ? Sur cette base, déduisez les **avantages et les inconvénients** de cette approche.

Questions d'interprétation

1. Comment le point contre-factuel choisi **modifie-t-il les caractéristiques** par rapport à l'observation originale ?

2. Quelle(s) caractéristique(s) a/ont le **plus changé** pour atteindre la prédiction souhaitée ?

3. Que pouvez-vous conclure sur la **sensibilité du modèle** à certaines caractéristiques à partir du contre-factuel ?