Outils agnostiques pour l'interprétation des modèles

Nous travaillons sous PYTHON avec notamment les packages « shap » et « h2o » (qui nécessite une version de numpy < 2) qu'il faudra installer avant de commencer les exercices. **Ou bien**, plus simplement, vous pouvez aussi créer un environnement dédié à l'aide du fichier « envtd_interpretation.yml ».

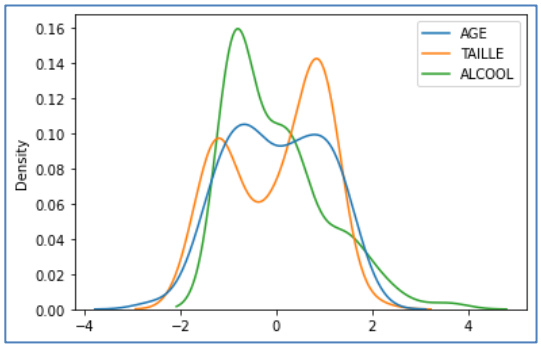
Deux tutoriels vidéo pour nous guider :

- « Machine Learning Interprétation (Python – Scikit-Learn) » (**Vidéo 1**)
- « Machine Learning – Model Explainability (H2O – Python) » (**Vidéo 2**)

A. Données Ronflement

On souhaite caractériser le ronflement (RONFLE ∈ {NON, **OUI**}) des personnes à partir de leur âge (en années), leur taille (en cm), et leur consommation d'alcool (en nombre de verres par jour). Nous disposons de 100 observations.

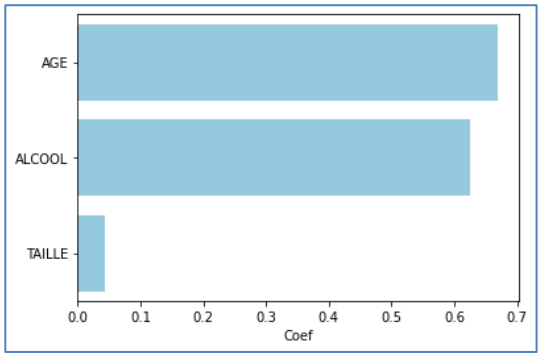
1. Chargez le fichier « **ronflement_interpretation.xlsx** ». Affichez ses caractéristiques (info).
2. Calculez les statistiques descriptives des variables explicatives (describe). Que constatez-vous en ce qui concerne les moyennes et écarts-type notamment ?
3. Chargez et affichez votre version de « Scikit-Learn » (1.6.1 en ce qui me concerne ; une version ≥ 0.24 est suffisante pour travailler).
4. Chargez et affichez votre version de « Numpy » (1.26.4 en ce qui me concerne ; il faut absolument que votre version soit < 2, sinon « h2o » rencontrera des problèmes).
5. Nous souhaitons standardiser ces variables (Vidéo 1, 14:28). Affichez de nouveau les statistiques descriptives. Les nouvelles variables sont certes centrées et réduites, mais que constatez-vous concernant l'étendue des valeurs (min et max) (très différentes d'une variable à l'autre) ?
6. Construire un graphique avec les fonctions de densité approximées pour vous faire une idée de leurs distributions (en utilisant la librairie « seaborn » peut-être ?)



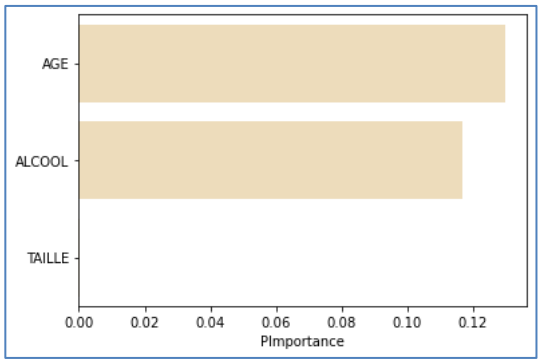
B. Régression logistique

Nous souhaitons analyser les relations entre la cible et les variables explicatives à l'aide d'une régression logistique dans un premier temps. Nous utilisons la totalité des données puisque l'interprétation prime dans notre étude, et non pas les performances prédictives.

7. Instanciez et entraînez une régression logistique de « Scikit-learn » sur les données centrées et réduites (Vidéo 1, 17:05). Fixez « random_state = 0 » pour que nous ayons des résultats identiques (pour peu que nous ayons les mêmes versions de la librairie). Affichez les coefficients. Comment les interpréter ? (en évolution du logit en fonction d'une variation d'un écart-type de chaque variable)
8. Passez les coefficients en valeur absolue. Triez-les de manière décroissante. Construisez alors un graphique retraçant l'importance des variables dans la régression.

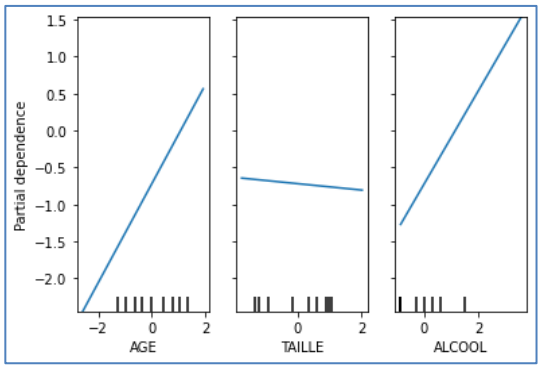


9. On souhaite mesurer l'importance des variables dans la régression à partir de l'outil « Permutation Feature Importance » de Scikit-learn (Vidéo 1, 19:30). Nous utilisons le critère (scoring = 'roc_auc'). Les résultats obtenus sont-ils cohérents avec la lecture que nous faisons des coefficients ? (oui).



10. Le sens de la liaison entre chaque explicative et la cible est exprimée par le signe du coefficient de la régression. Nous souhaitons l'analyser à l'aide de l'outil « Partial Dependence Plot ». Affichez les graphiques de dépendances partielles en agrégeant les valeurs (kind = 'average'), et en utilisant la fonction de décision (response_method = 'decision_function') comme réponse (Vidéo 1, 24:48). Les graphiques obtenus (sens de la liaison, importance de la liaison) sont-ils conformes à ce que l'on attendait à la lecture des coefficients de la régression ? (oui,

heureusement) (Remarque : utilisez PartialDependenceDisplay si votre version de Scikit-learn est ≥ 1.0 ; sinon, utilisez la commande plot_partial_dependence comme indiqué dans la vidéo).

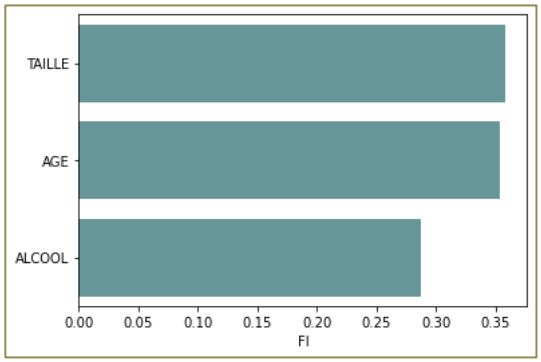


11. Affichez le même graphique mais avec les courbes individuelles (kind = 'both'). Utilisez la moitié des observations disponibles (subsample = 0.5). Que constatez-vous ? (les formes des relations sont identiques pour l'ensemble des individus, toutes linéaires, de par la nature même du modèle, mais à des « hauteurs » différentes en fonction des valeurs prises par les autres variables).
12. Affichez le graphique agrégé en utilisant les probabilités d'affectation cette fois-ci (response_method = 'predict_proba'). Observe-t-on une différence avec le graphique agrégé basé sur la fonction de décision ? (oui, mais à peine, on devine - plus qu'on ne l'observe réellement - la forme sigmoïde de la transformation logistique).

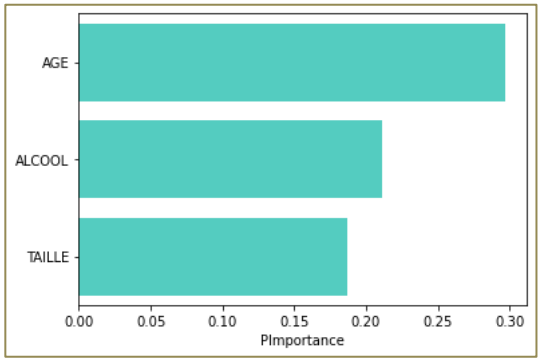
C. Gradient boosting

On souhaite réitérer l'analyse avec un classifieur non-linéaire qui ne fournit pas de coefficients interprétables, un « Gradient Boosting Machine » en l'occurrence (Vidéo 1, 32:50).

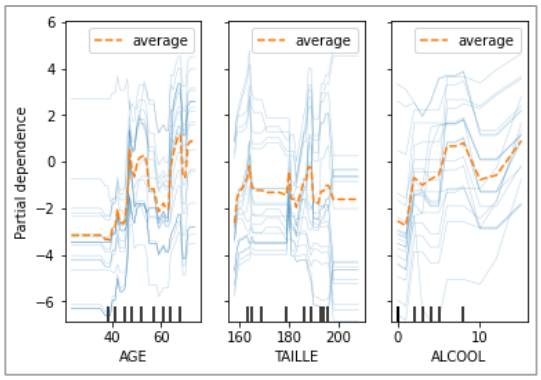
13. Instanciez et entraînez un gradient boosting (GradientBoostingClassifier) sur les données originelles (centrer et réduire les variables n'a aucun impact dans l'induction des arbres sous-jacents). Fixez (random_state = 0) pour que nous ayons des résultats comparables.
14. Récupérez, triez, et représentez graphiquement l'importance des variables (.feature_importances_) proposée nativement par la classe de calcul (basée sur les propriétés des arbres sous-jacents à la méthode, rappelons-le). Que constatez-vous par rapport à la régression logistique ? (« taille » prend de l'importance dans ce modèle).



15. Calculez et affichez l'importance des variables calculée ex-post avec « Permutation Feature Importance ». Les résultats sont-ils cohérents avec ceux de l'outil natif ? (le rôle de « taille » dans le modèle finalement…)



16. Calculez et affichez le graphique des dépendances partielles avec le critère « decision function ». Que peut-on en conclure ?



D. Analyse des prédictions du gradient boosting

Nous souhaitons caractériser les prédictions à l'aide des valeurs « Shapley » dans cette section, individuellement dans un premier temps, globalement dans un second temps pour situer d'une autre manière les relations entre les explicatives et la modalité cible (Vidéo 1, 49:05).

17. Calculez les probabilités d'appartenance aux classes des individus de notre base de données (predict_proba). Affichez les dimensions de la matrice générée (100, 2).

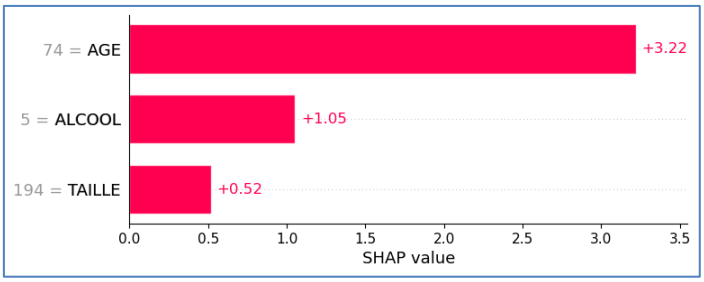
18. Dans cette matrice, repérez la colonne correspondant à la modalité (RONFLE = 'OUI') en affichant l'ordre des classes dans le modèle (.classes_). Quelle est la colonne qui nous intéresse pour la suite ? (la colonne n°1, associée à RONFLE = OUI).

19. Quel est l'individu le plus âgé dans notre jeu de données ? Affichez ses caractéristiques (n°7 ; 74 ans, 194 cm, alcool = 5). Quelle est sa probabilité de ronfler calculée par le modèle ? (0.973).

20. On souhaite comprendre les raisons qui ont conduit le modèle à lui attribuer une probabilité de ronflement aussi élevée. Importez la librairie « **shap** ». Définissez un « explainer » pour le modèle prédictif. Calculez les valeurs « Shapley » sur notre jeu de données (Vidéo 1, 52:00).

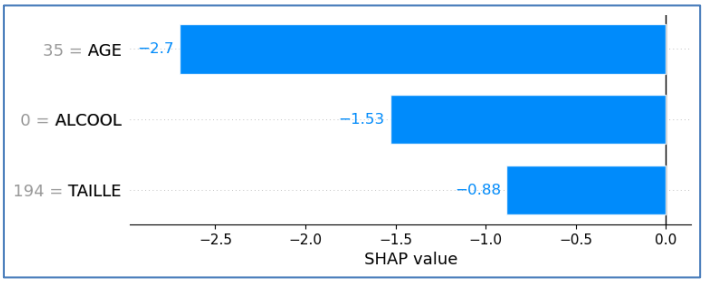
21. Affichez les valeurs « Shapley » pour notre individu le plus âgé. Représentez-les graphiquement.

Pour quelles raisons l'individu n°7 présente-t-il une probabilité élevée de ronflement au sens du modèle ? (essentiellement parce qu'il est âgé, ah ça ne pardonne pas hélas ...)

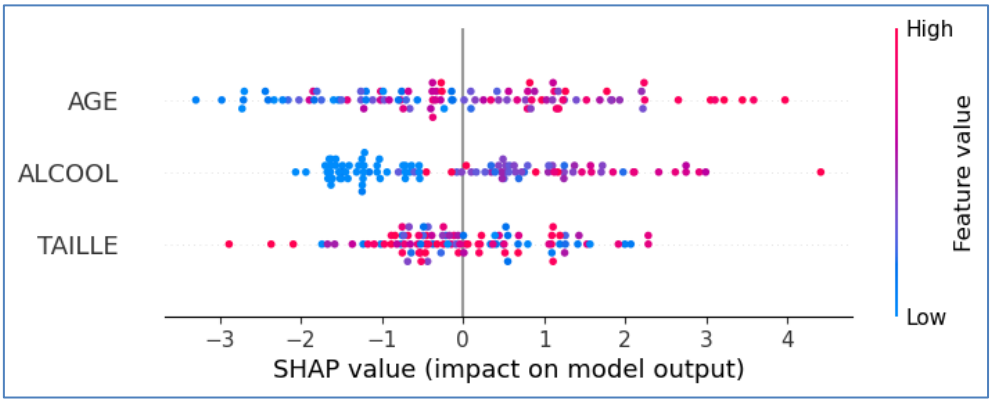


22. Intéressons-nous maintenant à l'individu qui présente le risque le plus faible de ronflement au sens du modèle (individu n°57 en ce qui me concerne).

23. Affichez ses caractéristiques. De la même manière que pour l'individu n°7, indiquez les raisons pour lesquelles le gradient boosting lui a attribué un aussi faible risque de ronflement (c'est parce qu'il est jeune, il ne boit pas, il est grand, il a tout pour lui... un peu fayot, vous voyez le genre...).



24. On aimerait disposer d'une vue globale des relations entre les explicatives et la modalité cible (RONFLE = OUI). Affichez le graphique « Beeswarm » (Vidéo 1, 57:05). Que peut-on conclure. Est-ce que ces résultats sont en cohérence avec les graphiques de dépendance partielle par exemple ? (oui, je dirais... à discuter...)



E. Modélisation et interprétation avec H2O

Nous souhaitons reproduire l'analyse avec la librairie « H2O » et une autre méthode prédictive (Random Forest).

25. Importez la librairie « h2o », affichez le numéro de version (**3.44.0.3** pour moi). Démarrez ensuite

le serveur de calcul (init) (Vidéo 2, 06:45).

26. Typez le data frame initial des données en un ensemble de données reconnu par H2O (H20Frame) (Vidéo 2, 08:45). Affichez le type de l'objet obtenu (type).
27. Instanciez un « Random Forest » (H2ORandomForestEstimator ; Vidéo 2, 09:07, sauf que nous partons sur un **Random Forest** donc). Mettez (seed = 0) pour que nous ayons des résultats directement comparables. Demandez également une validation croisée en 5 blocs pour obtenir directement une estimation fiable de l'erreur (nfolds = 5). Décryptez attentivement les sorties de l'outil. Elles sont abondantes. Regardez ce qui est annoncé concernant l'**importance des variables** (en toute fin des sorties).
28. Effectué un bilan en matière d'explicabilité du modèle (explain) (Vidéo 2, 13:25). Ici aussi, décryptez attentivement les abondantes sorties de l'outil (importance, shap, partial dépendance plot).
29. Effectuez la prédiction sur l'ensemble de données (predict) (Vidéo 2, 28:20). Récupérer le résultat sous forme de data frame (as_data_frame). Affichez les premières valeurs (head).
30. Affichez la ligne correspondant à l'individu n°7 (notre ancêtre). Quelle est sa probabilité de ronfler ? (0.91). Expliquez la décision du modèle pour cette individu (explain_row) (Vidéo 2, 29:24).
31. Réitérez l'analyse pour l'individu n°57 (le bellâtre).