ECUE2.1 : Science des données **Consignes pour le rapport de mini-projet numérique**

Date de rendu : 1er juillet 2020

Il s'agit dans ce projet de construire un modèle prédictif du risque de défaut de paiement sur un crédit, en fonction de variables décrivant la personne bénéficiant de ce prêt ainsi que l'historique des remboursements sur les 6 derniers mois.

Les données sont traitées dans le notebook `Sujet_mini_projet.ipynb`.

Quels documents rendre ? Si vous souhaitez que votre projet soit noté à la place ou en plus de l'examen écrit, vous devrez rendre :

- un rapport de projet, au format `.pdf` ou `.ipynb`;
- un fichier de prédictions au format `.csv`.

Si vous choisissez le format `.pdf` n'incluez que le code demandé (section 5 de ce document).

Quand rendre ces documents ? Le mercredi 1er juillet 2020 à midi.

Comment rendre ces documents ? À préciser.

Travail de groupe Le travail à plusieurs est autorisé. Néanmoins vous devrez rendre **un rapport et un fichier de prédictions par personne**. Le fichier de prédictions ainsi que le code et les figures inclus dans votre rapport pourront bien sûr être identiques à ceux de vos collaborateurs et collaboratrices ; les interprétations et réponses aux questions doivent être individuelles.

Contenu du rapport et grille de notation

Indiquez en haut du rapport **les noms des personnes avec lesquelles vous avez travaillé sur le projet, le cas échéant.**

1. Contexte [4 pts]

Les données sont proches de données réelles. Vous semblerait-il intéressant, pour une banque, de déployer un tel modèle, entrainé sur ses données ? Quelle performance vous semblerait acceptable pour votre cas d'usage ? Quels risques pourrait-on encourir à déployer un tel modèle ? Voyez-vous des sources de biais possibles, qui pourraient conduire à de mauvaises performances et/ou à des discriminations ? Quel(s) autre(s) usage(s) pourrait-on faire de ces données ? Quels sont des usages actuels de modèles prédictifs dans l'industrie bancaire ?

Longueur de la réponse attendue : environ une demi-page.

2. Prétraitement [2 pts]

Décrivez brièvement comment vous avez prétraité vos données. Incluez une représentation visuelle de vos données prétraitées (par exemple histogrammes ou diagrammes en barres pour chacune des variables).

3. Plus proche voisin [2 pts]

L'algorithme du plus proche voisin (kNN avec k=1) prédit l'étiquette d'une observation comme celle de son point le plus proche dans le jeu d'entraînement.

— Quelle est la classe des hypothèses ?

— S'agit-il d'un modèle paramétrique ou non ?

— Peut-on écrire cet algorithme sous la forme de la minimisation d'un risque empirique ? Si oui, précisez la classe des hypothèses, la fonction de coût et la technique d'optimisation utilisée.
Remarque : le point le plus proche d'un point du jeu d'entraînement est lui-même.

4. Sélection de modèle [8 pts]

1. Utilisez une validation croisée sur votre jeu d'entraînement `(X_train, y_train)` pour sélectionner les meilleurs hyperparamètres :

— d'une approche des k plus proches voisins (hyperparamètre = valeur de k) ;

— d'une régression logistique régularisée. Vous pouvez justifier le choix du type de régularisation soit par des arguments a priori, soit en considérant le type de régularisation comme un hyperparamètre. Dans les deux cas, la valeur du coefficient de régularisation est à choisir par validation croisée ;

— [facultatif] d'une ou plusieurs autres approches de classification de votre choix (forêts aléatoires, SVM, etc.).

2. Ré-entraînez ces méthodes (kNN avec k optimal ; régression logistique avec votre choix de régularisation et votre choix de coefficient de régularisation ; etc.) sur votre jeu d'entraînement `(X_train, y_train)` et appliquez les modèles ainsi appris à votre jeu de test `(X_test, y_test)`.

Dans le rapport, incluez

— Votre code ;

— Une ou plusieurs figures permettant de comparer les performances (selon la ou les mesures de performance de votre choix) des différentes approches, d'une part, en validation croisée, et d'autre part, sur votre jeu de test ;

— Une analyse statistique : les prédictions (valeurs de la fonction de décision) que vous obtenez sur le jeu de test sont-elles significativement différentes entre les différents modèles ? Une des façons de répondre à cette question consiste à utiliser un test de comparaison de deux distributions continues non-indépendantes tels que le test des rangs signés de Wilcoxon, ou *Wilcoxon signed-rank test*, implémenté dans `scipy.stats.wilcoxon`. Si vous comparez plus de deux modèles, n'oubliez pas d'utiliser une correction de tests d'hypothèses multiples.

— Quelques phrases pour analyser ces résultats et en conclure quel modèle final choisir.

Détail des points

Implémentation de la procédure de validation croisée	1 pt
Choix des grilles d'hyperparamètres	1 pt
Figures	2 pts
Test statistique	2 pts
Choix du modèle	2 pts

5. Prédictions finales [2 pts]

Entraînez votre modèle final sur l'ensemble des données publiques `(X_public, y_public)` et faites vos prédictions sur les données non-étiquetées disponibles dans `data/credit_private.csv`.

Votre fichier de prédictions doit comporter autant de lignes que `data/credit_private.csv` et deux colonnes : une pour des prédictions binaires (0 ou 1) et une pour des scores retournés par une fonction de décision (plus ce score est élevé, plus le risque de défaut est élevé). Chaque ligne de ce fichier correspondra à la même ligne de `data/credit_private.csv`.

La première ligne de ce fichier sera un en-tête : `Prediction_binaire Prediction_score`

Si vos prédictions binaires sont dans l'array numpy de dimension 1 `y_pred_binary` et vos scores de décision dans l'array numpy de dimension 1 `y_pred_scores`, vous pouvez créer votre fichier de prédiction en utilisant :

```
# Reshape 1-dimensional arrays to 2-dimensional and stack them in the same array
y_array_final = np.hstack((y_pred_binary.reshape((y_pred_binary.shape[0], 1)),
                           y_pred_scores.reshape((y_pred_scores.shape[0], 1)))

# Save array to file
np.savetxt("mon_fichier.csv",
           y_array_final,
           fmt=('%d', '%.3f'),
           header='Prediction_binaire\tPrediction_score',
           delimiter='\t', comments="")
```

6. Apprentissage profond [2 pts]

Pensez-vous qu'un réseau de neurones profond puisse être adapté à ce problème ? Expliquez pourquoi en quelques phrases.