Compte rendu

March 3, 2022

1 Contexte

Ce projet est réalisé dans le cadre de la formation Microsoft IA et permet de mettre en évidence, lors d'une classification supervisée, l'importance du choix des hyperparamètres d'un modèle dans la performance de la prédiction et plus particulièrement celui d'un classificateur KNN (K plus proches voisins), implémenté en python.

Dans cet atelier, nous appliquons la classification pour prédire comment une personne gère son stress.

Challenges:

- Collection d'une base de données.
- Analyse, prétraitement et visualisation des données.
- Préparation des données pour l'apprentissage.
- Conception d'un modèle KNN.
- Choix de la meilleure configuration pour le modèle KNN.
- Vérification de l'efficacité de ce modèle à des nouvelles données.
- Intégration de ce modèle dans l'application de test de personnalité.

2 Étapes de réalisation

Pour réaliser ce projet nous avons suivi les étapes suivantes :

- Prétraitement des données
  - Récupération des données
  - Conversion des caractères en miniscule
  - Conversion des coquilles
  - Remplacement des NaN par le mode
  - Recalcul du Score
  - Encodage des valeurs catégorielles
  - Séparation des données
- KNN From Scratch
  - Fonction distance
  - Fonction modèle
  - Expérimentation
    - Distance euclidienne
    - Distance manhattan

        * Distance minkowski
    - Performances
    - Matrice de confusion

- KNN Sklearn

    - Importation de la classe KNeighborsClassifier
    - Préparation des données
    - K-fold validation
    - Gridsearch
    - Performances
    - Matrice de confusion

- Comparaison des résultats
- Mise en place de la solution

3 Détails de l'application

Il est composé de :

- Run_Questionnaire : le dossier contenant les fichiers nécessaire à l'exécution du questionnaire

    - DataSet : le dossier contenant les fichiers .csv des réponses au questionnaire
    - ConcatenationDataset.py : le script permettant de générer un fichier .csv de toutes les réponses
    - DataSetTotal.csv : le fichier .csv récapitulant toutes les réponses
    - Run.ipynb : le fichier jupyter permettant de répondre au questionnaire
    - Test.py : le script permettant de généré le questionnaire

- .gitignore : le fichier git permettant d'ignorer des fichiers de l'espace de travail
- Compte rendu.ipynb : le fichier compte rendu du brief
- KNN - Test de personnalité.ipynb : le fichier permettant de générer un modèle prédictif à partir des fichiers
- ModeleKNN.pkl : le fichier du modèle sérialisé, généré par joblib
- README.md : ce fichier

4 Résultats

4.1 Prétraitement des données

Lors de la saisie des réponses, de nombreuses coquilles ont été enregistrées dans les fichiers réponses au questionnaire et se sont donc retrouvées dans le fichier regroupant toutes les réponses.

Nous avons alors décidé de prétraiter ces réponses en modifiant les réponses au questionnaire afin d'obtenir un dataset plus fourni et qui permettrait donc de potentiellement mieux réaliser des prédictions.

Pour ce faire, nous avons décidé de recalculer le score après prétraitement des données et d'effectuer l'entraînement avec ces nouvelles valeurs.

Cette étape n'étant pas spécifiée dans le descriptif de ce projet, il est important de noter que les prédictions peuvent diverger des interprétations car ces dernières ne prendront pas en compte ces modifications.

4.2 KNN From Scratch

Afin de détailler les étapes réalisées par scikit-learn lors de l'utilisation de la classe `KNeighborsClassifier` du module `sklearn.neighbors`, nous nous sommes attachés dans cette partie à la réalisation de la méthode des K plus proches voisins manuellement.

La première étape est d'implémenter une fonction permettant de calculer les distances suivantes, avec n le nombre de caractéristiques, (X, Y) un couple de vecteur:

- Euclidienne : $\sqrt{\sum_{i=1}^{n}(X_i - Y_i)^2}$
- Manhattan : $|\sum_{i=1}^{n}(X_i - Y_i)|$
- Minkowski : $(\sum_{i=1}^{n}|X_i - Y_i|^p)^{\frac{1}{p}}$ avec p le paramètre de la métrique Minkowski

Ensuite une fonction effectue la méthode des K plus proches voisins en suivant les étapes suivantes :

- Parcourir les données de test
- Calculer la distance entre chaque entrée de test avec chacune des entrées d'entraînement
- Récupérer les k premires index des distances triées dans un ordre croissant
- Récupérer les valeurs cibles associées
- Calculer la fréquence des résultats
- Récupérer la valeur la plus fréquente
- Prédire la classe des données de test

À partir de ces deux fonctions, on peut alors tester les différents hyperparamètres possibles et calculer les performances de chaque expérimentation, puis en déduire les meilleurs paramètres pour notre modèle et enfin étudier les classes le plus source d'erreurs.

4.3 KNN Sklearn

Dans cette partie nous nous sommes attachés à l'implémentation de la méthode des K plus proches voisins à l'aide des modules de la bibliothèque `Scikit Learn`.

Pour ce faire, nous avons suivi les étapes suivantes :

- Importer la classe `KNeighborsClassifier`
- Vérifier la validité des données à l'aide de la classe `KFold`
- Sélectionner les hyperparamètres à l'aide de la classe `GridSearch`
- Étudier les performances en terme de précision avec les meilleurs hyperparamètres trouvés
- Analyser les erreurs en fonction des différentes classes

Cette partie a été l'occasion d'implémenter manuellement la méthode des K-fold, tout en découvrant son utilisation au sein de la classe `GridSearch`, afin de trouver les meilleurs hyperparamètres et finalement analyser les performances du modèle retenu.

4.4 Comparaison des résultats

Après comparaison des résultats entre KNN from Scratch et KNN Sklearn, on obtient des résultats comparables, mais les hyperparamètres ne sont pas forcément les mêmes.

Cependant après exécution de plusieurs modèles, la précision obtenue avec le KNN from Scratch est un peu meilleure et permet une meilleure prédiction des valeurs les moins fréquentes.

Cela peut provenir du choix des hyperparamètres : pour le KNN from Scratch, ce choix est effectué manuellement après avoir réalisé différentes simulations, mais pour le KNN Sklearn, ce choix est effectué automatiquement suite à l'exécution d'un GridSearch.

Ainsi, le modèle KNN from Scratch semble être mieux adapté à nos valeurs, mais peut être sujet à de la surinterprétation, alors que le modèle KNN Sklearn aura hypothétiquement une meilleure prédiction sur de nouvelles valeurs.

4.5 Mise en place de la solution

Afin de terminer cet atelier, nous avons implémenté le modèle retenu au sein de l'application de Test de Personnalité en affichant la prédiction issue du dernier modèle jugé le plus performant, qu'on a au préalable sérialisé à l'aide de la bibliothèque `joblib` dans le fichier `KNNModele.pkl`.

Le résultat est affiché après avoir répondu à l'ensemble des questions posés, avant l'affichage de l'interprétation issue du calcul du score.

5 Conclusion

À travers cette étude, nous avons pu découvrir la méthode des K plus proches voisins, tout en revenant sur l'importance du prétraitement des jeux de données, mais aussi de souligner l'influence du choix des hyperparamètres sur les différentes métriques qui peuvent être utilisées afin d'évaluer les performances des modèles étudiés.