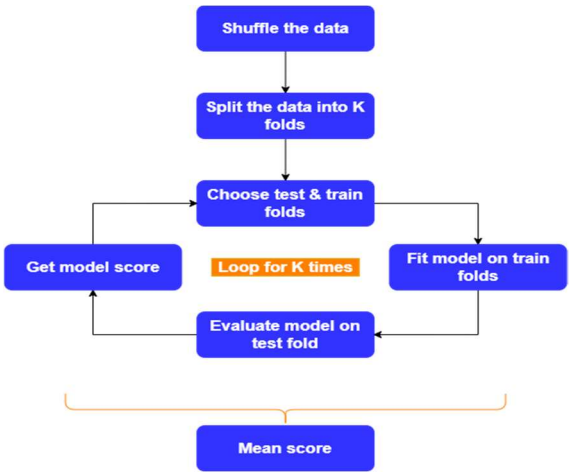
Université Mohammed Premier
École Nationale des Sciences Appliquées
Filière Génie Informatique

TP : Machine Learning

Partie 6 : La Validation croisée : Kfold, StratifiedKFold

La validation croisée

La validation croisée est une technique de validation de modèle qui permet d'estimer la performance d'un algorithme de machine learning sur un ensemble de données donné. Elle est utilisée pour évaluer la capacité de généralisation d'un modèle, c'est-à-dire sa capacité à bien se comporter sur de nouvelles données qu'il n'a jamais rencontrées auparavant.



Nous allons examiner les différents types de validation croisée disponibles, notamment la K-fold cross-validation et la Stratified K-fold cross-validation, ainsi que la manière d'interpréter les résultats de la validation croisée.

Les Types de validation croisée : K-fold cross-validation

La K-fold cross-validation est l'un des types de validation croisée les plus populaires. Elle consiste à diviser les données en k sous-ensembles de taille égale, appelés « folds ». L'algorithme est ensuite entraîné sur k-1 des folds et testé sur le fold restant. Cette procédure est répétée k fois en utilisant chaque fold comme jeu de test une fois. Les performances sont ensuite moyennées sur les k itérations. Cette technique est souvent utilisée lorsque le nombre d'échantillons disponibles est limité.

- **Utilisation de la fonction cross_val_score()**

La fonction cross_val_score() de scikit-learn est utilisée pour effectuer une K-fold cross-validation. Cette fonction prend en entrée le modèle à tester, les données et le nombre de folds k. Elle renvoie un tableau contenant les scores de chaque itération.

- **Choix du nombre de folds**

Le choix du nombre de folds dépend du nombre d'échantillons disponibles et de la complexité du modèle. Si le modèle est simple et qu'il y a suffisamment d'échantillons, k peut être fixé à 5 ou 10. Si le modèle est complexe et que le nombre d'échantillons est limité, k peut être fixé à une valeur plus élevée.

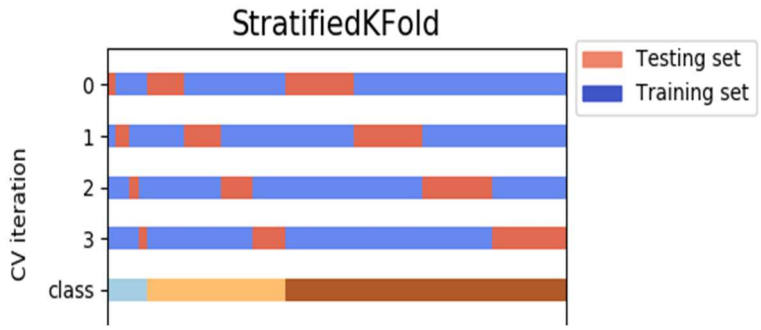
- **K-fold cross-validation avec scikit-learn**

Voici un exemple d'utilisation de la fonction cross_val_score() pour effectuer une K-fold cross-validation :

Entrée [1]:
```
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.datasets import load_iris

iris = load_iris()
X = iris.data
y = iris.target

model = LogisticRegression()

scores = cross_val_score(model, X, y, cv=5)
print(scores.mean())
```

Les Types de validation croisée : Stratified K-fold cross-validation

La Stratified K-fold cross-validation est similaire à la K-fold cross-validation, sauf que la répartition des classes dans les folds est conservée. Cela signifie que chaque fold contient une proportion égale de chaque classe. Cette technique est particulièrement utile lorsque les données sont déséquilibrées.



- **Utilisation de la fonction StratifiedKFold()**

La fonction StratifiedKFold() de scikit-learn est utilisée pour effectuer une Stratified K-fold cross-validation. Cette fonction prend en entrée le nombre de folds k et les données et renvoie un générateur qui produit des indices pour chaque fold.

- **Avantages par rapport à la K-fold cross-validation**

La Stratified K-fold cross-validation présente plusieurs avantages par rapport à la K-fold cross-validation :

✓ Elle est plus robuste lorsque les classes sont déséquilibrées, car elle garantit que chaque fold contient une proportion égale de chaque classe.

✓ Elle est plus précise que la K-fold cross-validation pour les données déséquilibrées, car elle garantit que chaque fold contient une proportion égale de chaque classe.

✓ Elle fournit des résultats plus stables que la K-fold cross-validation, car elle répartit les classes de manière équilibrée dans chaque fold.

- **Stratified K-fold cross-validation avec scikit-learn**

Voici un exemple d'utilisation de la fonction StratifiedKFold() pour effectuer une Stratified K-fold cross-validation :

Entrée [2]:
```
from sklearn.model_selection import StratifiedKFold
from sklearn.linear_model import LogisticRegression
from sklearn.datasets import load_iris

iris = load_iris()
X = iris.data
y = iris.target

model = LogisticRegression()

skf = StratifiedKFold(n_splits=5)

scores = []
for train_index, test_index in skf.split(X, y):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]

    model.fit(X_train, y_train)
    scores.append(model.score(X_test, y_test))
def mean(lst):
    return sum(lst)/len(lst)
print(mean(scores))
```

Interprétation des résultats de la validation croisée

Les résultats de la validation croisée peuvent être interprétés de différentes manières en fonction de l'objectif du modèle. Les scores renvoyés par la K-fold cross-validation ou la Stratified K-fold cross-validation sont des estimations de l'erreur de généralisation du modèle. Plus le score est élevé, meilleure est la performance du modèle. Les scores peuvent être utilisés pour comparer différents modèles ou pour ajuster les hyperparamètres d'un modèle.

Exercice N° 1

Supposons que nous avons le jeu de données ***diabetes*** de ***sklearn.datasets*** qui est un jeu de données sur le diabète de patients. Nous voulons entraîner un modèle de classification pour prédire si un patient a un diabète ou non en fonction de certaines caractéristiques.

1. Charger le jeu de données ***diabetes*** à partir de ***sklearn.datasets***
2. Définir le modèle de classification ***LogisticRegression*** pour entraîner et tester sur le jeu de données choisi.
3. Implémenter les deux méthodes de validation croisée pour évaluer votre modèle.
4. Calculez les scores des k-Folds ainsi que la moyenne des scores afin de comparer les résultats des deux méthodes.

Exercice N° 2

1. Appliquez la méthode StratifiedKFold de la validation croisée pour le modèle de K-plus proche voisin du TP précèdent (TP5).
2. Appliquez à nouveau la méthode de validation croisée StratifiedKFold à un autre modèle de classification (par exemple, ***LogisticRegression***) pour le jeu de données ('knn_data_2.csv') du TP précédent (TP5).
3. On compare les scores calculés par StratifiedKFold pour les deux modèles afin de déterminer lequel est le meilleur pour ce jeu de données.