Méthodologie informatique

Classification thématique avec Python

Kouamé Kouassi

Sorbonne Université

05/01/2023

- **Corpus Reuters 21578**
  - 22 fichiers en format SGML - 1 fichier DTD - 6 fichiers décrivant les catégories utilisées pour l'indexation des données
  - Chaque texte est associé à une catégorie - 21578 instances

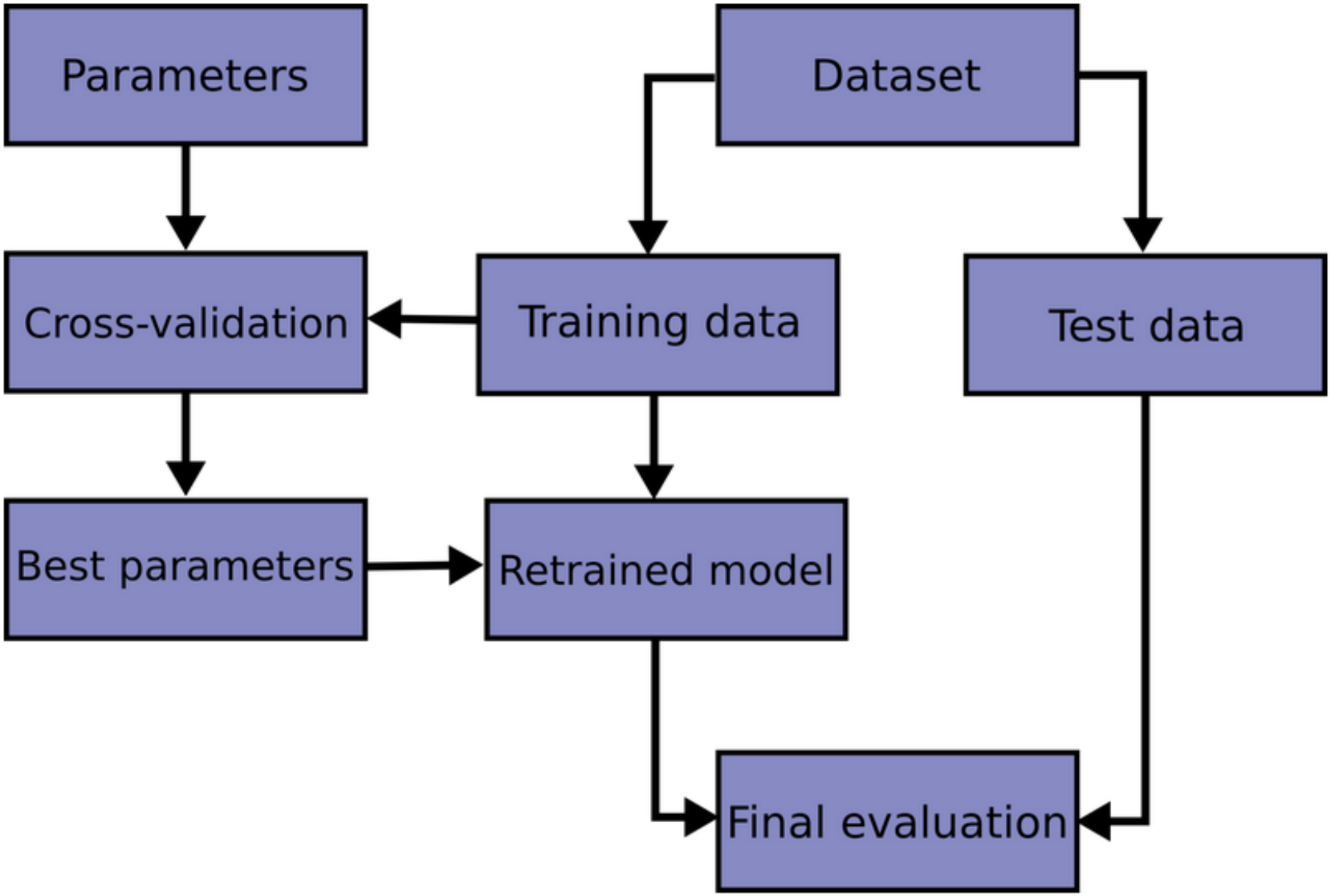
- **Catégories**
  - 5 attributs

1. EXCHANGES : 39 catégories
2. ORGS : 56 catégories
3. PEOPLE : 267 catégories
4. PLACES : 175 catégories
5. TOPICS : 135 catégories

- X = l'ensemble des textes
- y = les catégories
- 10 757 exemples d'entrainement
- 2690 dans l'ensemble de test

	REUTERS TOPICS	TOPICS	TITLE	BODY
0	YES	earn	ISLAND TELEPHONE SHARE SPLIT APPROVED	ISLAND TELEPHONE SHARE SPLIT APPROVED
1	YES	trade	U.K. GROWING IMPATIENT WITH JAPAN - THATCHER	U.K. GROWING IMPATIENT WITH JAPAN - THATCHER
2	YES	earn	QUESTECH INC <QTEC> YEAR NET	QUESTECH INC <QTEC> YEAR NET
3	YES	none	ASLK-CGER FINANCE ISSUES 10 BILLION YEN BOND	ASLK-CGER FINANCE ISSUES 10 BILLION YEN BOND
4	YES	crude	CANADA OIL EXPORTS RISE 20 PCT IN 1986	CANADA OIL EXPORTS RISE 20 PCT IN 1986
...	...	...	...	...

- Stratifier avec le paramètre *Stratify*
- Test size à 0.2
- Valider le modèle sur une validation set
- Technique de cross-validation


Parameters
Dataset
Cross-validation
Training data
Test data
Best parameters
Retrained model
Final evaluation

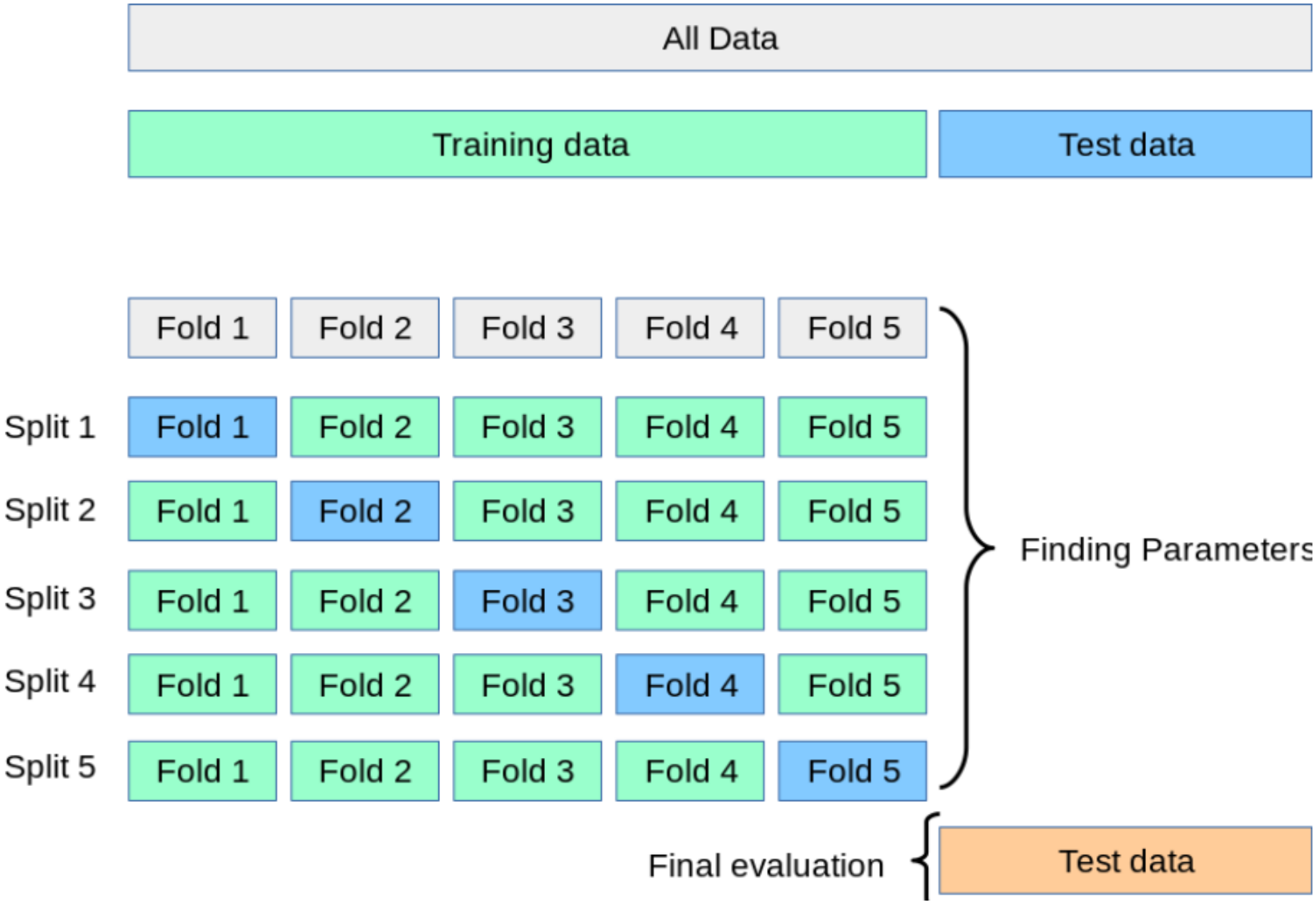
- Consiste à faire des splits de l'ensemble de développement
- 3 splits pour raccourcir le temps de calcul
- Donc on entraine 3 fois le modèle sur ces splits

- Module GridSearchCV et module Pipeline

- GridSearchCV automatise la recherche d'un optimum parmi les hyperparamètres en utilisant la validation croisée

- Appliquer séquentiellement une liste de transformations et un estimateur final

- Assembler plusieurs étapes qui peuvent être validées ensemble en définissant des paramètres


All Data
Training data
Test data
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 1
Split 2
Split 3
Split 4
Split 5
Finding Parameters
Final evaluation
Test data

- Transformations effectuées à l'aide du pipeline
- CountVectorizer et Tfidftransformer
- Suppression des mots vides
- Découper en mots avec *analyzer*
- Quelle groupe de n-grammes améliore le score avec (2,2),(1,2),(2,3),(1,3)

- Decision tree - Criterion = entropy - Random state = 6
- Random Forest - n estimators (nombre d'arbres) = 200 - criterion = entropy
- Naive Bayes - n estimators = 200 - criterion = entropy
- Perceptron multicouches - hidden layer sizes = 500 - solver = adam - early stopping = True - random state = 6 - max iter = 200

- **Arbre de décision**

  Beaucoup de bons résultats et peu d'erreurs
  - F1-score à 0.73
- Dans tous les classifieurs, les catégories comme " corn " , " carcass " sont des catégories pour lesquelles on a pas beaucoup d' occurrences dans l' ensemble de données (support) et pour lesquelles on obtient 0% de précision La catégorie la mieux fournie en support est " earn " avec un support de 795 pour laquelle on obtient :
  - Precision : 0.94
  - Rappel : 0.87
  - F1-score : 0.91
- Les catégories " hog " et " tin " qui sont parmi les catégories les moins fournies avec un support respectivement de 3 et de 6 obtiennent 100% de bonne classification.

- **Random Forest**

  - F1-score à 0.80 L'algorithme de forêts aléatoires est celui qui marche le mieux pour le dataset. L'algorithme d'arbre de décision a un meilleur score de précision sur la catégorie "earn" avec 0.94, tandis que random Forest a 0.85. Par contre il s'en sort mieux sur le rappel (ratio de bonnes réponses trouvées) avec 0.95 contre 0.87 pour Decision Tree.

- **Naive Bayes** - F1-score = 0.68

- Cross-validation utilisée pour ne pas avoir à tester le modèle sur les données de test
- Vectorisation et transformations utilisées
- Possibilité d'améliorer les classifieurs utilisés avec la technique de Boosting et de rétropropagation (backpropagation) pour perceptron multicouche
- Possibilité d'utiliser l'élagage pour les arbres de décision pour résoudre le problème d'overfitting avec un coût à complexité minimale dans Scikit-learn (Cost-Complexity Pruning)
- On pourrait également tester l'algorithme C4-5 qui paraît-il apporte des améliorations en produisant un arbre de décision dans Weka