TP : Classification de texte avec sklearn

31 mars 2025

Introduction

Dans ce TP, vous allez explorer des techniques de classification de texte en utilisant la bibliothèque `scikit-learn`. L'objectif est de construire des modèles de machine learning pour classer des textes en deux parties distinctes. Dans la première partie, vous utiliserez des approches classiques comme Bag of Words et TF-IDF. Dans la deuxième partie, vous explorerez une méthode plus moderne avec les embeddings générés par un modèle Sentence Transformers.

Vous devrez rédiger un rapport court (2-3 pages max) expliquant vos choix, vos résultats et une comparaison des approches. Le code doit être clair et commenté.

Prérequis

— Bibliothèques : `scikit-learn`, `numpy`, `pandas`, `sentence-transformers`, `matplotlib`.
— Jeu de données : Utilisez le dataset `SMS Spam Collection` disponible sur Google drive. Ce dataset contient des messages SMS étiquetés comme "spam" ou "ham" (non-spam).

1 Partie 1 : Classification avec Bag of Words et TF-IDF

Dans cette partie, vous allez transformer des textes en représentations numériques avec deux méthodes classiques : Bag of Words (BoW) et Term Frequency-Inverse Document Frequency (TF-IDF). Ensuite, vous entraînerez un modèle de classification.

Consignes

1. Chargez le dataset `SMS Spam Collection` à partir du fichier CSV disponible sur Kaggle.
2. Prétraitement :
   — Supprimez les caractères spéciaux et convertissez le texte en minuscules.
3. Bag of Words :
   — Utilisez CountVectorizer pour transformer les textes en une matrice de fréquences.

— Limitez le vocabulaire à 5000 mots maximum (`max_features=5000`).

4. TF-IDF :
   — Utilisez TfidfVectorizer pour transformer les textes en une matrice TF-IDF.
   — Gardez les mêmes paramètres que pour BoW.
5. Entraînez des modèles de classification (**Logistic regression**, **Random Forest**, **MLP**) sur les deux représentations (BoW et TF-IDF).
6. Évaluez les performances avec une validation croisée (5 folds) et calculez l'accuracy et le F1-score.

Questions

— Quelle méthode (BoW ou TF-IDF) donne les meilleurs résultats ? Pourquoi pensez-vous que c'est le cas ?
— Que se passe-t-il pour les metriques si vous diminuez `max_features` ? (Montrez un graphique)

2 Partie 2 : Classification avec Sentence Transformers

Dans cette partie, vous utiliserez un modèle pré-entraîné de Sentence Transformers pour générer des embeddings de texte, puis vous les utiliserez comme features pour un modèle de classification.

Les embeddings d'un texte représentent ce texte sous forme de vecteur. L'objectif est d'exprimer la sémantique du texte sous forme de vecteurs, c'est-à-dire de caractéristiques (features) qui peuvent ensuite être utilisées dans un modèle de sklearn.

Consignes

1. Installez la bibliothèque sentence-transformers.
2. Chargez un modèle léger comme 'all-MiniLM-L6-v2' pour rester "CPU friendly".
3. Transformez chaque texte du jeu de données en un embedding.
4. Utilisez ces embeddings comme features pour entraîner un modèle de classification (**Logistic regression**, **Random Forest**, **MLP**).
5. Évaluez les performances avec une validation croisée (5 folds) et comparez avec les résultats de la Partie 1.

Questions

— Comment les embeddings se comparent-ils à BoW et TF-IDF en termes de performance ?
— Quels sont les avantages et inconvénients d'utiliser des embeddings pré-entraînés ?

3 Partie 3 : Le Super Learner

Qu'est-ce que le Super Learner ?

Le Super Learner est une méthode ensembliste qui combine les prédictions de plusieurs modèles d'apprentissage automatique pour obtenir une performance optimale. Contrairement à un simple vote majoritaire, il utilise une validation croisée pour apprendre une combinaison pondérée des modèles de base (ou "base learners"), optimisant ainsi une métrique comme l'*accuracy*. Cette approche est particulièrement puissante car elle s'adapte aux forces et faiblesses des différents modèles en fonction des données.

Plus précisément, le Super Learner construit une combinaison linéaire des prédictions des modèles de base, comme une régression linéaire ou logistique, où chaque modèle M_i reçoit un poids w_i. Ces poids sont appris à partir des données via une validation croisée, de sorte que la prédiction finale est de la forme : $y_{\text{pred}} = w_1 M_1(x) + w_2 M_2(x) + \cdots + w_k M_k(x)$. "Apprendre les poids" signifie ajuster ces coefficients w_i pour minimiser l'erreur globale, en exploitant les forces de chaque modèle.

Algorithme en pseudo-code

Voici une description simplifiée de l'algorithme du Super Learner :

Algorithm 1 Super Learner

Require: Données d'entraînement (X, y), liste de modèles de base $M = \{M_1, M_2, \ldots, M_k\}$, modèle méta $M_{\text{méta}}$, nombre de plis K

Ensure: Prédictions combinées

1: Diviser (X, y) en K plis pour la validation croisée
2: **for** chaque pli $k = 1$ à K **do**
3: Entraîner chaque modèle M_i sur les $K - 1$ plis restants
4: Prédire sur le pli k avec chaque M_i, stocker les prédictions $Z_{i,k}$
5: **end for**
6: Construire une matrice Z où chaque colonne contient les prédictions d'un modèle M_i sur tous les échantillons
7: Entraîner le modèle méta $M_{\text{méta}}$ sur (Z, y) pour apprendre les poids des modèles de base
8: **Pour les nouvelles données X_{test} :**
9: Prédire avec chaque M_i pour obtenir Z_{test}
10: Appliquer $M_{\text{méta}}$ sur Z_{test} pour obtenir la prédiction finale

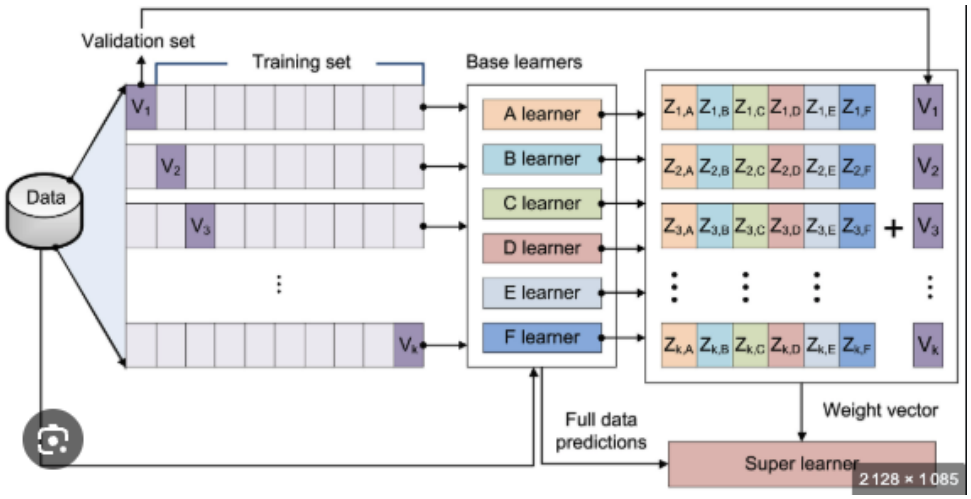

FIGURE 1 – Super Learner

Exemple pratique

Un exemple concret d'implémentation est disponible sur ce site. Dans cet exemple, plusieurs modèles sont combinés avec un modèle méta (souvent une régression logistique) pour prédire une classe. Vous pouvez vous en inspirer pour votre propre implémentation.

Consignes

1. **Implémentation du Super Learner** :
   - Implémentez le Super Learner en suivant l'exemple ci-dessus ou utilisez une bibliothèque comme `mlens` (optionnel).
   - Utilisez 3 modèles de base adaptés à la classification de texte : `LogisticRegression`, `RandomForestClassifier`, et `SVC`, en utilisant les features TF-IDF de la Partie 1.
   - Choisissez un modèle méta simple, comme une régression logistique (`LogisticRegression`).
2. **Tests de performance** :
   - Testez les performances (en *accuracy* et F1-score) du Super Learner avec les features TF-IDFs :
   - Présentez les résultats dans un tableau ou un graphique comparatif global (TF-IDF, Embeddings, Super Learner).

Questions

1. Commenter les graphiques obtenus
2. Quels sont les poids que votre méta-modèle a attribués à chaque modèle de base ?

4 Bonus

Il vous est possible de soumettre votre meilleur modèle de la partie 1 ou 2 pour participer à une compétition et comparer vos performances avec celles des autres. Un bonus de 10% et 5% est accordé aux premier et deuxième participants, respectivement. Vous pouvez soumettre votre code ici.

Livrables

— Un notebook Jupyter (ou script Python) avec le code commenté.
— Un court rapport (2 - 3 pages) résumant vos résultats et analyses, avec tableaux ou graphiques.

Ressources principales

— Documentation de scikit-learn
— mlens (optionnel)
— Implémentation pratique d'un Super Learner