Feature engineering: **Catégorisation de Texte**

**1. Introduction**

Ce projet porte sur l'application du Feature Engineering pour la classification de textes sportifs. L'objectif est d'explorer différentes approches de transformation et d'extraction des caractéristiques des textes afin d'améliorer la performance des modèles de classification.

Le jeu de données provient du répertoire bbcsport, contenant des articles classés en différentes catégories sportives :

* Football
* Rugby
* Cricket
* Tennis
* Athletics

Nous allons explorer les données, les nettoyer, extraire des caractéristiques, tester différents modèles de classification et comparer leurs performances.

**2. Lecture des données**

Les données sont constituées de fichiers texte organisés en sous-répertoires selon leurs catégories respectives. Le processus d'importation consiste à :

* Lire les fichiers texte et les stocker dans un DataFrame pandas avec deux colonnes : texte (contenu de l'article) et categorie (sport associé).
* Éliminer les doublons (10 doublons supprimés).
* Vérifier la répartition des catégories :
  + Football : 262 articles
  + Rugby : 146 articles
  + Cricket : 121 articles
  + Tennis : 99 articles
  + Athletics : 99 articles
* Rééquilibrage des classes à 149 observations par catégorie grâce au rééchantillonnage.
* Dimensions finales après nettoyage : 614 observations.

**3. Prédicteurs utilisés**

Les caractéristiques extraites sont classées en plusieurs catégories :

1. **Prédicteurs structurels**
   * Nombre de caractères, mots, phrases.
   * Longueur moyenne des mots et phrases.
   * Nombre de mots courts et mots longs.
2. **Prédicteurs linguistiques**
   * Nombre de verbes, noms, stopwords.
   * Richesse lexicale (rapport entre mots uniques et mots totaux).
   * Nombre de chiffres, majuscules, ponctuation.
3. **Analyse sémantique avec mots-clés**
   * Extraction de mots-clés spécifiques à chaque catégorie :
     + Football : goal, penalty, striker, league.
     + Tennis : serve, racket, wimbledon.
     + Cricket : bowler, batsman, wicket.
     + Rugby : scrum, tackle, conversion.
     + Athletics : 100m, sprint, marathon.
4. **Techniques d’extraction de texte**
   * TF-IDF (Term Frequency - Inverse Document Frequency).
   * Bag-of-Words (simple comptage des mots).
   * Word Embeddings (Word2Vec).
5. **Réduction du nombre de prédicteurs en utilisant Random Forest (60 à 20)**

**Robustesse face aux données bruitées**

* + Random Forest peut identifier et éliminer les variables inutiles ou redondantes.

**Prise en compte des interactions entre variables**

* + Contrairement à des méthodes comme la corrélation, il détecte les relations non linéaires entre les variables.

**Pas besoin de normalisation ou de transformation**

* + Fonctionne aussi bien avec des variables quantitatives que qualitatives sans preprocessing complexe.

**Amélioration de la performance du modèle**

* + Moins de variables = **moins de risque d’overfitting** et un modèle plus rapide à entraîner.

**Meilleure interprétabilité**

* + Réduit la complexité du modèle et facilite la compréhension des facteurs influents.

**4. Technique de nettoyage (Transformation)**

* Conversion en minuscules.
* Suppression de la ponctuation.
* Suppression des chiffres.
* Tokenization et suppression des stopwords.
* Lemmatisation des mots (réduction à leur forme de base).

**5. Algorithmes utilisés**

Les modèles de classification testés :

* **Régression Logistique**
* **Random Forest**
* **SVM (Support Vector Machine)**
* **KNN (K-Nearest Neighbors)**

Les modèles sont testés avec différentes techniques d'extraction de texte :

* TF-IDF
* Bag-of-Words
* Prédicteurs sélectionnés
* Word Embeddings

**6. Résultats obtenus**

Les performances sont évaluées selon **F1-score** et **Accuracy** :

| **Technique** | **Logistic Regression (F1, Accuracy)** | **Random Forest (F1, Accuracy)** | **SVM (F1, Accuracy)** | **KNN (F1, Accuracy)** |
| --- | --- | --- | --- | --- |
| **TF-IDF** | 98.37%, 98.38% | 96.77%, 96.75% | 97.58%, 97.56% | 93.60%, 93.50% |
| **Bag-of-Words** | 97.56%, 97.56% | 95.94%, 95.94% | 87.24%, 86.99% | 64.97%, 65.04% |
| **Word Embeddings** | 32.51%, 35.77% | 45.63%, 40.53% | 11.83%, 24.39% | 32.63%, 33.33% |
| **Prédicteurs de base** | 97.57%, 97.56% | 99.19%, 99.19% | 87.24%, 86.99% | 68.18%, 68.29% |
| **Prédicteurs sélectionnés** | 72.33%, 72.36% | 73.23%, 73.17% | 71.77%, 71.54% | 61.54%, 61.79% |

**Analyse des résultats :**

* TF-IDF et Random Forest sont les plus efficaces, atteignant **98.38% de précision**.
* Bag-of-Words fonctionne bien avec la Régression Logistique (97.55%), mais moins avec KNN.
* Word Embeddings sous-performe, probablement en raison d'un entraînement insuffisant.
* Les Prédicteurs sélectionnés sont utiles mais offrent des résultats inférieurs à TF-IDF.

**7. Difficultés rencontrées**

* **Déséquilibre des classes** : nécessité d’un rééchantillonnage.
* **Complexité des modèles** : Word2Vec nécessite plus de données.
* **Problèmes de surapprentissage** : nécessité d’optimiser les hyperparamètres.
* **Temps de calcul** : le nettoyage et l’extraction des caractéristiques sont gourmands en ressources.
* **Difficulté avec Word Embeddings** : obtenir de bons résultats avec Word Embeddings est complexe en raison du besoin d'un grand volume de données annotées et de la nécessité d'un entraînement plus long pour capter des relations sémantiques pertinentes.
* **Déséquilibre des classes** : nécessité d’un rééchantillonnage.
* **Complexité des modèles** : Word2Vec nécessite plus de données.
* **Problèmes de surapprentissage** : nécessité d’optimiser les hyperparamètres.
* **Temps de calcul** : le nettoyage et l’extraction des caractéristiques sont gourmands en ressources.

**8. Compétences acquises**

* Prétraitement avancé de texte (nettoyage, tokenization, lemmatisation).
* Application de techniques de Feature Engineering.
* Comparaison de plusieurs algorithmes de Machine Learning.
* Utilisation d'outils de visualisation et d’analyse de données.
* Interprétation des performances des modèles et sélection des meilleurs prédicteurs.

**Conclusion**

* **Random Forest + TF-IDF** donne les meilleurs résultats.
* **Les Prédicteurs sélectionnés** permettent d'améliorer la classification sans surcharger le modèle.
* **L’utilisation de Word Embeddings** nécessite davantage de données pour être efficace.

Ce projet illustre comment le Feature Engineering améliore les performances des modèles de classification de texte. 🚀