INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN NORMAND

Mastère spécialisé®
EXPERT EN SCIENCES DES DONNÉES

PROJET

Entraînement d'un algorithme de classification de sentiments avec Pyspark

Analyse des données massive

Année universitaire 2022-2023

Réalisé par :

AYENA Mahougnon

Prof:

M. Erwan KOFFI

Rapport

Le présent rapport décrit les méthodes utilisées dans le projet d'analyse des données massives, ainsi que l'utilisation de la distribution des calculs dans le cadre du traitement des données à grande échelle. Le code source utilisé pour le projet est en Python avec l'utilisation de la bibliothèque PySpark.

Le projet comprend les étapes suivantes :

- ✓ Lecture des données : Les données d'entraînement et de test sont lues à partir de fichiers JSON à l'aide de la bibliothèque PySpark. La session Spark est créée, et les données sont chargées dans des DataFrames.
- ✓ Prétraitement des données : Dans cette étape, les données sont préparées avant de les utiliser pour l'entraînement des modèles. Différentes méthodes d'extraction de caractéristiques sont utilisées, telles que HashingTF, TF-IDF et CountVectorizer, pour convertir les mots en vecteurs numériques. Les étiquettes de polarité sont également converties en valeurs numériques.
- ✓ Modélisation : Trois modèles de classification sont utilisés : la régression logistique, le Random Forest et le Naive Bayes. Le modèle de régression logistique a donné les meilleurs résultats, avec une précision de 73,46% sur les données de test. Une validation croisée est effectuée sur le modèle de régression logistique pour sélectionner les meilleurs paramètres.
- ✓ Autres méthodes d'extraction : Deux autres méthodes d'extraction de caractéristiques sont utilisées : TF-IDF et CountVectorizer. En utilisant CountVectorizer avec la régression logistique, le meilleur score obtenu est de 76,11%.
- ✓ Prédiction sur le fichier noclass.json : Le meilleur modèle, c'est-à-dire la régression logistique avec CountVectorizer, est utilisé pour faire des prédictions sur le fichier noclass.json. Les prédictions sont ensuite exportées dans un fichier JSON.

Le calcul distribué est utilisé à plusieurs étapes du projet, notamment lors du chargement des données, du prétraitement des données et de l'entraînement des modèles. La bibliothèque PySpark permet de tirer parti de la distribution des calculs pour accélérer le traitement des données massives.

- Chargement des données :

Le chargement des données à partir des fichiers JSON est réalisé en parallèle sur un cluster Spark distribué. Chaque nœud du cluster lit une partie des données, puis les données sont consolidées dans un DataFrame distribué. Le partitionnement des données peut être automatiquement effectué en fonction du nombre de nœuds et de la taille des données.

- Exploration des données :

Les opérations d'affichage des premières lignes (show) et de comptage des lignes (count) sont réalisées localement sur le DataFrame. Ces opérations n'impliquent pas de calcul distribué car elles sont effectuées sur des parties spécifiques du DataFrame stockées localement.

- Prétraitement des données :

Les étapes de prétraitement des données (tokenization, suppression des mots vides, transformation HashingTF) sont définies en utilisant les API de Spark ML. Lorsque le pipeline est appliqué aux données d'entraînement (pipelineFit.fit(train)), les transformations sont effectuées en parallèle sur chaque partition du DataFrame distribué. Les données sont

divisées en partitions et chaque partition est traitée indépendamment sur différents nœuds du cluster Spark. Les résultats intermédiaires des transformations sont stockés dans un nouveau DataFrame distribué.

- Modélisation :

L'entraînement des modèles (régression logistique, Random Forest, Naive Bayes) est effectué de manière distribuée. Chaque partition du DataFrame d'entraînement est traitée indépendamment sur différents nœuds du cluster Spark. Les calculs sont parallélisés et exécutés simultanément sur les différentes partitions du DataFrame. Les modèles entraînés sont stockés de manière distribuée.

- La validation croisée avec la régression logistique :

La validation croisée est effectuée en utilisant la classe CrossValidator de Spark ML. Le processus de validation croisée est effectué de manière distribuée. Différentes combinaisons de paramètres sont évaluées simultanément sur différentes partitions des données d'entraînement. Les performances des modèles sont évaluées sur différentes partitions des données de test. Les résultats de validation croisée sont agrégés et analysés pour sélectionner les meilleurs paramètres.

- Autres méthodes d'extraction de caractéristiques :

Les autres méthodes d'extraction de caractéristiques (TF-IDF, CountVectorizer) sont également réalisées de manière distribuée.Les transformations des données (tokenization, suppression des mots vides, TF-IDF, CountVectorizer) sont appliquées en parallèle sur chaque partition du DataFrame. Les résultats intermédiaires et les modèles entraînés sont stockés de manière distribuée.