Mini-projet Machine learning avec MLLIB

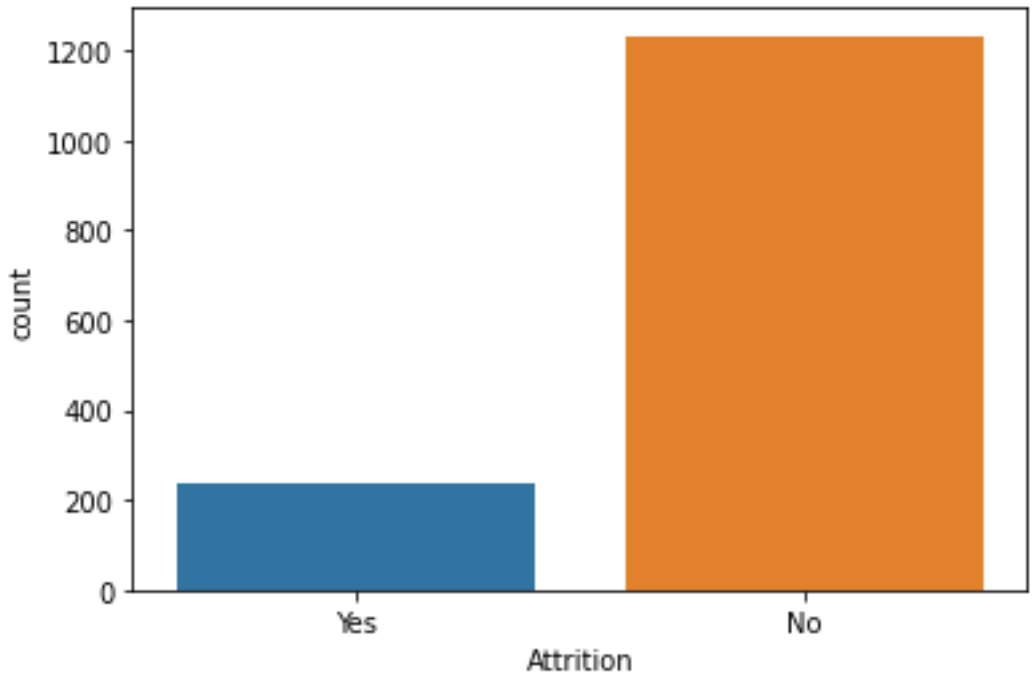
Le problème de prédiction de l'attrition des employés consiste à prédire si un employé quittera ou non son poste. Cela peut être utile pour les entreprises afin de prendre des mesures préventives pour retenir leurs employés précieux et réduire le taux d'attrition. L'attrition peut être coûteuse pour les entreprises en termes de recrutement et de formation de nouveaux employés, ainsi que de la perte de connaissances et d'expertise. L'utilisation de Spark MLLIB permet de construire des modèles prédictifs à grande échelle pour résoudre ce problème. En utilisant des techniques telles que l'encodage des caractéristiques, la visualisation des données et la construction de modèles Random Forests et Gradient Boosting, les data scientists peuvent développer des modèles prédictifs précis qui peuvent aider les entreprises à réduire le taux d'attrition et à améliorer la rétention des employés.

1. **Dataset :*Attrition et performance des employés IBM HR Analytics**

Prévoyez l'attrition de vos précieux employés

Il s'agit d'un ensemble de données fictif créé par les data scientists d'IBM. Nous devons explorer l'ensemble de données, comprendre les algorithmes et les techniques qui peuvent y être appliqués. Nous essaierons d'obtenir des informations significatives à partir de l'ensemble de données, comme quels sont les facteurs qui ont un impact sur l'attrition des employés' ll try to gain meaningful insights from the dataset, like what are the factors which have an impact on Employee Attrition.

```
"https://raw.githubusercontent.com/msellamiTN/Machine-Learning-with-
Python/master/data/HR-Employee-Attrition.csv
```



2. Travail demandé :

1. Importer les données : Tout d'abord, il est nécessaire d'importer les données dans un format exploitable par Spark MLLIB. Les données peuvent être stockées dans des fichiers CSV, des bases de données ou des formats de fichiers parquet, entre autres.
2. Explorer les données : Une fois les données importées, il est important d'en explorer la structure et les caractéristiques. Cela peut inclure l'analyse des valeurs manquantes, des distributions de variables et de l'existence de corrélations entre les variables.
3. Nettoyer les données : Après l'exploration des données, il est possible que certaines données soient incohérentes, mal formatées ou contiennent des valeurs aberrantes. Dans ce cas, il est important de nettoyer les données avant de les utiliser pour entraîner un modèle.
4. Préparer les données : Selon le modèle que vous souhaitez utiliser, vous devrez peut-être préparer les données pour le rendre compatible avec ce modèle. Cela peut inclure la normalisation des variables, la transformation des variables catégorielles en variables numériques et la séparation des données en ensembles de formation et de test.
5. Sélectionner les fonctionnalités : Si les données contiennent de nombreuses variables, vous pouvez envisager de sélectionner les fonctionnalités qui sont les plus pertinentes pour votre modèle. Cela peut aider à améliorer la précision de votre modèle et à réduire le temps nécessaire pour l'entraîner.
6. Équilibrer les données : Si les données contiennent un déséquilibre entre les classes, il peut être nécessaire d'équilibrer les données en utilisant des techniques telles que SMOTE (Synthetic Minority Over-sampling Technique) pour augmenter le nombre d'échantillons de la classe minoritaire.

7. Diviser les données : Une fois que les données sont nettoyées et préparées, il est important de diviser les données en ensembles de formation et de test. L'ensemble de formation est utilisé pour entraîner le modèle, tandis que l'ensemble de test est utilisé pour évaluer les performances du modèle.
8. Entraîner le modèle : Utilisez l'ensemble de formation pour entraîner le modèle en utilisant l'algorithme approprié. Les algorithmes populaires pour la classification incluent Random Forest, Gradient Boosting, Logistic Regression et Decision Tree.
9. Évaluer le modèle : Une fois que le modèle est entraîné, utilisez l'ensemble de test pour évaluer ses performances. Les métriques courantes pour évaluer les performances d'un modèle de classification incluent la précision, le rappel, le score F1 et la courbe ROC.
10. Optimiser le modèle : Si le modèle ne répond pas aux attentes, il est possible que les paramètres du modèle doivent être optimisés pour obtenir de meilleures performances. Cela peut inclure l'ajustement des hyperparamètres, la modification de l'algorithme ou la sélection de différentes fonctionnalités.
11. Appliquer le modèle : Une fois que le modèle est entraîné et testé, il peut être appliqué aux nouvelles données pour faire des prédictions sur la probabilité d'attrition d'un employé.
12. Déployer le modèle comme une API Flask en utilisant docker et docker-compose
13. Surveiller le modèle avec Elastic Search et utiliser Kibana pour créer un Dashbord

3. Livrable

- Lien vers github de projet
- Rapport explique les étapes

Dernier délai ferme : 30/04/2023 minuit

Tout travail envoyé après ce délai ne sera pris en considérations