Projet - Présentation
Idée foodApp pour la santé

Préambule

Dans ce TP, vous allez travailler sur un dataset avec des informations sur des produits alimentaires différents. Vous utiliserez Apache Spark pour charger, nettoyer et analyser les données. Enfin vous chercherez une idée d'application, en vous basant sur les données exploitées.

Objectifs

- Charger les données à l'aide d'Apache Spark.
- Effectuer une analyse exploratoire des données.
- Nettoyer les données en traitant les valeurs manquantes et les valeurs aberrantes.
- Explorer les caractéristiques des produits alimentaires.
- Générer un rapport d'analyse des données.

Instructions

1. Charger les données

1.1) Importez les packages requis pour utiliser les fonctionnalités de Spark SQL. Cela inclut SparkSession, DataFrame et les fonctions de manipulation de DataFrame telles que functions._.

1.2) Créez une session Spark avec le nom de l'application "FoodApp" et en mode local avec autant de threads que de cœurs disponibles.

1.3) Chargez les données à partir du fichier CSV `"data/datafoodfr.csv"` en spécifiant que la première ligne contient les noms de colonnes.

1.4) Affichez les cinq premières lignes du DataFrame chargé

2. Analyse exploratoire des données

2.1) Calculer / Afficher le nombre de lignes et de colonnes dans le DataFrame.

→ `data.count()` pour le nombre de lignes et `data.columns.length` pour le nombre de colonnes.

2.2) Identifiez les valeurs manquantes dans chaque colonne.

→
```
data.select(data.columns.map(c =>
sum(col(c).isNull.cast("int")).alias(c)): _*)
```

2.3) Définissez une fonction calculateMissingPercentages() qui prend un tableau de noms de colonnes comme argument et retourne un tableau de tuples contenant le nom de la colonne et le pourcentage de valeurs manquantes pour chaque colonne. À l'aide de la méthode filter() et de la fonction isNull(), comptez le nombre de valeurs manquantes pour chaque colonne, puis calculez le pourcentage en le divisant par le nombre total de lignes

2.4) Utilisez ce code pour Afficher les pourcentages de valeurs manquantes pour chaque segment de colonnes.

```
segments.foreach { case (title, columns) =>
  val missingPercentages = calculateMissingPercentages(columns)
  println(s"Taux de valeurs manquantes pour $title:")
  missingPercentages.foreach { case (colName, percentage) =>
    println(f"$colName%-30s ${percentage.toDouble}%10.6f")
  }
  println()
}
```

2.5) Effectuez une analyse descriptive des données pour comprendre la distribution des valeurs numériques.

→
```
data.describe().show()
```

2.6) Sélectionnez uniquement les colonnes pertinentes

→Utiliser .select() en utilisant la fonction map

3. Nettoyage des données

3.1) Créez une fonction pour remplacer les valeurs aberrantes par NaN

3.2) Exécutez-la sur la dataframe et faites une description des données nettoyées

3.3) Supprimez les lignes ayant des valeurs manquantes dans la colonne "product_name".

3.4) Remplacer les valeurs aberrantes de la colonne `salt_100g` par *null* via la fonction lit()

3.5) Décrire à nouveau les données nettoyées

3.6) Filtrer les produits qui ont un nom à l'aide .filter(), col() et .isNotNull

3.7) Chercher les produits contenant le mot "thé" dans le nom à l'aide .contains

3.8) Filtrer les produits qui ont une catégorie principale non nulle et créer une copie

3.9) Créer une nouvelle colonne "rang_category" combinant "main_category_fr" et "categories_fr" à l'aide de .withColumn()

4. Analyse des caractéristiques des produits alimentaires

4.1) Définir les catégories que nous allons garder

4.2) Filtrer les produits qui appartiennent aux catégories définies

4.3) Définir les nouvelles catégories renommées « renamed_category »

4.4) Créer un DataFrame avec les catégories originales et renommées

4.5) Le joindre avec le dataframe de travail

4.6) Supprimer la colonne "rang_category"

4.7) Sélectionner uniquement les colonnes nécessaires

4.8) Remplir les valeurs manquantes pour product_name et renamed_category avec les valeurs les plus fréquentes

4.9) Calculer les médianes pour les colonnes numériques et remplir les valeurs manquantes avec les médianes

5. Génération du rapport d'analyse

5.1) Réduisez le nombre de partitions du DataFrame à une seule. Utilisez `coalesce(1)` pour réduire le nombre de partitions.

5.2) Enregistrez le DataFrame nettoyé au format CSV. Utilisez `write.format("csv")` pour enregistrer le DataFrame au format CSV.

6. Analyse des données et Idée d'Application

Faites une analyse des features de votre dataframe (par ex. la répartition du nutriscore) et éventuellement une corrélation entre elles qui vous permettra d'expliquer votre idée d'application pour la santé