Atelier SD3: Catégorisation de produits (Cdiscount)

Résumé

1 Introduction

1.1 Objectif général

L'objectif général est l'affectation d'un produit à une catégorie à partir de son descriptif écrit *langage naturel*. Ce problème se rencontre sur tous les sites marchands dont Cdiscount à l'origine de ces données proposée à un concours du site datascience.net. L'objectif est donc d'apprendre sur une base d'apprentissage textuelle à déterminer la classe du produit pour l'appliquer à un échantillon test.

1.2 Les données

Les données sont accessibles sur demande auprès de Cdiscount, mais pas la solution de l'échantillon test utilisé lors du concours. La base d'apprentissage initiale comporte vingt millions de produits dont un échantillon test est extrait.

Nombre de catégories ? ?

1.3 Les solutions

Le descriptif des trois solutions gagnantes a fait l'objet d'une communication aux journées de Statistique (2016). Elles exploitent une "pyramide" de modèles de régressions logistiques pour refléter la structure hiérarchique des catégories à trois niveaux. Ce choix de méthode est également imposé par la nécessité de stocker tous les modèles dans un espace disque contraint (plusieurs GO). Des modèles plus complexes, obtenus par agrégation, comme c'est souvent le cas dans ce type de concours seraient trop volumineux. Un point important dans la qualité des modèles est la prise en compte du gtrand déséquiibre des effectifs des catégories. Des astuces de sous-échantillonnage des classes très fréquentes et de sur-échantillonnage de celles très peu fréquentes permet de rétablir une forme d'équilibre. Enfin dernière astuce d'une des solutions qui consiste à exécuter une forme de bagging : moyenne de modèles obtenus par en échantillonnant plusieurs fois les classes les plus fréquentes tout en conservant les produits des classes les moins fréquentes.

1.4 Étapes de l'analyse

La préparation des données est très importantes. Il s'agit d'un exemple caractéristique de fouille de données textuelles enchaînant généralement les étapes suivantes :

- nettoyer les données des caractères spéciaux,
- supprimer les mots de liaison inutiles (*stop words*) pour l'identificaiton des classes,
- réduire les mots à leur seule racine pour obtenir un dictionnaire réduit de tous ceux utilisés,
- simplifier encore le dictionnaire par une opération de hachage (*hashing tric*),
- remplacer les mots ou plutôt leur code de hachage par une quantification dépendant des nombres relatifs d'occurrences (TF-IDF).

Enfin, il reste simplement à estimer les modèles classiques de discrimination. Comme souvent, la prétraitement des données (*data munging*) est essentiel à la qualité du modèle finalement obtenu.

1.5 objectif de l'atelier

Attention, par souci de simplification, cet atelier ne s'intéresse qu'à la prévision du seul premier niveau des 39 catégories. L'objectif est de comparer les performances de deux stratégies, l'une programmée en python, l'autre utilisant la librairie MLlib de Spark et donc la possibilité de distribuer les données sur un cluster. l'objectif n'est pas de faire mieux que les solutions du concours. D'autre part, comme cela est moins criant que pour le 3ème niveau de catégories, le déséquilibre des classes n'est pas pris en compte.

Plusieurs méthodes de classification supervisée sont testées (logistiques, arbres, forêts aléatoires) mais même sur cette version simplifiée du problème, la batterie de régression fournit de meilleures résultats.

2 Déroulement de l'atelier

2.1 Ressources

Plusieurs tutoriels d'initiation sont disponibles afin d'acquérir les compétences techniques nécessaires en plus de celles du cours d'apprentissage statistique.

Pédagogiques

- Initiation à Python et les librairies pandas et scikit-learn.
- Introduction à pySpark et à la manipulation de bases de données résilientes.
- Introduction à MLlib qui implémente quelques algorithmes classiques pour données massives car *scalable* (passe à l'échelle volume) dont les forêts aléatoires.

D'autres compétences spécifiques au traitement des données textuelles sont requises.

2.1.1 Matériel

En fonction du contexte : ordinateur individuel multi-cœurs, cluster...

3 Traitement de données textuelles

3.1 Nettoyage

Mots d'arrêt

3.2 Racinisation

3.3 Hachage

3.4 Vectorisation

4 Solutions

4.1 Avec Python

4.2 Avec MLlib

Un calepin décline les instructions pyspark permettant d'opérer les fonctions de la librairies MLlib de Spark sur des bases de données résilientes et donc susceptibles de passer à l'échelle volume en s'adaptant aux capacités (nombre de machines ou de nœuds) d'un cluster.

C'est important pour assure le passageà l'échelle susceptible de prendre en compte des millions de produits à traiter

5 Conclusion

Comment répondre à la question suivante ? Est-il plus efficace de traiter un sous-échantillon des données avec par exemple Python ou est-il préférable de mettre en œuvre une plateforme plus sophistiquée avec les technologies de données distribuées Spark et Hadoop assurant en principe le passage à l'échelle.

Le choix s'opère à deux niveaux, celui de préparation des données, puis celui d'estimation du modèle de discrimination.