

Big Data avec R ? Mode d'emploi !

Publié le 17 janvier 2017

R, un langage de référence dans le domaine de l'assurance et de la banque

R est un langage de référence chez les actuaires et plus largement dans les domaines de la banque et de l'assurance. Il bénéficie en effet d'une communauté très active sur les réseaux sociaux (la popularité du hash tag #rstats sur twitter en témoigne) et d'un répertoire de package impressionnant (le CRAN), mettant ainsi à la disposition de chaque développeur R les dernières nouveautés. Il suffit d'une ligne de code pour accéder aux dernières trouvailles de Hadley Wickham (véritable rock star de la donnée). En 2015, deux fois plus d'articles scientifiques s'appuyaient sur R que sur Python pour traiter des sujets de Data Science. R a en effet su s'adapter à l'émergence des nouvelles problématiques et en particulier celle du volume de données de plus en plus grand.

Les domaines de la banque et de l'assurance sont fortement impactés par la Data Science

L'assurance et la banque sont des domaines où les données sont au cœur du métier. Les données sont utilisées pour tarifier les produits, détecter la fraude, prédire les comportements des clients, faire du ciblage marketing... L'accès à plus de données est donc avant tout une opportunité pour avoir des analyses toujours plus précises. Certes, le Big Data nécessite de nouvelles compétences, mais le jeu en

vaut la chandelle. En témoignent les data labs qui fleurissent chez les assureurs ou dans les banques. De même, des start-ups Assurtech/Fintech se lancent dans la course et essaient de s'emparer des nouvelles opportunités telles que la blockchain ou l'assurance paramétrique. Le point commun à tous ces changements est que la donnée occupe une place toujours plus importante.

R, Python, C++... sont limités par la taille de la mémoire vive

La quasi-totalité des langages de programmation sont des langages « in-memory », c'est-à-dire que tous les objets sont stockés dans la mémoire vive (RAM). Par conséquent, si vous avez comme moi 16GB de mémoire vive, vous ne pourrez pas travailler sur des bases de données de plus de 16GB. En fait, c'est même plus contraignant, car les algorithmes de Machine Learning requièrent de l'espace libre pour faire tourner les modèles. Vous pourrez donc travailler sur des data sets de l'ordre de 10Go ce qui correspond grosso modo à un CSV de 100 millions de lignes. C'est déjà bien, mais il est désormais courant de dépasser cette limite dans les domaines de l'assurance ou de la banque, d'où la question Comment faire du Big Data avec R ?

Comment faire du Big Data avec R ? Ou plus modestement comment gérer plus de données sous R ?

Cette question était au cœur de la conférence annuelle 'useR' de 2013. Depuis cette date, un engouement s'est créé autour de cette question. Les gurus de R se sont emparés de la question et ont proposé différentes solutions. La première solution relève du bon sens. Puisque qu'il n'y a plus de place dans la mémoire vive (RAM), utilisons le disque dur (ROM). On entre alors dans le monde de l'out-of-core programming.

Utiliser le disque dur pour stocker les données qui débordent de la RAM permet de gérer en local des bases de données beaucoup plus volumineuses. Les ordinateurs portables ont en effet couramment plus de 500Go de ROM. On peut donc gérer plus de données, sans dépenser un centime dans du nouveau matériel informatique. Cette solution reste néanmoins réservée aux experts en informatique. L'accès aux données sur le disque dur étant relativement lent, cette technique nécessite une gestion très fine de la mémoire. Un bon nombre d'algorithmes d'algèbre linéaire (factorisation matricielle LU, QR, Cholesky, analyse en composante principale...) ont été adaptés pour l'out-of-core programming. Mais, à ma connaissance, ce n'est pas le cas des algorithmes de Machine Learning. Donc si vous vous sentez l'âme de relever ce défi, les packages « ff » ou « ffbase » de R sont faits pour vous. Attendez-vous aussi à vous frotter à la notion de sérialisation, à bon entendeur...

Sinon je vous invite à lire la suite de l'article, car heureusement d'autres solutions plus simples existent. Que ce soit sur le cloud ou en local, la mémoire vive est désormais disponible à des prix abordables. Si vous êtes prêts à dépenser quelques euros pour abriter vos données dans la mémoire vive, alors suivez-moi ! Je vous propose un petit tour d'horizon des meilleures solutions dans le domaine.

Aujourd'hui, R offre la possibilité de faire du Big Data. Tour d'horizon des meilleures options selon votre infrastructure

Si vous comptez travailler sur une seule machine (que ce soit un serveur avec des centaines de Go de RAM ou votre ordinateur), alors il vous suffit de choisir les bons packages sur R et tout ira bien ! Les calculs seront automatiquement parallélisés sur vos processeurs. R supporte jusqu'à 8To de RAM. La limite proviendra donc sûrement plus du matériel informatique que du langage R. Si vous optez pour un serveur avec 100Go de RAM, alors vous serez limités à 100Go de mémoire pour importer vos données et faire tourner vos algorithmes de machine learning.

Quelques recommandations pour les packages à utiliser :

- Pour la manipulation de données : data.table et dplyr devraient faire l'affaire ;
- Pour la partie machine learning : XGboost (le plus performant pour faire du boosting) et H2O (très bonne performance et une librairie d'algorithmes de data science complète) ;
- Pour la partie visualisation : ggplot2 ou Rshiny si l'on veut des graphes interactifs.

En revanche, si vous souhaitez gérer des bases des données encore plus grandes (de l'ordre de plusieurs To) et travailler sur plusieurs serveurs, il faudra alors recourir au combo : Spark+Hadoop+H2O, soit en un mot Sparkling Water. La combinaison de Spark et Hadoop permet de traiter efficacement des volumes de données gigantesques grâce à une architecture distribuée. H2O apporte quant à lui les algorithmes de machine learning. La version 2.0 de Sparkling Water est sortie il y a tout juste quelques mois. Il reste certes encore quelques ajustements à faire pour assurer la stabilité de cette solution lorsqu'elle est déployée sur des grandes structures (plus de 20 nœuds). Ce problème devrait être résolu dans les semaines à venir, la version corrigeant ce problème

de stabilité étant déjà en phase de test. Dans cette solution, R va en fait servir de chef d'orchestre entre ces différentes technologies. Du côté du développeur R, peu de choses changeront. Il pourra tout installer en quelques lignes de code, puis continuer à coder en R comme à son habitude tout en profitant de la performance de Sparkling Water. Vous pourrez trouver ce nouveau package R sous le nom de Rsparkling.

On peut résumer tout cela en une équation :

$$R*(Spark+Hadoop+H2O) = Rsparkling$$

Python a bien sûr aussi résolu l'équation :

$$Python*(Spark+Hadoop+H2O) = pySparkling$$

Attention tout de même, comme Spark et Windows ne s'entendent pas très bien, cette dernière solution fonctionne mal (voire pas du tout) sous Windows. OS X et Linux sont donc à privilégier pour s'éviter un casse-tête pour tout installer correctement.

Let's test and learn now :)

Mathieu Léchine, Data Scientist chez DataSquare



More efficient. **Together**

Pour plus d'information datasquare.fr