De la Statistique à la Science des (grosses) Données

Résumé

Statistique, fouille ou Science des Données, les appellations changent le volume et la diversité des données explosent, les technologies se succèdent, les modèles et algorithmes se complexifient. L'estimation devient un apprentissage, la prévision remplace l'explication.

Le parcours pour devenir data scientist *est structuré en quatre parties :*

Saison 1 *(L3) Statistique élémentaire, descriptive* vs. *inférentielle.*

Saison 2 *(M1) Statistique Exploratoire multidimensionnelle et apprentissage non supervisé.*

Saison 3 *Apprentissage Statistique / Machine supervisé.*

Saison 4 *(M2) Technologies pour la Science des (grosses) Données.*

plus des réflexions sur : Statistique et Déontologie scientifique.

1 Origines de la *Data Science*

Le terme de *data scientist* à été "inventé" par Dhanurjay "DJ" Patil (LinkedIn) [1] et Jeff Hammerbacher (Facebook) en cherchant comment caractériser les métiers des données pour afficher des offres d'emploi : *Analyste, ça fait trop Wall Street ; statisticien, ça agace les économistes ; chercheur scientifique, ça fait trop académique. Pourquoi pas "data scientist" ?*

Une "définition" attribuée à J. Wills (Cloudera) est souvent reprise :*Data scientist (n) : Person who is better at statistics than any software engineer and better at software than any statistician*

La Science des Données n'est pas une nouvelle science créée *ex nihilo* mais l'association de compétences (informatique, mathématiques, métiers) résultat d'une longue évolution parallèle à celle des moyens de calcul et des volumes de données concernés. Cette évolution est passée par l'*analyse des données* en France, l'*Exploratory Data Analysis* ou EDA au USA, le *data mining* ou fouille des données puis la *Bioinformatique*.

En voici un bref résumé nécessairement schématique avec une chronologie linéaire :

1930-70 – hOctets Il était une fois la *Statistique* (inférentielle) : une question, (*e.g.* biologique), associée à une *hypothèse expérimentalement réfutable* H_0, une expérience *planifiée* avec un échantillon *représentatif* de $n \approx 30$ individus observés sur p (moins de 10) variables, un modèle *linéaire gaussien* supposé *vrai*, un test, une décision, donc une réponse qui peut être inférée à la population en contrôlant le risque (généralement 5%) de rejeter à tort H_0.

1970s – kO Les premiers outils informatiques se généralisant et, pour échapper à l'impérialisme du modèle linéaire, l'*analyse des données* (Caillez et Pages, 1976)[2] se développe en France ; l'*Exploratory Data Analysis* ou EDA aux États-Unis (Tukey 1977)[9]. L'objectif est alors de décrire ou explorer, prétendument sans modèle, des données déjà plus volumineuses.

1980s – MO En Intelligence Artificielle (IA), les *systèmes experts* expirent, supplantés par l'apprentissage des *réseaux de neurones*. La Statistique développe des modèles non-paramétriques ou fonctionnels.

1990s – GO *Data Mining* et *Premier changement de paradigme*. Les données ne sont plus *planifiées*, elles sont préalablement acquises et basées dans des entrepôts pour les objectifs usuels (*e.g.* comptables) de l'entreprise. L'aide à la décision les valorise : *From Data Mining to Knowledge Discovery* (Fayyad ; 1997)[4]. Les logiciels de fouille regroupent dans un même environnement des outils de gestions de bases de données, des techniques exploratoires et de modélisation statistique. C'est l'avènement du marketing quantitatif et de la gestion de la relation client (GRC ou CRM). L'IA se développe avec l'émergence du (*Machine Learning*) dont un sous-ensemble de méthodes est mis en exergue par le livre de Vapnik (1998) : *The Nature of Statistical Learning Theory*.

1. Entretien publié dans un article de l'Obs.

2000s –TO *Deuxième changement de paradigme*. Le nombre p de variables explose (de l'ordre de 10^4 à 10^6), notamment avec les biotechnologies omiques où $p >> n$ et la Bioinformatique. Le FDR (*False Discovery Rate*) de Benjamini et Hochberg (1995)[1] se substitue à la p-valeur et l'Apprentissage Statistique (Hastie et al. 2009)[5] sélectionne des modèles en optimisant leur complexité par un meilleur compromis *biais vs. variance* ; minimiser conjointement erreur d'*approximati* (biais) et erreur d'*estimation* (variance).

2010s – PO *Troisième changement de paradigme*. Dans les applications industrielles, le e-commerce, avec la géo-localisation, la *datafication* du quotidien où toutes les traces numériques sont enregistrées, c'est le nombre n d'individus qui explose ; les statistiques usuelles de test, toutes significatives, perdent leur utilité au profit des méthodes d'apprentissage non supervisées ou supervisées ; les bases de données se déstructurent et se stockent dans les nuages (*cloud computing*), les moyens de calculs se groupent (*cluster*), mais la puissance brute ne suffit plus à la voracité (*greed*) des algorithmes. Un troisième terme d'erreur est à prendre en compte : celle d'*optimisation*, induite par la limitation du temps de calcul ou celle du volume des données considéré ; leur flux nécessite la construction de décisions adaptatives ou séquentielles.

Une présentation plus détaillée de la "science des données" et ses implications notamment économiques est proposée par Besse et Laurent (2015).

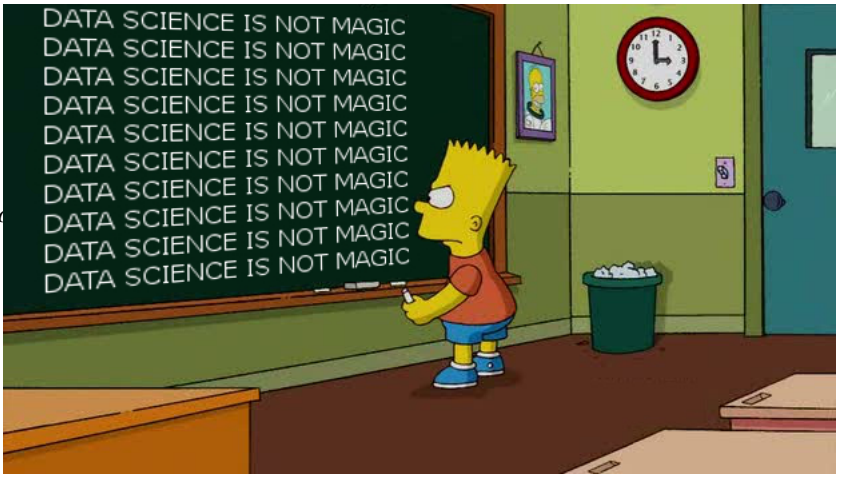

FIGURE 1 – *À copier 100 fois.*

2 Environnement logiciel

2.1 Logiciels de fouille de données

Dès les années 90, et provoquant l'avènement de la *fouille de données* (*data mining*), les éditeurs de logiciels commerciaux et les communautés de logiciels libres ont inclus dans leurs suites, en plus des modèles linéaires classiques, les différents algorithmes d'apprentissage au fur et à mesure de leur apparition. Ceux-ci ont été intégrés à un ensemble plus complet de traitement des données en connexion avec les gestionnaires de bases de données relationnelles, le tout pilotable par une interface graphique plus ou moins conviviale : *Clementine* de SPSS, *Enterprise Miner* de SAS, *Insightfull Miner* de Splus, KXEN, SPAD, *Statistica Data Miner*, Statsoft, WEKA... Leur apparente simplicité d'utilisation a largement contribué à la diffusion de méthodes sophistiquées dans des milieux difficilement perméables à une conceptualisation mathématique abstraite et peu armés pour des développements logiciels importants.

Dans un paysage en constante évolution ou révolution, les langages R (2015)[7] et Python (Rossum et Guido ; 1995)[8] jouent un rôle particulier. L'analyse des offres de stage et d'emploi montre de profonds changements dans les demandes. SAS, plébiscité jusqu'à la fin du siècle dernier, est largement supplanté par R et maintenant Python pour des raisons d'évidente économie mais aussi de flexibilité. Nous limiterons cette courte introduction à cs trois références.

2.2 SAS

Évolution

Le système SAS a acquis le siècle dernier et depuis sa mise en route au début des années 60, une situation dominante dans beaucoup de secteurs d'activités. En France, les grandes entreprises de l'énergie et administrations : INSEE,

EDF, GDF,..., toute l'industrie pharmaceutique l'ont largement adopté ainsi que les entreprises du tertiaire impliqués dans la gestion volumineuse de bases clientèles (banques, assurances, marketing, VPC...). Afin d'afficher une diversification de ses activités, SAS ne signifie plus *Statistical Analysis Sytem* ; le calcul statistique est devenu accessoire au regard des tâches d'ingénierie globale des systèmes d'information. Historiquement, SAS a suivi l'expansion des sites IBM sur lesquels il a été conçu et conserve, de cet environnement initial, les caractéristiques fondamentales : complexité, lourdeur, coût mais aussi puissance et efficacité.

SAS Institute, a en effet adopté la stratégie d'IBM pour fidéliser ses clients et s'attache à suivre le grand principe de la compatibilité verticale ; toute nouvelle version du logiciel est censée accepter les applicatifs conçus avec les versions antérieures. Cela semble à première vue positif pour les utilisateurs mais, en un demi-siècle de développement, ceci a aussi des conséquences très néfastes sur la clarté, la cohérence, la souplesse du ou plutôt des langages spécifiques à ce logiciel. Son apprentissage est donc long, fastidieux si l'on veut en maîtriser les subtilités.

L'extrême lourdeur, la complexité, la stratégie de vouloir prendre en charge toutes le étapes de l'acquisition des données à leur valorisation par l'aide à la décision ont fait de ce logiciel un *dinosaure* à la location excessivement coûteuse qui survit très difficilement aux bouleversements technologiques et économiques qui accompagnent les déluges de données. Les derniers algorithmes d'apprentissage sont trop longs à implémentés de même que des interfaces efficaces avec les nouveaux systèmes de bases de données *not only SQL* ainsi que les nouvelles interfaces graphiques comme celle de *RStudio* ou *Jupyter Notebook*. La conséquence directe de cette incapacité d'adaptation suffisamment rapide est que SAS perd de vastes parts de marché depuis le début du siècle au profit de l ?implantation de logiciels sous licence ouverte (GNU) et d'entreprises qui développent des services (*plateform as a service*) dans ces environnements. Comme l'ont également compris Microsoft ou IBM, la vente de logiciels ou de matériels ne génèrent plus suffisamment de marge bénéficiaire comparativement à la vente de services.

Toutes ces raisons font que l'apprentissage de ce logiciel a été mis de coté au profit de celui de R et Python. Néanmoins en cas de nécessité, les tutoriels d'initiation à SAS sont toujours accessibles en auto-apprentissage car certains secteurs, notamment dans la santé : Agences Régionales de Santé, industrie pharmaceutique pour les essais cliniques, ou encore l'INSEE vont encore l'utiliser pendant de nombreuses années.

En voici une brève introduction.

Structure de SAS

Le système SAS est un ensemble de modules logiciels pour la gestion et le traitement statistique des données. A travers différents types d'interfaces utilisateur. Il permet d'écrire, ou de générer automatiquement (SAS Enterprise Guide), des *Programmes SAS* qui exécutent :

- les saisies, importations, interrogations, manipulations, fusions, transformations de données ;
- les éditions d'états, tableaux de bord, de rapports, numériques et graphiques ;
- les analyses statistiques, modélisation, prévision ;
- des applications spécifiques définies sous forme de macro-commandes pouvant être pilotées par menus ou à partir d'un navigateur ;
- les éditions plus ou moins automatisées de rapports et pages web.

Depuis la version 8, SAS propose des *solutions* : *analyse guidée des données*, *analyse marketing*, *Prévision de séries chronologiques*... qui sont autant d'environnements de travail associés à une interface graphique spécifique et à une problématique. Ils permettent un traitement de l'information sans écrire une ligne de programme. Le module *Enterprise Miner* sont très élaborés en ce sens. Il serait certes possible, en première approche, de se contenter de cette utilisation élémentaire mais l'usage montre que ces solutions sont nécessairement limitées et qu'un usage professionnel, associé à des contraintes spécifiques, rend incontournable l'usage d'une programmation basique, notamment pour la production de graphiques adaptés aux objectifs.

Après saisie ou importation en provenance de fichiers ASCII ou d'un SGBD (Système Relationnel de Gestion de Base de Données), les données sont gérées par SAS sous la forme d'un ensemble de fichiers binaires appelés *SAS Data Set* ou en français *Table SAS*.

Un *programme SAS* est un enchaînements d'*étapes* de gestion des données (*Data Step*) et d'appels de *procédures*, décrivant, dans une syntaxe souvent spécifique à un *module*, les traitements à réaliser sous le couvert d'*options*

prises par défaut ou explicitement définies. Les différentes étapes ou procédures communiquent entre elles exclusivement par l'intermédiaire de tables SAS, permanentes ou temporaires, et avec l'extérieur par des tables SAS ou des fichiers textes usuels en un format quelconque.

```
/* exemple de programme SAS */
/* Lecture, impression et tabulation de données. */
data Europe;
    infile "edc.fun.overseas";
    input date $ 1-7 dest $ 8-10 boarded 11-13;
proc print data = europe;
proc tabulate data = europe;
    class date dest;
    var boarded;
    table date, dest*boarded*sum;
run;
```

La diffusion de ce logiciel est organisée sous la forme de location annuelle de modules dont les principaux sont :

`Base SAS` est le module de référence pour tous les traitements de gestion des donnés au sein d'une étape *Data*, Ce module propose également des procédures élémentaires de description statistique (uni, bivariée, tableau de bord, tri...), un macro langage pour l'écriture de macro-commandes, des instructions de requêtage (SQL) dans des bases de données, des outils de production automatique de rapports (tt .docx, .odt) ou de pages web (html, XML).

`SAS/Stat` propose un vaste ensemble de procédures statistiques avec une grande variété d'options : tous les modèles de régression, les classifications, les durées de vie, la statistique non-paramétrique, les analyses multidimensionnelles...

`SAS/Graph` propose de grandes possibilités graphiques en haute résolution avec d'innombrables options.

`SAS/IML` est module de calcul matriciel avec un langage interprété, comme Matlab ou R, et qui offre récemment une interface graphique analogue à celle de RStudio.

2.3 R

Présentation

Le logiciel R sous licence GNU est facile à installer à partir de la page du CRAN ou d'un site miroir ; ils contiennent toutes les ressources nécessaires à l'utilisateur de R, débutant ou expérimenté : fichiers d'installation, mises à jour, librairies, FAQ, newsletter, documentation... Ce langage est le plus utilisé de la communauté statistique académique et aussi d'utilisation croissante dans les services R&D des entreprises industrielles en concurrence avec les logiciels commerciaux. Son utilisation nécessite un apprentissage à travers des tutoriels comme par exemple ceux de ce dépôt mais il est facile de démarrer à partir de quelques notions de base sur son utilisation, notions décrites dans *Start-R*.

Dans sa structure, R est un langage de programmation *interprété* avec une syntaxe voisine à celle du langage C et capable de manipuler des objets complexes sous forme de matrice, scalaire, vecteur, liste, facteur et aussi *data frame*. Proposant donc une *programmation matricielle*, il offre des fonctionnalités analogues à Matlab. Toute méthode statistique ou d'apprentissage est implémentée en R sous la forme d'une librairie (*package*) librement accessible. C'est même le mode de diffusion privilégié de nouvelles méthodes statistiques par la communauté scientifique.

Il existe de nombreuses librairies (cf. 'Rcmdr') d'interface graphique par menu mais celles-ci sont contraignantes, trop limitées dans les choix et options, elles ne peuvent éviter une utilisation par lignes de commandes ; autant s'y mettre tout de suite, c'est le choix fait ici. Il existe également un environnement de programmation ou IDE : `RStudio` relativement efficace ; à l'utilisateur de faire ses choix.

R est, comme Matlab, un langage interprété ; même en utilisant des librairies spécifiques pour paralléliser certains calculs compilés en C, les temps d'exécution de R deviennent vite rédhibitoires avec des données un peu volumineuses. De plus, son utilisation est rendue impossible (ou très difficile) dès que les limites de la mémoire interne de l'ordinateur sont atteintes.

2.4 Python

Le langage Python (Rossum et Guido ; 1995)[8] est développé et diffusé par la *Python Software Foundation* selon une licence *GPL-compatible*. À partir d'applications, initialement de calcul scientifique en image, signal, finance, son utilisation s'est généralisée dans de nombreux domaines et notamment pour l'analyse statistique de données pouvant être volumineuses. Il est donc "libre", efficace en calcul numérique (librairie 'NumPy'), orienté objet et propose de la programmation fonctionnelle... Il bénéficie d'une communauté très active développant de nombreuses applications et librairies.

La version 3.6. de Python est celle actuellement la plus récente. Le passage à la version 3 introduisit une rupture de compatibilité par rapport à la version 2 qui est toujours en développement (2.7). Il reste actuellement nécessaire de pouvoir utiliser les 2 versions selon les librairies utilisées et applications recherchées. La version 2.7 inclut des ajouts permettant des éléments de rétro-compatibilité avec la version 3. Pour l'usage rudimentaire des tutoriels fournis, il semble que les deux versions soient compatibles.

Ce langage et les principales librairies y faisant référence permettent d'envisager de façon efficace le prétraitement de données volumineuses. De façon plus précise, la librairie `pandas` offrent des outils efficaces, comme le découpage automatique en morceaux (`chunks`) adaptés à la taille de la mémoire vive ou encore l'accès à des données au format binaire HDF5 (librairie `Pytable`), pour lire (format `.csv` ou fixe), gérer, pré-traiter, trafiquer (en jargon : *data munging* ou *wrangling*), visualiser des données volumineuses. Et, lorsque celles-ci deviennent trop volumineuses pour la taille du disque et sont distribuées sur les nœuds d'un *cluster* sous *Hadoop* c'est encore le langage Python (API PySpark) qui permet de passer à l'échelle en utilisant la technologie Spark.

La librairie *Scikit-learn* (Pedregosa et al. 2011)[6] met à disposition les principales méthodes d'apprentissage supervisées ou non. Cette librairie n'est pas ouverte au sens où le choix d'implémentation d'une méthode est décidé au sein du groupe des principaux développeurs. L'avantage est un développement intégré et homogène, l'inconvénient, qui peut être aussi un avantage, est un choix plus restreint de méthodes accessibles. Également interprété, Python s'avère beaucoup plus rapide que R en gérant par défaut les possibilités de parallélisation d'une machine, même sous Windows.

2.5 R *vs.* Python

Le choix entre ces deux environnements repose sur les quelques points suivants :

- R et ses librairies offrent beaucoup plus de possibilités pour une exploration, des sélections et comparaisons de modèles, des *interprétations* statistiques détaillées avec des graphes complets produits par défaut.
- Mise en œuvre souvent implicite des possibilités de parallélisation, même sous Windows, par les librairies de Python.
- *Scikit-Learn* ne reconnaît pas (ou pas encore ?) la classe *DataFrame* développée dans la librairie *pandas*. Cette classe est largement utilisée en R pour gérer des types hétérogènes de variables. C'est un problème dans *Scikit-Learn* pour la prise en compte de variables qualitatives complexes. Une variable binaire est simplement remplacée par une indicatrice $(0, 1)$ mais, en présence de plusieurs modalités, une variable qualitative est remplacée par l'ensemble des indicatrices (*dummy* variables $(0, 1)$) de ses modalités. Ceci complique les stratégies de sélection de modèles et rend obscure leur interprétation.

En résumé, préférer R pour modéliser et interpréter des modèles statistiques mais préférer Python pour des modélisations efficaces à seule fin prédictive au détriment de l'interprétation. Les deux approches pouvant d'ailleurs être traitées de façon complémentaire.

Enfin, si les données sont trop volumineuses pour la mémoire interne voire pour le disque d'un ordinateur, ou encore si les données sont déjà archivées sur une architecture distribuée, d'autres approches sont à considérer et abordées en saison 4 avec Spark.

2.6 Reproductibilité des analyses

Donoho (2015)[3] insiste à juste titre sur la question importante de la reproductibilité des analyses. Les médias se font régulièrement l'écho de manquements déontologiques et plus généralement du problème récurrent du manque de reproductibilité des résultats publiés dans des journaux ou revues que ce soit par exemple en Biologie ou en Psychologie. Pour un statisticien, contribuer à la prise en compte de ces problèmes consiste à produire des chaînes de traitements ou d'analyses (*pipeline*) facilement transmissibles pour être reproductibles sur des matériels standards. Deux environnements s'y prêtent particu-

lièrement. Le premier concerne l'automatisation de la production d'un rapport en intégrant des commandes R (librairie *sweave* ou *knitr*) ou Python (*pweave*) au sein d'un source LaTeX. Ces commandes, automatiquement exécutées, provoquent l'insertion de tableaux ou graphiques. Le deuxième, plus en amont, consiste à enregistrer systématiquement l'enchaînement des commandes et de leurs résultats numériques ou graphique dans un calepin (*Jupyter notebook*). La sauvegarde est faite sous un format ré-exécutable dans un environnement similaire ou sous forme de fichier au format `html, pdf`. Ce type de résultat est obtenu en exécutant le bon noyau (Python, R, Julia...) dans le même environnement *Jupyter* à partir d'un simple navigateur. C'est pour cette raison que tous les tutoriels sont exécutables sous la forme d'un calepin, notamment pour les

- Tutoriels d'initiation à R.
- Tutoriels d'initiation à Python.

À exécuter et approfondir parallèlement à la maîtrise des principales méthodes.

3 Méthodes de la Science des Données

3.1 Méthodes traitées

L'historique précédent illustre schématiquement une progression pédagogique car il est difficile d'analyser de grands ensembles de données sans maîtriser les outils de base développés pour des données plus modestes à condition de bien identifier et faire coïncider les objectifs d'une étude : exploratoire, explicatif ou prédictif, avec ceux des méthodes mis en œuvre. C'est aussi une progression méthodologique, des outils les plus simples aux plus sophistiqués, pour aborder un nouvel ensemble de données.

Cette présentation propose donc de découper schématiquement la progression de la formation d'un *data scientist*, du L3 au M2, en quatre étapes ou *saisons* regroupant chacune un ensemble de scénarios ou épisodes couplant présentation théoriques et tutoriels pratiques des différentes méthodes et donc compétences à acquérir.

Saison 1 (L3) Statistique élémentaire, descriptive *vs.* inférentielle.

Saison 2 (M1) Statistique Exploratoire multidimensionnelle et apprentissage non supervisé.

Saison 3 Apprentissage Statistique / Machine supervisé.

Saison 4 (M2) Technologies pour la Science des (grosses) Données.

N.B. Cette formation s'appuie sur des compétences parallèlement acquises en Statistique mathématique, calcul des Probabilités, Optimisation, Analyse Fonctionnelle pour une compréhension approfondie des méthodes et algorithmes utilisées, de leurs limites, et en Informatique pour leur mise en exploitation.

3.2 Méthodes auxquelles vous avez échappé

Certains points n'ont pas été intégrés à ce déroulement notamment en lien avec le **V** de *variété* ou celui de *vélocité*. Il faut se rendre à l'évidence qu'il n'est pas possible de former à bac+5 un mouton à 7 pattes supposé maîtriser toute la "science des données". Il a fallu faire des choix laissant de côté certains points :

- Méthodes d'apprentissage machine mais pas d'apprentissage statistique comme celles issues du domaine de la logique formelle. La recherche de règles d'associations (problème du panier de la ménagère) en est une. Elle consiste à identifier les co-occurences les plus fréquentes ou significatives par un ensemble de règles logiques associant variables et valeurs de celles-ci. Elle n'est pas adaptée à des volumétries importantes.
- Traitement de données structurées (variété) : l'analyse d'images ou signaux, nécessitant la projection des données sur des bases fonctionnelles adaptées (Fourier, ondelettes, splines) est abordée dans différents cas d'usage. En revanche, l'analyse de graphes (*e.g.* réseaux sociaux), arbres, trajectoires n'est pas abordée ; elle nécessite l ?utilisation d'algorithmes adaptés faisant appel à des notions de distance ou de noyau spécifiques.
- *Apprentissage par renforcement* et traitement de flux de données (vélocité). L'apprentissage se fait en ligne, voire en temps réel, et sans stockage par des algorithmes d'optimisation stochastique (systèmes dynamiques, contrôle stochastique...) pour produire des décisions séquentielles. Exemple : algorithme de bandit pour la recommandation de produits déterminée par le comportements (les clics) d'un internaute.

Références

[1] Y. Benjamini et Y. Hochberg, *Controlling the false discovery rate : a prac-*

tical and powerful approach to multiple testing, Journal of the Royal Statistical Society (1995), n° 85, 289–300.

[2] F. Caillez et J.M. Pages, *Introduction à l'Analyse des Données*, SMASH, 1976.

[3] David Donoho, *50 years of Data Science*, Princeton NJ, Tukey Centennial Workshop, 2015.

[4] U. M. Fayyad, *Editorial*, Data mining and Knowledge discovery **1** (1997), 5–10.

[5] T. Hastie, R. Tibshirani et J Friedman, *The elements of statistical learning : data mining, inference, and prediction*, Springer, 2009, Second edition.

[6] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot et E. Duchesnay, *Scikit-learn : Machine Learning in Python*, Journal of Machine Learning Research **12** (2011), 2825–2830.

[7] R Core Team, *R : A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2016, http://www.R-project.org.

[8] Guido Rossum, *Python Reference Manual*, Rap. tech. CS-R9525, CWI (Centre for Mathematics and Computer Science), Amsterdam, The Netherlands, The Netherlands, 1995.

[9] John W. Tukey, *Exploratory data analysis*, 1977.