Mettre les technologies cloud au service de la production statistique

Romain Avouac
Insee
romain.avouac@insee.fr

Thomas Faria
Insee
thomas.faria@insee.fr

Frédéric Comte
Insee
frederic.comte@insee.fr

2025-01-17

Résumé French

1 Introduction

L'exploitation de sources de données non traditionnelles afin d'améliorer le processus de production statistique est une orientation majeure du Système Statistique Européen (SSE). Cette évolution vers un modèle de *Trusted Smart Statistics* [1] s'accompagne d'innovations dans les processus statistiques, permettant de tirer parti du potentiel de ces sources — plus grande disponibilité, résolution spatio-temporelle accrue, etc. — tout en faisant face à leur complexité et à leurs limites. Parmi ces innovations figurent les méthodes d'apprentissage automatique et leurs applications prometteuses dans les domaines du codage et de la classification, des redressements et de l'imputation [2]. Les multiples défis auxquels font face les instituts statistiques dans ce contexte d'évolution sont abordés dans le *Mémorandum de Bucarest sur les statistiques publiques dans une société numérisée*, qui anticipe que « la variété des nouvelles sources de données, paradigmes computationnels et outils nécessitera des adaptations de l'architecture métier statistique, des processus, des modèles de production, des infrastructures informatiques, des cadres méthodologiques et de qualité, ainsi que des structures de gouvernance correspondantes », et invite en conséquence le SSE à évaluer les adaptations requises et à les prioriser [3]. Cette évolution est également largement visible dans le cadre du service statistique public (SSP), dont elle constitue l'une des lignes directrices de la stratégie à horizon 2025 [4].

Dans l'optique de ces transformations, de nombreux travaux ont été menés dans le cadre de projets successifs à l'échelle européenne pour opérationnaliser l'utilisation de sources de données non-traditionnelles dans la production de statistiques officielles. Dans le cadre du projet ESSnet Big Data II (2018-2020), les instituts statistiques nationaux (INS) ont travaillé sur une large gamme de thématiques (offres d'emploi en ligne, transactions financières, traces GPS, etc.) afin de constituer les briques nécessaires pour intégrer ces sources dans les processus de production et identifier leurs limites [5]. En France, les travaux sur l'exploitation des données mobiles [6] ou des données

de caisse [7] ont permis d'illustrer le potentiel de ces sources pour construire de nouveaux indicateurs ou raffiner des indicateurs existants. Néanmoins, si un travail considérable a été consacré au développement de cadres méthodologiques [8], [9], de lignes directrices sur la qualité [10], ainsi qu'à la conception de processus sécurisant l'acquisition de données auprès de tiers [11], les infrastructures informatiques et les compétences nécessaires pour gérer ces nouveaux objets sont restées peu abordées dans la littérature.

Pourtant, ces nouvelles sources présentent des caractéristiques qui rendent leur traitement informatique complexe. On qualifie souvent de *big data* ces données qui se distinguent par leur volume (souvent de l'ordre de plusieurs centaines de Go voire du To), leur vélocité (vitesse de génération, souvent proche du temps réel) ou de leur variété (données structurées mais aussi non structurées, telles que les textes et les images). Or les « compétences pour automatiser, analyser et optimiser ces systèmes complexes ne font souvent pas partie des compétences traditionnelles de la plupart des instituts statistiques nationaux » [12]. Au cours des dernières années, on observe un nombre croissant de statisticiens publics formés aux méthodes de *data science*, permettant d'envisager l'intégration de ces sources dans des processus de production. Dans ses multiples acceptions, le terme « *data scientist* » reflète en effet l'implication croissante des statisticiens dans le développement informatique et l'orchestration de leurs opérations de traitement des données, au-delà des seules phases de conception ou de validation [13]. Toutefois, il est clair en pratique, à l'Insee et dans d'autres organisations, que la capacité de ces profils à tirer parti des sources *big data* et des méthodes d'apprentissage automatique se heurte à plusieurs défis.

Un premier défi réside dans l'absence d'infrastructures informatiques adaptées aux nouvelles sources de données auxquelles les INS ont désormais accès, ainsi qu'au besoin croissant de nouvelles méthodes statistiques. Par exemple, les sources *big data* nécessitent d'énormes capacités de stockage et s'appuient souvent sur des infrastructures et des méthodes de calcul distribué pour être traitées [14]. De même, l'adoption de nouvelles méthodes statistiques basées sur des algorithmes d'apprentissage automatique requiert des capacités informatiques — en particulier des GPU (processeurs graphiques) dans le cadre du traitement du texte ou de l'image — pour paralléliser massivement les calculs [15]. De telles ressources sont rarement disponibles dans les infrastructures informatiques traditionnelles. Lorsque des infrastructures de calcul adaptées sont disponibles, comme les supercalculateurs (HPC) utilisés dans certains domaines de recherche, elles nécessitent des compétences spécifiques — notamment pour leur mise en place et leur maintenance — qui sont rarement disponibles au sein des INS.

Un autre défi majeur pour les statisticiens est de disposer d'environnements de développement leur permettant d'expérimenter plus librement. L'essence de l'innovation dans les travaux statistiques réside dans la capacité à intégrer rapidement de nouveaux outils et méthodologies. Cette agilité est limitée lorsque les statisticiens dépendent excessivement des départements informatiques pour provisionner des ressources ou installer de nouveaux logiciels. Dans les configurations traditionnelles — ordinateurs personnels ou bureaux virtuels sur des architectures centralisées[1] —

[1] `AUSv3` est un exemple d'une telle infrastructure. Le statisticien utilise son poste de travail comme point d'accès à un bureau virtuel qui « reproduit » l'expérience habituelle du poste de travail. Néanmoins, les calculs qui sont

les départements informatiques privilégient généralement la sécurité et la stabilité du système au détriment de la fourniture de nouveaux services, ce qui limite le potentiel d'innovation. De plus, ces environnements rigides rendent difficile la mise en œuvre de bonnes pratiques de développement, telles que le travail collaboratif — nécessitant des environnements permettant de partager facilement des expérimentations avec ses pairs — et la reproductibilité.

Un troisième défi concerne la difficulté de passer des expérimentations innovantes à des solutions en production. Même lorsque les statisticiens ont accès à des environnements leur permettant d'expérimenter aisément, la transition vers le déploiement d'une application ou d'un modèle reste généralement difficile. Les environnements de production diffèrent souvent des environnements de développement, ce qui entraîne des coûts de développement supplémentaires importants pour passer d'une preuve de concept à une solution industrialisée qui rend du service dans la durée. Par exemple, dans le cas des projets d'apprentissage automatique, les modèles déployés nécessitent un suivi rigoureux pour s'assurer qu'ils conservent leur précision et leur utilité au fil du temps, et requièrent généralement des améliorations périodiques ou continues. Ces besoins plaident pour des environnements plus flexibles permettant aux statisticiens de gérer de manière autonome le cycle de vie complet de leurs projets de *data science*.

Ces différents défis ont un thème sous-jacent commun : le besoin d'une plus grande autonomie. La capacité des méthodes de *data science* à améliorer et potentiellement transformer la production des statistiques officielles dépend crucialement de la capacité des statisticiens à mener des expérimentations innovantes plus librement. Pour ce faire, ils doivent avoir accès à des ressources informatiques substantielles et diversifiées leur permettant de gérer le volume et la diversité des sources *big data* et d'exploiter les méthodes d'apprentissage automatique. Ces projets expérimentaux nécessitent à leur tour des environnements de développement flexibles favorisant le travail collaboratif pour tirer parti de la diversité des profils et compétences des équipes de projet. Enfin, pour tirer pleinement parti de ces expérimentations, les statisticiens ont besoin d'outils pour déployer des applications sous forme de preuves de concept et orchestrer leurs opérations statistiques en toute autonomie.

Dans ce contexte, l'Insee a développé `Onyxia` : un projet open source permettant aux organisations de déployer des plateformes de *data science* favorisant l'innovation en offrant aux statisticiens une plus grande autonomie[2]. Cet article vise à décrire le processus de réflexion ayant conduit à ce projet et à illustrer comment il autonomise les statisticiens à l'Insee, devenant ainsi un pilier de notre stratégie d'innovation. La section 2 offre une analyse approfondie des derniers développements de l'écosystème de la donnée, mettant en lumière les choix technologiques qui ont façonné le développement d'un environnement moderne de *data science*, adapté aux besoins spécifiques des statisticiens. En particulier, nous montrons comment les technologies *cloud* — en particulier la conteneurisation et le stockage objet — sont essentielles pour créer des environnements évolutifs et flexibles qui favorisent l'autonomie tout en promouvant la reproductibilité des projets statistiques. Toutefois, malgré leurs atouts pour les applications modernes de *data science*, la complexité

lancés — via `R` ou `Python` par exemple — sont effectués sur des machines virtuelles (VM) de calcul dédiées, et non sur le poste de travail.

[2]https://github.com/InseeFrLab/onyxia

de configuration et d'utilisation des technologies *cloud* est souvent un obstacle à leur adoption. Dans la section 3, nous détaillons le projet `Onyxia` qui vise précisément à rendre les technologies *cloud* accessibles aux statisticiens grâce à une interface conviviale et un catalogue étendu d'environnements de *data science* prêts à l'emploi. Enfin, à travers l'étude de cas de la classification des activités des entreprises françaises (APE), la section 4 illustre comment l'utilisation de ces technologies a considérablement facilité la mise en production de modèles d'apprentissage automatique à l'Insee en permettant d'appliquer les meilleures pratiques issues du *MLOps*.

2 Principes pour construire une architecture de données moderne et flexible pour les statistiques publiques

Avec l'émergence de sources de données massives et de nouvelles méthodologies prometteuses pour améliorer le processus de production des statistiques publiques, les statisticiens formés aux techniques de *datascience* sont désireux d'innover. Cependant, leur capacité à le faire est limitée par plusieurs défis. L'un des principaux défis réside dans le besoin d'une plus grande autonomie — qu'il s'agisse de dimensionner la puissance de calcul en fonction des chaînes de productions statistiques, de déployer des preuves de concept avec agilité et de manière collaborative, etc. Dans ce contexte, notre objectif était de concevoir une plateforme de *datascience* qui non seulement gère efficacement les données massives, mais qui renforce également l'autonomie des statisticiens. Pour y parvenir, nous avons étudié l'évolution de l'écosystème des données afin d'identifier les tendances significatives susceptibles de surmonter les limitations mentionnées précédemment[3]. Nos conclusions indiquent que l'adoption des technologies *cloud*, en particulier les conteneurs et le stockage objet, est essentielle pour construire des infrastructures capables de gérer des ensembles de données volumineux et variés de manière flexible et économique. De plus, ces technologies améliorent considérablement l'autonomie, facilitant ainsi l'innovation et favorisant la reproductibilité dans la production des statistiques publiques.

2.1 Limites des architectures traditionnelles pour les big data

Au cours de la dernière décennie, le paysage des *big data* s'est transformé de manière spectaculaire. Suite à la publication des articles fondateurs de Google introduisant le *MapReduce* [16]–[18], les systèmes basés sur Hadoop sont rapidement devenus l'architecture de référence dans l'écosystème des données massives, salués pour leur capacité à gérer d'importants ensembles de données grâce aux calculs distribués. L'introduction d'Hadoop a marqué une étape révolutionnaire, permettant aux organisations de traiter et d'analyser des données à une échelle sans précédent. En substance, Hadoop offrait aux entreprises des capacités complètes pour l'analyse de *big data* : des outils pour la collecte, le stockage des données (HDFS) et des capacités de calcul (notamment Spark) [19], expliquant ainsi son adoption rapide dans les industries.

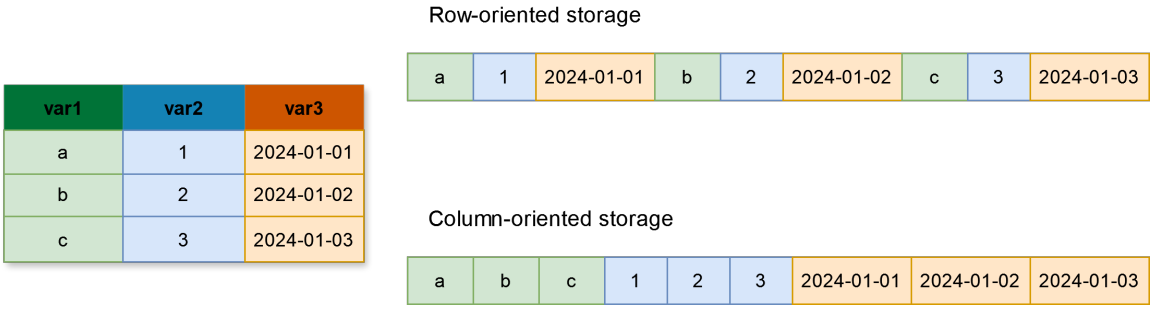
[3]En préambule à cette analyse, il convient de noter que, bien que nous ayons fait de notre mieux pour ancrer nos réflexions dans la littérature académique, une grande partie de nos observations provient de connaissances informelles acquises grâce à une veille technologique assidue et continue. Dans l'écosystème des données, qui est en constante évolution, les articles de recherche traditionnels cèdent de plus en plus la place aux billets de blog en tant que principales références pour les développements de pointe. Ce changement s'explique en grande partie par le rythme soutenu auquel les technologies et méthodologies liées aux données massives progressent, rendant souvent le processus de publication formelle trop long pour la diffusion de connaissances et d'innovations.

À la fin des années 2010, les architectures basées sur Hadoop ont connu un net déclin de popularité. Dans les environnements Hadoop traditionnels, le stockage et le calcul étaient co-localisés par construction : si les données sources étaient réparties sur plusieurs serveurs (scalabilité horizontale), chaque section des données était directement traitée sur la machine hébergeant cette section, afin d'éviter les transferts réseau entre serveurs. Dans ce paradigme, la mise à l'échelle de l'architecture impliquait souvent une augmentation linéaire à la fois des capacités de calcul et de stockage, indépendamment de la demande réelle. Dans un article volontairement provocateur et intitulé « Big Data is Dead » [20], Jordan Tigani, l'un des ingénieurs fondateurs de Google BigQuery, explique pourquoi ce modèle ne correspond plus à la réalité de la plupart des organisations centrées sur les données. Premièrement, parce que « dans la pratique, la taille des données augmente beaucoup plus rapidement que les besoins en calcul ». Alors que la quantité de données générées et nécessitant donc d'être stockées peut croître de manière linéaire au fil du temps, il est généralement vrai que nous n'aillons besoin d'interroger que les portions les plus récentes, ou seulement certaines colonnes et/ou groupes de lignes. Par ailleurs, Tigani souligne que « la frontière du *big data* ne cesse de reculer » : les avancées dans les capacités des serveurs et la baisse des coûts du matériel signifient que le nombre de charges de travail ne tenant pas sur une seule machine — une définition simple mais efficace du *big data* — a diminué de manière continue. En conséquence, en séparant correctement les fonctions de stockage et de calcul, même les traitements de données substantiels peuvent finir par utiliser « beaucoup moins de calcul que prévu […] et pourraient même ne pas avoir besoin d'un traitement distribué du tout ».

Ces observations concordent fortement avec nos propres constats à l'Insee au cours des dernières années. Par exemple, une équipe de l'Insee a mis en place un cluster Hadoop en tant qu'architecture alternative à celle déjà utilisée pour traiter les données des tickets de caisse dans le cadre du calcul de l'indice des prix à la consommation. Une accélération des opérations de traitement des données pouvant aller jusqu'à un facteur 10 a été obtenue, pour des opérations qui prenaient auparavant plusieurs heures [21]. Malgré cette amélioration des performances, ce type d'architectures n'a pas été réutilisé par la suite pour d'autres projets, principalement parce que l'architecture s'est révélée coûteuse et complexe à maintenir, nécessitant une expertise technique spécialisée rarement disponible au sein des Instituts Nationaux de Statistiques (INS) [22]. Bien que ces nouveaux projets puissent encore impliquer des volumes de données massifs, nous avons observé que des traitements efficaces pouvaient être réalisés à l'aide de logiciels conventionnels (R, Python) sur des systèmes à nœud unique, en tirant parti des récentes innovations importantes de l'écosystème des données. Tout d'abord, en utilisant des formats de stockage efficaces tels qu'Apache Parquet [23], dont les propriétés — stockage en colonnes [24] (voir Figure 1), optimisation pour les analyses « écrire une fois, lire plusieurs fois », possibilité de partitionner les données, etc. — le rendent particulièrement adapté aux tâches analytiques comme celles généralement effectuées dans les statistiques publiques [17]. Ensuite, en effectuant des calculs optimisés en mémoire tels qu'Apache Arrow [25] ou DuckDB [26] le proposent. Également basés sur une représentation en colonnes — travaillant ainsi en synergie avec les fichiers Parquet — ces deux logiciels améliorent considérablement les performances des requêtes de données grâce à l'utilisation de l'"évaluation paresseuse" (*lazy evaluation*) : au lieu d'exécuter de nombreuses opérations distinctes (par exemple, sélectionner des colonnes et/ou filtrer des lignes, puis calculer de nouvelles colonnes,

puis effectuer des agrégations, etc.), ils les traitent toutes en une fois de manière plus optimisée. En conséquence, les calculs se limitent aux données effectivement nécessaires pour les requêtes, permettant le traitement de données beaucoup plus importantes que la mémoire disponible sur des machines classiques à nœud unique.

Figure 1. – Représentation orientée ligne et orientée colonne d'un même jeu de données.



Note: De nombreuses opérations statistiques sont analytiques (OLAP) par nature : elles impliquent la sélection de colonnes spécifiques, le calcul de nouvelles variables, la réalisation d'agrégations basées sur des groupes, etc. Le stockage orienté ligne n'est pas bien adapté à ces opérations analytiques, car il nécessite de charger l'ensemble du jeu de données en mémoire afin d'effectuer une requête. À l'inverse, le stockage orienté colonne permet de ne lire que les colonnes de données pertinentes, ce qui réduit considérablement les temps de lecture et de traitement pour ces charges de travail analytiques. En pratique, les formats colonnes populaires tels que Parquet utilisent une représentation hybride : ils sont principalement orientés colonne, mais intègrent également un regroupement astucieux basé sur les lignes pour optimiser les requêtes de filtrage.

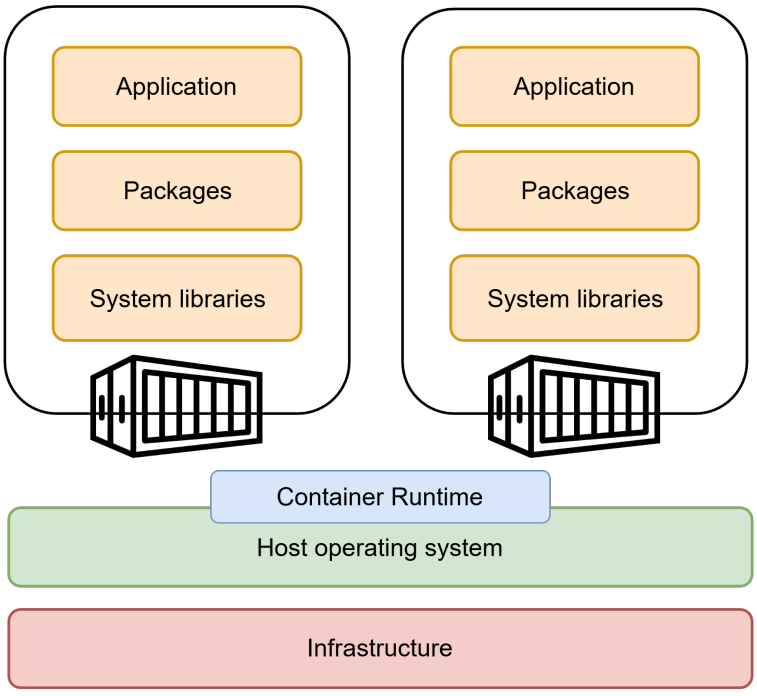
2.2 Adopter les technologies cloud

Suite à cette évolution de l'écosystème des big data, on observe un virage notable ces dernières années dans l'industrie vers des architectures plus flexibles et faiblement couplées. L'avènement des technologies *cloud* a joué un rôle déterminant dans cette transition. Contrairement à l'époque où Hadoop dominait, la latence réseau est devenue une préoccupation bien moindre, rendant le modèle traditionnel de solutions de stockage et de calcul sur site et co-localisées moins pertinent. Concernant la nature des données à traiter, on observe une évolution que certains ont qualifiée de passage « du big data aux données flexibles ». Les infrastructures modernes doivent non seulement être capables de traiter de grands volumes, mais aussi être adaptables sur de multiples dimensions. Elles doivent pouvoir prendre en charge diverses structures de données (allant des formats structurés et tabulaires aux formats non structurés comme le texte et les images), assurer la portabilité des données dans des environnements *multi-cloud* et *cloud* hybride, et prendre en charge une large gamme de calculs computationnels (des calculs parallèles aux modèles d'apprentissage profond nécessitant des GPU, ainsi que le déploiement et la gestion d'applications) [27]. Ces dernières années, deux technologies ont émergé comme des éléments fondamentaux pour atteindre cette flexibilité dans les environnements *cloud* : la conteneurisation et le stockage d'objets.

Dans un environnement *cloud*, l'ordinateur de l'utilisateur devient un simple point d'accès pour effectuer des calculs sur une infrastructure centrale. Cela permet à la fois un accès ubiquitaire et

une scalabilité des services, car il est plus facile de mettre à l'échelle une infrastructure centrale — généralement de manière horizontale, c'est-à-dire en ajoutant davantage de serveurs. Cependant, ces infrastructures centralisées présentent deux limites bien identifiées qui doivent être prises en compte : la concurrence entre utilisateurs pour l'accès aux ressources physiques et la nécessité d'isoler correctement les applications déployées. Le choix de la conteneurisation est fondamental, car il répond à ces deux enjeux [28]. En créant des « bulles » spécifiques à chaque service, les conteneurs garantissent l'isolement des applications tout en restant légers, puisqu'ils partagent le système d'exploitation de support avec la machine hôte (voir Figure 2). Pour gérer plusieurs applications conteneurisées de manière systématique, les infrastructures conteneurisées s'appuient généralement sur un logiciel orchestrateur — le plus connu étant Kubernetes, un projet open source initialement développé par Google pour gérer ses nombreuses charges de travail conteneurisées en production [29]. Les orchestrateurs automatisent le processus de déploiement, de mise à l'échelle et de gestion des applications conteneurisées, coordonnant leur exécution sur différents serveurs. De manière intéressante, cette propriété permet de traiter de très grands volumes de données de manière distribuée : les conteneurs décomposent les opérations de traitement des données massives en une multitude de petites tâches, organisées par l'orchestrateur. Cela minimise les ressources requises tout en offrant une flexibilité supérieure aux architectures basées sur Hadoop [30].

Figure 2. – Architecture d'un environnement conteneurisé.



Note: Un conteneur est un regroupement logique de ressources permettant d'encapsuler une application (par exemple, du code R), les bibliothèques utilisées (par exemple, ggplot, dplyr) et les bibliothèques système (l'interpréteur R, d'autres bibliothèques dépendantes du système d'exploitation, etc.) dans un seul package. Les applications conteneurisées sont isolées les unes des autres

grâce à la virtualisation, ce qui permet d'attribuer des ressources physiques spécifiques à chaque application tout en garantissant leur indépendance totale. Contrairement aux machines virtuelles, qui virtualisent également le système d'exploitation (OS), les conteneurs s'appuient sur une forme de virtualisation légère : le conteneur partage l'OS de l'infrastructure hôte via le runtime de conteneur (par exemple, Docker). En conséquence, les conteneurs sont beaucoup plus portables et peuvent être déployés et redistribués facilement.

L'autre choix fondamental dans une architecture de données concerne la nature du stockage de ces données. Dans l'écosystème *cloud*, le « stockage d'objets » est devenu la référence *de facto* [31][4]. Dans ce paradigme, les fichiers sont stockés sous forme d'"objets" composés de données, d'un identifiant et de métadonnées. Ce type de stockage est optimisé pour la scalabilité, car les objets ne sont pas limités en taille et la technologie sous-jacente permet un stockage rentable de fichiers (potentiellement très) volumineux. Le stockage d'objets joue également un rôle clé dans la construction d'une infrastructure découplée comme celle évoquée précédemment : les dépôts de données — appelés *« buckets »* — sont directement interrogeables via des requêtes HTTP standards grâce à une API REST normalisée. Dans un contexte où la latence réseau n'est plus le principal goulot d'étranglement, cela signifie que le stockage et le calcul n'ont pas besoin d'être sur les mêmes machines, ni même dans le même lieu. Ils peuvent ainsi évoluer indépendamment en fonction des besoins spécifiques de l'organisation. Enfin, le stockage d'objets est un complément naturel aux architectures basées sur des environnements conteneurisés. Il fournit une couche de persistance — les conteneurs étant par construction sans état (*stateless*) — et une connectivité facile, sans compromettre la sécurité, voire en renforçant celle-ci par rapport à un système de stockage traditionnel [32].

2.3 Exploiter les technologies cloud pour accroître l'autonomie et favoriser la reproductibilité

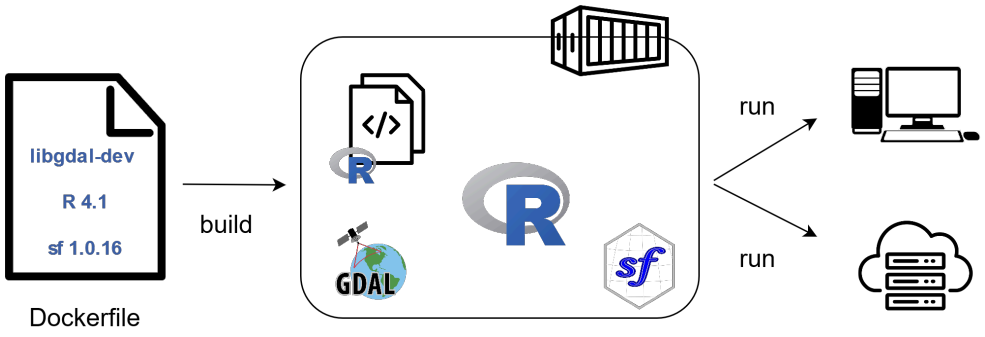
Comprendre comment les choix technologiques décrits dans la discussion technique ci-dessus sont pertinents dans le contexte des statistiques publiques nécessite un examen approfondi des pratiques professionnelles des statisticiens dans leur utilisation des environnements informatiques. À la fin des années 2000, alors que la micro-informatique était à son apogée, une grande partie des ressources techniques utilisées par les statisticiens de l'Insee étaient locales : le code et les logiciels de traitement étaient situés sur des ordinateurs personnels, tandis que les données étaient accessibles via un système de partage de fichiers. En raison de la scalabilité limitée des ordinateurs personnels, cette configuration restreignait considérablement la capacité des statisticiens à expérimenter avec des sources *big data* ou des méthodes statistiques intensives en calculs, et cela impliquait des risques de sécurité liés à la diffusion étendue des données au sein de l'organisation. Pour surmonter ces limitations, une transition a été opérée vers des infrastructures informatiques centralisées, regroupant toutes les ressources — et donc globalement beaucoup plus — sur des serveurs centraux. Ces infrastructures, mises à disposition des statisticiens via un environnement de bureau virtuel partagé pour faciliter leur utilisation, constituent encore la méthode dominante pour réaliser des calculs statistiques à l'Insee au moment de la rédaction de ces lignes.

[4] Principalement grâce à l'implémentation « S3 » (Simple Storage Service) d'Amazon.

À travers nos observations et nos discussions avec d'autres statisticiens, il est devenu évident que, bien que l'infrastructure informatique actuelle soutienne adéquatement les activités fondamentales de production statistique, elle restreint de manière notable la capacité des statisticiens à expérimenter librement et à innover. Le principal goulot d'étranglement dans cette organisation réside dans la dépendance des projets statistiques à la prise de décision centralisée en matière d'informatique, notamment en ce qui concerne l'allocation des ressources de calcul, l'accès au stockage partagé, l'utilisation de langages de programmation préconfigurés etc. En outre, ces dépendances conduisent souvent à un phénomène bien connu dans la communauté du développement logiciel, où les priorités des développeurs — itérer rapidement pour améliorer continuellement les fonctionnalités — entrent souvent en conflit avec l'objectif des équipes informatiques de garantir la sécurité et la stabilité des processus. À l'inverse, nous comprenons que les pratiques modernes en *datascience* reflètent une implication accrue des statisticiens dans le développement et l'orchestration informatique de leurs opérations de traitement de données, au-delà de la simple phase de conception ou de validation. Les nouvelles infrastructures de *datascience* doivent donc prendre en compte ce rôle élargi de leurs utilisateurs, en leur offrant plus d'autonomie que les infrastructures traditionnelles.

Nous soutenons que les technologies *cloud* sont une solution puissante pour offrir aux statisticiens une autonomie bien plus grande dans leur travail quotidien, favorisant ainsi une culture de l'innovation. Grâce au stockage d'objets, les utilisateurs obtiennent un contrôle direct sur la couche de stockage, leur permettant d'expérimenter avec des sources de données diverses sans être limités par les espaces de stockage souvent restreints et alloués par les départements informatiques. La conteneurisation permet aux utilisateurs de personnaliser leurs environnements de travail selon leurs besoins spécifiques — qu'il s'agisse de langages de programmation, de bibliothèques système ou de versions de packages — tout en leur offrant la flexibilité nécessaire pour adapter leurs applications à la puissance de calcul et aux capacités de stockage requises. Par construction, les conteneurs favorisent également le développement d'applications portables, permettant des transitions plus fluides entre les environnements (développement, test, pré-production, production), en garantissant que les applications peuvent être exécutées sans difficulté, évitant ainsi les problèmes liés aux incohérences d'environnement. Enfin, avec des outils d'orchestration tels que Kubernetes, les statisticiens peuvent déployer plus facilement des applications et des API, tout en automatisant l'ensemble du processus de construction. Cette capacité s'aligne avec l'approche DevOps, qui préconise la création de preuves de concept de manière itérative, plutôt que de chercher à développer la solution optimale (mais chronophage) pour un objectif préalablement défini [33].

Figure 3. – Par construction, les conteneurs favorisent la reproductibilité et la portabilité.



Note: Dans un environnement conteneurisé, les applications sont créées à partir de spécifications sous forme de scripts — un paradigme connu sous le nom d'*"infrastructure as code"*. Dans un fichier texte, conventionnellement nommé « Dockerfile », les *data scientists* peuvent spécifier l'environnement de travail de leur application : le code de l'application, les logiciels à inclure (par exemple, R), les packages utilisés pour leurs opérations de traitement (par exemple, le package R pour le calcul géospatial *sf*), ainsi que les bibliothèques système dépendant de l'OS appelées par ces packages (par exemple, GDAL, la bibliothèque qui permet de lire et de traiter les formats d'images géospatiales utilisée par la plupart des packages traitant des données géospatiales). Un point essentiel est que les versions des logiciels et des packages utilisés pour développer l'application peuvent être précisément spécifiées, ce qui garantit la reproductibilité des calculs effectués. Une étape de construction génère ensuite une image associée au Dockerfile, c'est-à-dire une forme empaquetée et compressée de l'environnement de travail de l'application. Les images créées de cette manière sont portables : elles peuvent être facilement distribuées — généralement via un registre de conteneurs — et exécutées de manière reproductible sur n'importe quelle infrastructure disposant d'un runtime de conteneur.

Outre la scalabilité et l'autonomie, ces choix architecturaux favorisent également la reproductibilité des calculs statistiques. Le concept de reproductibilité — à savoir la capacité de reproduire le résultat d'une expérience en appliquant la même méthodologie aux mêmes données — est un critère fondamental de validité scientifique [34]. Il est également très pertinent dans le domaine des statistiques publiques, car il constitue une base pour la transparence, essentielle pour établir et maintenir la confiance du public [35]. Favoriser la reproductibilité dans la production statistique implique de concevoir des solutions de traitement capables de produire des statistiques reproductibles, tout en étant partageables entre pairs [36]. Les infrastructures informatiques traditionnelles — qu'il s'agisse d'un ordinateur personnel ou d'une infrastructure partagée avec un accès à distance — sont insuffisantes à cet égard. Construire un projet ou calculer un simple indicateur statistique dans ces environnements implique généralement une série d'étapes manuelles (installation des bibliothèques système, des binaires du langage de programmation, des packages du projet, gestion des versions conflictuelles, etc.) qui ne peuvent pas être pleinement reproduites d'un projet à l'autre. En comparaison, les conteneurs sont reproductibles par définition, car leur processus de construction implique de définir précisément toutes les ressources nécessaires comme un ensemble d'opérations standardisées, allant de la « machine nue » à l'application en cours d'exécu-

tion [37]. De plus, ces environnements reproductibles peuvent être facilement partagés avec des pairs, car ils peuvent être publiés sur des registres ouverts (par exemple, un registre de conteneurs comme DockerHub) avec le code source de l'application (par exemple, sur une forge logicielle publique comme GitHub ou GitLab). Cette approche améliore considérablement la réutilisation des projets de code, favorisant un modèle de développement et d'innovation basé sur la collaboration communautaire.

3 Onyxia : un projet open source pour construire des plateformes de datascience sur des technologies cloud

Cette section examine comment Onyxia, un projet open source initié par l'Insee, démocratise l'accès aux technologies *cloud* pour les statisticiens en fournissant des environnements modernes de *datascience* favorisant l'autonomie. Nous analysons comment cette initiative s'inscrit dans l'objectif général de création des « connaissances communes » en promouvant et en développant des logiciels facilement réutilisables dans le domaine des statistiques publiques et ailleurs.

3.1 Rendre les technologies cloud accessibles aux statisticiens

Notre veille technologique et notre revue de la littérature ont mis en évidence les technologies *cloud*, en particulier la conteneurisation et le stockage d'objets, comme des éléments clés pour construire une plateforme de *datascience* à la fois scalable et flexible. En nous appuyant sur ces enseignements, nous avons mis en place notre premier cluster Kubernetes dans les locaux de l'Insee en 2020, en l'intégrant avec MinIO, un système de stockage d'objets *open source* conçu pour fonctionner de manière fluide avec Kubernetes. Cependant, nos premières expérimentations ont révélé un obstacle majeur à l'adoption généralisée des technologies *cloud* : la complexité de leur intégration. C'est une considération importante lorsqu'il s'agit de construire des architectures de données qui privilégient la modularité — une caractéristique essentielle pour atteindre la flexibilité que nous visons[5]. Toutefois, la modularité des composants architecturaux implique également que toute application de données lancée sur le cluster doit être configurée pour communiquer avec tous les autres composants. Par exemple, dans un environnement *big data*, la configuration de Spark pour fonctionner sur Kubernetes tout en interagissant avec des ensembles de données stockés dans MinIO nécessite de nombreuses et complexes configurations (définition des points d'entrée, des jetons d'accès, etc.), une compétence qui dépasse généralement l'expertise des statisticiens.

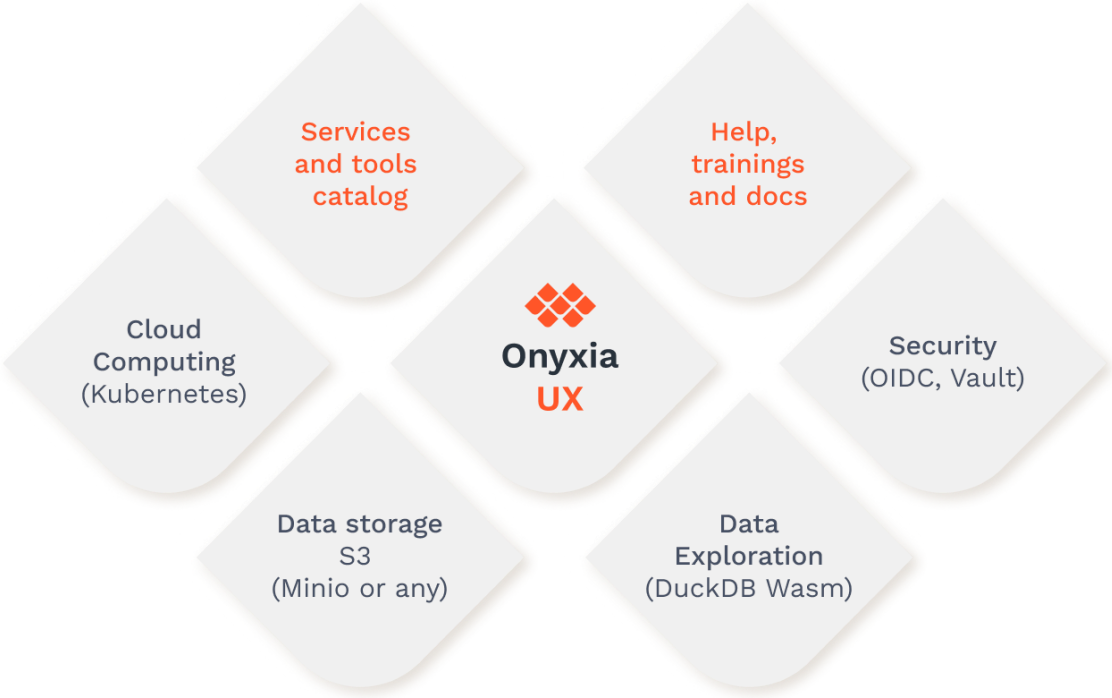
For instance, due to MinIO's compatibility with the Amazon S3 API, the storage source could easily be switched to one managed by another public cloud provider, without requiring substantial modifications.

This insight is really the base of the Onyxia project: choosing technologies that foster autonomy will not actually foster autonomy if their complexity acts as a barrier from widespread adoption

[5] A telling example of the importance of building a modular architecture is the ability to switch between storage sources (on-premise, public cloud provider, etc.). The storage solution we chose, MinIO, is compatible with Amazon's S3 API, which has become a de facto standard in the cloud ecosystem due to the success of Amazon's AWS S3 storage solution. As a result, organizations that choose to use Onyxia are not tied to a specific storage solution: they can choose any solution that complies with the standards defined by the S3 API.

in the organization. In recent years, statisticians at Insee already needed to adapt to a changing environment in terms of their everyday tools: transitioning from proprietary software (SAS®) to open-source ones (R, Python), acculturating to technologies that improve reproducibility (version control with Git), consuming and developing APIs, etc. These changes, making their job more and more akin to the one of software developers, already imply significant training and changes in daily work practices. Against this background, adoption of cloud-technologies was utterly dependent on making them readily accessible.

Figure 4. – Onyxia is the technical binder between cloud-native modular components



To bridge this gap, we developed Onyxia, an application that essentially acts as interface between the modular components that compose the architecture (see Figure 4). The main entry point of the user is a user-friendly web application[6] that enables users to launch services from a data science catalog (see Chapitre 3.3) as running containers on the underlying Kubernetes cluster. The interface between the UI and Kubernetes is done by a lightweight custom API[7], that essentially transforms the application request of the user into a set of manifests to deploy Kubernetes resources. For a given application, these resources are packaged under the form of Helm charts, a popular way of packaging potentially complex applications on Kubernetes [38]. Although users can configure a service to tailor it to their needs, they will most of the time just launch an out-of-the-box service with default settings and start developing straight away. This point really illustrates the added value of Onyxia in facilitating the adoption of cloud technologies. By injecting authentication information and configuration into the containers at the initialization, we ensure

[6]https://github.com/InseeFrLab/onyxia-ui
[7]https://github.com/InseeFrLab/onyxia-api

that users can launch and manage data science services in which they can interact seamlessly with the data from their bucket on MinIO, their sensitive information (tokens, passwords) in a secret management tool such as Vault, etc. This automatic injection, coupled with the pre-configuration of data science environments in Onyxia's catalogs of images[8] and associated helm-charts[9], make it possible for users to execute potentially complex workloads — such as running distributed computations with Spark on Kubernetes using data stored in S3, or training deep-learning models using a GPU — without getting bogged down by the technicalities of configuration.

3.2 Architectural choices aimed at fostering autonomy

The Onyxia project is based on a few structuring principles, with a central theme: fostering autonomy, both at the organizational and individual levels. First, at the level of the organization by preventing vendor lock-in. In order to get a competitive edge, many commercial cloud providers develop applications and protocols that customers need to use to access cloud resources, but that are not interoperable, greatly complexifying potential migrations to another cloud platform [39]. Recognizing these challenges, there is a trend towards endorsing cloud-neutral strategies [40] in order to reduce reliance on a single vendor's specific solutions. In contrast, the use of Onyxia is inherently not restrictive: when an organization chooses to use it, it chooses the underlying technologies — containerization and object storage — but not the solution. The platform can be deployed on any Kubernetes cluster, either on-premise or in public clouds. Similarly, Onyxia was designed to be used with MinIO because it is an open-source object-storage solution, but is also compatible with objects storage solutions from various cloud providers (AWS, GCP).

Onyxia also fosters autonomy at the level of users. Proprietary softwares that have been used intensively in official statistics — such as SAS or STATA — also produce a vendor lock-in phenomenon. The costs of licensing are high and can evolve quickly, and users are tied in certain ways of performing computations, preventing progressive upskilling. On the contrary, Onyxia aspires to be removable; we want to enhance users' familiarity and comfort with the underlying cloud technologies rather than act as a permanent fixture in their workflow. An illustrative example of this philosophy is the platform's approach to user actions: for tasks performed through the UI, such as launching a service or managing data, we provide users with the equivalent terminal commands, promoting a deeper understanding of what actually happens on the infrastructure when triggering something. Furthermore, all the services offered through Onyxia's catalog are open-source.

[8] https://github.com/InseeFrLab/images-datascience

[9] https://github.com/InseeFrLab/helm-charts-interactive-services

Figure 5. – Launching a service through Onyxia's UI.

Note: Services from Onyxia's catalog can either be used vanilla or configured by the users to tailor them to their specific needs. In order to limit the dependence of users on Onyxia, each action performed by the user on the UI is accompanied by the actual command that is executed on the Kubernetes cluster.

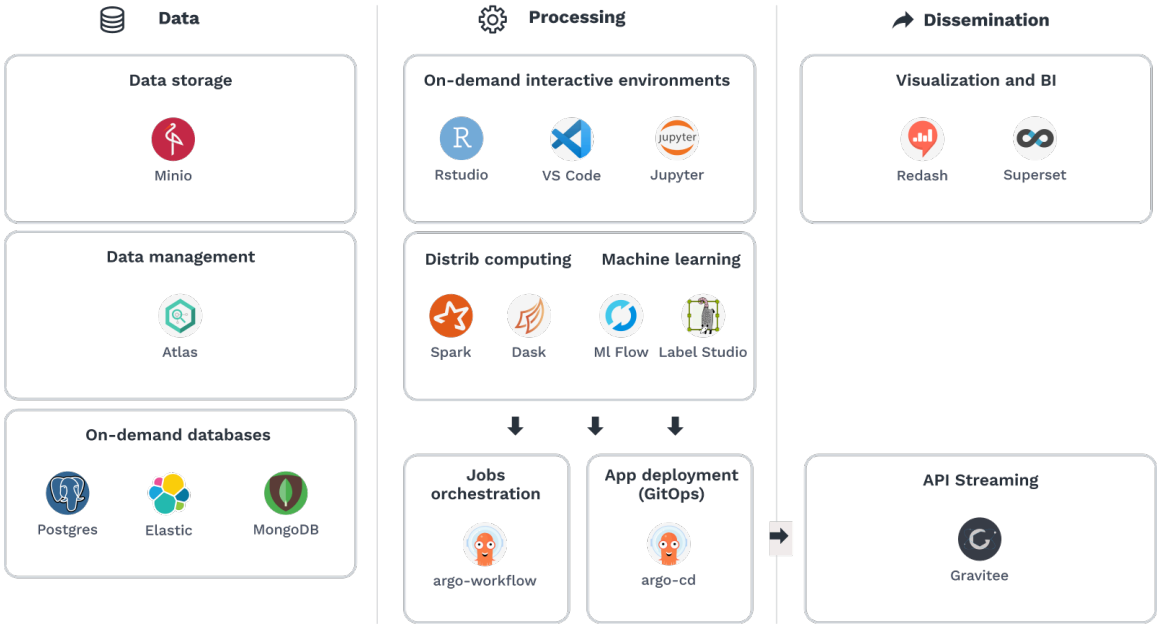
Naturally, the way Onyxia makes statisticians more autonomous in their work depends on their needs and familiarity with IT skills. Statisticians that just want to have access to extensive computational resources to experiment with new data sources or statistical methods will have access in a few clicks to easy-to-use, pre-configured data science environments, so that they can directly start to experiment. However, many users want to go deeper and build actual prototypes of production applications for their projects: configuring initialization scripts to tailor the environments to their needs, deploying an interactive app that delivers data visualization to users of their choice, deploying other services than those available in our catalogs, etc. For these advanced users to continue to push the boundaries of innovation, Onyxia gives them access to the underlying Kubernetes cluster. This means that users can freely open a terminal on an interactive service and interacts with the cluster - within the boundaries of their namespace - in order to apply custom resources and deploy custom applications or services.

Besides autonomy and scalability, the architectural choices of Onyxia also foster reproducibility of statistical computations. In the paradigm of containers, the user must learn to deal with resources which are by nature ephemeral, since they only exist at the time of their actual mobilization. This fosters the adoption of development best practices, notably the separation of the code — put on an internal or open-source forge such as GitLab or GitHub — the data — stored on a specific storage solution, such as MinIO — and the computing environment. While this requires an entry cost for users, it also helps them to conceive their projects as pipelines, i.e. a series of sequential steps with well-defined inputs and outputs (akin to directed acyclic graph (DAG)). The projects developed in that manner are usually more reproducible and portable — they can work seamlessly on different computing environments — and thus also more readily shareable with peers.

3.3 An extensive catalogue of services to cover the entire lifecycle of data science projects

In developing the Onyxia platform, our intention was to provide statisticians with a comprehensive environment designed to support end-to-end development of data science projects. As depicted in Figure 6, the platform offers a vast array of services that span the complete lifecycle of a data science project.

Figure 6. – Onyxia's catalog aims at covering the entire lifecycle of data science projects



The primary usage of the platform is the deployment of interactive development environments (IDE), such as RStudio, Jupyter, or VSCode. These IDEs come equipped with the latest kernels of major open-source programming languages commonly employed by public statisticians (R, Python, Julia), as well as an extensive collection of packages commonly used in data science for each language. In order to ensure that services remain up-to-date and consistent between them, we maintain our own stack of underlying Docker images and rebuild it weekly. The stack of images is fully open-source[10] and can thus be reused outside Onyxia.

As discussed in previous sections, the persistence layer of these interactive environments is mainly carried out by MinIO, Onyxia's default object storage solution. As it is based on a standardized REST API, files can be easily queried directly from R or Python using high-level packages. This in itself is an important step of ensuring reproducibility: the input files of a project are not mounted manually and then specified via paths adherent to a specific infrastructure and filesystem. Rather, files are specified as HTTP queries, making the overall structure of projects much more extendable. In our experience, the object-storage paradigm covers very well the needs of most statistical projects we accompany. However, additional database services such as PostgreSQL and

[10]https://github.com/InseeFrLab/images-datascience

MongoDB are available for applications with specific needs, such as those requiring online transaction processing (OLTP) capabilities or document-oriented storage.

As Onyxia was developed to allow experimentation with big data sources and machine learning methods, we also provide services optimized for scalability. For instance, frameworks like Spark and Trino that enable to perform distributed computations within Kubernetes. These services come pre-configured to integrate seamlessly with S3 storage, thus facilitating building integrated and efficient data pipelines.

Beyond mere experimentation, our goal is to empower statisticians to transition from trial phases to production-grade projects. In lines with principles from the DevOps approach, this involves facilitating the deployment of prototypes and their continuous improvement over time. To this end, we provide a set of open-source tools aimed at automatizing and industrializing the process of deploying data-intensive applications (ArgoCD, Argo-Workflows, MLflow). For projects leveraging machine-learning models, statisticians can serve their models through APIs, deploy them using the aforementioned tools, and manage their lifecycle using an API manager (e.g. Gravitee). Section 4 will illustrate how these tools, particularly MLflow, have been central in putting machine learning models in production at Insee, in accordance with MLOps principles.

In Chapitre 3.2, we stressed that one of Onyxia's fundamental design principle was to avoid vendor lock-in. In line with this idea, organizations that implement Onyxia are free to customize catalogs to suit their specific requirements, or even opt to construct their own catalogs independent of Onyxia's default offerings. This flexibility ensures that organizations are not confined to a single solution or provider, and can adapt the platform to their evolving needs.

3.4 Building commons: an open-source project and an open-innovation platform

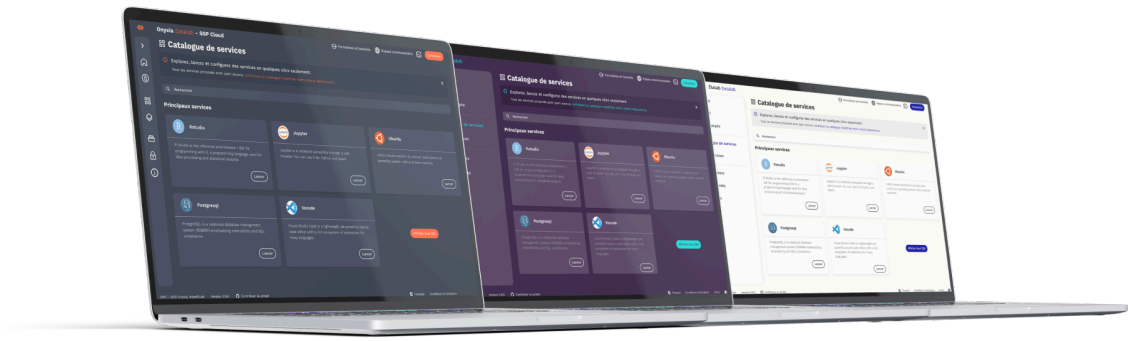
As a fully open-source initiative, the Onyxia project aims at building « knowledge commons » by promoting and building software that can be easily reused in official statistics and beyond [41]. This concerns, first of all, the components on which Onyxia are based: both its constitutive technological bricks (Kubernetes, MinIO, Vault) as well as all the services from the catalog are open-source. But more crucially, all the code of the project is available openly on GitHub[11]. Alongside an in-depth documentation[12], this greatly facilitates the potential for other organizations to create instances of data science platforms built upon the Onyxia software and tailor it to their respective needs (see Figure 7). This enabled the project to attract a growing community of contributors from official statistics (Statistics Norway), NGOs (Mercator Ocean), research centres and even industry, thus transitioning progressively towards a more decentralized governance of the project. In the next years, the involvement of NSIs from the European Statistical System is expected to increase as Onyxia was chosen as the reference data science platform in the context of the AIML4OS project, a « One-Stop-Shop » for Artificial Intelligence/Machine Learning for Official Statistics in the European Statistical System[13].

[11] https://github.com/InseeFrLab/onyxia

[12] https://docs.onyxia.sh/

[13] More information on this project available at https://cros.ec.europa.eu/dashboard/aiml4os

Figure 7. – One project, multiple instances: the UI is adaptable to the graphic identity of the organization



Another major way in which we try to build commons is by developing and maintaining a showcase instance of the Onyxia project, the SSP Cloud [42]. This platform, equipped with extensive and scalable computational resources[14], is designed to be a sandbox for experimenting with cloud technologies and new data science methods. The full catalog of services of Onyxia is available on the platform, enabling motivated users to go beyond mere experimentation by producing « proof of concepts », with full autonomy regarding the configuration and orchestration of their services.

Beyond its technical capabilities, the SSP Cloud is an endeavour at embodying the principles of open-innovation [43]. Deployed on internet[15], it is open not only to Insee employees, but also more broadly to French governmental agencies, French Universities and other European NSIs, and is dedicated to experimenting with data science methods using open data. Thus, the projects carried out on this platform showcase the growing abundance of datasets published openly by organizations. The fundamentally collaborative nature of the SSP Cloud has proven especially beneficial for organizing innovative events such as hackathons — both at the national and international levels — and in the academic sphere. It has become an integral resource for several universities and Grandes Ecoles in France, fostering the use of cloud-native and reproducible environments, and preventing vendor lock-in effect due to the over-reliance of educational organizations on proprietary cloud solutions. As a result, the platform is now widely used in the French National Statistical System and beyond, with about 800 unique users per month in 2024. These users form a dynamic community thanks to a centralized discussion canal; they help improve the user experience by reporting bugs, suggesting new features, and thus contribute directly to the project.

4 Cas d'usage

5 Discussion

Bibliographie

[14] On the physical side, the SSP Cloud consists in a Kubernetes cluster of about 20 servers, for a total capacity of 10 TB of RAM, 1100 CPUs, 34 GPUs and 150 TB of storage.

[15] https://datalab.sspcloud.fr/

[1] F. Ricciato, A. Wirthmann, K. Giannakouris, M. Skaliotis, et others, « Trusted smart statistics: Motivations and principles », *Statistical Journal of the IAOS*, vol. 35, nº 4, p. 589-603, 2019.

[2] T. Gjaltema, « High-Level Group for the Modernisation of Official Statistics (HLG-MOS) of the United Nations Economic Commission for Europe », *Statistical Journal of the IAOS*, vol. 38, nº 3, p. 917-922, 2022.

[3] DGINS, « Bucharest Memorandum on Official Statistics in a Datafied Society ». 2018.

[4] INSEE, « Horizon 2025 ». 2016.

[5] EUROSTAT, « ESSnet Big Data 2 - Final Technical Report ». 2021.

[6] B. Sakarovitch, M.-P. d. Bellefon, P. Givord, et M. Vanhoof, « Estimating the residential population from mobile phone data, an initial exploration », *Economie et Statistique*, vol. 505, nº 1, p. 109-132, 2018.

[7] M. Leclair, I. Léonard, G. Rateau, P. Sillard, G. Varlet, et P. Vernédal, « Scanner data: advances in methodology and new challenges for computing consumer price indices », *Economie et Statistique*, vol. 509, nº 1, p. 13-29, 2019.

[8] P. Descy, V. Kvetan, A. Wirthmann, et F. Reis, « Towards a shared infrastructure for online job advertisement data », *Statistical Journal of the IAOS*, vol. 35, nº 4, p. 669-675, 2019.

[9] D. Salgado, L. Sanguiao-Sande, S. Barragán, B. Oancea, et M. Suarez-Castillo, « A proposed production framework with mobile network data », in *ESSnet Big Data II - Workpackage I - Mobile Network Data*, 2020.

[10] A. Kowarik et M. Six, « Quality Guidelines for the Acquisition and Usage of Big Data with additional Insights on Web Data », in *4th International Conference on Advanced Research Methods and Analytics (CARMA 2022)*, 2022, p. 269-270.

[11] F. Ricciato, F. De Meersman, A. Wirthmann, G. Seynaeve, et M. Skaliotis, « Processing of mobile network operator data for official statistics: the case for public-private partnerships », in *104th DGINS conference*, 2018.

[12] A. Ashofteh et J. M. Bravo, « Data science training for official statistics: A new scientific paradigm of information and knowledge development in national statistical systems », *Statistical Journal of the IAOS*, vol. 37, nº 3, p. 771-789, 2021.

[13] T. H. Davenport et D. Patil, « Data scientist », *Harvard business review*, vol. 90, nº 5, p. 70-76, 2012.

[14] L. Liu, « Computing infrastructure for big data processing », *Frontiers of Computer Science*, vol. 7, p. 165-170, 2013.

[15] A. Saiyeda et M. A. Mir, « Cloud computing for deep learning analytics: A survey of current trends and challenges. », *International Journal of Advanced Research in Computer Science*, vol. 8, nº 2, 2017.

[16] S. Ghemawat, H. Gobioff, et S.-T. Leung, « The Google file system », in *Proceedings of the nineteenth ACM symposium on Operating systems principles*, 2003, p. 29-43.

[17] A. I. Abdelaziz, K. A. Hanson, C. E. Gaber, et T. A. Lee, « Optimizing large real-world data analysis with parquet files in R: A step-by-step tutorial », *Pharmacoepidemiology and Drug Safety*, 2023.

[18] J. Dean et S. Ghemawat, « MapReduce: simplified data processing on large clusters », *Communications of the ACM*, vol. 51, nº 1, p. 107-113, 2008.

[19] B. Dhyani et A. Barthwal, « Big data analytics using Hadoop », *International Journal of Computer Applications*, vol. 108, nº 12, p. 975-8887, 2014.

[20] J. Tigani, « Big data is dead ». [En ligne]. Disponible sur: https://motherduck.com/blog/big-data-is-dead/

[21] M. Leclair et others, « Using Scanner Data to Calculate the Consumer Price Index », *Courrier des statistiques*, vol. 3, p. 61-75, 2019.

[22] S. Vale, « International collaboration to understand the relevance of Big Data for official statistics », *Statistical Journal of the IAOS*, vol. 31, nº 2, p. 159-163, 2015.

[23] A. S. Foundation, « Apache Parquet ». 2013.

[24] D. Abadi, P. Boncz, S. Harizopoulos, S. Idreos, S. Madden, et others, « The design and implementation of modern column-oriented database systems », *Foundations and Trends® in Databases*, vol. 5, nº 3, p. 197-280, 2013.

[25] A. S. Foundation, « Apache Arrow ». 2016.

[26] M. Raasveldt et H. Mühleisen, « Duckdb: an embeddable analytical database », in *Proceedings of the 2019 International Conference on Management of Data*, 2019, p. 1981-1984.

[27] Y. Li *et al.*, « Big data and cloud computing », *Manual of digital earth*, p. 325-355, 2020.

[28] O. Bentaleb, A. S. Belloum, A. Sebaa, et A. El-Maouhab, « Containerization technologies: Taxonomies, applications and challenges », *The Journal of Supercomputing*, vol. 78, nº 1, p. 1144-1181, 2022.

[29] R. Vaño, I. Lacalle, P. Sowiński, R. S-Julián, et C. E. Palau, « Cloud-native workload orchestration at the edge: A deployment review and future directions », *Sensors*, vol. 23, nº 4, p. 2215-2216, 2023.

[30] Q. Zhang, L. Liu, C. Pu, Q. Dou, L. Wu, et W. Zhou, « A comparative study of containers and virtual machines in big data environment », in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, 2018, p. 178-185.

[31] S. Samundiswary et N. M. Dongre, « Object storage architecture in cloud for unstructured data », in *2017 International Conference on Inventive Systems and Control (ICISC)*, 2017, p. 1-6.

[32] M. Mesnier, G. R. Ganger, et E. Riedel, « Object-based storage », *IEEE Communications Magazine*, vol. 41, nº 8, p. 84-90, 2003.

[33] L. Leite, C. Rocha, F. Kon, D. Milojicic, et P. Meirelles, « A survey of DevOps concepts and challenges », *ACM Computing Surveys (CSUR)*, vol. 52, nº 6, p. 1-35, 2019.

[34] M. McNutt, « Reproducibility », *Science*, vol. 343, nº 6168, p. 229-230, 2014.

[35] European Commission, « European Statistics Code of Practice — revised edition 2017 ».

[36] S. Luhmann, J. Grazzini, F. Ricciato, M. Mészáros, J.-M. Museux, et M. Hahn, « Promoting reproducibility-by-design in statistical offices », in *2019 New Techniques and Technologies for Statistics (NTTS) conference*, 2019, p. . doi: 10.5281/zenodo.3240198.

[37] D. Moreau, K. Wiebels, et C. Boettiger, « Containers for computational reproducibility », *Nature Reviews Methods Primers*, vol. 3, nº 1, p. 50-51, 2023.

[38] S. Gokhale *et al.*, « Creating helm charts to ease deployment of enterprise application and its related services in kubernetes », in *2021 international conference on computing, communication and green engineering (CCGE)*, 2021, p. 1-5.

[39] J. Opara-Martins, R. Sahandi, et F. Tian, « Critical analysis of vendor lock-in and its impact on cloud computing migration: a business perspective », *Journal of Cloud Computing*, vol. 5, p. 1-18, 2016.

[40] J. Opara-Martins, M. Sahandi, et F. Tian, « A holistic decision framework to avoid vendor lock-in for cloud saas migration », *Computer and Information Science*, vol. 10, nº 3, 2017.

[41] C. M. Schweik, « Free/open-source software as a framework for establishing commons in science », 2006.

[42] F. Comte, A. Degorre, et R. Lesur, « SSPCloud: a creative factory to support experimentations in the field of official statistics », *Courrier des Statistiques, INSEE*, vol. 7, p. 68-85, 2022.

[43] H. W. Chesbrough, *Open innovation: The new imperative for creating and profiting from technology*. Harvard Business Press, 2003.