Chapitre 1-4
Installer son environnement de travail

1. Introduction

Il ne s'agit ici que de CPython, l'implémentation de référence de Python, et non de PyPy ou Jython.
Quel que soit votre système d'exploitation, vous pouvez installer Python en lisant ce chapitre puis, dans un second temps, installer des bibliothèques tierces au gré de vos besoins (cf. section Installer une bibliothèque tierce) et vous pourrez créer des environnements virtuels (cf. section Créer un environnement virtuel).
Si vous souhaitez installer d'un seul coup Python ainsi que Jupyter (anciennement IPython) et la plupart des bibliothèques scientifiques ou d'analyse de données, vous pouvez aller directement à la section Installer Anaconda, pour installer celui-ci en lieu et place de Python. Vous disposerez alors d'autres méthodes pour gérer les environnements virtuels et pour installer des bibliothèques tierces.

2. Installer Python

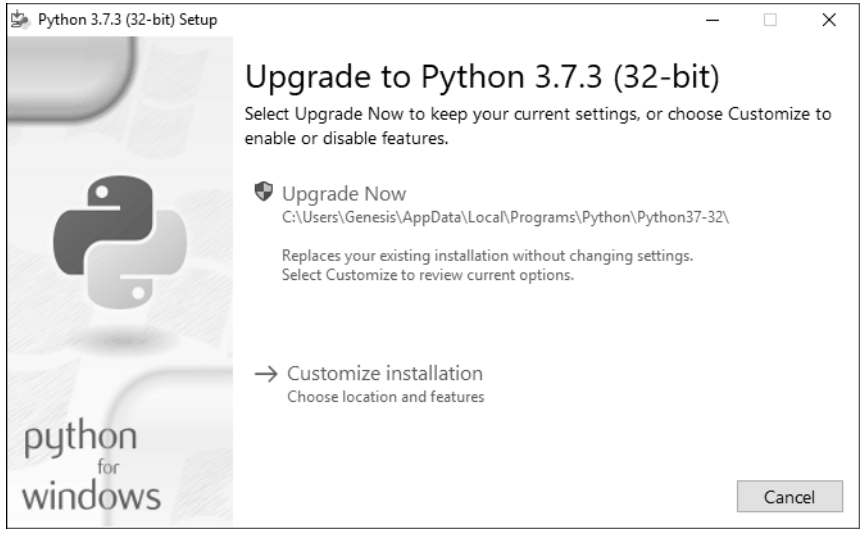
2.1 Pour Windows

Le système d'exploitation Windows requiert usuellement l'utilisation d'un installateur pour pouvoir installer un logiciel quel qu'il soit. Si vous disposez de Windows, vous devriez en avoir l'habitude. Python ne déroge pas à la règle.
Pour installer Python, vous devez donc aller sur le site officiel (https://www.python.org/downloads/) pour télécharger l'installateur adéquat. Comme vous pourrez le constater, on vous met en avant un accès rapide à la dernière version (au moment où ces lignes sont écrites, la 3.11.0), puis un accès aux dernières versions encore actives (actuellement la version 3.10 qui reçoit encore des corrections d'anomalies, puis les versions 3.9 à 3.7 qui reçoivent des corrections de sécurité uniquement).

Il est également possible de télécharger la toute dernière version de la branche 2.7 qui est en fin de vie (elle n'est plus mise à jour), car il existe encore de nombreux projets n'ayant pas encore migré.
Le support correctif dure 2 ans après la première sortie de la version et le support de sécurité dure 5 ans.
Pour notre part, nous vous conseillons la dernière 3.x, mais vous êtes libre d'installer celle que vous souhaitez ou même d'en installer plusieurs suivant vos contraintes, il n'y a pas d'objection à cela.
Une fois le téléchargement effectué, vous devez lancer l'installateur (et éventuellement passer quelques protections de votre système qui vous demande d'accorder votre confiance à cet installateur), pour observer l'écran suivant :

Comme vous pouvez le constater, il est possible de personnaliser l'installation en choisissant le chemin d'installation du logiciel ou en choisissant de ne pas sélectionner quelques fonctionnalités, mais nous ne le conseillons pas.
Nous vous recommandons en revanche de cocher la case **Add python.exe to PATH** afin de configurer la variable PATH du terminal pour rendre Python accessible plus facilement.

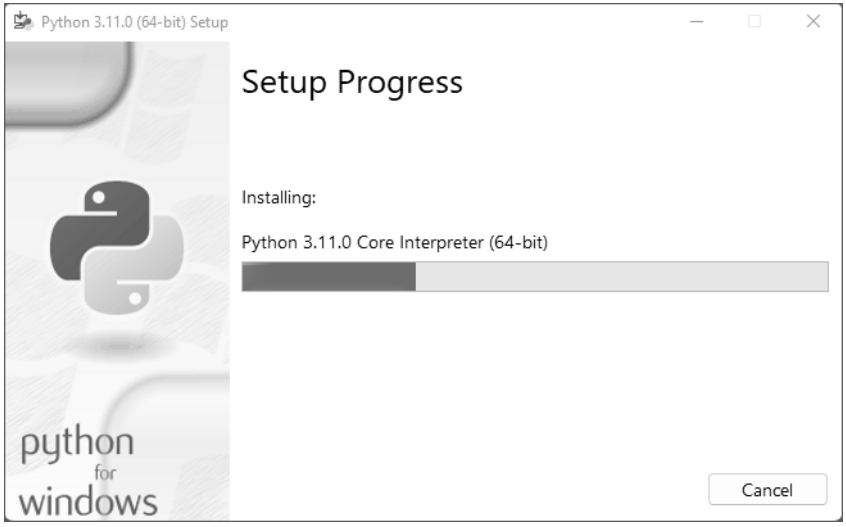
Si vous avez déjà une ancienne version de Python installée de la même branche (dans cet exemple, Python 3.7.2 est déjà installé), vous pourrez la mettre à jour à l'aide du même installeur :



Par contre, si vous avez déjà la version 3.7.1 et que vous installez la version 3.11, cette dernière ne viendra pas remplacer la précédente, mais s'installera à côté. Si vous souhaitez remplacer, il vous faudra donc désinstaller proprement la toute dernière version installée, ce qu'il est possible de faire en relançant l'installeur d'origine.

Nous vous encourageons à garder les installeurs sur votre PC, car ils pourraient devenir indisponibles au téléchargement si trop vieux.

Quel que soit le scénario, vous arriverez devant un écran vous montrant la progression de l'installation et vous n'aurez qu'à fermer la fenêtre une fois celle-ci terminée :



Vous êtes maintenant prêt à utiliser Python.

2.2 Pour Mac

Il faut savoir qu'une version de Python est déjà préinstallée sur Mac, car Mac OS X l'utilise pour ses propres besoins et Python est intégré à son propre cycle de développement. Cependant, si vous souhaitez une version différente de celle qui est déjà présente, vous pouvez l'installer, sachant qu' il n'y a pas de contre-indication à posséder plusieurs versions de Python sur la même machine.

Pour installer Python sur Mac OS X, la procédure à suivre est similaire à celle pour Windows. Il faut donc se rendre sur le site officiel (https://www.python.org/downloads/mac-osx/), télécharger un installateur correspondant à sa configuration et suivre les étapes.

Pour les utilisateurs de Mac, sachez que Python dispose d'une bonne intégration de ses spécificités, en particulier vis-à-vis de Objective-C, le langage de programmation avec lequel est développé Mac OS X, et Cocoa, interface de programmation de Mac OS X.

2.3 Pour GNU/Linux et BSD

Les différentes distributions libres utilisent nativement Python, notamment pour des parties sensibles. Python y est donc tout naturellement déjà installé, généralement sous la dernière version de la branche 2.x. Cependant, ici comme ailleurs, il n'y a pas d'objections à utiliser plusieurs versions de Python.
Le plus simple reste d'utiliser votre gestionnaire de paquets, ce qui peut se faire via un outil graphique, comme Synaptic pour Debian :

Il suffit alors de faire une recherche sur le mot-clé **`python`** pour voir les différentes versions (sur une ancienne Debian, ce sont Python 2.6, 2.7 et 3.2).
Par contre, tous les paquets python3-xxxxx que vous pouvez voir ici sont des bibliothèques tierces et non Python lui-même. Nous en parlerons plus tard dans ce chapitre.
Une fois les paquets souhaités sélectionnés, il ne manque plus qu'à les installer en cliquant sur le bouton **Appliquer**.
Notez que tout ceci peut se faire par la simple ligne de commande, toujours en utilisant votre gestionnaire de paquets qui peut être apt-get, aptitude, yum, emerge, pkg_add ou autre.
Voici par exemple pour une distribution Debian ou Ubuntu :

```
$ sudo aptitude install python3
```

Ceci ne permet cependant pas de choisir la version que l'on souhaite, à moins d'aller trouver des sources alternatives. Si l'on veut avoir la toute dernière version de Python, il faudra la plupart du temps passer par la compilation.

2.4 Par la compilation

Compiler Python n'est pas en soi une tâche très complexe. C'est par contre souvent une tâche imposée lorsque l'on ne travaille pas avec des conteneurs. En effet, en entreprise, on développe souvent des applications qui sont destinées à être hébergées. Il est alors impératif de travailler sur votre propre poste avec une version de Python qui soit la même que celle existante sur la machine de production.

Sous GNU/Linux, mais aussi sous d'autres systèmes, il est possible de compiler la version de Python que l'on souhaite. Après tout, Python n'est rien d'autre qu'un programme écrit en C. Pour ce faire, il faut aller télécharger le code source (https://www.python.org/downloads/source/), qui prend la forme d'une archive, puis décompresser celle-ci, se placer dans le répertoire ainsi obtenu et taper ces quelques commandes :

```
$ ./configure –prefix=/path/to/my/python/directory
$ make
$ sudo make altinstall
```

Notez que dans cette dernière ligne, nous n'utilisons pas la commande **`make install`**, qui aurait pour effet de remplacer votre Python système par le Python que vous compilez, ce qui pourrait avoir des conséquences indésirables voire désastreuses.

Notez également que vous choisissez lors de la configuration le chemin dans lequel vous placerez vos bibliothèques Python. En général, l'usage veut que l'on utilise **`/opt`**, mais il n'y a pas de règle, tout dépend des pratiques de votre entreprise ou votre expérience en la matière.

Si vous venez d'installer Python 3.5 par cette méthode, vous aurez alors maintenant accès à ce programme en l'appelant ainsi, depuis votre terminal :

```
$ python3.5
```

Par cette même méthode, vous pouvez installer les dernières versions (https://www.python.org/download/pre-releases/) de Python qui ne sont pas encore sorties (alphas ou betas), ce qui vous permet de les tester en avant-première !

Notons que, par cette méthode, toutes les bibliothèques de Python ne fonctionneront pas. En effet, lorsqu'elles ont besoin d'autres bibliothèques C, il faut effectuer des compilations croisées et utiliser les différents en-têtes de ces bibliothèques. C'est le cas par exemple pour faire fonctionner Curses, ReportLab (génération de fichiers PDF) ou encore PyUSB (accès aux ports matériels USB).

Dans ce cas-là, la commande **`./configure`** devra recevoir des arguments supplémentaires et vous devrez trouver un tutoriel en ligne pour vous indiquer la démarche, laquelle peut être plus ou moins complexe.

2.5 Pour un smartphone

Installer une machine virtuelle Python sur un smartphone est possible. Pour Android, la procédure est assez simple puisqu'il existe un produit dédié (http://qpython.com/), tout comme sur Windows Phone (https://apps.microsoft.com/store/detail/python-39/9P7QFQMJRFP7). Pour iOS, c'est une autre paire de manches (https://github.com/linusyang/python-for-ios) étant donné que l'utilisateur est enfermé dans un système sur lequel il n'a aucun contrôle.

3. Installer une bibliothèque tierce

Remarque

Si vous abhorrez le terminal, sachez que vous pouvez installer une bibliothèque tierce depuis votre IDE, ce qui sera probablement plus aisé pour vous.

3.1 À partir de Python 3.4

Pour installer une bibliothèque tierce, vous devez simplement connaître son nom. Celui-ci est généralement assez intuitif. Par exemple, la bibliothèque permettant de communiquer avec un serveur Redis s'appelle redis.
Il peut y avoir des variations. Par exemple, la bibliothèque de référence pour traiter du XML est lxml et, plus complexe, celle pour BeautifulSoup est bs4. En recherchant comment répondre à un besoin sur le Net ou sur PyPi (https://pypi.python.org/pypi), vous trouverez rapidement une bibliothèque de référence.
Sur des sujets plus confidentiels, il vous arrivera de trouver plusieurs petites bibliothèques. Vous pourrez alors les tester et choisir celle que vous utiliserez pour votre projet.
Sachez que vous pouvez aussi conduire une recherche directement depuis votre terminal :

```
$ pip search xml
$ pip search soup
```

Cela vous donnera une liste de bibliothèques accompagnée d'une courte description, à la manière de ce que font les gestionnaires de paquets sous Linux (lesquels sont écrits en Python, au passage).
Sachez que **`pip`** existe quel que soit votre système d'exploitation (vous devez être familier avec le terminal de votre système, cependant) et que depuis la version 3.4 de Python, il est installé automatiquement avec celui-ci. Si ce n'est pas votre cas, consultez la section suivante : Pour une version inférieure à Python 3.4.
`pip` est un outil formidable. Si vous utilisez une version de Python qui est celle du système, vous utiliserez alors la commande **`pip`** pour gérer les bibliothèques. Si vous utilisez une autre version, telle que Python 3.5, alors vous utiliserez la commande **`pip-3.5`**. Pour Python 3.3, ce sera **`pip-3.3`**. Dans les exemples suivants, il vous faudra prendre en compte cette particularité.
Cet outil vous permettra d'installer une bibliothèque à sa dernière version ainsi que toutes les bibliothèques dépendantes. En effet, il n'est pas rare qu'une bibliothèque de Python ait besoin d'une autre bibliothèque (ou de plusieurs) pour fonctionner. Par exemple, l'installation de redis se fait par cette commande :

```
$ pip install redis
```

On peut aussi choisir la version à installer :

```
$ pip install -Iv redis==2.10.5
```

Ou mettre à jour la bibliothèque à une version précise :

```
$ pip install -U redis==2.10.5
```

Ou à la dernière version :

```
$ pip install -U redis
```

Et on peut la désinstaller :

```
$ pip uninstall redis
```

Une fonctionnalité très importante permet d'obtenir la liste des bibliothèques déjà installées (quelle que soit la manière dont elles ont été installées) :

```
$ pip freeze
```

Ce que l'on peut mettre dans un fichier :

```
$ pip freeze > requirements.txt
```

Pour installer tous les paquets ainsi listés, il faut procéder ainsi :

```
$ pip install -r requirements/base.txt
```

Cette méthode est particulièrement utile dans le cadre d'un environnement virtuel ; nous y reviendrons.
Il est possible de retrouver des informations sur un paquet déjà installé :

```
$ pip show django-redis
---
Name: django-redis
Version: 4.3.0
Location: /path/to/my/env/lib/python3.4/site-packages
Requires: redis
```

On voit ici que le paquet **`django-redis`** a une dépendance vers **`redis`** : en l'installant, on installe automatiquement **`redis`**.
Mettre à jour ce paquet met à jour automatiquement les dépendances :

```
$ pip install -U django-redis
```

Si on ne veut pas mettre à jour les dépendances, on peut procéder ainsi :

```
$ pip install -U --no-deps django-redis
```

On peut aussi installer plusieurs bibliothèques en même temps :

```
$ pip install django-redis==4.3.0 bs4 lxml
```

Cette commande installera donc automatiquement redis s'il n'est pas installé, car il est déclaré comme dépendance.
Cette commande a cependant des limites. En effet, si vous installez une bibliothèque tierce qui utilise une bibliothèque C, vous devrez disposer des en-têtes C correspondants (paquets **`dev`** pour Debian ou **`devel`** pour Fedora). Il faut donc avoir un peu de pratique dans ce genre de situation pour savoir déjouer ces pièges.

Chapitre 3
Préparer vos données pour en exploiter le potentiel

1. Qualité des données : rappel

La qualité des données est un élément fondamental à considérer avant d'aborder les techniques de nettoyage et de traitement des données. Pour toute organisation cherchant à prendre des décisions éclairées et à tirer le meilleur parti de ses informations, cet aspect ne peut être négligé. Toutes les procédures que nous examinerons dans ce chapitre ont un seul but : mettre à la disposition des différentes équipes des données fiables !

1.1 Qu'est-ce que la qualité des données ?

La qualité des données reflète la capacité d'une organisation à maintenir l'exactitude et la pérennité de ses informations au cours du temps. En tant qu'experts du domaine, nous devons fournir des données irréprochables tout en nous appuyant sur des indicateurs clairs et facilement interprétables. Nous commencerons par examiner en détail les six critères qui définissent la qualité des données (QDD).

Cette notion englobe à la fois les caractéristiques intrinsèques des données et les méthodes mises en œuvre pour les garantir. En essence, la qualité des données se définit par leur aptitude à servir l'usage auquel elles sont destinées.

Une initiative de qualité des données s'inscrit dans la durée et s'intègre à l'ensemble du cycle de vie des données. Elle requiert une évolution culturelle dans la façon dont l'organisation gère ses données. C'est une approche globale qui impacte l'ensemble de l'entreprise et ses pratiques quotidiennes.

Il est important de noter que des données erronées en entrée d'un processus produiront inévitablement des résultats inexacts en sortie. Par conséquent, une stratégie fondée sur des données de piètre qualité aboutira à des décisions inefficaces, avec des conséquences directes sur le retour sur investissement.

Crédits : https://dataedo.com/

1.2 Pourquoi est-ce que la QDD est importante ?

La qualité des données est souvent compromise par différents facteurs. On peut citer par exemple les erreurs humaines lors de la saisie initiale. En effet, les fautes de frappe, les conventions de nommage différentes entre les sources de données ou des abréviations incorrectes sont une source fréquente de problèmes. De plus, les informations initialement exactes peuvent devenir obsolètes au fil du temps en raison de l'évolution du contexte.

La qualité des données est souvent compromise par divers facteurs, entraînant des conséquences coûteuses pour les entreprises. Voici quelques exemples concrets de problèmes fréquemment rencontrés et leurs impacts :

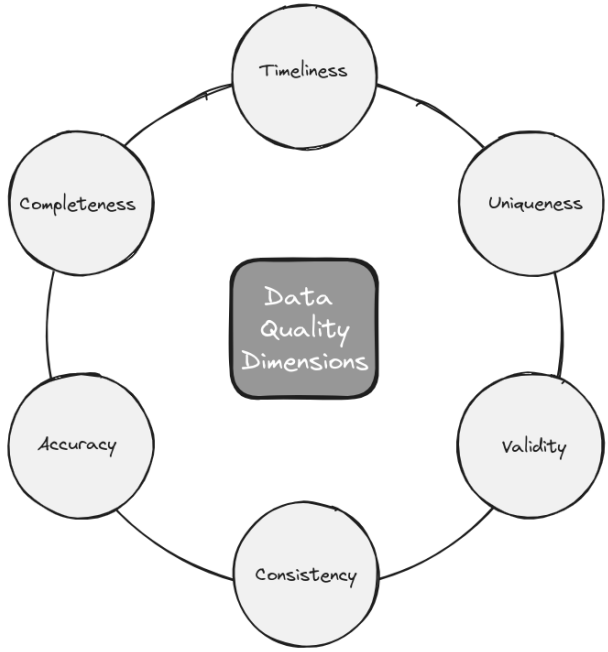
– Erreurs de saisie : en 2018, un employé de Samsung Securities a commis une erreur de saisie monumentale lors de la distribution de dividendes, émettant par inadvertance 2,8 milliards d'actions « fantômes », soit environ trente fois le nombre total d'actions existantes. Cette erreur a provoqué une perturbation massive du marché et une perte de confiance envers la gestion de l'entreprise, nécessitant des mesures correctives coûteuses. [Source : https://www.sirfull.com/blog/impacts-mauvaise-gestion-donnees/]
– Incohérences de données : en 2022, Unity Technologies a fait face à un problème majeur avec son outil de publicité ciblée Pinpointer. Des données inexactes provenant d'un grand client ont corrompu les modèles d'IA, entraînant une perte de revenus de 110 millions de dollars et une chute de 37 % de la valeur des actions de l'entreprise. [Source : https://zeenea.com/fr/quelles-sont-les-principales-erreurs-liees-a-la-data-quality-et-comment-les-resoudre/]

- Données obsolètes : Tesco, le géant britannique de la distribution, a connu des problèmes liés à des erreurs dans ses données de stock, conduisant à des ruptures fréquentes dans ses magasins. Ces inexactitudes ont entraîné une perte de ventes, une insatisfaction des clients et un impact négatif direct sur le chiffre d'affaires de l'entreprise. [Source : https://www.sirfull.com/blog/impacts-mauvaise-gestion-donnees/]
- Erreurs dans les données critiques : entre mars et juillet 2017, Equifax a généré des scores de crédit incorrects pour des millions de consommateurs en raison de données erronées. L'entreprise a dû faire face à des amendes réglementaires, des poursuites judiciaires et une perte de crédibilité, mettant en danger les décisions de prêt et érodant la confiance du public. [Source : https://www.datagalaxy.com/fr/blog/big-data-attention-aux-donnees-de-mauvaise-qualite/]
- Données de localisation inexactes : Royal Dutch Shell a rencontré des erreurs dans les données de localisation de ses puits de pétrole, conduisant à des forages inefficaces. Ces erreurs ont entraîné des coûts supplémentaires de plusieurs millions de dollars et une perte de temps considérable due à des forages incorrects. [Source : https://solutions-business-intelligence.fr/data-quality-enjeux-et-bonnes-pratiques/]

Chacun de ces problèmes de qualité des données a eu des répercussions significatives sur les opérations, la réputation et les résultats financiers des entreprises concernées. Ces exemples soulignent l'importance d'investir dans des processus robustes de gestion de la qualité des données pour éviter ces pièges coûteux.

1.3 Les principaux critères de la QDD

Découvrons les principaux critères de la QDD. Ces critères essentiels sont au cœur de toute stratégie efficace de gestion des données.



1.3.1 Exactitude (accuracy)

L'exactitude est le degré de conformité des données stockées avec les valeurs réelles qu'elles sont censées représenter. Elle mesure la proximité entre la valeur enregistrée et la valeur correcte ou acceptée comme étant vraie.

Dans le secteur bancaire, imaginons une erreur dans le calcul des intérêts d'un prêt immobilier. Si le taux d'intérêt est incorrectement enregistré à 3,5 % au lieu de 3,05 %, cela pourrait entraîner une surfacturation pour des milliers de clients. Non seulement cela pourrait conduire à des pertes financières importantes pour la banque en cas de remboursement, mais cela pourrait aussi gravement nuire à sa réputation et potentiellement entraîner des sanctions réglementaires.

Pour assurer l'exactitude, les entreprises peuvent mettre en place des processus de validation des données, des contrôles croisés automatisés, et des audits réguliers. L'utilisation de l'intelligence artificielle pour détecter les anomalies peut également être efficace.

1.3.2 Exhaustivité (completeness)

L'exhaustivité est la mesure dans laquelle tous les éléments de données nécessaires sont présents dans un ensemble de données spécifique. Elle évalue le degré auquel toutes les valeurs requises sont incluses et tous les enregistrements qui devraient être présents le sont effectivement.

Dans le domaine de la recherche médicale, imaginons une étude sur l'efficacité d'un nouveau traitement contre le cancer. Si les données de suivi post-traitement sont incomplètes pour un groupe significatif de patients, cela pourrait fausser les conclusions de l'étude. Des décisions cruciales concernant l'approbation ou le rejet du traitement pourraient être basées sur des informations partielles, affectant potentiellement la vie de nombreux patients.

Utiliser des champs obligatoires dans les formulaires de saisie, mettre en place des alertes pour les données manquantes, et implémenter des processus de collecte de données systématiques peuvent améliorer l'exhaustivité.

1.3.3 Cohérence (consistency)

La cohérence fait référence à l'absence de contradictions dans les données au sein d'un ensemble de données ou entre différents ensembles de données. Elle assure que les données sont uniformes et logiquement compatibles dans tous les systèmes, applications et processus de l'organisation.

Dans une multinationale, imaginons que le département des ressources humaines et le département finance utilisent des systèmes différents pour gérer les informations des employés. Si un employé change de poste et que cette information n'est mise à jour que dans le système RH, cela pourrait conduire à des erreurs dans la paie, les avantages sociaux, et même dans la planification stratégique des ressources humaines.

L'utilisation d'un système d'information intégré, la mise en place de processus de synchronisation automatique entre différents systèmes, et l'établissement de règles de gestion des données cohérentes à l'échelle de l'entreprise peuvent améliorer la cohérence.

1.3.4 Actualité (timeliness)

L'actualité mesure le degré auquel les données sont à jour et disponibles dans le délai requis pour leur utilisation prévue. Elle évalue la fraîcheur des données par rapport au moment de leur création ou de leur dernière mise à jour, ainsi que leur disponibilité au moment où elles sont nécessaires pour les processus métier.

Dans le domaine du trading haute fréquence, où des décisions d'achat et de vente sont prises en millisecondes, l'actualité des données est critique. Un retard de quelques secondes dans la mise à jour des prix des actions pourrait entraîner des pertes financières considérables. Par exemple, si un événement majeur affecte le cours d'une action, mais que cette information n'est pas reflétée immédiatement dans les données utilisées par les algorithmes de trading, cela pourrait conduire à des décisions d'investissement désastreuses.

L'utilisation de systèmes de traitement en temps réel, la mise en place de processus de mise à jour automatique des données, et l'optimisation des flux de données peuvent améliorer l'actualité.

1.3.5 Validité (validity)

La validité est la mesure dans laquelle les données sont conformes aux règles métier définies, aux formats spécifiés et aux contraintes du domaine. Elle assure que les valeurs des données respectent les critères syntaxiques et sémantiques établis pour le type de données en question.

Dans le secteur de l'aviation, imaginons un système de réservation qui accepte une date de vol antérieure à la date actuelle. Cela pourrait conduire à des problèmes majeurs dans la planification des vols, la gestion des équipages, et potentiellement compromettre la sécurité si ces données invalides sont utilisées dans d'autres systèmes critiques.

L'implémentation de contrôles de validation stricts dans les interfaces de saisie, l'utilisation de contraintes au niveau de la base de données, et la mise en place de processus de nettoyage régulier des données peuvent améliorer la validité.

1.3.6 Unicité (uniqueness)

L'unicité est la propriété selon laquelle chaque entité distincte du monde réel est représentée une et une seule fois dans l'ensemble de données. Elle garantit l'absence de doublons ou de redondances non intentionnelles dans les enregistrements de données.

Dans le secteur de la santé, imaginons un patient ayant plusieurs dossiers médicaux en raison de doublons dans la base de données. Cela pourrait conduire à des erreurs graves dans le traitement si un médecin n'a pas accès à l'historique médical complet du patient. Par exemple, des allergies ou des interactions médicamenteuses pourraient être manquées, mettant en danger la santé du patient.

L'utilisation d'identifiants uniques, la mise en place de processus de déduplication, et l'implémentation de contrôles stricts lors de la création de nouveaux enregistrements peuvent améliorer l'unicité des données.

2. Nettoyage de données

2.1 Premiers pas avec la librairie pandas

Pandas est une bibliothèque Python puissante et polyvalente, conçue pour la manipulation et l'analyse des données. Elle a été développée par Wes McKinney, un chercheur qui a commencé à construire ce qui allait devenir Pandas. Le nom « pandas » est dérivé du terme « Panel Data », un terme d'économétrie pour les jeux de données qui comprennent des observations sur plusieurs périodes.

Pandas est particulièrement adaptée pour travailler avec des données tabulaires, similaires à une feuille de calcul Excel ou une table SQL. Les principales structures de données gérées par cette bibliothèque sont les séries, qui stockent des données selon une dimension, et les DataFrames, qui stockent des données selon deux dimensions (lignes et colonnes). Ces structures de données facilitent la manipulation des données, ainsi que le nettoyage, le prétraitement, l'analyse et la visualisation.

L'utilisation de pandas est largement répandue dans le domaine de l'analyse de données. Elle est souvent présentée comme l'outil idéal pour manipuler des données qui peuvent être organisées sous forme de lignes et de colonnes. De plus, la maîtrise de pandas est une compétence recherchée par les employeurs, car de nombreuses entreprises de tous secteurs utilisent de plus en plus la science des données.

Il existe plusieurs alternatives à pandas, on peut citer notamment polars, dask et cudf. Chacune de ces solutions présente un intérêt, en particulier la vitesse de traitement par rapport à pandas. Nous n'en parlerons pas dans cet ouvrage, car Pandas demeure la bibliothèque la plus utilisée pour l'analyse de données en Python. Sa richesse et sa polyvalence, ainsi que sa large adoption dans la communauté d'analystes de données, en font un outil incontournable.

2.2 Présentation de notre jeu de données

Au cours de ce chapitre, nous allons travailler avec un jeu de données disponible en libre accès sur la plateforme Kaggle.

Le commerce électronique est devenu un nouveau canal pour soutenir le développement des entreprises. Grâce au commerce électronique, les entreprises peuvent accéder à un marché plus large et établir une présence plus importante en fournissant des canaux de distribution moins coûteux et plus efficaces pour leurs produits ou services. Le commerce électronique a également changé la façon dont les gens achètent et consomment des produits et des services. De nombreuses personnes se tournent vers leurs ordinateurs ou leurs appareils intelligents pour commander des biens, qui peuvent être facilement livrés à leur domicile.