Partie 3
Les statistiques

Chapitre 3-1
Statistiques

1. Objectif du chapitre

Les statistiques regroupent un ensemble de méthodes dédiées à l'échantillonnage de données ainsi qu'à leur analyse afin de tirer des conclusions et de comprendre les phénomènes sous-jacents à ces données. Ces méthodes statistiques font partie intégrante de la Data Science.

Il est quasiment impossible d'aborder l'ensemble des méthodes statistiques en un seul ouvrage vu leur diversité. Il existe plusieurs livres qui traitent des statistiques. L'objectif de ce chapitre est double : le premier est la présentation des outils statistiques élémentaires que tout Data Scientist devrait connaître, et le deuxième objectif est d'attirer l'attention du lecteur sur l'intérêt des statistiques et leur relation avec la Data Science. Ainsi, nous allons porter une attention particulière à la partie inférentielle des statistiques.

À la fin de ce chapitre, le lecteur aura abordé :

– les statistiques descriptives,
– les lois de probabilité,
– la loi normale et la loi normale centrée réduite,
– le principe de l'échantillonnage,
– le théorème central limite,
– l'estimation ponctuelle,

– l'estimation par intervalle de confiance,
– les tests d'hypothèses,
– le paradoxe de Simpson,
– les séries temporelles.

2. Les statistiques descriptives

Les statistiques descriptives permettent de résumer un ensemble de données de manière concise. Avec les statistiques descriptives, et pour un échantillon de valeurs $E = \{x_1, x_2, \dots x_n\}$, nous pouvons calculer certains paramètres afin de cerner la nature de la distribution associée aux valeurs x_i.

Ainsi, nous distinguons deux types de paramètres que nous pouvons calculer sur une série statistique de type quantitative : les paramètres de position et les paramètres de dispersion présentés dans les sous-sections suivantes.

2.1 Paramètres de position

Les paramètres de position permettent d'avoir une idée précise sur la nature du domaine de définition d'un ensemble de données. Ces paramètres de position, également appelés indicateurs de position, sont des nombres réels utilisés comme référence pour un ensemble $E = \{x_1, x_2, \dots x_n\}$.

2.1.1 La moyenne

La moyenne $\overline{X}$ associée à une série de valeurs $S = (x_1, x_2, \dots, x_n)$ se calcule comme suit $\overline{X} = \frac{1}{n}\sum_{i=1}^{n} x_i$.

La moyenne ainsi calculée correspond à la moyenne arithmétique. Il existe d'autres types de moyennes telles que la moyenne harmonique, la moyenne quadratique ou encore la moyenne géométrique. Généralement, en statistiques, la moyenne utilisée est la moyenne arithmétique.

Remarque

Si la série S est un échantillon issu d'une population plus grande, alors il ne faut pas confondre la moyenne $\overline{X}$ calculée sur cet échantillon avec la moyenne de la population ! Dans la suite de ce chapitre, nous allons revenir sur la relation entre la moyenne d'un échantillon et la moyenne de la population.

2.1.2 Le mode

Le mode d'une série de valeurs est tout simplement la valeur qui apparaît le plus fréquemment.

Par exemple, dans la série de valeurs $S=(1, 2, 5, 2, 5, 5, 6, 8, 5, 9, 5)$, on dira que le mode est la valeur 5, car c'est bien cette valeur qui apparaît avec le plus d'occurrences. La valeur 5 apparaît cinq fois, la valeur 2 deux fois, puis les autres valeurs apparaissent une fois chacune.

La série S de notre exemple est dite unimodale, car elle dispose d'un seul mode. La série $S^1=(1, 2, 5, 2, 5, 5, 2, 2, 5, 2, 5)$ est dite bimodale, car elle dispose de deux modes, à savoir le mode 5 et le mode 2.

2.1.3 La médiane

La médiane associée à une série de valeurs rangée dans l'ordre croissant $S = (x_1, x_2, \ldots, x_n)$ avec $x_i < x_{i+1}\ \forall\, i$ est une valeur qui partage toutes les valeurs de S en deux groupes de valeurs de taille égale. La valeur de la médiane peut ou pas faire partie de S.

Si les valeurs x_i de S sont des valeurs discrètes, alors la médiane se calcule comme suit :

– Si la taille n de la série S est impaire, alors nous pouvons écrire $n=2p+1$. Comme les valeurs de S sont ordonnées, la médiane est la $(p+1)^{\text{ième}}$ valeur. Dans ce cas, la médiane fait partie de la série S.

  Par exemple, pour la série $S=(1, 2, 5, 8, 15, 55, 68, 82, 125, 199, 500)$, la médiane est la valeur 55, alors que la moyenne est égale à 96,36.

– Si la taille n de la série S est paire, alors nous pouvons écrire $n=2p$. Comme les valeurs de S sont ordonnées, la médiane est la moyenne entre la $(p+1)^{\text{ième}}$ valeur et la $p^{\text{ième}}$ valeur. Dans ce cas, la médiane ne fait pas partie de la série S.

  Par exemple, pour la série $S=(1,2,5,8,15,55,68,82,125,199)$, la médiane est la valeur $\frac{15+55}{2} = 35$ et la moyenne est égale à 56.

Remarquez que dans des deux derniers exemples, les différences entre les moyennes et les médianes étaient importantes !

Si les valeurs x_i de S sont des valeurs continues, alors la médiane correspond à la valeur centrale que nous pouvons calculer par interpolation linéaire du centre des effectifs cumulés sur les x_i.

Par exemple, soit le tableau récapitulatif des notes obtenues par un groupe de 100 étudiants :

Notes	Effectifs
[0;5[	15
[5;7[	30
[7;11[	20
[11;16[	25
[16;20[	10
Total	100

À partir de ce tableau, nous allons procéder au calcul des effectifs cumulés comme suit :

Notes	Effectifs	Effectifs cumulés
[0;5[	15	15
[5;7[	30	45
[7;11[	20	65

Notes	Effectifs	Effectifs cumulés
[11;16[	25	90
[16;20[	10	100

D'après la colonne des effectifs cumulés, nous avons :

– 15 étudiants qui ont une note en dessous de 5.

– 45 étudiants qui ont une note en dessous de 7.

– 65 étudiants qui ont une note en dessous de 11.

– 90 étudiants qui ont une note en dessous de 16.

– 100 étudiants qui ont une note en dessous de 20.

Le nombre total des étudiants est égal à 100, donc la moitié est égale à 50 étudiants. L'intervalle sur l'axe des effectifs cumulés où se situe la médiane est l'intervalle [45;65], puisque 50 appartient à cet intervalle.

Nous pouvons calculer la valeur de la médiane par interpolation linéaire comme sur la figure 6-1 suivante :

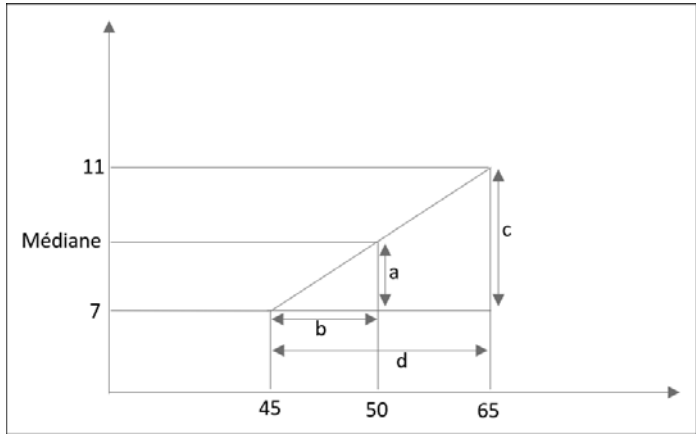


Figure 6-1 : Calcul de la médiane par interpolation linéaire du centre des effectifs cumulés

Grâce au théorème de Thalès, nous savons que $\frac{a}{b} = \frac{c}{d}$ (voir la figure précédente).

a = médiane - 7.

b = 50 - 45 = 5.

c = 11 - 7 = 4.

d = 65 - 45=20.

En procédant au remplacement de a, b, c et d par leurs valeurs respectives, on obtient $\frac{médiane - 7}{5} = \frac{4}{20}$.

Donc la médiane est égale à 8.

2.1.4 Les quartiles

Pour une série de valeurs rangées dans l'ordre croissant $S = (x_1, x_2, \ldots, x_n)$ avec $x_i < x_{i+1} \ \forall \ i$, les quartiles sont définis par trois valeurs qui partagent les valeurs de la série S en quatre groupes de valeurs de même taille. Ces trois valeurs sont définies comme suit :

– Le deuxième quartile correspond à la valeur de la médiane définie ci-dessus.

– Le premier quartile partage les valeurs de S situées entre la première valeur x_1 et la médiane en deux groupes de valeurs de même taille. En d'autres termes, le premier quartile est la médiane de la série S_1 constituée des valeurs entre x_1 et la médiane.

– De même que pour le premier quartile, le troisième quartile partage les valeurs de S situées entre la médiane et la dernière valeur xn en deux groupes de valeurs de même taille. En d'autres termes, le troisième quartile est la médiane de la série S_2 constituée des valeurs entre la médiane et x_n.

2.2 Paramètres de dispersion

Les paramètres de dispersion permettent de comprendre la variabilité associée à une série de données quantitatives.

2.2.1 La variance

La variance associée à un échantillon de données est une mesure qui nous renseigne sur la moyenne des carrés des écarts à la moyenne. On peut également dire que la variance est une valeur qui nous donne une idée sur la dispersion d'un ensemble de valeurs autour de leur moyenne.

Pour une série statistique $X = (x_1, x_2, \dots, x_n)$, la variance *Var(X)* se calcule comme suit : $Var(X) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{X})^2$.

Avec $\overline{X}$ la moyenne de la série X et qui, pour rappel, se calcule comme suit $\overline{X} = \frac{1}{n}\sum_{i=1}^{n} x_i$

À partir de la formule de *Var(X)*, on voit bien que la variance est une distance moyenne au carré qui sépare chaque élément x_i de sa moyenne $\overline{X}$.

2.2.2 Calcul de la variance avec la formule de Koenig

Le calcul de la variance *Var(X)* avec la formule vue précédemment peut être compliqué à réaliser sans l'aide d'un logiciel. La formule de Koenig est une simplification de la formule précédente et qui permet de calculer la variance comme suit : $Var(X) = \frac{1}{n}\sum_{i=1}^{n} {x_i}^2 - \overline{X}^2$.

Avec cette formule, la variance est la différence entre la moyenne des carrés des x_i et le carré de la moyenne x_i.

Avec la formule de Koenig, le calcul de la variance devient plus facile et une simple calculatrice serait largement suffisante.

2.2.3 L'écart-type

Nous avons vu que la variance est la distance moyenne au carré d'un ensemble de valeurs par rapport à leur moyenne. L'écart-type σ est tout simplement la distance moyenne, ou la distance type, qui sépare les valeurs de leur moyenne et il se calcule à partir de la variance comme suit : $\sigma = \sqrt{Var(X)}$.

2.2.4 L'écart interquartile

L'écart interquartile correspond à la distance entre le troisième et le premier quartile.

Par exemple, pour la série $S=(1, 2, 5, 8, 15, 55, 68, 82,125, 199, 500)$, les quartiles sont les suivants :

– Le premier quartile est égal à 5.

– Le deuxième quartile est égal à 55, ce qui est également la médiane.

– Le troisième quartile est égal à 125.

Pour cette série S, l'espace interquartile est donc égal à $125\text{-}5 = 120$.

La valeur de l'écart interquartile, aussi appelé l'espace interquartile ou bien l'étendue interquartile, nous renseigne sur l'ampleur de la dispersion des valeurs d'une série.

En combinant les mesures de la moyenne, du mode, de la médiane, de la variance, de l'écart-type et de l'écart interquartile, nous obtenons un résumé de l'ensemble des valeurs étudiées. Toutes ces mesures peuvent être calculées sur un ensemble de valeurs associées à une seule variable quantitative. Dans le chapitre Analyse en composantes principales, nous aborderons l'analyse en composantes principales, qui est une méthode permettant de résumer un ensemble de données définies avec plusieurs variables.

3. Les lois de probabilité

Une loi de probabilité permet de cerner le comportement d'une variable aléatoire. Dans le domaine des probabilités, une variable aléatoire dépend du hasard. Justement, c'est le comportement de ce hasard que l'on tente de décrire avec une loi de probabilité. Avec une loi de probabilité, nous pouvons calculer la probabilité qu'une variable aléatoire soit fixée à une valeur donnée.

Par exemple, si nous considérons une variable X associée au résultat obtenu après le lancer d'un dé à six chiffres, alors cette variable X sera appelée une variable aléatoire, puisque la survenue de l'un des six chiffres est un événement aléatoire.

Si nous supposons que notre dé est parfait, c'est-à-dire que chacun des six chiffres est équiprobable avec une probabilité de $\frac{1}{6}$, alors la loi de la variable X est tout simplement $F(X) = \frac{1}{6}$.

Le choix d'une loi de probabilité est en fonction de la nature de la variable aléatoire étudiée et en fonction du phénomène associé à cette variable aléatoire. En effet, une variable aléatoire X peut être discrète ou continue et elle peut être définie dans un intervalle fini, semi-fini ou infini. Le phénomène associé à une variable aléatoire peut concerner des durées, des quantités comme le poids, des longueurs comme la taille ou toute autre mesure qui peut être observée et relative à un événement quelconque.

En résumé, une loi de probabilité décrit la manière dont sont distribuées les valeurs possibles d'une variable aléatoire X.

Il existe un nombre important de lois de probabilité. Loin d'être exhaustifs, parmi les plus usuelles nous pouvons citer :

- Loi de Bernoulli : décrit la distribution d'une variable aléatoire associée à un événement à deux issues possibles.
- Loi binomiale : décrit la distribution d'une variable aléatoire associée à un événement à deux issues possibles répété plusieurs fois avec remise, c'est-à-dire que le même événement peut survenir plusieurs fois.
- Loi de Poisson : décrit la distribution d'une variable aléatoire associée au nombre d'événements se produisant dans un intervalle de temps donné. Par exemple, le nombre de personnes supplémentaires dans une file d'attente au bout d'un temps donné.
- Loi uniforme : décrit la distribution d'une variable aléatoire associée à un ensemble d'événements équiprobables.
- Loi exponentielle : décrit la distribution d'une variable aléatoire associée au délai de la survenue d'un événement. Par exemple, le délai de l'arrivée d'une personne supplémentaire dans une file d'attente.

Chapitre 3
La pile technologique en Python

1. Les outils de la Data Science

Un bon artisan a besoin de bons outils. C'est également vrai en Data Science et en Machine Learning.

Il existe trois grandes catégories d'outils :

– Des outils « intégrés » qui contiennent ce qui est nécessaire pour un projet : charger les données, les analyser, créer des modèles, les évaluer, les déployer, créer des rapports…

– Des outils « Auto ML » qui simplifient le processus pour les non-experts, en automatisant au maximum les différentes phases.

– Des outils de « développement » avec lesquels il est possible de faire plus de choses, à condition de tout coder, par exemple en Python.

1.1 Les outils intégrés

Il existe de nombreux outils dans la première catégorie, et chacun nécessite un apprentissage particulier permettant de s'en servir au mieux. De plus, ayant chacun leurs points forts et points faibles, il est important de pouvoir choisir le bon logiciel selon le but recherché.

Il est possible de citer : Dataiku DSS, SAS, QlikView, Tableau, Power BI, Snowflake, etc. Certains sont très orientés statistiques (SAS), d'autres visualisation de données (Tableau), mais tous couvrent au moins une partie du processus. Il s'agit principalement d'applications téléchargeables et accessibles via une interface web.

Remarque

Attention, la plupart de ces logiciels sont payants. Il existe bien souvent des versions gratuites, mais qui sont bridées dans leurs fonctionnalités et qui limitent souvent leur utilisation à un usage personnel. Pour être en règle, il est donc important de se rapprocher des principaux éditeurs.

1.2 L'auto ML

Les outils d'auto ML sont des outils généralement accessibles via une interface web. L'utilisateur y rentre ses données brutes et la tâche voulue : classification, régression, etc. L'application se charge ensuite du reste : choix de la préparation des données à effectuer, choix des modèles, choix des paramètres. Le meilleur modèle créé est alors retourné à l'utilisateur.

Ces outils ont l'énorme avantage d'être utilisables sans connaissance en Machine Learning. Cependant, ils manquent de souplesse et ne sont souvent pas très aimés des Data Scientists qui préfèrent avoir le contrôle de leur processus de modélisation.

Ils permettent en revanche d'avoir un premier modèle très rapidement, et un Data Scientist peut ensuite consacrer du temps à d'autres modèles et à l'optimisation des résultats de l'auto ML. La majorité des acteurs du cloud ont leurs propres outils.

1.3 Les outils de développement

Il est tout à fait possible de coder les algorithmes de Machine Learning dans tous les langages de programmation. De nombreux frameworks ou librairies existent, prêts à être incorporés dans des projets.

Les outils de développement classiques sont donc tout à fait utilisables. Tous les éditeurs de code comme Atom, Visual Studio Code ou encore Sublim Text peuvent servir pour un projet.

De plus, le code a l'avantage de plus facilement être partagé et synchronisé entre plusieurs personnes grâce aux gestionnaires de versions comme Git. Par ailleurs, les outils de CI/CD (Intégration continue/Déploiement continu) comme GitLab CI, Jenkins ou Ansible peuvent faciliter le déploiement et la mise à jour. Là encore, chaque cloud provider propose ses outils supplémentaires, comme la série des services Amazon Web Services (AWS) dont le nom commence par « Code » : CodeCommit, CodeBuild... ou encore Azure DevOps.

Ils sont de plus en plus utilisés aujourd'hui.

2. Langage Python

2.1 Présentation

Python est un langage de programmation dont la première version est sortie en 1991. C'est donc un langage mature, connu depuis longtemps des développeurs.

Il possède plusieurs caractéristiques :

– **Interprété** : cela signifie qu'il n'y a pas de compilation avant de pouvoir exécuter le code. Il est donc facile à déboguer mais cela le rend moins efficace en termes de temps d'exécution que des langages compilés comme C/C++. Cependant, la majorité des librairies dites « bas niveau » étant codée en C, le code reste performant.

- **Multiplateforme** : un code en Python ne dépend pas de la plateforme sur laquelle il s'exécute, ce qui le rend très portable et facile à partager. Il est en particulier compatible avec Windows, Unix, Mac OS, Android, iOS.
- **Multiparadigme** : Python permet différentes formes de programmation et s'adapte donc à la majorité des développeurs. Il est ainsi possible d'écrire :
  - En programmation impérative, forme de programmation la plus ancienne
  - En programmation orientée objet, forme aujourd'hui préférée des développeurs tous langages confondus
  - Et en programmation fonctionnelle, permettant une évaluation de fonctions mathématiques et la manipulation de listes de manière simplifiée
- **Lisible** : le langage Python ne contient pas de délimiteurs de blocs contrairement à d'autres langages, comme des accolades en Java ou des mots-clés begin – end en Pascal. La structure se définit par l'indentation des blocs. De cette façon, il est plus court et lisible et, avantage non négligeable, forcément bien indenté.
- **Facile à apprendre** : que vous soyez débutant en programmation ou que vous connaissiez un autre langage, il est facile d'apprendre à coder en Python, même si maîtriser complètement le langage demande plusieurs années d'expérience.

Toutes ces caractéristiques en font un très bon langage pour la Data Science, bien qu'il ne soit pas le seul utilisable.

2.2 Brève présentation de R

R est un langage de programmation destiné aux statistiques et à la Data Science. Il s'agit d'un langage libre et open source. Ce langage a été fortement développé depuis 1997. C'est la force de sa communauté et les nombreux packages disponibles (plus de 15000) qui en font un langage si apprécié.

En effet, quel que soit l'analyse ou le calcul souhaité sur des données, il existe sûrement un package R permettant de le faire.

Sa syntaxe est cependant particulière, avec par exemple pour l'affectation le symbole <-, mais les tableaux et matrices sont très bien gérés et le code optimisé.

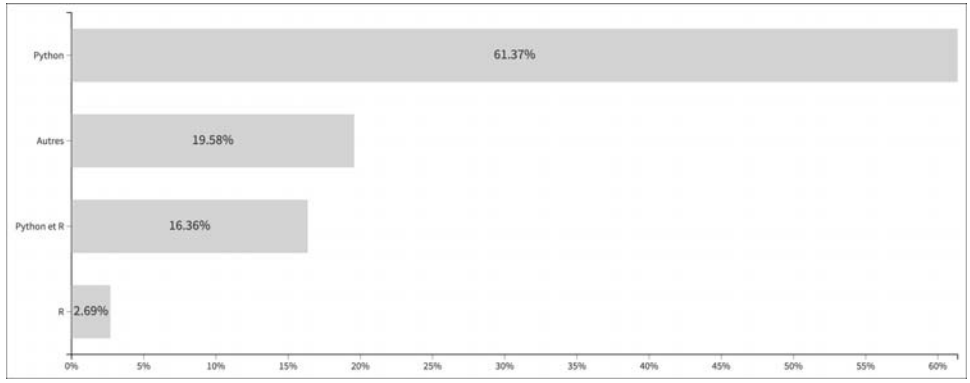
Tout comme Python, il est interprété, multiplateforme et lisible.

2.3 Python ou R ?

Python et R sont devenus les deux langages préférés des Data Scientists. Les deux permettent de coder facilement la majorité des algorithmes de Machine Learning et disposent de librairies pour accéder aux principaux formats ou sources de données.

En tant que langage de programmation, Python est particulièrement apprécié des Data Engineers, et tous les algorithmes existent dans des frameworks Python, souvent codés en C pour des questions d'optimisation.

D'après l'enquête de Kaggle sur les langages utilisés par les Data Scientists (« Data Science and Machine Learning Survey »), Python devance R depuis 2017. Le sondage réalisé fin 2022 donne les résultats suivants, ce qui fait de Python le langage préféré par trois quarts des Data Scientists :



2.4 Python 2 vs Python 3

Il existe deux versions majeures de Python :

– Python 2, supportée jusqu'à fin 2019, et dont la dernière version est la version 2.7, sortie en 2010

– Python 3, actuellement en version 3.12, sortie en octobre 2023

Les différences entre la version 2 et la version 3 sont importantes, rendant souvent la compatibilité entre du code 2.7 et 3.x impossible. De plus, de nombreuses librairies ne sont compatibles qu'avec une seule des versions.

Depuis début 2020, la version 2 n'est plus supportée et ne doit donc plus être utilisée. Il existe cependant quelques cas exceptionnels, comme du code commencé en 2.7 qui doit être maintenu en attendant une migration vers 3.x, ou du code faisant appel à une librairie non disponible en 3.x (ce qui est de moins en moins le cas).

Attention aussi au choix de la version en 3.x. La version 2.7 ayant été très longtemps la version mise en production, les éditeurs de librairies n'ont pas forcément tous créé des versions compatibles avec toutes les versions 3.x. Par exemple TensorFlow, librairie la plus utilisée pour le Deep Learning, n'est compatible avec Python 3.8 que depuis Tensorflow 2.2 et Python 3.9 que depuis sa version 2.5. À l'heure où nous écrivons ces lignes, Tensorflow n'est officiellement pas supporté pour les versions de Python supérieures à 3.9.

De même, il peut exister des contraintes sur les versions disponibles sur un système donné en particulier si le déploiement doit avoir lieu sur du matériel de faible puissance, comme des objets connectés.

Il est donc important de définir les librairies externes et les capacités des systèmes utilisés pour le déploiement en amont du projet, afin de ne pas faire de choix bloquants au moment du déploiement.

3. Jupyter

3.1 Caractéristiques de Jupyter

Jupyter est un logiciel qui propose de créer des « notebooks ». Chaque notebook est en réalité une page de taille non fixée, dans laquelle vont s'enchaîner des cellules.

Chaque cellule peut être de différents types :

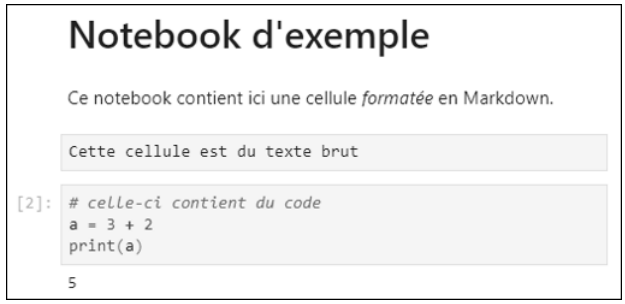
– Code, qui pourra être exécuté et dont le résultat s'affichera immédiatement sous la cellule à l'exécution.
– Texte brut, non formaté et affiché tel quel.
– Texte en Markdown, pour de la mise en page et même des formules en LaTeX.

Remarque

*Le Markdown est un langage de formatage léger créé en 2004 et utilisé dans de nombreux logiciels. Par exemple, les titres de niveau 1 commencent par #, ceux de niveau 2 par ##, etc. Les textes sont mis en italique et en gras en les entourant d'astérisques : *italique*, **gras**.*

Le LaTeX est quant à lui un langage de composition de documents. Fortement utilisé dans les milieux académiques, il permet de dissocier l'écriture du texte de sa mise en page. En effet, celle-ci est calculée lors d'une compilation du texte brut, en respectant les contraintes typographiques et le modèle voulu. Le langage LaTeX est très réputé pour sa capacité à écrire facilement des équations complexes.

Voici un exemple de rendu d'un notebook :



Chaque notebook doit être associé à un kernel. Il s'agit d'un environnement contenant le langage de programmation principal du notebook ainsi que les packages installés. Cela permet d'avoir en parallèle différentes versions d'un même langage.

Il est ainsi possible d'avoir dans le même Jupyter un kernel en Python 3.6 et un en Python 3.9, avec différents modules. En effet, chaque kernel peut avoir ses propres librairies ou des versions différentes de celles-ci, ce qui évite en partie les conflits potentiels.

Actuellement, Jupyter supporte plus de quarante langages dont Python, R, Julia, Scala… De plus, il s'intègre avec Spark, qui permet de gérer des gros volumes de données partitionnés sur plusieurs nœuds.

Le format d'enregistrement des notebooks est un fichier .ipynb qui est en réalité un fichier JSON contenant toutes les informations nécessaires : contenu des cellules et des retours s'ils sont enregistrés. Ce notebook peut ensuite être exporté dans les principaux formats d'échange, dont PDF, HTML, EPS…

Le code JSON du notebook montré précédemment est le suivant :

```
{
"cells": [
 {
  "cell_type": "markdown",
  "metadata": {},
  "source": [
   "# Notebook d'exemple\n",
```