# [Décrivez et nettoyez votre jeu de données](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees)

Table des matières

[Décrivez et nettoyez votre jeu de données 1](#_Toc57624740)

[Tirez parti de ce cours 7](#_Toc57624741)

[L'organisation du cours 7](#_Toc57624742)

[Qu'allez-vous faire dans ce cours ? 7](#_Toc57624743)

[Selon votre niveau de motivation 8](#_Toc57624744)

[Python ou R ? 8](#_Toc57624745)

[Le code informatique et les données 9](#_Toc57624746)

[Installez R ou Python 10](#_Toc57624747)

[Ce dont vous aurez besoin 10](#_Toc57624748)

[Comment s'orienter dans ce vaste chapitre ? 10](#_Toc57624749)

[R ou Python: lequel choisir ? 10](#_Toc57624750)

[Notebook or not Notebook ? 11](#_Toc57624751)

[Installer R sans notebook 11](#_Toc57624752)

[Installer R avec notebook 15](#_Toc57624753)

[Installer la totale : Python + librairies + notebook (distribution Anaconda) 16](#_Toc57624754)

[Installer Python sans notebook 17](#_Toc57624755)

[Installer Python avec notebook 17](#_Toc57624756)

[Tester si Python est installé 18](#_Toc57624757)

[Lancer et tester Jupyter 18](#_Toc57624758)

[Ouvrir une console 22](#_Toc57624759)

[Erreurs fréquentes 24](#_Toc57624760)

[Découvrez les statistiques : vocabulaire et tour d’horizon 25](#_Toc57624761)

[Le vocabulaire 25](#_Toc57624762)

[Comment représenter un échantillon ? 25](#_Toc57624763)

[Petit tour d’horizon des statistiques 26](#_Toc57624764)

[Aller plus loin : Plus de cours ! 27](#_Toc57624765)

[Aller plus loin : Data Analyst vs Data Scientist 28](#_Toc57624766)

[Téléchargez les données 29](#_Toc57624767)

[Les données et le code informatique 29](#_Toc57624768)

[C'est parti ! 29](#_Toc57624769)

[Ce que nous allons faire 30](#_Toc57624770)

[C'est parti ! 30](#_Toc57624771)

[Et si ça a planté...? 38](#_Toc57624772)

[Découvrez les 4 types de variables 39](#_Toc57624773)

[Les variables quantitatives 39](#_Toc57624774)

[Les variables qualitatives 39](#_Toc57624775)

[Aller plus loin : Les dates, quantitatives ou qualitatives ? 40](#_Toc57624776)

[Représentez la distribution empirique d'une variable 42](#_Toc57624777)

[Représenter une distribution empirique 42](#_Toc57624778)

[Du côté du code 46](#_Toc57624779)

[Aller plus loin : La fonction de répartition empirique 47](#_Toc57624780)

[Aller plus loin : Nombre optimal de classes pour l'agrégation 48](#_Toc57624781)

[Présentez une variable sous forme de tableau 49](#_Toc57624782)

[Donnons des noms à ce que nous étudions 49](#_Toc57624783)

[Représenter les variables sous forme de tableau 50](#_Toc57624784)

[Et le code ? 51](#_Toc57624785)

[Testez vos connaissances sur les statistiques descriptives 53](#_Toc57624786)

[Compétences évaluées 53](#_Toc57624787)

[ Question 1 53](#_Toc57624788)

[ Question 2 53](#_Toc57624789)

[ Question 3 54](#_Toc57624790)

[ Question 4 54](#_Toc57624791)

[ Question 5 54](#_Toc57624792)

[ Question 6 54](#_Toc57624793)

[Laissez-vous guider par les indicateurs 56](#_Toc57624794)

[Qu'est-ce qu'une statistique ? 56](#_Toc57624795)

[A quoi servent les indicateurs et indices statistiques ? 56](#_Toc57624796)

[Que nous disent les indicateurs ? 57](#_Toc57624797)

[Découvrez les mesures de tendance centrale 58](#_Toc57624798)

[Les mesures de tendance centrale 58](#_Toc57624799)

[Du côté du code 60](#_Toc57624800)

[Aller plus loin : Sur un histogramme 62](#_Toc57624801)

[Aller plus loin : Distributions plurimodales 62](#_Toc57624802)

[Comprenez les mesures de dispersion 64](#_Toc57624803)

[Réfléchissons 64](#_Toc57624804)

[Les mesures de dispersion 65](#_Toc57624805)

[Du côté du code 67](#_Toc57624806)

[Aller plus loin : La variance empirique corrigée 68](#_Toc57624807)

[Aller plus loin : Calculs avec la variance empirique 68](#_Toc57624808)

[Aller plus loin : Le coefficient de variation 69](#_Toc57624809)

[Aller plus loin : Autres mesures de dispersion 69](#_Toc57624810)

[Appréhendez les mesures de forme 71](#_Toc57624811)

[Réfléchissons 71](#_Toc57624812)

[Les mesures de forme 73](#_Toc57624813)

[Du côté du code 74](#_Toc57624814)

[Aller plus loin : Quelques mots sur l'asymétrie 76](#_Toc57624815)

[Aller plus loin : Les moments 76](#_Toc57624816)

[Familiarisez-vous avec les mesures de concentration 78](#_Toc57624817)

[Les mesures de concentration 78](#_Toc57624818)

[Du côté du code 82](#_Toc57624819)

[Aller plus loin : La médiale 83](#_Toc57624820)

[Abordez encore plus de mesures 84](#_Toc57624821)

[Taux de croissance 84](#_Toc57624822)

[Moyennes 85](#_Toc57624823)

[Aller plus loin : La formule générale d'une moyenne 86](#_Toc57624824)

[Les mesures statistiques 87](#_Toc57624825)

[Compétences évaluées 87](#_Toc57624826)

[ Question 1 87](#_Toc57624827)

[ Question 2 87](#_Toc57624828)

[ Question 3 87](#_Toc57624829)

[ Question 4 88](#_Toc57624830)

[ Question 5 88](#_Toc57624831)

[ Question 6 89](#_Toc57624832)

[Entrez dans le monde de l’analyse bivariée 90](#_Toc57624833)

[Pourquoi l'analyse bivariée ? 90](#_Toc57624834)

[Aller plus loin : Un autre exemple 92](#_Toc57624835)

[Recherchez les corrélations 94](#_Toc57624836)

[La causalité 94](#_Toc57624837)

[Réfléchissons... 95](#_Toc57624838)

[Le tableau de contingence 96](#_Toc57624839)

[Analysez la corrélation entre deux variables quantitatives 98](#_Toc57624840)

[2 variables quantitatives : graphiques 98](#_Toc57624841)

[2 variables quantitatives : indicateurs numériques 99](#_Toc57624842)

[Du côté du code 101](#_Toc57624843)

[Aller plus loin : Alternative au diagramme de dispersion 101](#_Toc57624844)

[Aller plus loin : Propriétés de la covariance empirique 103](#_Toc57624845)

[Analysez deux variables quantitatives par régression linéaire 104](#_Toc57624846)

[Étape préliminaire 104](#_Toc57624847)

[Modélisons ! 105](#_Toc57624848)

[Critiquons ce résultat ! 108](#_Toc57624849)

[Aller plus loin : Analyser la qualité du modèle 109](#_Toc57624850)

[Aller plus loin : Estimation de **a** et **b** 110](#_Toc57624851)

[Analysez une variable quantitative et une qualitative par ANOVA 113](#_Toc57624852)

[Quelles questions se poser ? 113](#_Toc57624853)

[Des graphiques ! 113](#_Toc57624854)

[Modélisons 115](#_Toc57624855)

[Évaluation du modèle : les variables sont-elles corrélées ? 115](#_Toc57624856)

[Aller plus loin : Les expressions de SCT, SCE et SCR 117](#_Toc57624857)

[Analysez deux variables qualitatives avec le Chi-2 119](#_Toc57624858)

[Quelles questions nous posons-nous ? 119](#_Toc57624859)

[Représentation 119](#_Toc57624860)

[Résultat du code ci-dessus : le tableau de contingence 120](#_Toc57624861)

[Des statistiques ! 120](#_Toc57624862)

[Aller plus loin : "C'est un peu corrélé mais pas trop...!" 124](#_Toc57624863)

[Entraînez-vous à réaliser des analyses bivariées 125](#_Toc57624864)

[À vous de jouer 125](#_Toc57624865)

[Questions 128](#_Toc57624866)

[Traduction du code en R 131](#_Toc57624867)

[Repérez les différents types d'erreurs 133](#_Toc57624868)

[D'où proviennent les erreurs ? 133](#_Toc57624869)

[Les différents types d'erreurs 133](#_Toc57624870)

[Que faire de toutes ces erreurs ? 134](#_Toc57624871)

[Aller plus loin : Les dates 135](#_Toc57624872)

[Aller plus loin : Ressources externes 136](#_Toc57624873)

[Traitez les valeurs manquantes, les outliers et les doublons 137](#_Toc57624874)

[Les valeurs manquantes 137](#_Toc57624875)

[Les outliers 139](#_Toc57624876)

[Et les doublons ? 139](#_Toc57624877)

[Aller plus loin : Conséquence de la suppression d'individus 140](#_Toc57624878)

[TP : Nettoyez votre jeu de données 142](#_Toc57624879)

[Les méthodes apply et map 142](#_Toc57624880)

[Attaquons ! 144](#_Toc57624881)

[Appliquons toutes ces fonctions 147](#_Toc57624882)

[Aller plus loin : Les compréhensions de listes 148](#_Toc57624883)

[Aller plus loin : le traitement des tailles 148](#_Toc57624884)

[Nettoyez votre échantillon 150](#_Toc57624885)

[Compétences évaluées 150](#_Toc57624886)

[ Question 1 150](#_Toc57624887)

[ Question 2 150](#_Toc57624888)

[ Question 3 150](#_Toc57624889)

[ Question 4 152](#_Toc57624890)

[ Question 5 152](#_Toc57624891)

Quel est le point commun entre un biologiste, un journaliste et un analyste marketing ? Ils communiquent tous à l’aide de graphiques, de moyennes ou de pourcentages, bref : ils font des statistiques !

Avoir des données à disposition, c'est bien, mais savoir en tirer les informations principales, c'est essentiel. Les données sont l'or noir de notre époque : on en analyse des quantités astronomiques chaque seconde dans le monde grâce à des algorithmes. Vous avez probablement déjà entendu parler de Big Data ou de Data Science.

Dans ce cours, vous apprendrez les bases de la Data Science : chercher les informations essentielles dans des données, notamment à l'aide de graphiques. Vous avez déjà entendu parler de moyennes, de variances ou d’histogrammes. Nous allons ici redécouvrir tous ces concepts, mais nous irons encore plus loin en recherchant les relations qui existent à l’intérieur de vos données. Mais avant tout, nous verrons comment nettoyer des données : vous verrez en effet que vos données ne seront jamais directement exploitables sans une phase préalable.

Petit bonus : je vous propose d'analyser vos comportements de consommation à partir de vos relevés de compte bancaire, si vous en avez. Sinon, vous pourrez en télécharger un. Vous êtes-vous déjà posé les questions suivantes ?

* Quand vous faites vos courses, à quelle vitesse consommez-vous vos produits ?
* Combien faites-vous de stock ?
* Consommez-vous plus en début ou en fin de mois ? les week-ends ?
* Êtes-vous plus dépensier lorsque vous avez beaucoup d'argent sur votre compte ?
* etc.

Si vous souhaitez vous lancer dans la Data Science, ce cours présente les bases essentielles : il est un très bon point de départ. Si vous êtes un simple curieux, ce cours vous conviendra également.

Quel que soit votre jeu de données, il a forcément quelque chose à vous dire !

**Objectifs pédagogiques :**

* Nettoyer un jeu de données
* Représenter les variables
* Réaliser une analyse univariée
* Réaliser une analyse bivariée

**Prérequis :**

* Avoir quelques notions de mathématiques : multiplication, division, puissance, nombres entiers, nombres réels, équation de droite par fonction affine ( f(x)=ax+b ), coordonnées d'un point dans un graphique à 2 dimensions
* Optionnellement, savoir programmer en langage R ou en langage Python (niveau basique) et savoir manipuler les objets Dataframe (disponibles nativement sous R, ou en Python via la librairie Pandas). Cela vous sera utile pour quelques quiz et activités évaluées. Les cours permettant d'acquérir ces prérequis sont listés dans le premier chapitre de ce cours.

Ce cours fait partie du parcours [Data Analyst](https://openclassrooms.com/paths/data-analyst). Il comporte aussi des compétences pour la formation [architecture big data](https://openclassrooms.com/fr/paths/64-data-architect).

## Tirez parti de ce cours

### L'organisation du cours

Ce cours est-il long ?

Même si les chapitres sont nombreux, ils sont relativement courts.

Allez, disons-le : dès que l'on arrive sur la page d'un chapitre, on le déroule jusqu'en bas pour voir s'il est long ou pas. Ne vous laissez pas décourager, car certains chapitres contiennent beaucoup de parties optionnelles (appelées Aller plus loin) et paraissent donc longs, mais ils ne le sont en fait pas !

De plus, les chapitres sont assez variés. Vous trouverez :

* **2 chapitres** contenant des interviews de Data Analysts (analystes de données)
* **14 chapitres** sur la théorie des statistiques descriptives, et comment l'appliquer en Python et en R
* **5 chapitres** de contextualisation : de la culture générale sur les statistiques
* **2 chapitres** "fiches techniques" : détaillant comment installer R ou Python, et comment préparer vos données

### Qu'allez-vous faire dans ce cours ?

Vous apprendrez à analyser un jeu de données (parties 1 à 3) puis à le nettoyer (partie 4). Il y aura des chapitres plutôt théoriques, sur la théorie des statistiques, et des chapitres pratiques. Ces derniers vous invitent à appliquer les statistiques en langage R ou en Python.

Tout au long de ce cours, je vous propose de vous entraîner sur 2 jeux de données :

1. Un jeu de données de relevés bancaires, qui nous occupera durant 3 parties.
2. Un petit jeu de données plein d'erreurs, qu'il vous faudra nettoyer. Il nous occupera durant 1 partie.

Est-ce que ce sera fun ?

Je l'espère ! J'ai essayé de prendre un angle original dans ce cours : celui de vous faire analyser des données qui viennent ... de vous ! Si vous avez un compte bancaire avec une carte de paiement, et que vous pouvez consulter vos comptes sur internet, alors je vous propose de télécharger vos relevés et de répondre à certaines questions telles que :

* Quand vous faites vos courses, à quelle vitesse consommez-vous vos produits ?
* Combien faites-vous de stock ?
* Consommez-vous plus en début ou en fin de mois ? les week-ends ?
* Etes-vous plus dépensier lorsque vous avez beaucoup d'argent sur votre compte ?
* etc.

Si vous ne possédez pas de compte bancaire avec une carte, vous pourrez télécharger un jeu de données factice.

### Selon votre niveau de motivation

On a tous des attentes et des motivations différentes en arrivant sur un cours Openclassrooms. Alors, en fonction de votre niveau de motivation, voici comment il est possible de suivre ce cours :

1. Motivation minimum : ne lire que les chapitres théoriques, sur les statistiques descriptives
2. Motivation basse : lire les chapitres avec le code informatique
3. Motivation normale : installer R ou Python sur votre ordinateur, et tester les exemples du cours
4. Motivation haute : lire les sections Aller plus loin en bas des chapitres
5. Motivation de super-héros : vous entraîner sur les 2 langages en même temps, R et Python

Pour obtenir le certificat de cours, vous devrez passer et réussir les quiz et les activités de fin de partie (il y en a 4 au total). Il est donc préférable pour cela d'avoir le niveau de motivation 3, c'est-à-dire vous entraîner sur le code R ou Python.

Si vous vous formez au métier de Data Analyst, alors il est fortement conseillé d'avoir le niveau de motivation 4. Le niveau 5 (connaître à la fois R et Python) est conseillé si vous souhaitez devenir [**Data Scientist**](https://openclassrooms.com/fr/paths/164-data-scientist).

A la fin de certains chapitres, il y a des sections Aller plus loin. Ce sont des approfondissements qui sont facultatifs. Ne vous inquiétez pas, ils ne seront pas évalués dans les quiz et les activités de fin de partie. ;)

### Python ou R ?

Python et R sont les deux langages les plus utilisés en analyse de données.

Ce cours est plus orienté vers Python, car les chapitres sont illustrés dans ce langage. Mais si vous préférez R, sachez que tout le code Python de ce cours est traduit en R, vous n'aurez donc aucun mal à suivre !

Si vous n'en maîtrisez aucun, vous aurez du mal à suivre ce cours, à moins que vous ne souhaitiez consulter que les parties théoriques. Pour choisir entre Python ou R, rendez-vous au chapitre intitulé Installez Python ou R.

### Le code informatique et les données

Le code informatique (dans ses versions R et Python) ainsi que les données utilisées tout au long de ce cours sont présentes dans un fichier .zip, au chapitre [Téléchargez les données](https://openclassrooms.com/courses/nettoyez-et-decrivez-votre-jeu-de-donnees/telechargez-les-donnees).

## Installez R ou Python

Rassurez-vous, ce chapitre est plus rapide qu'il n'y paraît ! Pas besoin de tout lire. ;)

### Ce dont vous aurez besoin

Pour ce cours, vous aurez besoin d'utiliser soit Python soit R.

Si vous avez déjà installé R dans le passé, alors vous n'avez rien à faire, sauf si vous souhaitez travailler avec un [notebook](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927891).

Si vous avez déjà installé Python, alors vous aurez besoin des librairies suivantes :

1. Pandas
2. Matplotlib
3. Numpy
4. Scipy

Si cependant vous n'avez jamais installé Python, alors il existe une [solution miracle](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927979), qui installera directement Python, toutes les librairies nécessaires, ainsi que le notebook Jupyter

### Comment s'orienter dans ce vaste chapitre ?

Tout d'abord, faites votre choix entre Python ou R grâce à la section [R ou Python: lequel choisir ?](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927886)

Ensuite, choisissez si vous souhaitez utiliser un notebook grâce à la section [Notebook or not Notebook ?](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927891)

Une fois ces 2 choix faits, orientez-vous vers l'une de ces 4 sections :

* [Installer R sans notebook](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927944)
* [Installer R avec notebook](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927955)
* [Installer Python sans notebook](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927981)
* [Installer Python avec notebook](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927984)

Si vous rencontrez une erreur d'installation, rendez-vous à la section [Erreurs fréquentes](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927999).

### R ou Python: lequel choisir ?

R et Python sont les deux langages les plus réputés pour l'analyse et le traitement des données. Maîtriser l'un de ces langages, c'est s'armer d'une compétence indispensable dans le domaine de la data science. En plus, ces deux langages se développent depuis de nombreuses années... Ils sont donc fiables et soutenus par de vastes communautés d'utilisateurs prêts à vous aider !

Si vous souhaitez faire un choix informé, nous vous conseillons de consulter ces ressources (en anglais) :

* [Différences entre les deux langages](https://www.quora.com/Which-is-better-for-data-analysis-R-or-Python-Is-R-still-a-better-data-analysis-language-than-Python-Has-anyone-else-used-Python-with-Pandas-to-a-large-extent-in-data-analysis-projects)
* [Complémentarités entre R et Python](http://www.kdnuggets.com/2016/03/r-python-learning-both-datacamp.html)

Pour faire court, disons que R est le plus utilisé parmi les Data Analysts et Python parmi les [Data Scientists](https://openclassrooms.com/fr/paths/164-data-scientist). R est un langage très spécifique aux statistiques et y est vraiment très adapté, alors que Python est très généraliste : il permet également de [créer des applications](https://openclassrooms.com/paths/developpeur-se-dapplication-python), des sites webs, etc. Python est plus agréable à utiliser (en terme de simplicité et de lisibilité du code), mais il n'est pas directement fourni avec les librairies utilisées en Data Science : il faut les installer. L'avantage avec R, c'est que vous pouvez directement manipuler vos données via la structure du dataframe, qui est native du langage.

### Notebook or not Notebook ?

Sans notebook, vous devrez écrire votre programme R ou Python dans un fichier, l'enregistrer, puis le lancer à l'aide de la console avec la ligne de commande  python monfichier.py  ou  Rscript monfichier.R.

Cependant, travailler de cette manière sera fastidieux. En effet, si votre programme affiche des graphiques (ce que nous ferons beaucoup dans ce cours !), ils s'afficheront soit tous en même temps dans des fenêtres séparées, soit ils s'afficheront les uns après les autres au fur et à mesure que vous fermez les fenêtres graphiques. De plus, il sera difficile de revenir en arrière dans votre programme sans tout relancer depuis le début.

Un notebook est la solution idéale. Déjà, il est plus convivial car vous tapez votre code directement dans un navigateur web (type Firefox, Chrome, Safari, Spartan, feu Internet Explorer, etc.). Vous pourrez facilement revenir en arrière en exécutant les lignes de code que vous souhaitez. Enfin, les graphiques s'affichent dans le navigateur, et c'est bien plus pratique ! Vous pouvez essayer le notebook Jupyter [ici](https://try.jupyter.org/), ou avoir un aperçu de son utilisation dans la vidéo de ce [cours](https://openclassrooms.com/courses/evaluez-et-ameliorez-les-performances-d-un-modele-de-machine-learning/tp-selectionnez-le-nombre-de-voisins-dans-un-knn).

### Installer R sans notebook

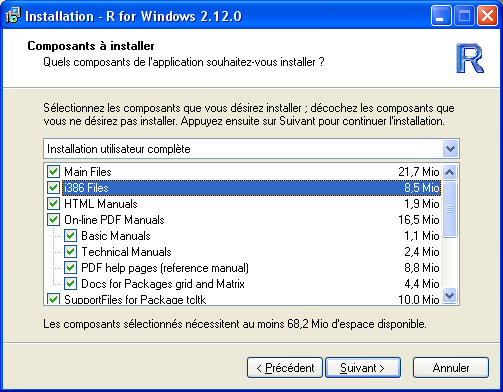
R est disponible sous différentes plateformes : Windows, Mac OS et Linux.

Rendez vous sur la [page de téléchargement de R](https://cran.r-project.org/mirrors.html) (page en anglais) pour sélectionner un des miroirs de téléchargement.

Un miroir est une copie du site située en un lieu différent. Afin de bénéficier de la meilleure vitesse de téléchargement, il est préférable de sélectionner un miroir situé à proximité de chez vous. Quelque soit le lien choisi, le site et son contenu seront identiques.

#### Installation sous Windows

Une fois votre miroir choisi, cliquez sur le lien Download R for Windows puis sur le lien base. En haut de page apparaît alors un lien Download R x.x.x for Windows (où x.x.x représente le numéro de version), cliquez dessus pour le télécharger.  
Une fois le téléchargement fini, lancez le programme d'installation et suivez les instructions données par les différentes boîtes de dialogue vous permettant, par exemple, de choisir la langue du protocole d'installation ou le dossier dans lequel sera installé R.  
Il vous est alors demandé de choisir le type d'installation. L'installation complète ne demande qu'une soixantaine de Mo d'espace, il est donc conseillé de la sélectionner comme illustré ci dessous.



(Pour l'installation complète, assurez-vous d'avoir sélectionné Installation utilisateur complète).

Si vous disposez de très peu d'espace sur votre machine, vous pouvez vous contenter de l'option par défaut. L'installation minimale est quant à elle peu recommandée dans la mesure où elle n'installe pas certains fichiers d'aide. Enfin, si vous pensez comprendre tous les éléments indiqués dans la liste, vous pouvez essayer d'effectuer une installation personnalisée mais cette fonctionnalité est plutôt réservée aux utilisateurs avancés.

Une fois cette étape réalisée, continuer à suivre les indications des boîtes de dialogue en gardant les options par défaut. Après quelques clics, vous aurez installé R, félicitations. ;)

Testez maintenant si cela a fonctionné, en continuant à la section [Testez R](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927928).

#### Installation sous Ubuntu

Sous Ubuntu ou autre distribution dérivée de Debian, vous pouvez trouver R via apt-get ou synaptic. Le paquet s'appelle r-base et vous pouvez donc l'installer en lançant la ligne de commande suivante :

sudo apt-get install r-base

Si vous disposez d'un autre type de distribution ou que vous souhaitez compiler directement les sources, choisissez le lien Linux sur la page de téléchargement du miroir que vous aurez choisi et sélectionnez le dossier correspondant à votre distribution pour télécharger les sources. Les instructions d'installation se trouvent alors dans un fichier d'aide présent dans le dossier téléchargé.

Sous Ubuntu, il est possible que la version de R que vous installez grâce à la ligne de commande ci-dessus ne soit pas la dernière. Ce n'est pas grave, mais pour remédier à cela, suivez [**ces instructions**](https://cran.r-project.org/bin/linux/ubuntu/).

Testez maintenant si cela a fonctionné, en continuant à la section [Testez R](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927928). Si ce n'est pas le cas, rendez-vous sur [cette page](https://cran.r-project.org/bin/linux/), sélectionnez votre distribution Linux, puis consultez le fichier README correspondant.

#### Installation sous une autre distribution Linux

Rendez-vous sur [cette page](https://cran.r-project.org/bin/linux/), sélectionnez votre distribution Linux, puis consultez le fichier README correspondant. Ensuite, [testez R](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927928).

#### Installation sous Mac OS X

Une fois le miroir choisi, cliquez sur Download R for (Mac) OS X , puis dans la section Files, cliquez sur le premier lien proposé (du type R-x.xx.x.pkg où x.xx.x représente le numéro de version).  
Lancez alors le fichier téléchargé en prenant soin de vérifier que votre compte dispose des droits nécessaires. Suivez les instructions données par les boîtes de dialogue. Vous n'aurez normalement pas à changer d'option et pouvez donc valider chaque étape.  
Votre mot de passe vous sera normalement demandé lors de l'installation.  
Après quelques secondes ou minutes d'attente, R sera installé.

Testez maintenant si cela a fonctionné, en continuant à la section [Testez R](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927928).

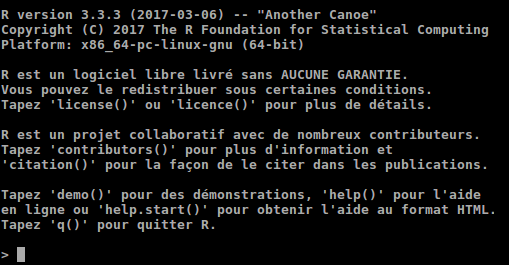
(Merci à [Philippe Julien](https://openclassrooms.com/membres/ptitlu-14857) pour la rédaction de cette partie)

#### Testez R

Pour tester R, ouvrez une console (si vous ne savez pas comment faire, c'est [par ici](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927997)), et tapez-y tout simplement :

R

... puis validez grâce à la touche Enter de votre clavier. Il devrait normalement s'afficher quelque chose qui ressemble vaguement à cela :



Si c'est le cas, c'est que vous êtes en train d'interagir avec R ! Tapez maintenant ceci :

3 + 2

Validez grâce à la touche Entrer. Si R vous répond  5, c'est que tout est bon ! :D

Si par contre il vous répond  6, c'est qu'il déconne totalement, mais cela ne devrait pas arriver ;) (cette phrase est une boutade bien entendu).

Pour quitter R, il faut taper  q(), ou utiliser le raccourci clavier Ctrl + D. Il faudra alors répondre à la question "Save workspace image?", ce à quoi vous pouvez répondre "y" pour oui, ou "n" pour non, en fonction de si vous souhaitez sauvegarder votre travail (c'est-à-dire toutes les variables que vous avez définies au cours de votre session de travail).

#### Testez R avec un fichier de script

A ce stade, vous devez choisir si vous souhaitez utiliser R avec un notebook ou pas. Si ce n'est pas le cas, il vous faudra écrire votre programme dans un fichier. Mais dans tous les cas, il est indispensable de savoir comment faire :

* Créez un fichier et nommez-le  monscript.R.
* Ouvrez ce fichier avec un éditeur de texte, écrivez-y print(3+2), puis sauvegardez le fichier.
* Ouvrez [une console](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927997) et placez-vous dans le répertoire (le dossier) dans lequel vous venez de créer votre fichier.
* Dans la console, saisissez la commande suivante :

Rscript monscript.R

Normalement, il devrait s'afficher le résultat du calcul 3+2 :

[1] 5

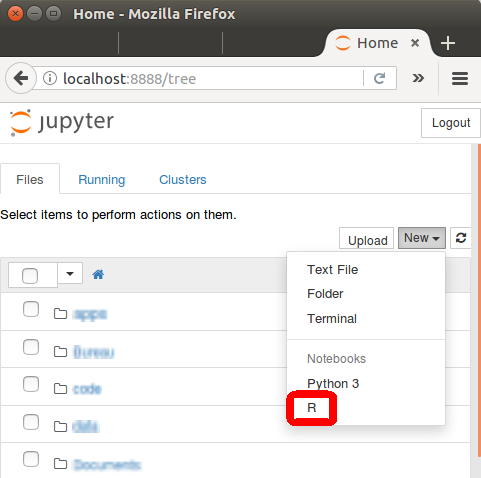
### Installer R avec notebook

Il vous faudra tout d'abord... [installer R sans notebook](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927944), allez hop hop hop, on remonte de quelques lignes !

Ensuite, il vous faudra installer Jupyter (le comble, c'est que Jupyter est codé en Python, alors il vous faudra aussi installer Python !). Pour installer Jupyter, rendez-vous à la section [Installer Python avec notebook](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927984).

Ensuite, rendez-vous sur cette page : <https://irkernel.github.io/installation/>, et suivez les instructions. Il s'agit de lignes de code à taper dans R.

Pour vérifier si cela a fonctionné, [lancez Jupyter](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4929093). Une fois sur la page d'accueil de Jupyter (à laquelle vous pouvez accéder depuis un navigateur en saisissant l'url <http://localhost:8888/tree>), en haut à droite, il y a un bouton "New" : si vous cliquez dessus, une option "R" devrait vous être proposée. Si vous cliquez dessus, un nouveau notebook R sera créé :



Tester le notebook Jupyter avec R

Une alternative au notebook R est le logiciel **[RStudio](https://www.rstudio.com/" \t "_blank)**, qui est **très** populaire parmi les utilisateurs de R. Il est d'ailleurs plus utilisé que le notebook Jupyter R.

### Installer la totale : Python + librairies + notebook (distribution Anaconda)

Si vous n'avez jamais installé Python, alors autant installer la distribution Anaconda.

En gros, une distribution, c'est un langage de programmation + certaines librairies et autres fonctionnalités.

Anaconda est donc une distribution Python, faite pour la Data Science.

Il installera donc :

* Python
* les librairies de Data Science dont nous aurons besoin : Matplotlib, Scipy, Numpy, Pandas
* le [notebook Jupyter](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927891), que je vous conseille vivement d'utiliser

Pour télécharger la distribution Anaconda, c'est par ici : <https://www.anaconda.com/download/>

Les instructions d'installation se trouvent sur cette page : [https://docs.anaconda.com/anaconda/install](https://docs.anaconda.com/anaconda/install/)

Pour tester si l’installation s'est bien déroulée, vous pouvez :

1. [Tester Python](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4929113)
2. [Tester le notebook Jupyter](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4929093)

### Installer Python sans notebook

Pour savoir si python est déjà installé sur votre ordinateur, ou pour le tester une fois que vous l'aurez installé, [rendez-vous ici](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4929113).

Normalement, il est préférable d'installer toutes les fonctionnalités d'un coup : Python, les librairies que nous utiliserons, et le notebook Jupyter. Je vous conseille donc de vous rendre [à cette section](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927979). Si pour une quelconque raison, vous souhaitez tout installer séparément, alors vous trouverez [dans ce cours](https://openclassrooms.com/courses/demarrez-votre-projet-avec-python/installez-python) les instructions pour installer Python. :)

### Installer Python avec notebook

Pour installer python avec le notebook Jupyter, il est fortement recommandé d'installer la distribution Anaconda. Pour cela, consultez la section [**Installer la totale : Python + librairies + notebook**](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927979).

Si vous ne souhaitez pas installer Anaconda, vous pouvez suivre les instructions suivantes, après avoir  [installé Python](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927981) :

Les instructions suivantes proviennent de cette page :[**http://jupyter.org/install.html**](https://jupyter.org/install.html)

Assurez-vous que le programme  pip  est installé sur votre ordinateur. Pou cela, tapez tout simplement  pip  dans une [console](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927997). Normalement, le [programme pip](https://docs.python.org/fr/3.5/installing/index.html) s'est installé en même temps que Python.

Tapez ensuite ces lignes de code l'une après l'autre :

python -m pip install --upgrade pip

python -m pip install jupyter

Pour vérifier si l'installation s'est bien déroulée, rendez-vous à la section [Lancer et tester Jupyter](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4929093).

### Tester si Python est installé

Pour tester Python, ouvrez une console (si vous ne savez pas comment faire, c'est [par ici](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927997)), et tapez-y tout simplement :

Python

... puis validez grâce à la touche Enter de votre clavier. Il devrait normalement s'afficher quelque chose qui ressemble vaguement à cela :



Si c'est le cas, c'est que vous êtes en train d'interagir avec Python ! Tapez maintenant ceci :

3 + 2

Validez grâce à la touche Entrer. Si Python vous répond  5, c'est que tout est bon !

Pour quitter Python, il faut utiliser le raccourci clavier Ctrl + D.

#### Tester Python avec un fichier de script

A ce stade, vous devez choisir si vous souhaitez utiliser Python avec un [notebook](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927891) ou pas. Si ce n'est pas le cas, il vous faudra écrire votre programme dans un fichier. Mais dans tous les cas, il est indispensable de savoir comment faire :

* Créez un fichier et nommez-le  monscript.py  .
* Ouvrez ce fichier avec un éditeur de texte, écrivez-y  print(3+2)  , puis sauvegardez le fichier.
* Ouvrez [une console](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927997) et placez-vous dans le répertoire (le dossier) dans lequel vous venez de créer votre fichier.
* Dans la console, saisissez la commande suivante :

python monscript.py

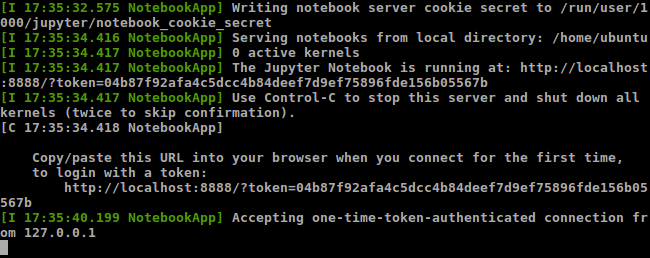
Normalement, il devrait s'afficher le résultat du calcul 3+2, c'est-à-dire 5.

### Lancer et tester Jupyter

Pour lancer Jupyter, ouvrez une [console](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927997), puis tapez-y ceci :

jupyter notebook

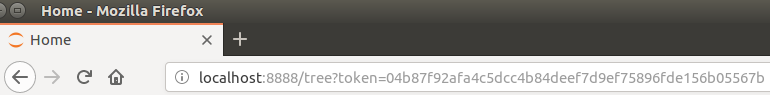
Il devrait s'afficher quelque chose qui ressemble à ceci :



Normalement, votre navigateur (Firefox, Chrome, Safari, Internet Explorer, Edge, etc...) devrait se lancer automatiquement.

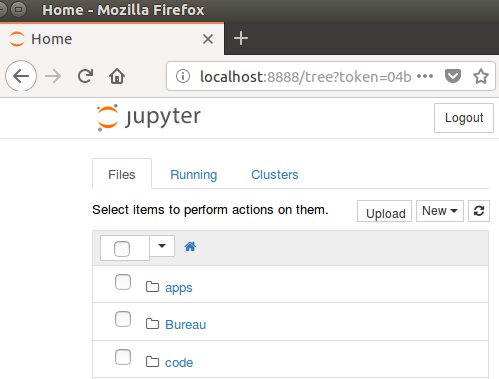
Si ce n'est pas le cas, lancez-le vous-même (si vous lisez cette page en ligne, votre navigateur internet est déjà lancé ! ;) ) et ouvrez l'URL donnée dans votre console: sur la capture d'écran ci-dessus, il s'agit de

http://localhost:8888/?token=04b87f92afa4c5dcc4b84deef7d9ef75896fde156b05567b



Il est possible qu'un mot de passe vous soit demandé. Celui-ci est aussi donné dans la console : 04b87f92afa4c5dcc4b84deef7d9ef75896fde156b05567b (sur la capture d'écran ci-dessus)

Dans votre navigateur devrait s'afficher la liste des dossiers et fichiers présents dans le répertoire à partir duquel vous avez lancé le programme Jupyter :



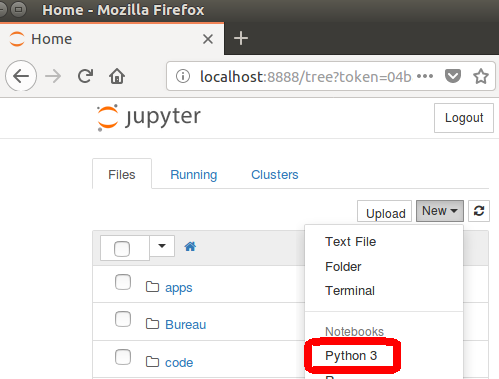
Lancer Jupyter

Bravo ! Vous avez lancé le programme Jupyter. Pour tester les fonctionnalités du notebook avec Python, poursuivez votre lecture ci-dessous, en créant un nouveau notebook.

Pour interrompre Jupyter , placez-vous dans la console à partir de laquelle vous avez lancé Jupyter, et saisissez le raccourci-clavier Ctrl + C.

#### Créer un nouveau notebook

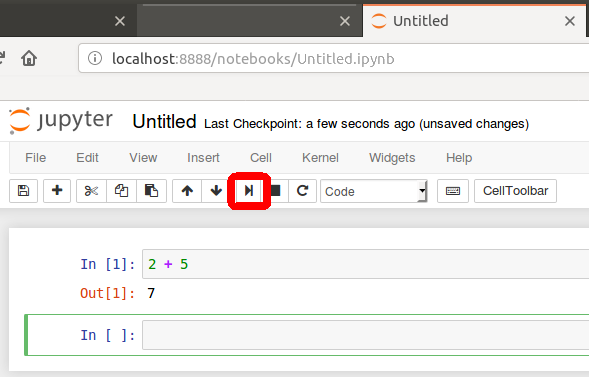
A partir de la fenêtre principale de Jupyter, cliquez sur "New" puis sur "Python 3", comme ceci :



Créer un notebook Python

Un fichier nommé  Untitled.ipynb  s'est normalement créé dans le répertoire à partir duquel vous avez lancé Jupyter.

Votre navigateur devrait vous afficher le notebook créé. Pour le tester, tapez 2 + 5  dans la case (cellule) vide au centre de la fenêtre. Cliquez ensuite sur ce bouton :



Exécuter une cellule du notebook

Il devrait s'afficher le résultat de l'opération : 7.

#### Lancer un notebook déjà créé

Une fois votre notebook créé, vous pouvez y ré-accéder à tout instant en ouvrant une [console](https://openclassrooms.com/fr/courses/4525266-decrivez-et-nettoyez-votre-jeu-de-donnees/4927821-installez-r-ou-python#r-4927997), en vous plaçant dans le répertoire qui contient votre notebook (que vous appelons ici  Untitled.ipynb  ) puis en saisissant dans la console cette ligne :

jupyter notebook Untitled.ipynb

Normalement, votre navigateur devrait s'ouvrir automatiquement et afficher votre notebook !

### Ouvrir une console

#### Sous Windows

Sous Windows, la console s'appelle "Invite de commandes". Les dernières versions de Windows étant très différentes, je vous invite à consulter cette page pour savoir comment lancer la console : <https://fr.wikihow.com/ouvrir-l%27Invite-de-commandes-sous-Windows>

##### Changer de répertoire dans une console Windows

Quand votre console s'ouvre, vous êtes situé dans un certain dossier (ou "répertoire"). On l'appelle "Répertoire (ou dossier) courant". Pour savoir quel est votre répertoire courant, tapez  cd  dans la console. Pour changer de répertoire, tapez  cd Documents  pour vous déplacer dans le dossier "Documents" par exemple. Bien entendu, remplacez "Documents" par le dossier dans lequel vous souhaitez vous déplacer. Si vous ne savez pas par où vous déplacer, tapez  dir  , qui affichera tous des dossiers et tous les fichiers présents dans le dossier courant. Pour "remonter" d'un répertoire en arrière (appelé "Dossier parent"), tapez  cd ..  ("cd" suivi de 2 points). Voilà, vous savez l'essentiel !

#### Sous Linux

Sous Ubuntu, tapez ce raccourci clavier : Ctrl + Alt + T

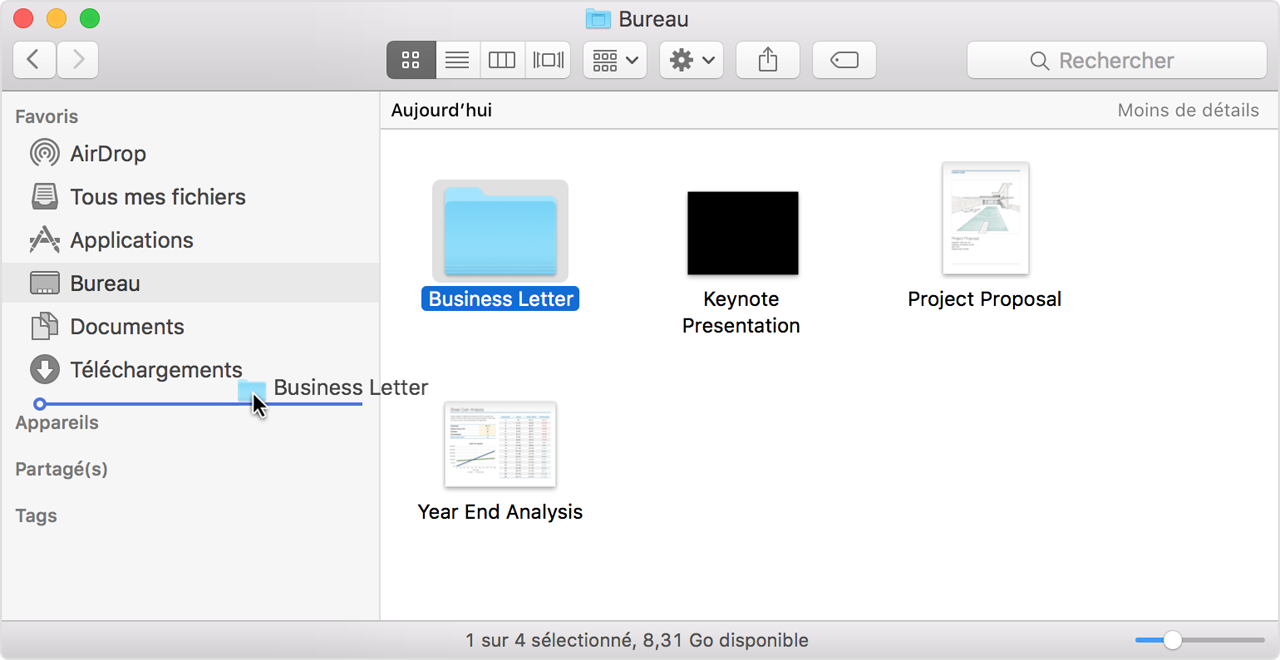
##### Changer de répertoire dans une console Linux

Quand votre console s'ouvre, vous êtes situé dans un certain dossier (ou "répertoire"). On l'appelle "Répertoire (ou dossier) courant". Pour savoir quel est votre répertoire courant, tapez  pwd  dans la console. Pour changer de répertoire, tapez  cd Documents  pour vous déplacer dans le dossier "Documents" par exemple. Bien entendu, remplacez "Documents" par le dossier dans lequel vous souhaitez vous déplacer. Si vous ne savez pas par où vous déplacer, tapez  ls  , qui affichera tous des dossiers et tous les fichiers présents dans le dossier courant. Pour "remonter" d'un répertoire en arrière (appelé "Dossier parent"), tapez  cd ..  ("cd" suivi de 2 points). Voilà, vous savez l'essentiel !

#### Sous Mac

Lancez le Finder grâce à cette icône :**

Une fenêtre similaire à celle-ci devrait s'ouvrir :



(source : support.apple.com)

Cliquez ensuite sur "Applications" disponible dans la partie gauche de la fenêtre, puis sur "Utilitaires" (ou "Utilities" en anglais) disponible dans la partie centrale. Dans le dossier Utilitaires devrait se trouver un programme nommé "Terminal". lancez-le, vous êtes dans votre console !

##### Changer de répertoire dans ne console Mac

Quand votre console s'ouvre, vous êtes situé dans un certain dossier (ou "répertoire"). On l'appelle "Répertoire (ou dossier) courant". Pour savoir quel est votre répertoire courant, tapez  pwd  dans la console. Pour changer de répertoire, tapez  cd Documents  pour vous déplacer dans le dossier "Documents" par exemple. Bien entendu, remplacez "Documents" par le dossier dans lequel vous souhaitez vous déplacer. Si vous ne savez pas par où vous déplacer, tapez  ls  , qui affichera tous des dossiers et tous les fichiers présents dans le dossier courant. Pour "remonter" d'un répertoire en arrière (appelé "Dossier parent"), tapez  cd ..  ("cd" suivi de 2 points). Voilà, vous savez l'essentiel !

### Erreurs fréquentes

Aucune erreur fréquente recensée à ce jour.

## Découvrez les statistiques : vocabulaire et tour d’horizon

Bon, l’heure est venue de savoir de quoi on parle ! Nous définirons dans ce chapitre les concepts clés du cours, et si vous souhaitez plonger dans le vaste monde de la Data Science, je vous propose un petit tour d’horizon du domaine des statistiques, c’est cadeau.

### Le vocabulaire

En statistiques, on étudie des trucs, des bidules et des choses.

Super. Merci pour l’info ! Mais encore ?

Je détaille ! Ces « choses », on les appelle des **individus**. Ces individus peuvent être des objets, des personnes, des animaux, des mesures physiques, etc. L’individu, c’est l’unité d’observation.

Des individus ont des caractéristiques : on les appelle des caractères, ou des **variables**.

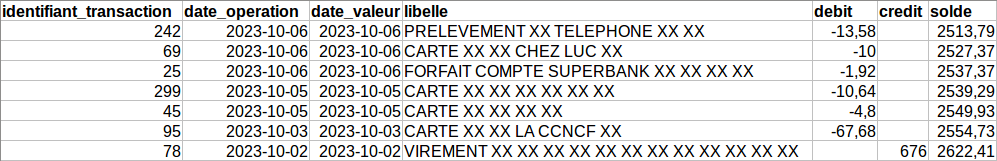
L’ensemble des individus s’appelle la **population**. On note souvent sa taille N, correspondant au nombre d’individus de la population. Il est très fréquent de ne pas connaître la taille exacte d'une population.

Lorsque l’on sélectionne certains individus d’une population, on obtient un **échantillon**. Sa taille est souvent notée n.

On utilisera souvent le terme de **jeu de données**, (ou **data set** en anglais). Ces termes n'ont pas de définition très précise, mais dans ce cours, ils seront équivalents à **échantillon**.

### Comment représenter un échantillon ?

On représente en général un échantillon sous forme de tableau, où chaque ligne correspond à un individu, et chaque colonne représente une variable. Cette représentation est à l’origine du format de fichier [CSV](https://en.wikipedia.org/wiki/Comma-separated_values) (comma separated values). Ce format peut être ouvert avec les logiciels tableurs (Microsoft® Excel, OpenOffice Calc), et est facilement interprétable par les langages R et Python.

Représentation d'un échantillon

Cette représentation est très similaire à [**celle des bases de données relationnelles**](https://openclassrooms.com/courses/initiez-vous-a-lalgebre-relationnelle-avec-le-langage-sql/decouvrez-le-concept-de-relation).

### Petit tour d’horizon des statistiques

#### Statistiques et probabilités

Les statistiques et les probabilités, c’est la même chose non ?

Eh bien… non ! Certes, ces deux domaines sont étroitement liés, mais ils sont distincts. Quand on ne fait qu’observer et décrire objectivement un phénomène, alors on fait des **statistiques**.

Mais dès lors que l'on modélise, on fait le lien entre ce qu'on observe et le domaine théorique que constituent les **probabilités** : on passe alors dans le domaine de la statistique dite inférentielle.

En statistiques, les données que l'on observe sont appelées **observations**, ou parfois **réalisations**. A partir de ces observations, on peut modéliser. Modéliser, c'est essayer de trouver les lois mathématiques qui régissent les données observées. Dans le domaine des probabilités, on manipule des variables aléatoires, des lois de probabilité, etc.

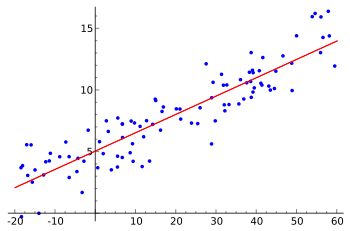
Si vous étudiez la proportion femmes/hommes d'un pays, vous sélectionnez un échantillon dans lequel vous observez ces proportions : par exemple 55% de femmes et 45% d'hommes. Ce sont des statistiques. Mais si vous dites ensuite dans ce pays, un enfant qui naît a une probabilité de 55% d’être une fille, alors vous faites des probabilités !

Dans ce cours, pas de probas : que des stats !

#### Les différents domaines de la statistique

##### Les statistiques descriptives

C’est le sujet ce cours ! Il s’agit de présenter, décrire et résumer le jeu de données, à l’aide de graphiques et de mesures (moyenne, écart-type, etc.). En statistique descriptive, chaque graphique (ou chaque mesure) est calculé(e) sur 1 ou 2 variables à la fois, pas plus. Pourquoi pas plus ? Parce que représenter les relations entre 2 variables sur un graphique est assez simple sur du papier ou sur un écran, car ceux-ci sont en 2 dimensions (longueur-largeur).



Graphique en 2 dimensions avec 1 axe horizontal et 1 axe vertical (source: Wikipedia)

##### L'analyse de données

L’analyse exploratoire de données est le prolongement des statistiques descriptives, sauf que là, on étudie plutôt les relations entre 3 variables ou plus. Représenter des graphiques avec 3, 4, 5 ou 100 dimensions n’est plus possible sur du papier à 2 dimensions. Il faut donc utiliser des techniques spéciales pour continuer à [décrire et explorer les données](https://openclassrooms.com/courses/explorez-vos-donnees-avec-des-algorithmes-non-supervises).

Le terme anglophone Data Analysis n'est pas l'équivalent du terme français analyse de données. Data Analysis est un terme beaucoup plus large qui englobe les statistiques descriptives, le nettoyage et la transformation des données, la modélisation, etc.

##### Les statistiques inférentielles

Ici, il s’agit d’analyser les données d’un sous-ensemble d’une population pour en déduire les caractéristiques globales de la population. Si vous entendez un jour parler d'**estimateurs** ou de **tests statistiques**, il s'agira de statistiques inférentielles.

##### La modélisation statistique

Il s’agit d’observer les caractéristiques d’un échantillon, puis de formaliser ces observations par des règles mathématiques. Cette formalisation s’appelle un modèle probabiliste. Une fois que l'on a décrit un phénomène par un modèle, on peut faire de la prédiction ou de la prévision.

### Aller plus loin : Plus de cours !

Sur OpenClassrooms, vous retrouverez des cours relatifs aux domaines vus ci-dessus, dans le parcours [Data Analyst](https://openclassrooms.com/paths/data-analyst).

### Aller plus loin : Data Analyst vs Data Scientist

Mais quelle est la différence entre un [**Data Scientist**](https://openclassrooms.com/fr/paths/164-data-scientist) et un [**Data Analyst**](https://openclassrooms.com/fr/paths/data-analyst) ?

La frontière entre ces deux métiers est assez floue, mais on peut dire que le Data Analyst pratique les statistiques descriptives, exploratoires et inférentielles. Le Data Scientist doit maîtriser ces 3 domaines, et doit également être capable de modéliser des phénomènes. Il a à sa disposition une batterie d'algorithmes qui permettent de trouver la modélisation la plus performante pour le problème qu'il doit traiter. Pour plus de précisions, vous pouvez faire un tour [ici](https://openclassrooms.com/courses/initiez-vous-au-machine-learning/comment-resoudre-un-probleme-de-data-science).

## Téléchargez les données

Ne fuyez pas devant l’apparente longueur de ce chapitre : beaucoup de cas particuliers sont traités, mais la réalisation du programme reste très rapide ! Vous avez même la possibilité de sauter ce chapitre en téléchargeant le fichier final appelé operations\_enrichies.csv (il est présent dans le dossier analyse de l'archive téléchargeable ci dessous).

Le code informatique (dans ses versions R et Python), ainsi que les données (au format CSV) utilisées tout au long de ce cours sont regroupées dans une archive .zip disponible [**ici**](https://course.oc-static.com/courses/4525266/code.zip). Il vous faudra décompresser l'archive afin de pouvoir utiliser les données.

### Les données et le code informatique

Dans l'archive que vous venez de télécharger, il y a 2 dossiers. Le dossier analyse vous sera utile dans les 3 premières parties du cours, et le dossier nettoyage dans la dernière partie.

Il est composé de fichiers au format :

* **.csv** : ce sont les données que nous utiliserons pour illustrer ce cours
* **.py** : c'est le code informatique cité tout au long de ce cours, en langage Python.
* **.r** : il s'agit du même code que dans les fichiers .py, mais traduits en langage R
* **.ipynb** : il s'agit encore du même code, mais cette fois-ci dans le format du notebook Jupyter. Ces notebooks sont utilisables soit en R soit en Python. Pour savoir comment les exécuter, retournez [dans la section Lancez un notebook déjà créé du chapitre Installez R ou Python](https://openclassrooms.com/courses/nettoyez-et-decrivez-votre-jeu-de-donnees/installez-r-ou-python-1#r-4929708) ! ;)

### C'est parti !

Pour que ce cours soit plus attrayant, je vous propose d'analyser des données que **vous avez générées** ! Si vous possédez un compte bancaire consultable en ligne, et qu'il y a sur celui-ci des opérations bancaires fréquentes (c'est le cas notamment si vous utilisez fréquemment une carte bancaire), alors vous pouvez analyser vos comportements de consommation.

Un relevé bancaire est composé de lignes, que l'on appellera **opération**, **opération bancaire** ou **transaction**. Une opération sera donc un individu.

Je vous propose donc dans ce chapitre de télécharger vos relevés de compte, de les charger dans Python, et d'enrichir votre échantillon de nouvelles variables. Si tout se passe bien, cela ne devrait pas prendre beaucoup de temps. Mais si vous rencontrez des erreurs, il vous sera possible de télécharger un fichier final afin de continuer le cours. ;)

Si vous n'avez pas de relevé de comptes, commencez par ouvrir le fichier operations.csv (il est présent dans l'archive téléchargeable en haut du chapitre, dans le dossier analyse.). Il s'agit d'un faux relevé, pour lequel tous les libellés d'opération ont été partiellement censurés.

### Ce que nous allons faire

Tous les relevés bancaires ont en commun (pour chaque opération) 3 informations :

* la date de l'opération
* le libellé de l'opération
* le montant de l'opération

A partir de ces 3 informations, nous allons créer plusieurs variables :

* **date\_operation**
* **libelle**
* **montant**
* **sens** : indique si l'opération est un crédit ou un débit
* **solde\_avt\_operation** : elle indique le solde restant avant que l'opération ne soit effectuée
* **categ** : qui indique la catégorie de l'opération, par exemple : "courses", "loyer", "facture", etc.
* **type** : indiquant le type d'opération, exemple : "virement", "paiement par carte", 'retrait", etc.
* **tranche\_depense** : si l'opération est une dépense, indique si celle-ci est petite, moyenne, etc.
* **annee** : l'année, déduite de date\_operation
* **mois** : le mois, déduit de date\_operation
* **jour** : le jour du mois (compris entre 1 et 31)
* **jour\_sem** : le jour de la semaine (lundi, mardi, etc.)
* **jour\_sem\_num** : le numéro du jour de la semaine (compris entre 1 et 7)
* **weekend** : indique si la date d'opération se situe sur un weekend
* **quart\_mois** : vaut 1, 2, 3 ou 4, et indique l'avancée dans le mois (1 : début, ..., 4 : fin de mois)
* **attente** : indique pour chaque opération de catégorie "courses", la durée (en jours) séparant cette opération de la précédente opération de catégorie "courses". Celle-ci sera en fait calculée [plus tard](https://openclassrooms.com/courses/nettoyez-et-decrivez-votre-jeu-de-donnees/analysez-2-variables-quantitatives-par-regression-lineaire), dans le chapitre sur la régression linéaire.

Certaines de ces variables peuvent être calculées automatiquement, mais certaines nécessiterons votre intervention. Par exemple, vous devrez indiquer vous-même les opérations qui sont des courses ou des paiements de loyers. Ne vous inquiétez pas, je vous faciliterai la tâche afin de ne pas faire ceci entièrement "à la main". ;)

### C'est parti !

#### Télécharger les données

En général, si vous avez accès à vos comptes bancaires à partir d'internet, il existe un endroit où vous pouvez télécharger le relevé de vos opérations.

Si vous avez le choix, choisissez le format CSV, et téléchargez le plus d'opérations possibles. Si seuls les formats de fichiers XLS ou ODS vous sont proposés, alors il faudra ouvrir le fichier dans un tableur (Excel ou [OpenOffice Calc](https://fr.wikipedia.org/wiki/Calc_(tableur))), puis l'enregistrer sous un format CSV. Pour plus de précision, cliquez [ici](https://www.youtube.com/watch?v=e_-YYRuFDA4).

Vos relevés de comptes peuvent présenter des informations confidentielles, n'utilisez donc pas de convertisseur XLS -> CSV ou ODS -> CSV en ligne ! ;)

Une fois votre fichier CSV obtenu, placez-le dans le même répertoire (dossier) que celui dans lequel se trouve votre script (Python ou R), ou votre notebook. Ensuite, renommez-le en  operations.csv.

#### Charger vos données dans votre programme

Ouvrez le fichier CSV avec un éditeur de texte (pas avec un tableur !). Sous Windows, utilisez par exemple l'application Bloc-notes, sous Mac Text Edit, sous Linux nano, vim ou gedit.

Votre fichier devrait ressembler à ceci :

identifiant\_transaction,date\_operation,date\_valeur,libelle,debit,credit,solde

242,2023-10-06,2023-10-06,FORFAIT COMPTE SUPERBANK XX XX XX XX,-1.92,,2513.79

69,2023-10-06,2023-10-06,CARTE XX XX CHEZ LUC XX,-10.00,,2515.71

Une fois ouvert, il faut y repérer certaines informations :

* Quelle(s) colonne(s) contiennent des dates ?
* Quel est le format des dates ?
* Pour les nombres, le délimiteur des décimales est-il la virgule  3,24  ou le point 3.24 ?
* Les colonnes sont-elles délimitées par le caractère  ,  ,  ;  ou par une [tabulation](http://www.commentcamarche.net/contents/1709-word-tabulations)\t  ?
* Le fichier contient-il un en-tête ?

Souvent, la toute première ligne indique le nom des colonnes (on l'appelle en-tête ou header), et les suivantes correspondent aux opérations bancaires. Si l'en-tête est absent, ce n'est pas grave. Mais s'il contient des accents, mieux vaut les remplacer (à la main) par des lettres sans accents.

Prêt à coder ? C'est parti ! Ouvrez un notebook ou un script Python, et utilisez la méthode  read\_csv  pour charger votre fichier csv :

Sur la ligne 2, il vous faudra compléter le  [...]  en fonction du format de votre CSV. Continuez à lire, les instructions sont données juste après ;)

import pandas as pd

data = pd.read\_csv("operations.csv", [...] )

print(data)

Ce code est disponible en R et en Python dans l'archive téléchargeable en haut du chapitre, dans le dossier analyse, dans ces différents fichiers : R\_preprocessing et python\_preprocessing.

Pour compléter le  [...]  , voici 3 exemples pour vous inspirer :

data = pd.read\_csv("operations.csv",parse\_dates=[1,2])

data = pd.read\_csv("operations.csv",parse\_dates=[1,2],header = None)

data = pd.read\_csv("operations.csv",parse\_dates=[1,2],sep= ';',

decimal= ',', dayfirst=True)

Bien entendu, ces 3 lignes sont des exemples : vous n'aurez besoin d'utiliser read\_csv  qu'une seule fois, pas 3 !  Voici une aide à propos des différents arguments à ajouter dans  read\_csv  :

* parse\_date  et  dayfirst  : Dans tous les cas, il faudra indiquer quelles sont les colonnes qui représentent des dates grâce à  parse\_date  . Attention, en Python, on indexe à partir de 0 : ainsi, la première colonne du CSV a le numéro 0. Le format des dates sera déduit automatiquement, sauf si les dates sont au format  01/02/2020. Si c'est le cas, il faut spécifier si le  01  indique le jour ou le mois (en Europe, cette date est interprétée comme "1er février 2020", alors qu'aux USA, ce serait "2 janvier 2020"). Si c'est le jour, alors il faut préciser  dayfirst=True, comme en ligne 4 ci-dessus.
* sep  et  decimal  : Si vos colonnes ne sont pas séparées par des virgules, ou si le délimiteur des décimales n'est pas le point, alors il faudra préciser les paramètres  sep  et/ou  decimal, comme en lignes 3 et 4.
* header  : Si votre fichier ne contient pas d'en-tête, alors il faudra préciser  header=None, comme en ligne 4.

Si vous utilisez [**le fichier operations.csv**](https://raw.githubusercontent.com/OCCourses/nettoyez_et_decrivez_votre_jeu_de_donnees/master/analyse/operations.csv) fourni dans l'archive zip, alors il faut utiliser la ligne 1.

Vérifiez que l'import s'est bien passé en affichant la variable data en tapant cette ligne, qui affichera un tableau :

print(data)

Pour utiliser le code qui suivra, nous devons tous avoir les mêmes noms de colonnes dans notre dataframe. Renommons donc nos colonnes :

data.columns = ['identifiant\_transaction','date\_operation','date\_valeur',

'libelle','debit','credit','solde']

Adaptez cette ligne à votre cas, mais assurez-vous bien que les colonnes indiquant la date d'opération, le libellé, le montant des débits et le montant des crédits aient le même nom qu'ici. Si au lieu des colonnes  credit  et  debit, votre fichier ne contient qu'une unique colonne, appelez-la  montant.

#### Détecter les opérations fréquentes

Vous avez probablement vos habitudes de consommation, par exemple, vous fréquentez sûrement les mêmes endroits pour faire vos courses. Dans ce cas, beaucoup d'opérations auront un libellé contenant le nom de votre magasin préféré.

Nous souhaitons ici déterminer des catégories et des types d'opérations à partir des libellés. Pour cela, je vous ai concocté une petite fonction qui repère les mots les plus fréquents parmi vos libellés. Testez-la, vous m'en direz des nouvelles :

from collections import Counter

def most\_common\_words(labels):

words = []

for lab in labels:

words += lab.split(" ")

counter = Counter(words)

for word in counter.most\_common(100):

print(word)

most\_common\_words(data['libelle'].values)

Voici le début de ce qu'elle affiche :

('XX', 1428)

('CARTE', 247)

('VIREMENT', 29)

('LES', 20)

('ANCIENS', 20)

('ROBINSON', 20)

('CHEZ', 16)

('LUC', 16)

Je vois que le mot "carte" est présent 247 fois dans mes relevés. Quand je regarde mes relevés, je vois qu'il s'agit de toutes les opérations qui représentent un paiement par carte bancaire. Nous appellerons le mode de paiement le **type** d'opération.

Aussi, je vois qu'il y a 16 fois les mots "CHEZ" et "LUC". Dans mes relevés, je vois que j'ai mangé 16 fois au restaurant "CHEZ LUC". Ici, nous dirons que le restaurant est la **catégorie** de l'opération.

Faites cet exercice à partir de vos relevés, puis créez deux variables :  TYPE  et  CATEGS, en suivant cet exemple :

CATEGS = {

'LOYER': 'LOYER',

'FORFAIT COMPTE SUPERBANK': 'COTISATION BANCAIRE',

'LES ANCIENS ROBINSON': 'COURSES',

"L'EPICERIE DENBAS": 'COURSES',

'TELEPHONE': 'FACTURE TELEPHONE',

'LA CCNCF': 'TRANSPORT',

'CHEZ LUC': 'RESTAURANT',

'RAPT': 'TRANSPORT',

'TOUPTIPRI': 'COURSES',

'LA LOUVE': 'COURSES',

'VELOC': 'TRANSPORT'

}

TYPES = {

'CARTE': 'CARTE',

'VIR': 'VIREMENT',

'VIREMENT': 'VIREMENT',

'RETRAIT': 'RETRAIT',

'PRLV': 'PRELEVEMENT',

'DON': 'DON',

}

Choisissez vos propres catégories, mais si vous payez vos courses en carte bancaire, essayez d'avoir au moins la catégorie "COURSES".

Attention aux majuscules et minuscules : en effet, le programme cherchera dans le libellé la chaîne de caractères exacte. Si le libellé d'une opération est  paiement 2738 Chez LUC s.a., alors elle ne sera pas classée dans la catégorie restaurant, à cause des minuscules dans "Chez".

#### Dernière étape avant le décollage !

Encore 3 lignes à personnaliser et on pourra lancer le programme :

EXPENSES = [80,200] # Bornes des catégories de dépense : petite, moyenne et grosse

LAST\_BALANCE = 2400 # Solde du compte APRES la dernière opération en date

WEEKEND = ["Saturday","Sunday"] # Jours non travaillés

Tout d'abord, les bornes des catégories de dépenses (ligne 1). Ici, j'ai considéré qu'une dépense inférieure à 80 € est une petite dépense, qu'une dépense de plus de 200 € est une grosse dépense, et qu'une dépense moyenne se situe entre ces 2 bornes. Ces 2 valeurs sont subjectives, alors personnalisez-les ! Elles doivent être exprimées dans la même devise que celle de vos relevés de compte : si vos relevés ne sont pas en euros, il vous faudra probablement changer ces valeurs.

Lorsque vous téléchargez vos relevés, retenez-bien la somme d'argent présente sur votre compte après la dernière opération de votre fichier CSV. Si votre fichier contient la toute dernière opération que vous avez effectuée, alors indiquez dans  LAST\_BALANCE  la somme d'argent actuellement présente sur votre compte.

Indiquez enfin dans  WEEKEND  s'il y a des jours durant lesquels votre quotidien change. Pour quelqu'un qui travaille du lundi au vendredi, le quotidien changera les samedis et dimanches car il ne travaille pas. Vous pouvez laisser cette liste vide en écrivant  WEEKEND = []. Choisissez ces valeurs parmi celles-ci, en respectant les majuscules/minuscules :

['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday']

#### Décollage !

C'est quasi-fini ! Il ne vous reste plus qu'à copier-coller ce code les yeux fermés :

# Controle des colonnes

for c in ['date\_operation','libelle','debit','credit']:

if c not in data.columns:

if (c in ['debit','credit'] and 'montant' not in data.columns) or \

(c not in ['debit','credit']):

msg = "Il vous manque la colonne '{}'. Attention aux majuscules "

msg += "et minuscules dans le nom des colonnes!"

raise Exception(msg.format(c))

# Suppression des colonnes innutiles

for c in data.columns:

if c not in ['date\_operation','libelle','debit','credit','montant']:

del data[c]

# Ajout de la colonne 'montant' si besoin

if 'montant' not in data.columns:

data["debit"] = data["debit"].fillna(0)

data["credit"] = data["credit"].fillna(0)

data["montant"] = data["debit"] + data["credit"]

del data["credit"], data["debit"]

# creation de la variable 'solde\_avt\_ope'

data = data.sort\_values("date\_operation")

amount = data["montant"]

balance = amount.cumsum()

balance = list(balance.values)

last\_val = balance[-1]

balance = [0] + balance[:-1]

balance = balance - last\_val + LAST\_BALANCE

data["solde\_avt\_ope"] = balance

# Assignation des operations a une categorie et a un type

def detect\_words(values, dictionary):

result = []

for lib in values:

operation\_type = "AUTRE"

for word, val in dictionary.items():

if word in lib:

operation\_type = val

result.append(operation\_type)

return result

data["categ"] = detect\_words(data["libelle"], CATEGS)

data["type"] = detect\_words(data["libelle"], TYPES)

# creation des variables 'tranche\_depense' et 'sens'

def expense\_slice(value):

value = -value # Les dépenses sont des nombres négatifs

if value < 0:

return "(pas une dépense)"

elif value < EXPENSES[0]:

return "petite"

elif value < EXPENSES[1]:

return "moyenne"

else:

return "grosse"

data["tranche\_depense"] = data["montant"].map(expense\_slice)

data["sens"] = ["credit" if m > 0 else "debit" for m in data["montant"]]

# Creation des autres variables

data["annee"] = data["date\_operation"].map(lambda d: d.year)

data["mois"] = data["date\_operation"].map(lambda d: d.month)

data["jour"] = data["date\_operation"].map(lambda d: d.day)

data["jour\_sem"] = data["date\_operation"].map(lambda d: d.day\_name)

data["jour\_sem\_num"] = data["date\_operation"].map(lambda d: d.weekday()+1)

data["weekend"] = data["jour\_sem"].isin(WEEKEND)

data["quart\_mois"] = [int((jour-1)\*4/31)+1 for jour in data["jour"]]

# Enregistrement au format CSV

data.to\_csv("operations\_enrichies.csv",index=False)

Ce code supprimera les colonnes dont nous n'avons pas besoin (par exemple date\_valeur), et vérifiera que votre dataframe contient bien les colonnes dont nous aurons besoin (leur nom doit être correctement orthographié). Ensuite, il créera toutes les variables citées tout en haut du chapitre !

Le résultat sera enregistré dans une nouveau fichier CSV :  operations\_enrichies.csv.

Voilà, vous êtes fin prêts à analyser vos comportements de consommation !

### Et si ça a planté...?

Si vous rencontrez des erreurs, vous pourrez utiliser pour les chapitres suivants le fichier [operations\_enrichies.csv](https://raw.githubusercontent.com/OCCourses/nettoyez_et_decrivez_votre_jeu_de_donnees/master/analyse/operations_enrichies.csv) (il est présent dans l'archive téléchargeable en haut du chapitre, dans le dossier analyse) .

## Découvrez les 4 types de variables

Observons les variables que nous venons de créer. Elles ne sont pas toutes de même type. Certaines sont des nombres (la variable "montant"), certaines sont des tranches de nombres (la variable "tranche\_depense"), certaines sont des mots (comme la catégorie d'opération : « loyer », « courses », etc.).

 Mettons un peu d’ordre dans nos esprits !

Il y a 2 types de variables, chacun d’eux est sub-divisé en 2 groupes.

### Les variables quantitatives

Ce sont les variables qui prennent des valeurs **numériques** (des nombres quoi !), à condition que ces valeurs expriment une quantité et aient un sens lorsque l’on y applique des opérations arithmétiques.

Par exemple, si vous additionnez tous les montants des dépenses de votre relevé bancaire, vous saurez combien vous aurez dépensé au total : ceci a un sens. C’est une quantité d’argent. Cependant, l’identifiant d’une opération, bien que numérique, n’est pas une variable quantitative. En effet, effectuer la somme des identifiants de vos opérations n’a aucun sens, l’identifiant ne représente pas une quantité. Je ne devrais pas vous le dire (car vous verrez ceci après) mais l’identifiant est une variable qualitative (mais chuuut, je ne vous ai rien dit!).

Une variable quantitative est soit **discrète**, soit **continue**.

Si le nombre de valeurs possibles (et probables) d'une variable est très grand, alors on peut la considérer comme **continue**. Sinon, on la considère comme **discrète**.

Dans nos relevés de compte, le montant des opérations peut prendre beaucoup de valeurs. Si vous prenez le montant d'une opération au hasard, il a de grandes chances d'être compris entre 0 € et 1 000 €. Entre ces 2 valeurs, il y a 100 000 valeurs possibles : 0,00 €, 0,01 €, 0,02 €, etc.

Dans le monde merveilleux des ordinateurs, une variable n’est jamais vraiment continue. Pour représenter une variable continue, il faudrait pouvoir stocker dans l’ordinateur un nombre avec une infinité de chiffres après la virgule, ce qui n’est pas possible. Notre variable "montant" est discrète (il n’y a pas de valeur possible entre 1,22 € et 1,23 €), mais nous la considérerons quand même comme continue, car on considère qu’un écart de 1 centime, c’est petit !

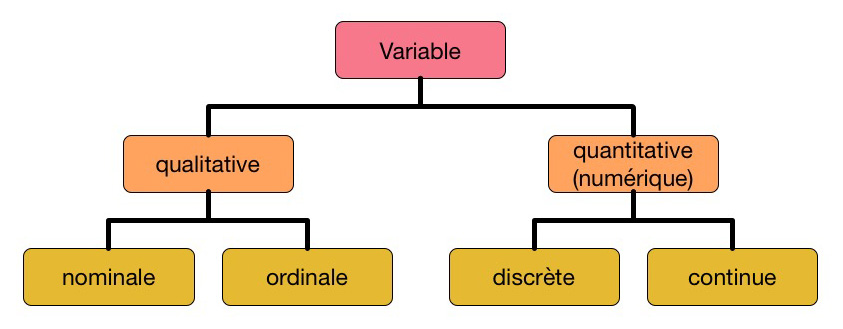
### Les variables qualitatives

Disons que ce sont toutes les variables qui ne sont pas quantitatives ;). Les valeurs qu’elles prennent sont appelées des catégories, ou **modalités**. Ces dernières sont exprimées sous forme littérale (par un mot, une phrase ou un code) ou par un codage numérique sur lequel les opérations arithmétiques n’ont aucun sens.

Une variable qualitative est **nominale** ou **ordinale**.

Une variable est **ordinale** si ses modalités peuvent être ordonnées. La variable "tranche\_depense" est ordinale, car on peut dire qu’une dépense de la tranche « petite dépense » est plus petite qu’une « dépense moyenne », elle-même plus petite qu’une « grosse dépense ». Dans un autre cadre, les mentions attribuées à un examen (moyen, bien, très bien) sont aussi une variable ordinale.

L’identifiant d’une opération est **nominal**, car on ne peut pas dire que l’opération numéro 1 est « inférieure » à l’opération numéro 40 (on suppose ici que les identifiants ne sont pas forcément classés par date d’opération).



Types de variables

Voilà vous savez tout ! :D

Ah si… les variables dites dichotomiques. Ce sont des variables qualitatives qui ne prennent que 2 modalités (0/1, oui/non, true/false). On les appelle souvent variables binaires, ou **booléennes** (abrégé « bool »).

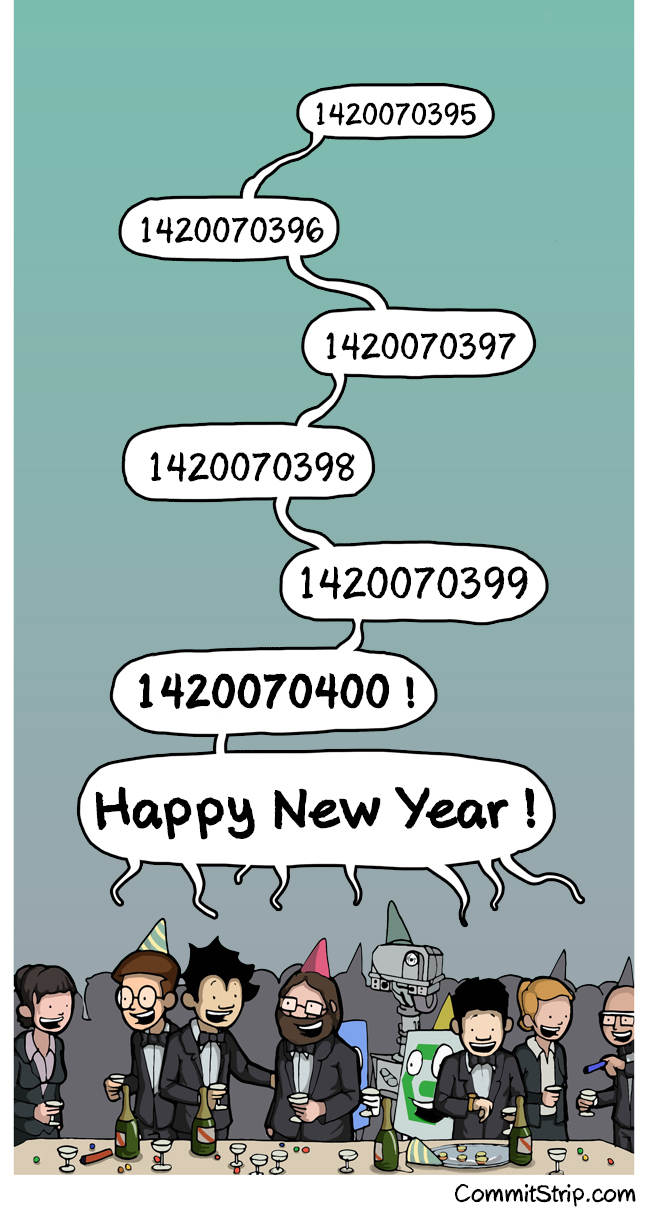
### Aller plus loin : Les dates, quantitatives ou qualitatives ?

… et les dates ?

Dans un ordinateur, les dates sont stockées sous forme de nombre entier, appelé **timestamp**. Il comptabilise le nombre de secondes (ou parfois de millisecondes) écoulées depuis le 1er janvier 1970. Par exemple, la date du 23 septembre 2020 est codée par le timestamp 1600819200. Cependant, additionner des timestamps n'a pas vraiment de sens : une date sera donc considérée comme **qualitative ordinale**.

Si vous rencontrez la date 01/01/1970 dans des données, sachez qu’il s’agit certainement d’une absence d’information pour laquelle le timestamp 0 a été mis par défaut...

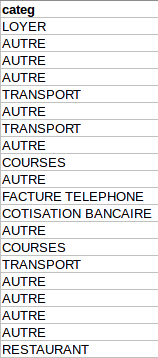
... et si vous rencontrez des timestamps comme sur cette image, sachez les convertir sur [www.epochconverter.com](http://www.epochconverter.com/) !



## Représentez la distribution empirique d'une variable

Comment peut-on représenter une variable ?

Jusqu'à maintenant, nous avons vu comment [afficher un échantillon](https://openclassrooms.com/courses/nettoyez-et-decrivez-votre-jeu-de-donnees/decouvrez-les-statistiques-vocabulaire-et-tour-dhorizon-1) (sous forme de tableau où chaque ligne représente un individu, et chaque colonne une variable). Pour représenter la variable categ par exemple, on pourrait sélectionner la colonne categ du tableau, et l'afficher telle quelle :



Mais il faut avouer que c'est assez illisible ! :waw: En plus, il est fréquent d'avoir des échantillons de 1000 individus ou plus. Une colonne avec 1000 valeurs dedans, c'est très moche et très difficile à interpréter. Il y a une solution bien meilleure, qui consiste à dire :

Il y a 39 fois la valeur COURSES, 212 fois la valeur AUTRE, 21 fois la valeur TRANSPORT, etc.

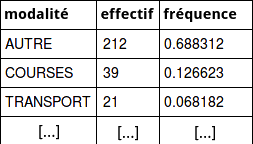
Cette formulation est appelée **distribution empirique**. C'est cette distribution que l'on se propose de représenter graphiquement ici.

### Représenter une distribution empirique

Les différentes "possibilités" que l'on peut observer pour la variable categ sont ses **modalités**. Les modalités de la variable categ sont : courses, transport, autre, loyer, etc. Pour une variable quantitative cependant, on les appelle les **valeurs** possibles. On associe à chaque modalité (ou valeur) un **effectif**. L'effectif de la modalité courses est .

En divisant un effectif par le nombre d'individus de l'échantillon (noté **n**), on obtient une **fréquence**.

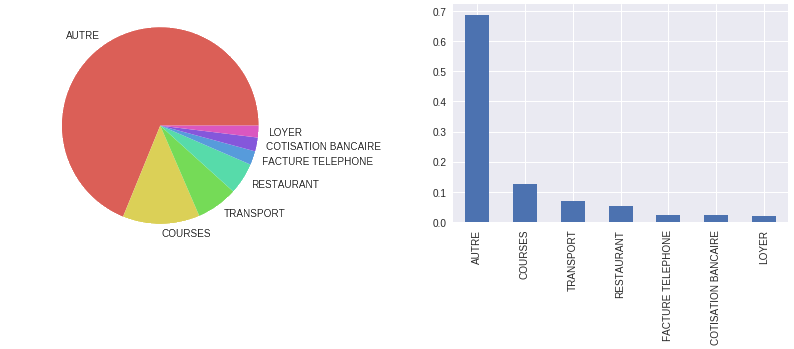
La distribution empirique d'une variable, c'est l’ensemble des valeurs (ou modalités) prises par cette variable, ainsi que leurs effectifs associés. On trouve aussi une autre version : l’ensemble des valeurs (ou modalités) prises par cette variable, ainsi que leurs fréquences associées. On peut présenter ceci sous forme de tableau. Nous approfondirons cette présentation dans le chapitre suivant :



Passons maintenant aux représentations graphiques.

#### Cas des variables qualitatives

Voici 2 représentations possibles de la distribution de la variable categ :



A gauche, vous avez le diagramme en secteurs, plus connu sous le nom de diagramme en camembert. Si les francophones y voient un camembert (fleuron de la gastronomie française), les anglophones y voient plutôt une tarte, et l'appellent donc **pie chart**. Ici, l'angle de chaque secteur est proportionnel à l'effectif de chaque modalité.

A droite, c'est le diagramme en tuyaux d'orgue, appelé en anglais **bar chart**. La hauteur des tuyaux est égale à l'effectif de chaque modalité, ou bien (au choix) égale à la fréquence de chaque modalité comme c'est le cas ici.

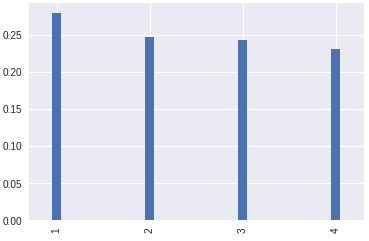
La hauteur des tuyaux représente le nombre (ou la fréquence) d'opérations bancaires d'une catégorie donnée, et non pas la somme de leurs montants. Si vous avez 2 opérations de catégorie transport, l'une de 20 € et l'autre de 300 €, la hauteur du tuyau sera de 2 (ou de 2/n), pas de 320.

Si la variable est qualitative ordinale, alors il suffit de classer sur le graphique les modalités en ordre croissant.

#### Cas des variables quantitatives

##### Variables discrètes

Pour les variables discrètes, on les représente par un équivalent du diagramme en tuyaux d'orgue : le diagramme en bâtons. Cependant, avec les variables qualitatives, on pouvait placer les tuyaux un peu n'importe où sur l'axe horizontal. Mais avec une variable quantitative, on est contraint à placer précisément les bâtons sur l'axe horizontal. Comme on doit être précis, on préfère que les bâtons soient très fins.



##### Variables continues

Prenons l'exemple de la taille d'une personne : c'est une variable continue. On peut très bien avoir une personne de taille 1,47801 m et une autre de 1,47802 m. Ces deux tailles sont différentes : faut-il alors afficher sur notre graphique 2 bâtons, un pour chacune des 2 tailles ?

Tu chipotes, 1,47801 m et 1,47802 m, c'est quasiment la même valeur, il faut donc que tu les considères comme égales !

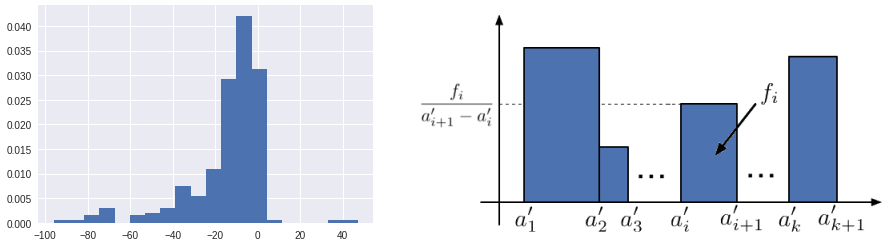
Tout à fait ! Considérer que 1,47801 m et 1,47802 m sont presque égales, c'est regrouper ces valeurs. On dit alors que l'on **agrège** des valeurs en **classes**. Si on décide d'agréger en classes de taille 0,2 m, alors ces 2 valeurs seront toutes les deux situées dans la classe .

Notez qu'il est possible d'agréger en classes de largeurs inégales. Par exemple, on peut avoir des classes de largeur 0,5 m pour les tailles inférieures à 1 m, puis des classes de largeur 0,2 m pour les tailles supérieures à 1 m. On aura ceci : 

Le fait d'agréger une variable s'appelle la **discrétisation** (en anglais : **binning, bucketing** ou**discretization**).

Ainsi, pour les variables continues, on utilise l'**histogramme**, dans lequel les valeurs sont agrégées. Ici, comme on représente des classes (ou des intervalles si vous préférez), on n'utilise plus de fins bâtons, mais des rectangles dont la largeur correspond à la largeur de la classe.

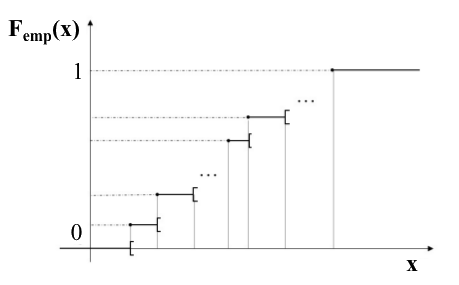
Aussi, l'effectif de la classe ne sera plus représenté par la hauteur du rectangle, mais par sa surface. Cela est dû au fait que les classes n'ont pas forcément la même largeur.



Si cependant, vous ne souhaitez pas agréger les valeurs, il existe une autre solution : représenter la **fonction de répartition empirique**. Il faut vous la représenter comme un escalier. Pour la représenter, on parcourt l'axe horizontal des petites valeurs vers les grandes valeurs. A chaque fois que l'on rencontre une valeur qui est présente dans notre échantillon, on monte d'une marche. Il y aura donc autant de marches que de valeurs, et d'ailleurs autant que d'individus. Toutes les marches ont la même hauteur.

Enfin... presque toutes les marches ont la même hauteur. En effet, s'il y a 2 valeurs égales ou plus, alors la marche est d'autant plus grande. Mais dans le cas de variables continues, il est assez rare d'avoir des valeurs égales.

Une fois parcourues toutes les valeurs de l'échantillon, on aura atteint le haut de l'escalier. On dit (arbitrairement) que le haut de l'escalier est d'une hauteur de 1.



Fonction de répartition empirique

Pour en savoir plus sur la fonction de répartition, rendez-vous au bas de ce chapitre (section Aller plus loin.)

### Du côté du code

Voici le code qui a généré les graphiques ci-dessus. Pour chacun des graphiques, 2 lignes suffisent :

import matplotlib.pyplot as plt

# VARIABLE QUALITATIVE

# Diagramme en secteurs

data["categ"].value\_counts(normalize=True).plot(kind='pie')

# Cette ligne assure que le pie chart est un cercle plutôt qu'une éllipse

plt.axis('equal')

plt.show() # Affiche le graphique

# Diagramme en tuyaux d'orgues

data["categ"].value\_counts(normalize=True).plot(kind='bar')

plt.show()

# VARIABLE QUANTITATIVE

# Diagramme en bâtons

data["quart\_mois"].value\_counts(normalize=True).plot(kind='bar',width=0.1)

plt.show()

# Histogramme

data["montant"].hist(density=True)

plt.show()

# Histogramme plus beau

data[data.montant.abs() < 100]["montant"].hist(density=True,bins=20)

plt.show()

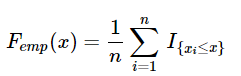
Ce code est disponible dans l'archive téléchargeable au chapitre [**Téléchargez les données**](https://openclassrooms.com/courses/nettoyez-et-decrivez-votre-jeu-de-donnees/telechargez-les-donnees), dans le dossier analyse, dans ces différents fichiers : R\_analysis et python\_analysis.

Ici, nous reprenons le même raisonnement qu'au début de ce chapitre. On commence par sélectionner la colonne souhaitée  data['categ'], puis on compte le nombre d'apparitions de chaque modalité :  data['categ'].value\_counts(). Pour obtenir les fréquences, on peut éventuellement ajouter  normalize=True. On obtient donc la distribution empirique. Pour l'afficher, on fait appel à la méthode  plot, à laquelle on spécifie le type de graphique souhaité (  pie  ou  bar  ).

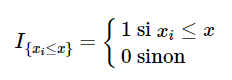
Comme nous l'avons dit plus haut, pour une variable quantitative, on regroupe les valeurs en classes. Utiliser  value\_counts()  n'aurait donc pas vraiment de sens : on utilise donc la méthode  hist(), qui s'occupe elle-même de regrouper les valeurs en classes. La ligne 20 crée un histogramme un peu trop étalé, car il y a des montants très grands et très petits. On filtre donc ici les montants compris entre -100 € et 100 € grâce à  data[data.montant.abs() < 100]  (on utilise pour cela la [valeur absolue](https://fr.wikipedia.org/wiki/Valeur_absolue)). Enfin, on peut aussi spécifier le nombre de classes voulues grâce au mot clé  bins  : ici 20.

### Aller plus loin : La fonction de répartition empirique

La fonction de répartition empirique s'exprime de la manière suivante :



 où  I est la fonction indicatrice, et où



### Aller plus loin : Nombre optimal de classes pour l'agrégation

Pour l'histogramme, il existe des règles pour déterminer le nombre optimal de classes. Par exemple, la règle de Sturges (1926) considère comme nombre optimal de classes :



où **n** est la taille de l'échantillon.

## Présentez une variable sous forme de tableau

Il n'y a pas que les histogrammes dans la vie !

Il est aussi possible de présenter des variables sous forme de tableau. C'est un peu moins beau certes, mais dans certains cas, cette représentation est mieux adaptée ou complémentaire d'une représentation graphique. Nous allons étudier ici les 4 cas correspondant aux 4 types de variables.

### Donnons des noms à ce que nous étudions

Pour que nous puissions communiquer vous et moi, nous devons parler un langage commun. Nous allons donc nommer les différents objets que nous manipulerons dans ce chapitre.

Nous travaillerons ici avec l'échantillon des relevés bancaires, composés d'opérations. Notons **n** le nombre d'opérations bancaires ; c'est la taille de notre échantillon.

Ensuite, la variable que nous étudierons s’appellera **X** .

**X** n'a rien de très concret : c'est juste une **variable**. Par exemple, montant est une variable.

Dans notre jeu de données, on a plusieurs **valeurs** pour la variable montant : 1.43, 80, 2.20, etc. Ici, c'est beaucoup plus concret, on a de vraies valeurs sous les yeux. Le nombre de valeurs présentes dans notre échantillon est de **n**. Ainsi, on peut noter ces valeurs  .

Quand on parle de variable, on préfère la nommer avec une majuscule. Par contre, ce qui est une observation (= une réalisation) de la variable, on le note par des lettres minuscules. Ici,  est une réalisation observée de la variable aléatoire **X** .

#### Cas des variables qualitatives et quantitatives discrètes

Si **X** est qualitative (ou même quantitative discrète), cela signifie qu'elle peut prendre plusieurs modalités. Par exemple, categ peut prendre les modalités "COURSES", "LOYER", "TRANSPORTS", etc. Ces modalités, nous les appellerons , où **k** est le nombre de modalités.

#### Cas des variables quantitatives continues

Pour présenter les variables quantitatives continues, nous agrégerons les valeurs de la variable **X** en classes, qui seront au nombre de **k** . Ces classes seront notées ainsi : 

### Représenter les variables sous forme de tableau

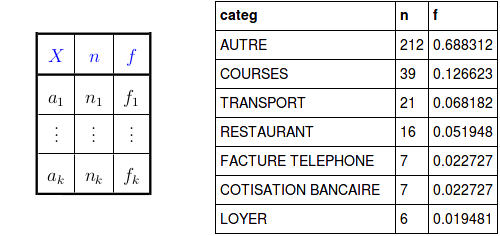
#### Pour les variables qualitatives

Pour les variables qualitatives, il suffit juste de compter le nombre de valeurs pour chaque modalité. Ce nombre est appelé effectif de la modalité.

Ainsi, pour une modalité **ai** (où ii est compris entre **1** et **k** bien entendu !), on appelle l'effectif **ni**. Si on additionne les effectifs de toutes les modalités, on retombe sur n : la taille de l'échantillon.

Si on divise l'effectif par **n**, on obtient alors la **fréquence**, qui est un nombre compris entre 0 et 1. Vous l'aurez deviné, si on additionne les fréquences des toutes les modalités, on obtient 1 !

Voici donc comment on présente usuellement une variable qualitative : formellement, et avec l'exemple de la variable categ :

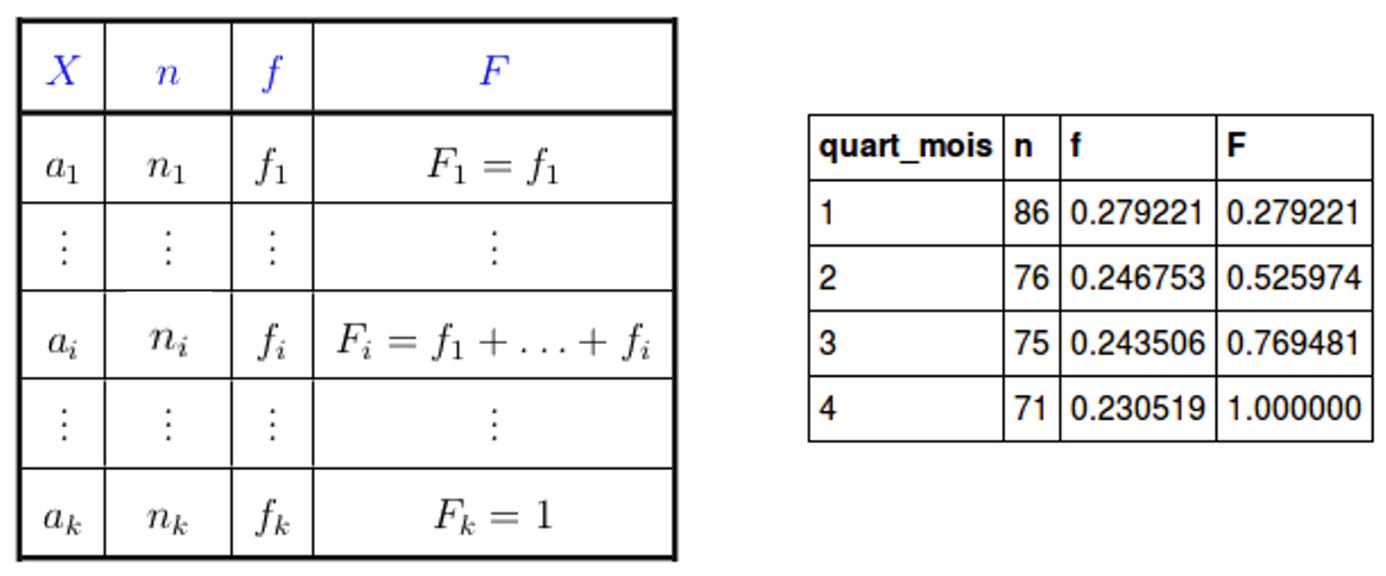


 Cet exemple vaut pour une variable qualitative. Plus précisément, si elle est qualitative ordinale, alors il suffit d'ordonner dans l'ordre croissant.

#### Pour les variables quantitatives

##### Variables discrètes

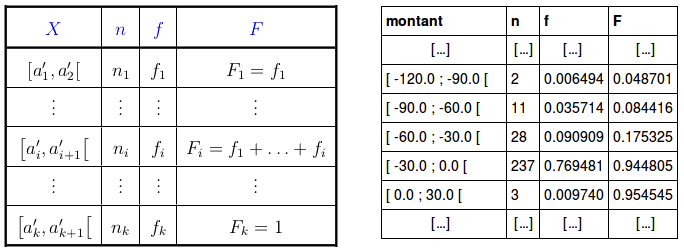
Pour les variables quantitatives discrètes, on peut reprendre le tableau précédent, et y ajouter une colonne qui donne la **fréquence cumulée**. La fréquence cumulée d'une modalité **ai** , c'est juste la somme des fréquences de toutes les modalités inférieures ou égales à **ai** . On la note **F** . Voici l'exemple de la variable quart\_mois :



La fréquence cumulée de la dernière modalité (  ) est 1.

##### Variables continues

Pour les variables continues, il suffit de remplacer les par les classes, comme nous l'avons dit plus haut. Voilà ce que ça donne, avec la variable montant :



### Et le code ?

En Python, c'est plutôt simple. Il suffit (presque) d'une ligne de code par colonne. Voici le code qui a généré le tableau récapitulatif de la variable quart\_mois.

effectifs = data["quart\_mois"].value\_counts()

modalites = effectifs.index # l'index de effectifs contient les modalités

tab = pd.DataFrame(modalites, columns = ["quart\_mois"]) # création du tableau à partir des modalités

tab["n"] = effectifs.values

tab["f"] = tab["n"] / len(data) # len(data) renvoie la taille de l'échantillon

Pour calculer les effectifs, on fait appel à  value\_counts()  sur la variable à étudier. Cette méthode renvoie un objet Series dont les valeurs sont les effectifs, et dont l'index contient les modalités. (lignes 1 et 2)

Besoin d'un petit rappel sur les objets Series et leurs index ? [**C'est par ici**](https://pandas.pydata.org/pandas-docs/stable/dsintro.html#series) ;).

A partir des modalités, on crée le tableau  tab  (ligne 4), auquel on ajoute la colonne des effectifs (ligne 5) puis la colonne des fréquences (ligne 6).

Pour calculer les fréquences cumulées, il suffit de 2 lignes en plus. L'une trie les valeurs, et l'autre calcule la somme cumulée des fréquences :

tab = tab.sort\_values("quart\_mois") # tri des valeurs de la variable X (croissant)

tab["F"] = tab["f"].cumsum() # cumsum calcule la somme cumulée

# Testez vos connaissances sur les statistiques descriptives

Bravo ! Vous avez réussi cet exercice !

### Compétences évaluées

* Représenter les différents types de variables

### Question 1

**Un variable qualitative peut être :**

* + 

soit nominale, soit ordinale

* + 

soit nominale, soit continue

* + 

soit discrète, soit ordinale

* + 

soit discrète, soit continue

*Les variables qualitatives sont soit nominales soit ordinales, et les variables quantitatives sont soit discrètes, soit continues.*

### Question 2

**Si une variable prend ses modalités dans la liste "petit", "moyen", "grand", alors elle est :**

* + 

ordinale

* + 

nominale

* + 

discrète

* + 

continue

*Comme les modalités ne sont pas numériques, cette variable est qualitative. Comme on peut ordonner ses modalités par ordre croissant, elle est ordinale.*

### Question 3

**La fréquence d'une modalité est calculée par :**

* + 

le nombre d'individus de cette modalité divisé par la taille de l'échantillon

* + 

le nombre d'individus de cette modalité multiplié par la taille de l'échantillon

*Il faut la diviser. En effet, on s'attend à ce qu'une fréquence soit comprise entre 0 et 1.*

### Question 4

**Avant de calculer une fréquence cumulée, quelle précaution faut-il prendre ?**

* + 

Ordonner les modalités (ou les classes d'agrégation) dans l'ordre croissant (ou décroissant).

* + 

Vérifier que la taille de l'échantillon est inférieure (ou égale) à la taille de la population.

*Vérifier que la taille de l'échantillon est inférieure à la taille de la population n'a pas de sens. En effet, un échantillon est tiré d'une population, sa taille est donc forcément inférieure ou égale à la taille de la population.*

### Question 5

**Un histogramme représente :**

* + 

la distribution d'une variable

* + 

pas grand chose, mais c'est joli quand même

* + 

la représentation des fréquences cumulées

*Un histogramme représente la distribution d'une variable.*

### Question 6

**Trouvez la phrase fausse :**

* + 

Un individu comporte plusieurs échantillons.

* + 

Une population est composée d'individus.

* + 

Une variable caractérise un individu.

* + 

Un échantillon provient d'une population

*Un****échantillon****comporte plusieurs****individus****, et non pas l'inverse. ;)*

## Laissez-vous guider par les indicateurs

### Qu'est-ce qu'une statistique ?

Quel est le point commun entre la moyenne d'âge de la population chinoise en 2010, le taux de réussite au quiz de fin de la partie 1 de ce cours, et l'indice d'érosion des sols de la région des Hauts de France ? o_O

Réponse : ce sont toutes des **statistiques** !

Formellement, une **statistique**, c'est un indicateur numérique calculé à partir d'un échantillon. La moyenne d'âge est calculée à partir des habitants d'un pays, le taux de réussite à un quiz est calculé à partir des réponses données par les étudiants, et l'indice d'érosion des sols est calculé à partir de relevés effectués sur des parcelles de terrain.

Autrement dit, dès que l'on calcule un nombre à partir d'un échantillon, on calcule une **statistique** !

Une statistique est utile car elle nous permet de résumer un grand échantillon en un seul nombre ! Certes, vous vous doutez bien qu'il y a une grande perte d'information quand on calcule une statistique : on peut calculer le taux de réussite à partir des réponses des étudiants, mais on ne peut pas retrouver les réponses des étudiants uniquement avec le taux de réussite !

Ainsi, une statistique est un indicateur, plus ou moins efficace, d'une certaine propriété d'un échantillon.

Dans le monde, on calcule tellement de **statistiques** dans tous les domaines, que ce terme possède beaucoup de synonymes : **donnée statistique**, **indicateur statistique**, **mesure statistique**, etc.

On trouve également le terme d'**indice statistique**. Un indice statistique, c'est une statistique construite à partir d'une certaine vision, à partir de connaissances d'un domaine (par ex : l'économie). En quelques sortes, un indice est une statistique "entourée" d'une certaine philosophie. A la différence d'un indice, un **indicateur** est quant à lui très neutre, comme une moyenne par exemple.

Dans les chapitres suivants, nous ne verrons que certains indicateurs (ou mesures) statistiques, mais pas d'indices car ils nécessiteraient l'apport de connaissances extérieures d'un domaine précis.

### A quoi servent les indicateurs et indices statistiques ?

Si on calcule autant d'indicateurs et d'indices, c'est parce qu'ils sont censés nous guider (comme leur nom l'indique !). Ils nous aident à prendre des décisions. Les indicateurs et indices économiques, écologiques, sociologiques, etc. aident par exemple à prendre des décisions politiques.

Certains indicateurs et indices résultent d'un calcul très simple, comme le chiffre d'affaires d'une entreprise (il suffit d'additionner toutes ses recettes).

D'autres au contraire résultent d'un calcul plus complexe, comme ceux qui conjuguent plusieurs caractéristiques d'une population. C'est le cas de l'indice de développement humain ([IDH](https://fr.wikipedia.org/wiki/Indice_de_d%C3%A9veloppement_humain)), calculé à partir du PIB par habitant, l'espérance de vie à la naissance et le niveau d'éducation. On trouve également l'[indicateur de capacité relationnelle](http://www.ferdi.fr/fr/indicateur/nouvel-indicateur-de-capacit%C3%A9-relationnelle), qui mesure la qualité des relations entre les personnes et le niveau de leur autonomisation relationnelle.

Dans le domaine de l'environnement, on trouve les indices de "biocapacité" et d'empreinte écologique de l'Homme, eux-même calculés à partir de données concernant les forêts, les terrains construits, les champs cultivés, etc.

### Que nous disent les indicateurs ?

Sur une même population, on peut bien sûr calculer plusieurs indicateurs. Chacun d'entre eux nous donnera une indication sur une caractéristique différente de la population. Par exemple, la moyenne d'une classe à un examen nous indiquera si l'examen a été bien réussi ou pas. Mais sur cette même population, l'écart-type des notes (nous verrons cette notion prochainement) nous indiquera s'il y a de grandes disparités de notes parmi les étudiants.

Il ne faut jamais faire confiance à 100% à un indicateur. Vous vous imaginez bien que résumer une réalité complexe en un seul nombre, c'est forcément faire l'impasse sur certains aspects importants de cette réalité. Ainsi, quand on utilise un indicateur, il faut toujours savoir ce qu'il mesure, et ce qu'il ne mesure pas !

## Découvrez les mesures de tendance centrale

Dans cette partie, nous allons effectuer des analyses univariées. Une analyse univariée est une analyse effectuée sur une variable à la fois.

Vous devez vous rendre en voiture à un entretien d'embauche, loin de chez vous, dans une autre ville que la vôtre. Vous vous demandez à quelle heure il faut partir pour arriver là bas à 15h. Comme vous avez beaucoup de choses à faire le matin, vous ne souhaitez pas partir trop tôt, mais vous voulez quand même être sûr d'être à l'heure.

Le trajet que vous devrez faire, vous le connaissez peu. Mais heureusement, l'un de vos amis le parcourt tous les jours, et il connaît la route par cœur.

Vous lui demandez donc :

Combien de temps dure le trajet entre les deux villes ? Il vous répond : Tout dépend si la circulation est bonne ou pas. La plupart du temps, je met entre 40 min et 45 min.

Penchons-nous sur cette phrase. Votre ami a l'habitude du trajet : il l'a parcouru peut-être 1000 fois ! A chaque fois, il a retenu (plus ou moins inconsciemment) le temps de trajet. Nous avons donc ici un échantillon de taille 1000, avec une variable quantitative continue : le temps de trajet entre les 2 villes.

Même si le temps de ce trajet peut en théorie prendre des valeurs comprises entre 0 et l'infini, vous vous doutez bien qu'elles se concentrent quand même autour d'une certaine valeur. Ce que vous souhaitez savoir ici, c'est avoir un ordre d'idée d'où (sur un axe de 0 à l'infini) se concentrent les valeurs des temps de trajet.

Tu emploies le terme se concentrent. Dans concentrer, il y a le mot centre non ?

Tout à fait ! Nous arrivons donc à l'objet de ce chapitre : les **mesures de tendance centrale**. Nous allons en voir 3, et devinez quoi : elles commencent toutes par un M !

### Les mesures de tendance centrale

#### Le mode

**La plupart du temps, je mets entre 40 et 45 min.**

Quand votre ami vous dit cela, il vous donne une mesure de tendance centrale qui s'appelle **le mode**.

Pour les variables qualitatives, ou pour les variables quantitatives discrètes, le mode est la modalité ou la valeur la plus fréquente. Dans notre relevé bancaire, le mode de la variable categ est "Autre", car la modalité "Autre" est présente 212 fois dans l'échantillon, et toutes les autres modalités ("loyer", "courses", etc.) sont présentes moins de fois.

Pour les variables quantitatives continues, on travaille dans le cas agrégé, en regroupant les valeurs par classes. La **classe modale** est la classe la plus fréquente. Votre ami a découpé sa variable en tranches de 5 minutes, et a déterminé que la tranche la plus fréquente était**[40min;45min[** .

#### La moyenne

Vous répondez donc à votre ami : Oui, mais je ne peux pas me contenter de la durée la plus fréquente : car si la deuxième durée la plus fréquente est de 65 à 70 minutes, il faut que je parte beaucoup plus tôt ! Il répond alors :

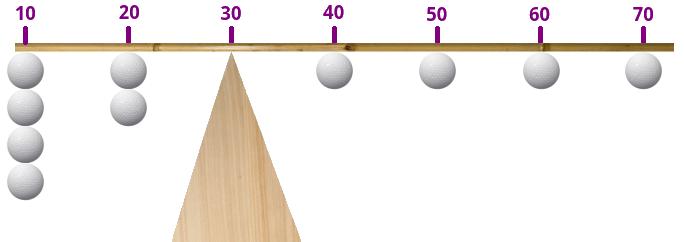
**Oui tu as raison. En fait je mets en moyenne 60 minutes par trajet, car il y a souvent des embouteillages.**

Cela change tout : heureusement que vous lui avez demandé de préciser, vous seriez arrivé en retard ! Ici, votre ami vous a répondu en termes de **moyenne**.

La moyenne, vous la connaissez tous. Pour calculer la moyenne de valeurs, on additionne celles-ci, puis on divise le résultat par le nombre de valeurs.

Il est courant d'associer la notion de moyenne à la notion d'équilibre et de centre de gravité. Pourquoi ? Imaginez que vous avez 10 valeurs numériques. Vous prenez un bâton, que vous graduez. Sur cette graduation, vous marquez au feutre l'emplacement de vos 10 valeurs, puis vous fixez au bâton une balle sur chacune des 10 marques. Après avoir calculé la moyenne des 10 valeurs, vous l'inscrivez également sur votre bâton. Si vous souhaitez faire tenir votre bâton en équilibre, à l'horizontale, il vous faudra trouver son centre de gravité. Vous me voyez venir : le centre de gravité sera pile-poil là où vous aurez placé la moyenne des 10 valeurs !

 En supposant que le poids du bâton soit négligeable par rapport au poids des balles, qui quant à elles, sont toutes de même poids.

Le centre de gravité est situé à 30.

Oui mais sur notre bâton... si l'une des valeurs est très différente des autres, sa balle sera très éloignée des autres sur le bâton, et ce dernier sera totalement déséquilibré, juste à cause d'une seule valeur !

#### La médiane

Derrière le problème de bâton déséquilibré, vous aurez reconnu le concept d'outlier. Comme nous venons de le voir, la moyenne est une mesure peu robuste aux outliers.

Alors, à propos de votre trajet à parcourir, vous demandez à votre ami : Quand tu me dis que tu mets en moyenne 60 min, j'imagine que tu considères dans ton calcul les rares fois durant lesquelles il y avait de la neige, et que tu as mis 4 h à faire la route ? Ce sont des outliers, autant ne pas les prendre en compte car c'est l'été, et il n'y aura pas de neige ! Ce à quoi il vous répond :

**Oui effectivement. Je vais alors formuler la chose autrement : disons que la moitié des trajets que j'ai effectué ont pris plus de 55 min, et l'autre moitié ont pris moins de 55 min.**

 Ici, votre ami vous parle en termes de **médiane**.

La médiane, (notée Med), est la valeur telle que le nombre d’observations supérieures à cette valeur est égal au nombre d’observations inférieures à cette valeur.

En gros, pour trouver la médiane de vos **n** valeurs, il faut commencer par les trier. Une fois triées, on appelle la première valeur, la deuxième valeur, ... , et  la dernière valeur. La médiane, c'est la valeur qui sera exactement au milieu du classement, soit

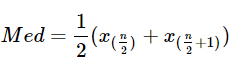


Ainsi, sur n=**999** trajets, la médiane est = 55 min.

Ton calcul marche bien car 999 est impair. Mais s'il y a 1000 trajets, la médiane est-elle la 500e valeur ou la 501e valeur ? Si on choisit la 500e, alors il y a 499 valeurs en dessous et 500 valeurs au-dessus. Mais si on choisit la 501e, alors il y a 500 valeurs en dessous et 499 valeurs au-dessus : dans les 2 cas, c'est déséquilibré !

Effectivement. Dans ce cas, on coupe la poire en deux : on place la médiane au centre de la 500e et de la 501e valeur. Ainsi, s'il y a eu 1000 trajets, et que la 500e valeur vaut = 54 min 30 sec, et que la 501e vaut = 55 min 30 sec, alors on coupe en deux et on prend 55 min.

Plus formellement, si **n** est pair, la médiane vaut



### Du côté du code

Difficile de faire plus simple ici : une ligne de code par indicateur ! Prenons l'exemple de la variable montant de nos relevés de comptes :

data['montant'].mean()

data['montant'].median()

data['montant'].mode()

Chacune de ces lignes renvoie une valeur, sauf la ligne 3 qui renvoie un  pd.Series, car une distribution peut avoir plusieurs modes (voir la section Aller plus loin).

Les montants des opérations sont très hétérogènes : il y a des dépenses (à montant négatif) parfois grosses (les loyers par exemple), souvent petites (courses, téléphone, etc.), et il y a des rentrées d'argent (à montant positif), peu fréquentes mais grosses. Difficile donc d'interpréter la moyenne (très sensible aux valeurs atypiques) qui vaut ici **2,87 €**. On a le même problème pour la médiane qui vaut **-9,6 €**. Le fait qu'elle soit négative nous indique cependant qu'il y a plus de dépenses que d'entrées d'argent. Par contre, le mode nous indique que la plupart des opérations tournent autour de -**1,6 €**. Ici, les 3 mesures sont très éloignées les unes des autres.

Pour avoir des montants d'opérations plus homogènes, je vous propose de calculer ces 3 mesures pour chaque catégorie d'opération. Au sein d'une catégorie, les montants devraient être moins éparpillés puisque les opérations sont de même nature. Je vous propose donc une boucle for, qui itérera sur chacune des catégories :

for cat in data["categ"].unique():

subset = data[data.categ == cat] # Création du sous-échantillon

print("-"\*20)

print(cat)

print("moy:\n",subset['montant'].mean())

print("med:\n",subset['montant'].median())

print("mod:\n",subset['montant'].mode())

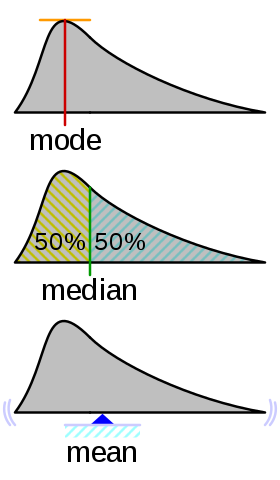
subset["montant"].hist() # Crée l'histogramme

plt.show() # Affiche l'histogramme

Pour chaque catégorie, on crée un sous-échantillon (subset) qui contient uniquement les opérations de la catégorie en cours. On affiche en lignes 5 à 7 les 3 mesures, et on affiche également l'histogramme pour mettre en perspectives les 3 mesures. A vous d'interpréter vos résultats !

### Aller plus loin : Sur un histogramme

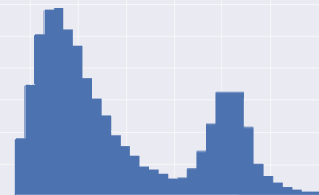
Sur un histogramme, le mode est le "point le plus haut" de la distribution, la médiane est la valeur qui divise la surface en deux et la moyenne est le centre de gravité de la distribution, comme sur cette illustration :



(Source : commons.wikimedia.org, licence GFDL)

### Aller plus loin : Distributions plurimodales

Plus généralement, il arrive que l'on extrapole la définition du mode en l'assimilant au(x) pic(s) d'une distribution. Le mode n’est donc pas obligatoirement unique. Lorsqu’une distribution n’a qu’un seul pic, on parle de distribution unimodale. Il arrive aussi qu’une distribution présente deux ou plusieurs pics : on parle alors de distribution bimodale ou plurimodale. Donner le nombre de modes d'une distribution est intéressant dans une statistique en présence d'une distribution plurimodale.



Une distribution bimodale

## Comprenez les mesures de dispersion

Il y a des formules mathématiques dans ce chapitre, mais ne vous enfuyez pas, elles sont expliquées pas-à-pas ;).

Au chapitre précédent, votre ami vous a donné une estimation de la durée du trajet. Mais il vous a donné des mesures de tendance centrale, comme par exemple la moyenne, qui est de 60 minutes par trajet.

Ce qui vous manque maintenant, c'est de savoir si les durées que votre ami a effectué sont très "resserrées" autour de 60 min (exemple :  [58, 60, 62, 59, 57, ...] ), ou bien si elles s'en écartent beaucoup (exemple :  [40, 70, 78, 43, ...] ).

Quel intérêt ?

Si les valeurs sont très resserrées autour de 60 minutes, alors prévoyez de partir 75 minutes à l'avance. Ainsi, il est probable que vous arriverez 5 ou 10 minutes avant votre entretien. Mais si les valeurs sont très écartées, alors prévoyez plutôt de partir 100 minutes à l'avance, car il est tout à fait possible que le trajet dure 80 minutes !

J'ai compris ! Mesurer l'espacement des valeurs... j'imagine qu'il y a une mesure statistique pour cela non ?

Tout à fait ! :D Il y en a même plusieurs. On les appelle les **mesures de dispersion**.

### Réfléchissons

Essayons de construire notre propre indicateur de dispersion, pas-à-pas. Pour illustrer, prenons les valeurs suivantes (70, 60, 50, 55, 55, 65, 65), et donnons-leur chacun un nom : **xi** , avec **i** allant de 1 à 7. Ainsi, nos valeurs portent les noms de **x1** à **x7** .

Formellement, on écrit  .

Remarquons que la moyenne de ces valeurs vaut 60, on la note , et on prononce "x barre".

Facile de faire une mesure de dispersion ! Prenons toutes nos valeurs, et calculons pour chacune d'entre elles l'écart qu'elles ont avec la moyenne. Puis additionnons tous ces écarts !

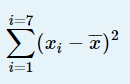
C'est un bon début. Comme notre moyenne est de 60, les écarts des **xi** à la moyenne sont :  Sauf que... si nous faisons la somme de ceux-ci, on obtient 0 ! On peut même le démontrer mathématiquement : quelle que soit la dispersion de vos valeurs, la somme des écarts à la moyenne vaudra toujours 0. Pas très efficace donc...

Si ça vaut 0, c'est parce qu’il y a des nombres positifs et des nombres négatifs. Évitons cela, et mettons-les tous au carré. Un nombre mis au carré, c'est toujours positif n'est-ce pas ?

Exact ! Voici ce que ça donne : .

Maintenant, si on fait la somme de toutes ces valeurs, on obtient 300.

Ici, on a fait la somme de tous les , avec **i** allant de 1 à 7. Mathématiquement, on note cette somme comme ceci :



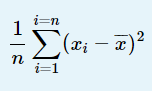
Bon. Il y a encore un problème. Ici, on a 7 valeurs, tout simplement parce que nous sommes un peu paresseux, et nous n'en avons relevé que 7. Mais en statistiques, plus on fait de relevés, plus on a une idée précise de ce que l'on décrit. Ainsi, on aurait dû retenir 10, 100 ou même 1000 valeurs !

Mais avec 1000 valeurs, notre mesure exploserait ! Il passerait de 300 avec 7 valeurs à peut-être 40000000000 avec 1000 valeurs. C'est problématique.

Alors, plutôt que de calculer la somme, et avoir un indicateur qui explose, prenons plutôt la moyenne. Ainsi, qu'il y ait 7 valeurs ou 1000 valeurs, la moyenne n'explosera pas.

Bonne idée. La moyenne de (100,0,100,25,25,25,25) est 42,86.

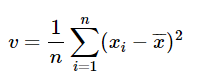
Pour obtenir cette moyenne de 42,86, on a juste divisé 300 par le nombre de valeurs : 7. On a multiplié la formule précédente par 1/n , ce qui nous donne



### Les mesures de dispersion

#### La variance empirique

Devinez quoi ! L'indicateur que nous venons de construire est l'un des plus utilisés en statistiques ! :soleil: Il s'appelle la **variance empirique**. Comme nous venons de le voir, elle est égale à



Pour approfondir l'aspect calculatoire, rendez-vous à la section Aller plus loin, au bas du chapitre. Aussi, vous trouverez souvent une version "corrigée" de la variance empirique, que l'on qualifie de non biaisée. Je vous renvoie ici aussi à la section Aller plus loin.

#### L'écart-type empirique

L'écart-type empirique, c'est juste la racine carrée de la variance empirique. On l'appelle en anglais standard deviation, souvent abrégé std. En fait, quand on calcule la variance empirique des temps de trajet, le résultat a pour unité la minute au carré, ce qui n'est pas très intelligible. En prenant la racine carrée, l'unité redevient la minute. Ici, notre écart-type vaut 6,55 minutes. On le note **s**.



Mais lorsque vous faites un trajet, un écart-type de 6,55 minutes sur un trajet de 1 h (1h en moyenne), ce n'est pas la même chose qu'un écart-type de 6,55 minutes sur un trajet de 24 h (24h en moyenne) ! On a donc créé le **coefficient de variation**, disponible dans la section Aller plus loin.

#### L'écart inter-quartiles

Vous vous souvenez de la médiane ? C'est la valeur au-dessous de laquelle se trouvent la moitié des valeurs.

Un **quartile**, c'est la même chose, mais avec la proportion d'un quart. Il existe 3 quartiles, notés **Q1** (premier quartile), **Q2** (deuxième quartile) et **Q3** (troisième quartile). Ainsi :

* 1/4 des valeurs se trouvent en dessous de **Q1** et 3/4 au-dessus
* 2/4 se trouvent en dessous de **Q2**, et 2/4 au-dessus (**Q2** est la médiane !)
* 3/4 se trouvent en dessous de **Q3**, et 1/4 au-dessus

La généralisation de ce concept s'appelle le **quantile d'ordre α**. Ainsi, la médiane est le quantile d'ordre 0,5, Q1 le quantile d'ordre 0,25, Q3 le quantile d'ordre 0,75. Il y a également les **déciles** (quantiles d'ordre 0.1, 0.2, etc.), ou les **centiles**, aussi appelés percentiles (quantiles d'ordre 0,01, 0.02, etc.)

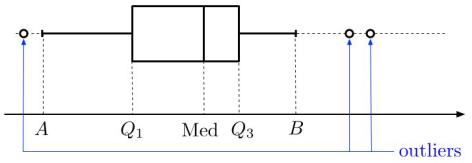
L'écart inter-quartile est la différence entre le 3e quartile et le 1e quartile :



#### La boîte à moustaches (boxplot)

Boîte à moustaches, quel nom rigolo ! :lol: Les anglophones l'appellent boxplot.

Elle permet de représenter schématiquement une distribution, en incluant sa dispersion. La boîte est délimitée par **Q1** et **Q3**, et on représente souvent la médiane à l’intérieur de la boîte. On dessine ensuite des moustaches à cette boîte, qui vont de la valeur minimale à la valeur maximale... à condition que la moustache (d'un côté ou de l'autre) ne mesure pas plus de 1,5 fois l'écart inter-quartiles. Si certaines valeurs sont au-dessous de  **Q1−1.5IQ** ou au-dessus de **Q3+1.5IQ** , alors on les considère comme des outliers, et on ne les inclut pas dans la moustache :



Les boîtes à moustaches et les histogrammes ont la même vocation : donner un aperçu d'une distribution empirique. La boîte à moustache est plus grossière que l'histogramme, mais elle permet de comparer plus facilement la distribution de plusieurs variables (deux boîtes à moustaches sont plus simples à comparer que 2 histogrammes).

### Du côté du code

Pour la variance empirique et l'écart-type empirique, c'est le même principe qu'au chapitre précédent : on appelle les méthodes var() et  std()  sur la variable étudiée. A vrai dire, ces 2 méthodes renverront des résultats un peu différents des calculs faits avec les formules données ci-dessus. C'est une histoire d'estimateur biaisé (pour les motivés, voir la section Aller plus loin : variance empirique). Pour obtenir les résultats décrits ci-dessus, il faut ajouter ddof=0 (lignes 8 et 9).

Nous allons afficher les boîtes à moustaches avec les histogrammes, pour que vous puissiez les comparer ;). Ligne 12, le mot clé vert=False signifie que nous souhaitons que la boîte à moustaches ne soit pas à la verticale (donc à l'horizontale !). En reprenant le code du chapitre précédent, voilà ce que cela donne :

for cat in data["categ"].unique():

subset = data[data.categ == cat]

print("-"\*20)

print(cat)

print("moy:\n",subset['montant'].mean())

print("med:\n",subset['montant'].median())

print("mod:\n",subset['montant'].mode())

print("var:\n",subset['montant'].var(ddof=0))

print("ect:\n",subset['montant'].std(ddof=0))

subset["montant"].hist()

plt.show()

subset.boxplot(column="montant", vert=False)

plt.show()

La variance du montant des opérations de la catégorie LOYER est nulle, ce qui signifie que le montant du loyer est toujours resté fixe.

### Aller plus loin : La variance empirique corrigée

La meilleure manière d'estimer la variance d'une variable aléatoire (i.e. la variance théorique) n'est pas d'utiliser la variance empirique.

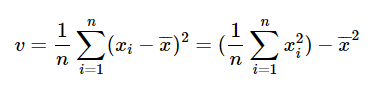
Étonnant, non ? Oui. En fait, quand on se plonge dans les calculs, on s'aperçoit que la variance empirique donne des valeurs qui (en moyenne) sont inférieures à la variance de la variable aléatoire.

Il s'agit de la notion de [biais](https://fr.wikipedia.org/wiki/Estimateur_(statistique)#Biais) d'un estimateur. Un estimateur sans biais est meilleur qu'un estimateur biaisé. La variance empirique est un estimateur biaisé de la variance de la variable aléatoire.

Pour corriger ce biais, on a créé la [variance empirique corrigée](https://fr.wikipedia.org/wiki/Estimateur_(statistique)#Estimateur_de_la_variance_de_Y), ou **variance empirique sans biais**. Elle est souvent notée , et est égale à , où **v** est la variance empirique, et **n** la taille de l'échantillon. Quand la taille de l'échantillon est grande, la variance empirique et la variance empirique corrigée sont presque égales.

### Aller plus loin : Calculs avec la variance empirique

On peut montrer par le calcul que la variance empirique **v** peut aussi s'écrire sous une forme très pratique :



C'est la Théorème de König-Huygens. Pour la démonstration, c'est [par ici](https://fr.wikipedia.org/wiki/Th%C3%A9or%C3%A8me_de_K%C3%B6nig-Huygens) ;).

Si on crée une nouvelle variable **Y** à partir d'une variable **X** dont on connaît la variance  , et que**Y=aX+b** , alors on peut connaître la variance de **Y** notée  ! Elle est donnée par cette relation :

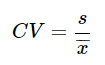


([Voir la démonstration](https://fr.wikipedia.org/wiki/Variance_(math%C3%A9matiques)#Transformation_affine_2))

### Aller plus loin : Le coefficient de variation

Lorsque vous faites un trajet, un écart-type de 6,55 minutes sur un trajet de 1 h, ce n'est pas la même chose qu'un écart-type de 6,55 minutes sur un trajet de 24 h ! Dans le premier cas, l'écart-type sera vu comme assez grand, alors que dans le second cas, il sera négligeable face aux 24 h.

Pour rendre compte de cela, on a créé le **coefficient de variation**, qui est l'écart-type empirique divisé par la moyenne :



### Aller plus loin : Autres mesures de dispersion

Quand au début du chapitre, nous avons dit :

Mettons-les tous au carré. Un nombre mis au carré, c'est toujours positif n'est-ce pas ?

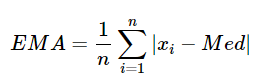
Peut-être vous êtes-vous dit :

On peut prendre la valeur absolue aussi plutôt que le carré non ?

Tout à fait. Quand on fait cela, on calcule l'**écart moyen absolu**.

Il y a deux versions : l'une où on mesure les écarts à la moyenne, l'autre où on mesure les écarts à la médiane.

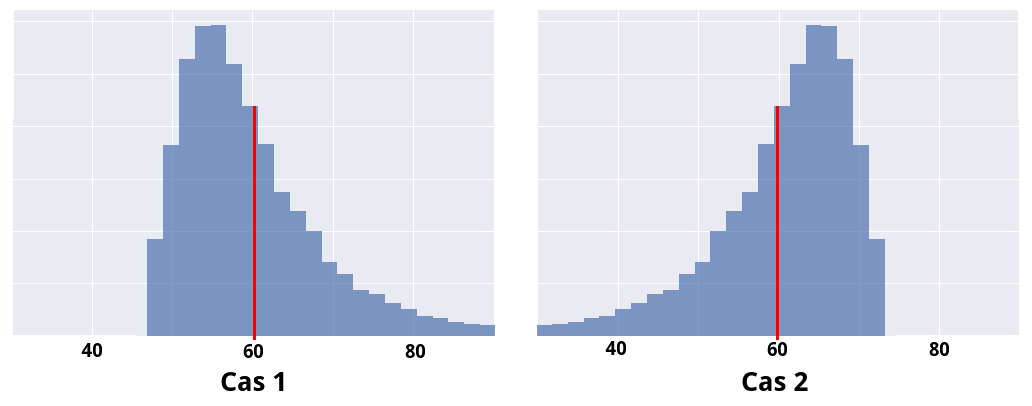
Voici la version avec la médiane :



Si on souhaite une mesure plus robuste, on définit également le MAD qui est la médiane des écarts absolus par rapport à la médiane.

## Appréhendez les mesures de forme

Bon, votre ami vous a donné la moyenne des temps de trajets, ainsi que l’écart-type. Vous êtes déjà plus serein. Mais... il y a quelque chose que vous n'avez pas prévu. Regardez ces 2 distributions :



Elles ont la même moyenne empirique (60 minutes), et le même écart-type. Cependant, le cas 1 est plus "risqué" que le cas 2. En effet, dans le cas 2, il est très peu probable que votre trajet dure plus de 75 minutes : pas de risque d'être en retard ! Par contre, dans le cas 1, il est tout à fait possible que votre trajet dure 80 minutes, ou même beaucoup plus.

Vous remarquez donc que connaître la moyenne et l’écart-type ne suffit pas ici. Ce qu'il vous faut connaître, c'est la forme de la distribution : est-ce qu'elle s'étale plutôt vers la gauche ou plutôt vers la droite ?

Il y a des mesures statistiques pour cela ! On les appelle les **mesures de forme**.

### Réfléchissons

Construisons notre propre indicateur de forme ! Nous souhaitons savoir si la distribution s'étale plutôt à gauche ou à droite de la moyenne.

Ceci est équivalent à savoir si la majorité des valeurs est plus petite ou plus grande que la moyenne.

Je vous propose de reprendre celui que nous avons construit au chapitre précédent. Au départ, nous avions eu cette idée :

Prenons toutes nos valeurs, et calculons pour chacune d'entre elles l'écart qu'elles ont avec la moyenne. Puis additionnons tous ces écarts !

L'écart entre une valeur et la moyenne, nous l'avons écrit . Si cet écart est positif, cela signifie que **xi** est supérieur à la moyenne, s'il est négatif, **xi** est inférieur à la moyenne.

En additionnant tous ces écarts, nous nous sommes aperçus que la somme valait toujours 0. Nous avons donc mis cette quantité au carré : . Avec le carré, cette grandeur est toujours positive. Si elle est toujours positive, on perd l'information qui nous dit si **xi** est supérieur ou inférieur à la moyenne. Or ici, nous voulons garder cette information !

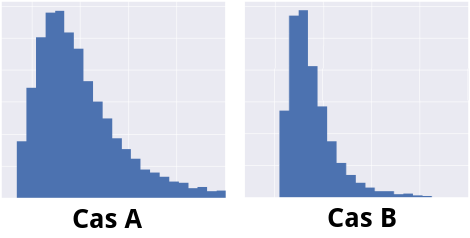
Bon, si le carré ne convient pas, mettons-la au cube pour voir !

Bien vu ! Quand on met l'écart au cube, on obtient . Contrairement au carré, le cube conserve le signe de . Ensuite, prenons la moyenne de tous ces écarts au cube, on obtient :



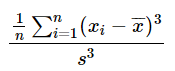
Nous avons atteint notre objectif : cette grandeur sera négative si la majorité des valeurs est plus petite que la moyenne, et positive sinon !

Mais nous pouvons faire encore mieux. Regardez ces deux distributions :



Elles ont la même forme, mais pas le même écart-type (la distribution A est plus étendue que B, A a un écart-type 2 fois supérieur à B). Comme elles ont la même forme, on voudrait que notre indicateur donne la même valeur pour ces deux distributions.

Mais actuellement, ce n'est pas le cas. Dans le cas A, les écarts à la moyenne sont 2 fois plus importants que dans le cas B. Comme on met ces écarts au cube, notre indicateur sera donc  fois plus grand pour A que pour B. Or nous les souhaitons égaux. Pour corriger cela, il faut annuler l'effet de l'écart-type. On va donc diviser notre indicateur par l'écart-type mis au cube :



### Les mesures de forme

#### Le Skewness empirique

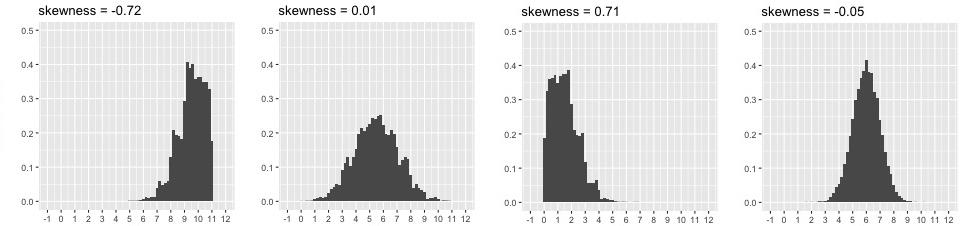
Devinez quoi ! L'indicateur que nous venons de créer est utilisé par les statisticiens, et s'appelle le **skewness empirique**. En général, on a l'habitude de nommer le skewness **γ1** et son numérateur **μ3** :



avec  

Le skewness est une mesure d'asymétrie. L’asymétrie d’une distribution traduit la régularité (ou non) avec laquelle les observations se répartissent autour de la valeur centrale. On interprète cette mesure de cette manière :

* Si **γ1=0** alors la distribution est symétrique.
* Si **γ1>0** alors la distribution est étalée à droite.
* Si **γ1<0** alors la distribution est étalée à gauche.

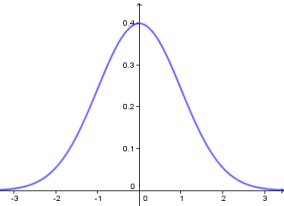


Relation entre la forme de la distribution et le skewness

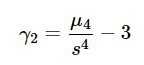
Pour en savoir plus sur la notion d'**asymétrie**, rendez-vous à la section Aller plus loin, au bas de ce chapitre. Vous y découvrirez pourquoi comparer la médiane à la moyenne mesure aussi l'asymétrie.

#### Le Kurtosis empirique

Le **kurtosis empirique** n'est pas une mesure d'asymétrie, mais c'est une mesure d'**aplatissement**. L’aplatissement peut s’interpréter à la condition que la distribution soit symétrique. En fait, on compare l'aplatissement par rapport à la distribution la plus célèbre, appelée **distribution normale** (parfois "courbe de Gauss" ou "Gaussienne"). Vous l'avez probablement déjà vue, elle ressemble à cela :

Distribution normale

Le kurtosis est souvent noté **γ2** , et se calcule par :

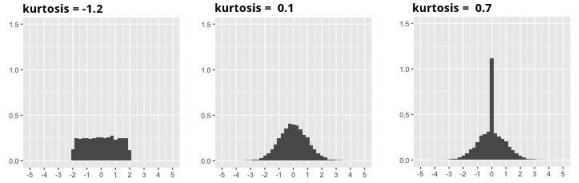


avec  

Mais que sont vraiment ces mystérieux **μ3** et **μ4** dans les formules du skewness et du kurtosis ? On les appelle des **moments**. Pour plus de précisions, reportez-vous à la section Aller plus loin, au bas du chapitre ;).

Il s’interprète comme ceci :

* Si **γ2=0** , alors la distribution a le même aplatissement que la distribution normale.
* Si **γ2>0** , alors elle est moins aplatie que la distribution normale : les observations sont plus concentrées.
* Si **γ2<0** , alors les observations sont moins concentrées : la distribution est plus aplatie.



Relation entre la forme de la distribution et le kurtosis

### Du côté du code

Vous connaissez maintenant le principe ! Au code du chapitre précédent, on ajoute les lignes 10 et 11, qui calculent le skewness empirique et le kurtosis empirique :

for cat in data["categ"].unique():

subset = data[data.categ == cat]

print("-"\*20)

print(cat)

print("moy:\n",subset['montant'].mean())

print("med:\n",subset['montant'].median())

print("mod:\n",subset['montant'].mode())

print("var:\n",subset['montant'].var(ddof=0))

print("ect:\n",subset['montant'].std(ddof=0))

print("skw:\n",subset['montant'].skew())

print("kur:\n",subset['montant'].kurtosis())

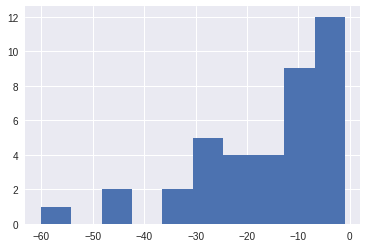
subset["montant"].hist()

plt.show()

subset.boxplot(column="montant", vert=False)

plt.show()

Pour toutes les opérations dont la catégorie regroupe principalement des dépenses (téléphone, courses, etc.), il est fort probable que le skewness empirique soit négatif. La distribution des dépenses s'étale vers la gauche, car on fait fréquemment des petites dépenses, et moins souvent des dépenses qui parfois peuvent être très importantes (donc très à gauche) :



Distribution empirique des montants des dépenses de catégorie COURSES

### Aller plus loin : Quelques mots sur l'asymétrie

Vous vous souvenez de cette phrase, plus haut dans le chapitre ?

Nous souhaitons savoir si la majorité des valeurs est plus petite ou plus grande que la moyenne.

Quand on dit la majorité, on entend plus de 50 % des valeurs. Vous vous souvenez que la médiane est construite de telle manière à ce que 50 % des valeurs lui soient supérieures. Ainsi, la phrase ci-dessus est équivalente à dire : Nous souhaitons savoir si la médiane est plus petite ou plus grande que la moyenne.

Chercher qui de la médiane ou de la moyenne est la plus grande, c'est donc étudier l'**asymétrie** de la distribution !

Une distribution est dite symétrique si elle présente la même forme de part et d’autre du centre de la distribution. Dans ce cas : .  
Une distribution est étalée à droite (ou oblique à gauche, ou présentant une asymétrie positive) si : .

De même, elle est étalée à gauche (ou oblique à droite) si .

### Aller plus loin : Les moments

La moyenne empirique, la variance empirique, **μ3** et **μ4** sont tous des **moments**.

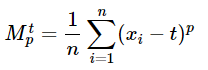
La notion de moment est ici très similaire à celle des moments d'inertie, dont la définition selon [M. Wikipedia](https://fr.wikipedia.org/wiki/Moment_d%27inertie) est la suivante :

Le moment d'inertie est une grandeur physique qui caractérise la géométrie des masses d'un solide, c'est-à-dire la répartition de la matière en son sein. Il quantifie également la résistance à une mise en rotation de ce solide.

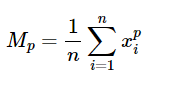
La moyenne, la variance et les mesures de forme que nous avons vu caractérisent la géométrie de la distribution, d'où la similitude avec la définition du moment d’inertie.

Ceux qui étudient la mécanique sont habitués à calculer des moments. Par exemple, si on prend une règle graduée, que l'on attache un poids à chacun des endroits correspondant à des observations **(x1,...,xn)** , puis que l'on fait tourner cette règle autour de la valeur moyenne, alors le moment d'inertie se calculera de la même manière que la variance des **(x1,...,xn)** !

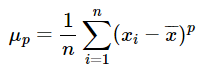
En statistiques, le **moment général empirique** d'ordre **p** par rapport à **t** est donné par la relation



Le **moment simple empirique** est le moment général par rapport à **t=0** :



Le **moment centré empirique** est le moment général par rapport à la moyenne, soit  :



On retrouve ici la formule générale de nos **μ3** et **μ4** , et on voit également que **μ2** est la variance empirique, que **μ1=0** (la somme des écarts à la moyenne est toujours nulle comme nous l'avons vu au chapitre précédent). Enfin, la moyenne est le moment simple d'ordre 1 : .

## Familiarisez-vous avec les mesures de concentration

Bonne nouvelle, nous en avons fini avec cette histoire d’entretien d'embauche et de cet ami qui, au lieu de vous dire clairement combien de temps il faut prévoir pour que vous arriviez à l'heure, vous parle en termes de médiane, moyenne, variance, skewness et tout le tralala !

Revenons sur vos relevés bancaires, et étudions vos dépenses.

Une dépense, c'est une somme d'argent. Cela tombe bien, car les **mesures de concentration** sont le plus souvent utilisées pour des sommes d'argent ! Étudier la concentration d'argent, c'est regarder si l'argent est répartie de manière égalitaire ou pas.

Ce que l'on va regarder, c'est si tout l'argent que vous dépensez se concentre en quelques opérations bancaires, ou si au contraire, il est bien réparti parmi les opérations. Dire que votre argent se concentre sur quelques opérations signifie que généralement, vous faites de très nombreuses petites dépenses, et que parfois, il vous arrive de faire quelques dépenses énormes.

Au contraire, l'argent que vous dépensez est bien réparti si toutes vos opérations bancaires (sortantes) ont à peu près le même montant.

Pour visualiser cela, nous utilisons la **courbe de Lorenz**.

### Les mesures de concentration

#### La courbe de Lorenz

Pour illustrer la courbe de Lorenz, imaginons la population d'un pays. Concentrons-nous sur les personnes qui ont des revenus : ceux qui gagnent de l'argent.

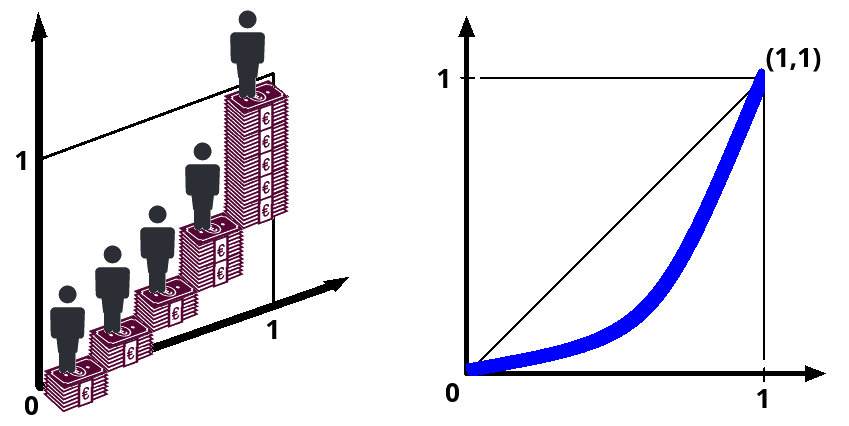
Il faut vous imaginer la courbe de Lorenz comme un podium, non pas avec 3 places, mais avec autant de places que de gens. Ce podium ressemble à un escalier, sur lequel on place l'individu qui gagne le plus d'argent tout en haut, et celui qui gagne le moins d'argent tout en bas.

Seulement, cet escalier n'est pas régulier : la hauteur d'une marche donnée, par rapport à la marche précédente, correspond au revenu de l'individu placé sur cette marche. Ainsi, quelqu'un qui gagne beaucoup d'argent sera placé sur une marche très haute par rapport à celle de la personne en dessous de lui.

Question : quelle sera la hauteur totale de l'escalier ?

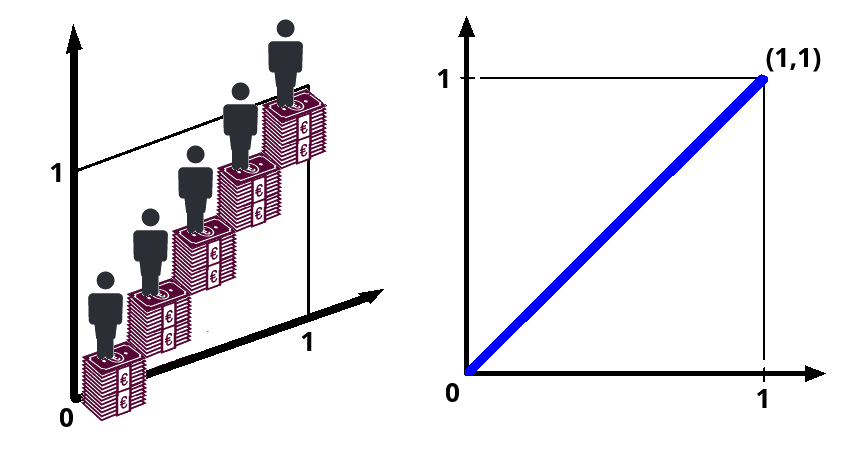
La hauteur de l'escalier est la somme des hauteurs des marches. La somme des hauteurs des marches correspond à la somme de tous les revenus des individus. Par exemple, si 10.000 € ont été distribués parmi la population, la hauteur de l'escalier sera de 10 m (si on considère que 1 m correspond à 1.000 €).

La courbe de Lorenz représente tout simplement cet escalier, à cela près que la hauteur de l'escalier est ramenée à 1, et que la longueur de l'escalier (projetée au sol) est aussi ramenée à 1.



Que se passe-t-il si tous les gens reçoivent la même somme d'argent ?

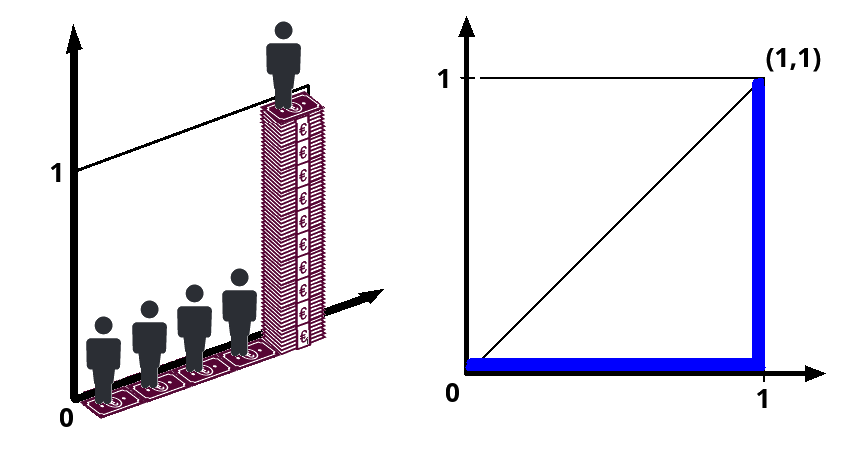
Dans ce cas, la répartition est la plus égalitaire possible. L'escalier se présente comme ceci (à gauche) :



On voit que les marches sont régulières, et que toutes les personnes sont alignées sur une droite appelée première bissectrice, c'est-à-dire qu'elle passe par les points (0,0) et (1,1). Elle est représentée en bleu sur le graphique de droite.

Et si une seule personne concentre en sa possession l'ensemble de la richesse ?

Nous sommes dans l’extrême inverse du cas précédent. Ici, la répartition est la plus inégalitaire possible :



Ici, la courbe de Lorenz ne suit plus du tout la première bissectrice, mais elle s'en éloigne au maximum !

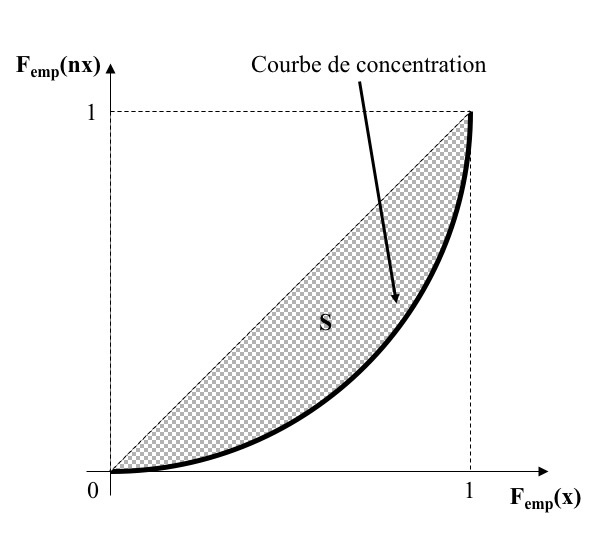
En résumé, plus la courbe de Lorenz est proche de la première bissectrice, plus la répartition est égalitaire.

#### L'indice de Gini

La courbe de Lorenz n'est pas une statistique, c'est une courbe ! Du coup, on a créé l'indice de Gini, qui résume la courbe de Lorenz.

Il mesure l'aire présente entre la première bissectrice et la courbe de Lorenz. Plus précisément, si on note **S** cette aire, alors

**gini=2×S**



Plus la distribution du revenu est concentrée, plus la fraction la plus riche de la population reçoit une grande partie des revenus, et plus la fraction la plus pauvre reçoit une petite partie des revenus. A l'inverse, dans une distribution du revenu parfaitement égalitaire, 10 % de la population reçoit exactement 10 % des revenus totaux.

#### D'autres manières d'exprimer des concentrations

A la radio ou dans les journaux, l'indice de Gini n'est pas très parlant pour le public. Une autre manière d'exprimer les inégalités est plus intelligible, il s'agit de dire :

* Les X % les plus riches possèdent Y % de la richesse mondiale, ou bien
* Les X % les plus riches possèdent autant que les Y % les plus pauvres.

La première de ces deux phrases est de la même forme que la [loi du 80-20](https://fr.wikipedia.org/wiki/Principe_de_Pareto), liée à l'[indice de Pareto](https://fr.wikipedia.org/wiki/Indice_de_Pareto).

### Du côté du code

Voici le code permettant d'afficher la courbe de Lorenz :

import numpy as np

depenses = data[data['montant'] < 0]

dep = -depenses['montant'].values

n = len(dep)

lorenz = np.cumsum(np.sort(dep)) / dep.sum()

lorenz = np.append([0],lorenz) # La courbe de Lorenz commence à 0

plt.axes().axis('equal')

xaxis = np.linspace(0-1/n,1+1/n,n+1) #Il y a un segment de taille n pour chaque individu, plus 1 segment supplémentaire d'ordonnée 0. Le premier segment commence à 0-1/n, et le dernier termine à 1+1/n.

plt.plot(xaxis,lorenz,drawstyle='steps-post')

plt.show()

On y sélectionne tout d'abord le sous-échantillon de travail que l'on appelle depenses  . Comme évoqué plus haut, il faut trier les individus dans l'ordre croissant des valeurs de la variable ; nous le faisons ici grâce à np.sort(dep)0 , car  dep  contient les observations de la variable montant. Ensuite, nous calculons la somme cumulée grâce à np.cumsum() . Pour normaliser et faire en sorte que le haut de la courbe soit à 1, on divise le tout par dep.sum(). La variable lorenz contient les ordonnées des points, mais il nous faut maintenant leurs abscisses : celles-ci s'étendent de 0 à 1 (comme évoqué précédemment) à intervalle réguliers. C'est ce que produit np.linspace(0,1,len(lorenz))  .

Le calcul de l'indice de Gini est un peu plus complexe à comprendre, je laisse les plus courageux s'y plonger ;) :

AUC = (lorenz.sum() -lorenz[-1]/2 -lorenz[0]/2)/n # Surface sous la courbe de Lorenz. Le premier segment (lorenz[0]) est à moitié en dessous de 0, on le coupe donc en 2, on fait de même pour le dernier segment lorenz[-1] qui est à moitié au dessus de 1.

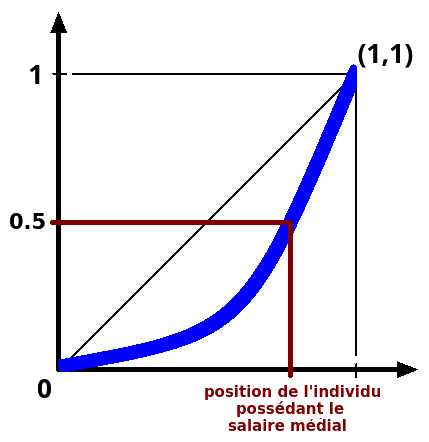
S = 0.5 - AUC # surface entre la première bissectrice et le courbe de Lorenz

gini = 2\*S

gini

### Aller plus loin : La médiale

Nous avons dit que la courbe de Lorenz est un escalier de hauteur 1. Le salaire médial, c'est simplement le salaire de la personne qui se trouve à la moitié de la hauteur : 0,5.



La valeur du salaire **médial** n'est pas directement visible sur la courbe de Lorenz, mais on peut y voir la personne qui a le salaire **médial**. De même, la personne qui a le salaire **médian** est située exactement à 0,5 sur l'axe horizontal.

On savait que 50 % des salaires sont inférieurs au **salaire médian**, et 50 % sont supérieurs. Maintenant on sait que la somme des salaires inférieurs au **salaire médial** vaut 50 % de la somme de tous les salaires, et évidemment, la somme des salaires supérieurs au **salaire médial** vaut 50 % de la somme de tous les salaires.

La somme des salaires est appelée masse salariale.

## Abordez encore plus de mesures

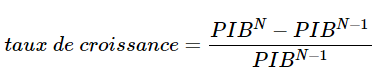
Il existe beaucoup d'autres mesures couramment utilisées en statistiques. Ce chapitre vous en donne quelques-unes supplémentaires.

Ce chapitre n'est pas très important, il dresse juste une liste de mesures statistiques. Il est bon de savoir qu'elles existent, mais pas nécessairement de chercher à les retenir.

### Taux de croissance

Vous entendez souvent parler de la "croissance économique" n'est-ce pas ? La croissance d'un pays, c'est l'augmentation de son [Produit Intérieur Brut](https://www.youtube.com/watch?v=6hURwg4j76U) (PIB) entre une année **N** et l'année précédente **N−1**.

Il est donné par :

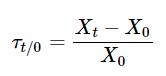


(où  est le **PIB** à l'année **N**)

Si on veut l'exprimer en pourcentage, il suffit de le multiplier par 100.

Par exemple, si un pays a un PIB de 2M€ en 2020, et qu'il monte à 6M€ en 2021, son PIB double. Son taux de croissance sera de 200 %.

On peut généraliser cela à n'importe quelle variable X (au lieu du PIB) et à n'importe quelle durée (au lieu de l'année). Si on note **xt** la valeur observée d la variable **X** à l'instant **t** , alors le taux de croissance (empirique) entre l'instant 0 et l'instant **t** vaut :



Ainsi, si :

* , c'est qu'il y a une hausse de la variable X entre l'instant 0 et **t**.
* , c'est qu'il y a une baisse de la variable X entre l'instant 0 et **t**.

Évitez à tout prix l'erreur classique suivante : "S'il y a une hausse de 100 %, puis une baisse de 100 %, alors on revient à la valeur initiale" ! En effet, la phrase correcte est : "***S'il y a une hausse de 100 %, puis une baisse de 50 %, alors on revient à la valeur initiale***".

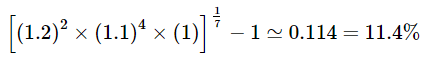
On dit qu'il n'y a pas de symétrie entre hausse et baisse.

### Moyennes

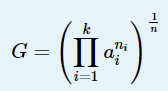
Il existe de nombreuses moyennes différentes. Par défaut, le terme "moyenne" désigne la **moyenne arithmétique**, que nous avons déjà vu dans le chapitre sur les tendances centrales.

Les formules données ici sont données dans le cas de variables discrètes, où les **ai** sont les **k** modalités, et les **ni** les effectifs des modalités **ai**.

Il existe également la **moyenne géométrique**, notamment utilisée dans le calcul du taux de croissance moyen. Elle permet par exemple de calculer la croissance annuelle moyenne du chiffre d'affaires d'une entreprise : si celui-ci augmente de 20 % les 2 premières années puis de 10 % les 4 années suivantes, puis reste stable la dernière année, alors le calcul à réaliser est le suivant :



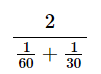
La moyenne géométrique est donnée par la formule suivante :



J'ai aussi en stock la **moyenne harmonique**, notamment utilisée dans le calcul des moyennes de pourcentages et des rapports. Par exemple, si un automobiliste roule à 60 km/h à l’aller et à 30 km/h au retour, alors sa vitesse moyenne n’est pas égale à la moyenne arithmétique (45 km/h) mais à la moyenne  
harmonique (40 km/h) :



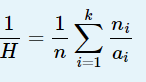
Elle est aussi utilisée quand on veut trouver un bon compromis entre deux mesures **a** et **b** (toutes deux positives) où **a** grandit quand **b** diminue. Pour rester dans les exemples d'automobilistes, disons que **a** désigne la vitesse à laquelle on roule, et **b** une mesure de sécurité de la conduite. A partir d'un certain point, si **a** est trop important, alors la conduite sera très peu sûre et donc **b** sera très bas. On veut trouver un bon compromis entre vitesse et sécurité. On peut le faire par la moyenne harmonique :



Ainsi, si **a** est très proche de 0 ou si **b** est très proche de 0, alors la moyenne harmonique sera aussi proche de 0 : le compromis est mauvais. Si **a** et **b** sont tous les deux assez grands, alors la moyenne harmonique sera grande ; une grande vitesse qui conserve cependant une grande sécurité est un bon compromis.

Si vous faites du machine learning, vous devrez évaluer la performance de vos modèles statistiques. Vous serez confronté à la F-mesure, qui cherche à faire un compromis entre la mesure de rappel et la mesure de précision. [La F-mesure utilise donc la moyenne harmonique du rappel et de la précision](https://openclassrooms.com/courses/evaluez-et-ameliorez-les-performances-d-un-modele-de-machine-learning/evaluez-un-algorithme-de-classification-qui-retourne-des-valeurs-binaires).

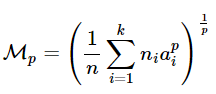
La moyenne harmonique est donnée par la formule suivante :



Pour finir, il y a la **moyenne quadratique**, qui est notamment utilisée pour évaluer la performance d'un modèle statistique. Vous pouvez la retrouver [ici](https://openclassrooms.com/courses/evaluez-et-ameliorez-les-performances-d-un-modele-de-machine-learning/evaluez-un-algorithme-de-regression), sous le nom de RMSE (Root Mean Square Error).

### Aller plus loin : La formule générale d'une moyenne

La formule générale d'une moyenne est donnée par :



On remarque ici que :

*  est la moyenne arithmétique  ;
*  est la moyenne quadratique ;
* est la moyenne harmonique **H** .

# Les mesures statistiques

Bravo ! Vous avez réussi cet exercice !

### Compétences évaluées

* Utiliser les principales mesures statistiques

### Question 1

**Voici les valeurs observées pour une variable donnée :**{1,10,1,10,1,1,5,1,5,5}{1,10,1,10,1,1,5,1,5,5} **Quelles phrases sont correctes ?**

*Attention, plusieurs réponses sont possibles.*

* + 

La moyenne est 4.

* + 

La médiane est 3.

* + 

Le mode est 1

*La moyenne est calculée par (1+10+1+10+1+1+5+1+5+5)/10 = 4, la médiane se situe à mi-chemin entre la 5e et la 6e valeur, soit à mi-chemin entre 1 et 5, soit (1+5)/2 = 3. Le mode est la valeur la plus fréquente : 1.*

### Question 2

**Soit ces deux distributions :**A={6,4,6,4,6,4,6,4}A={6,4,6,4,6,4,6,4}B={1,4,1,4,1,4,1,4}B={1,4,1,4,1,4,1,4} **Laquelle a la plus grande variance?**

* + 

A

* + 

B

*La variance mesure les écarts à la moyenne. La distribution A a une moyenne de 5, et toutes ses valeurs s'en éloignent de 1. La distribution B a une moyenne de 2,5, et toute ses valeurs s'en éloignent de 1,5.*

### Question 3

**Une boîte à moustaches, c'est :**

* + 

une entreprise dans laquelle les employés ont interdiction de se raser

* + 

une représentation de la distribution d'une variable

*C'était un jeu de mots...*

### Question 4

**Le "skewness" est une mesure :**

* + 

de concentration

* + 

de forme

* + 

de dispersion

* + 

de tendance centrale

*Le skewness est un indicateur de la forme de la distribution.*

### Question 5

**Pour un jour donné, dans une région donnée, s'il ne pleut que dans un seul village et qu'il ne pleut pas dans les autres, quel sera l'indice de Gini de la variable "pluviométrie" dans l'échantillon des villages de la région ?**

* + 

0

* + 

0,5

* + 

1

* + 

2

*L'indice de Gini mesure les inégalités de concentration d'une quantité. Ici, la quantité étudiée est l'eau tombée en 1 jour. Un indice de Gini à 1 signifie la répartition la plus inégalitaire qui soit : celle où un village a reçu toute l'eau tombée.  
Remarque : l'indice de Gini est toujours compris entre 0 et 1.*

### Question 6

**Que dire d'une distribution dont le skewness est égal à 0 ?**

* + 

La distribution est symétrique

* + 

L'écart-type de la distribution est lui aussi égal à 0

*Le skewness est une mesure de symétrie*

## Entrez dans le monde de l’analyse bivariée

Vous êtes maintenant capable d’étudier toutes les variables l’une après l’autre. Bravo, vous êtes donc un pro de l’analyse **univariée**.

J’ai étudié toutes les variables, c'est tout bon ! Mais alors pourquoi il reste des chapitres à ce cours ? :o

Et bien non, ce n’est pas fini. Dans cette troisième partie, nous allons étudier les relations entre variables. C’est l’analyse **bivariée**. Pour certains chapitres, il faudra vous accrocher. Mais si vous êtes arrivé jusqu'ici, vous ne devriez normalement pas avoir trop de soucis. De plus, c'est à partir d'ici que l'analyse de vos relevés de comptes devient intéressante !

### Pourquoi l'analyse bivariée ?

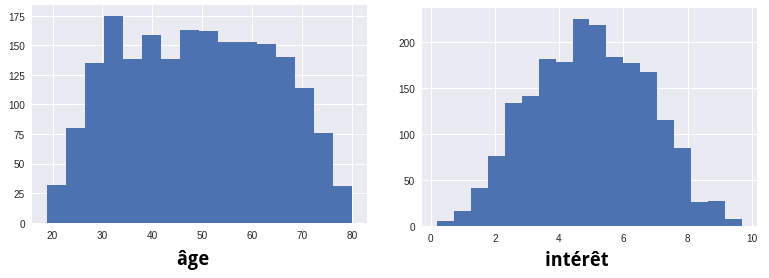
Pourquoi étudier les relations entre variables ?

Petit exemple. Vous travaillez pour un site web de e-commerce. Vous avez accès à la base de données des clients du site, ainsi qu'aux données de navigations sur le site. Grâce aux données de navigation, vous pouvez savoir quel client a consulté quelle page sur le site, combien de temps il a passé sur chaque page, etc. Dans le but de créer un algorithme de recommandation (qui proposera de nouveaux produits aux clients), vous décidez de faire une petite étude préliminaire.

Grâce aux données de navigation, vous pouvez sélectionner un échantillon de clients qui consultent souvent les derniers albums musicaux de variété française. Vous décidez alors de déterminer l'intérêt qu'ils portent au nouvel album d'un chanteur populaire, en modélisant cet intérêt par un score allant de 0 à 10 sur une échelle continue. Si un client donné n'a jamais visité la page qui présente ce nouvel album, vous lui attribuez le score intermédiaire de 5. S'il a souvent visité la page de cet album, qu'il y est resté longtemps, et qu'il a finalement acheté l'album vous lui attribuez le score de 10. Au contraire, s'il a consulté la page, qu'il n'y est pas resté longtemps, et qu'il n'a pas acheté l'album lors de sa dernière commande sur le site, alors c'est probablement qu'il semble ne pas aimer ce nouvel album. Vous lui attribuez donc le score de 0.

Vous connaissez l'âge de chaque client. Vous obtenez donc un échantillon de clients caractérisés par 2 variables : l'âge et le niveau d'intérêt.

Vous décidez donc d'étudier ces 2 variables séparément, avec des histogrammes :



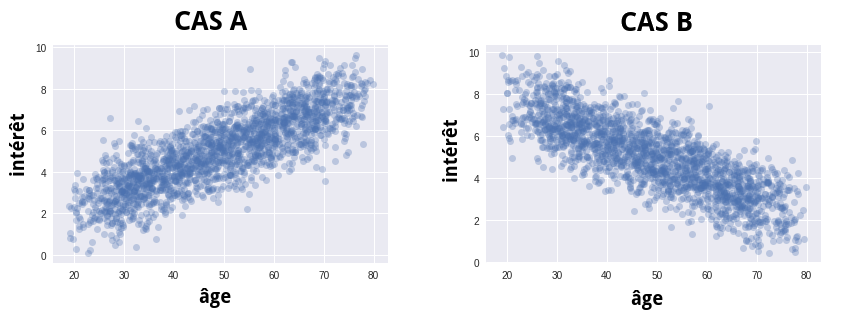
Ces histogrammes montent que les âges sont assez bien répartis sur cet échantillon : il y a à peu près autant de personnes jeunes que de personnes plus âgées. Quant au niveau d'intérêt, il y a également autant de personnes qui s'intéressent au nouvel album que de personnes qui n'y portent pas d'intérêt.

Bon. C’est déjà bien de savoir cela, mais nous allons voir que nous pouvons faire beaucoup mieux !

Maintenant, plaçons sur un graphique en 2 dimensions les individus de notre échantillon. Chaque point de ce graphique représente une personne. La position de chaque point peut être repérée selon 2 axes : l’axe des abscisses (horizontal), et l’axe des ordonnées (vertical). L’abscisse d’un point, c’est un nombre. Si ce nombre est grand, alors ce point sera très à droite du graphique, mais s’il est proche de 0, alors il sera très à gauche. C’est pareil pour l’ordonnée du point : si elle est élevée, le point sera très haut, mais si elle est proche de 0, le point sera très bas.

Ici, on place la variable âge en abscisse et celle du niveau d'intérêt en ordonnée. Un point qui sera en haut à droite représentera donc une personne plutôt âgée très intéressée par le nouvel album. Au contraire, un point qui sera en bas à gauche représentera une personne jeune n'aimant pas l'album.

En fait, plusieurs cas sont possibles. Voici 2 exemples extrêmes :



Remarquons qu'aucun de ces 2 graphiques n'entre en contradiction avec les 2 histogrammes obtenus plus haut !

**Dans le cas A**, beaucoup de personnes âgées aiment ce nouvel album, et beaucoup de personnes jeunes ne l'aiment pas. Ainsi, votre algorithme de recommandation devra conseiller ce nouvel album aux personnes plutôt âgées, et ne pas le recommander aux personnes jeunes (mieux vaudra leur proposer des produits qu'elles sont plus susceptibles d'aimer).

**Dans le cas B**, c’est l’inverse. Il faut conseiller cet album aux personnes jeunes et ne pas le faire pour les personnes âgées.

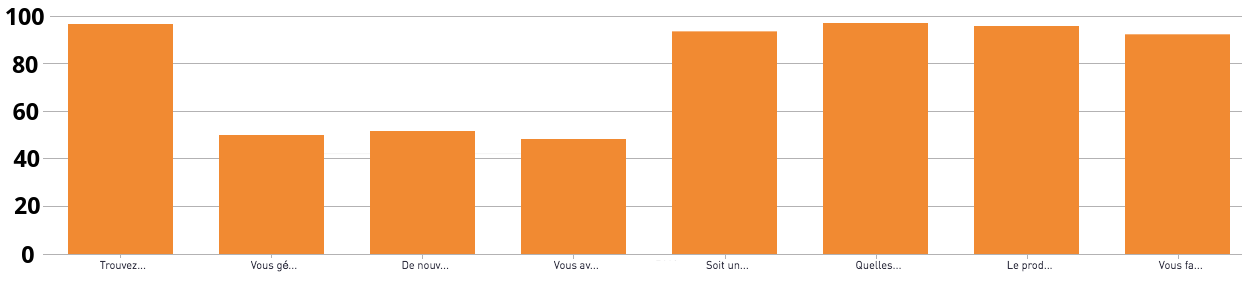
Chacun de ces 2 graphiques est appelé « diagramme de dispersion » ou « scatter plot » en anglais.

Vous l’aurez compris, on obtient en général beaucoup plus d’information en étudiant les relations entre 2 variables qu’en étudiant séparément 2 variables ! Sans l’analyse bivariée, vous auriez été incapable de savoir à qui recommander (ou non) l'album !

Ici, étudier la relation entre les 2 variables, c’est se poser la question suivante : « Sachant qu’une personne est plutôt jeune (ou plutôt âgée), a-t-elle plus de chances d’aimer ce nouvel album ? »

### Aller plus loin : Un autre exemple

Un autre exemple : un célèbre site de formations en ligne publie des cours, dans lesquels les étudiants doivent répondre à des quiz. Pour réussir un quiz, il faut 70 % de réponses correctes. Pour un quiz de 8 questions, il faut donc répondre correctement à au moins 6 questions pour réussir. L’échantillon des étudiants ayant répondu au quiz est un échantillon à 8 variables. Elles sont toutes binaires (réponse bonne/réponse fausse). Pour l’un des quiz du cours intitulé « [Initiez-vous à l’algèbre relationnelle avec SQL](https://openclassrooms.com/courses/initiez-vous-a-lalgebre-relationnelle-avec-le-langage-sql) », voici ces 8 variables représentées :



Considérons que ces 8 questions sont numérotées de 1 à 8, de gauche à droite.

5 questions sur 8 ont un taux de réussite proche de 100 %. Les 3 autres questions ont un taux de réussite proche de 50 %. Ce graphique montre 8 analyses univariées. Mais ici, il nous faudrait étudier les relations entre ces variables. En effet, parmi les 50 % d’étudiants qui ont raté la question 2, je ne sais pas combien ont réussi la question 3, et c’est problématique car :

* Si les 50 % qui ont raté la question 2, les 50 % qui ont raté la 3, et les 50 % qui ont raté la 4 sont les mêmes étudiants, alors cela signifie que 50 % d’étudiants au total ont raté le test (avec chacun 3 réponses fausses). Le taux de réussite globale au test est donc de 50 %, et il faudrait alors simplifier l’énoncé du quiz.
* Si cependant les 50 % qui ont raté la question 2 sont tous parmi les 50 % qui ont réussi la question 3, alors ceux-ci auront probablement tous réussi le quiz (quel que soit leur résultat à la question 4, ils auront presque tous un score global de 6/8 ou 7/8). Ainsi, le taux de réussite globale du quiz sera proche de 100 %, ce qui est un bon taux !

Ici, étudier les relations entre les variables, c’est se demander par exemple : « Sachant qu’un individu a eu faux à la question 2, a-t-il de grandes chances d’avoir répondu faux ou vrai à la question 3 ? »

## Recherchez les corrélations

Vous l’aurez compris, étudier les relations entre les variables, c’est important.

Pour être plus formel, la notion de relation entre variables est appelée **corrélation**. Dire que deux variables sont corrélées signifie que si on connaît la valeur d’une variable, alors il est possible d’avoir une indication (plus ou moins précise) sur la valeur d’une autre variable.

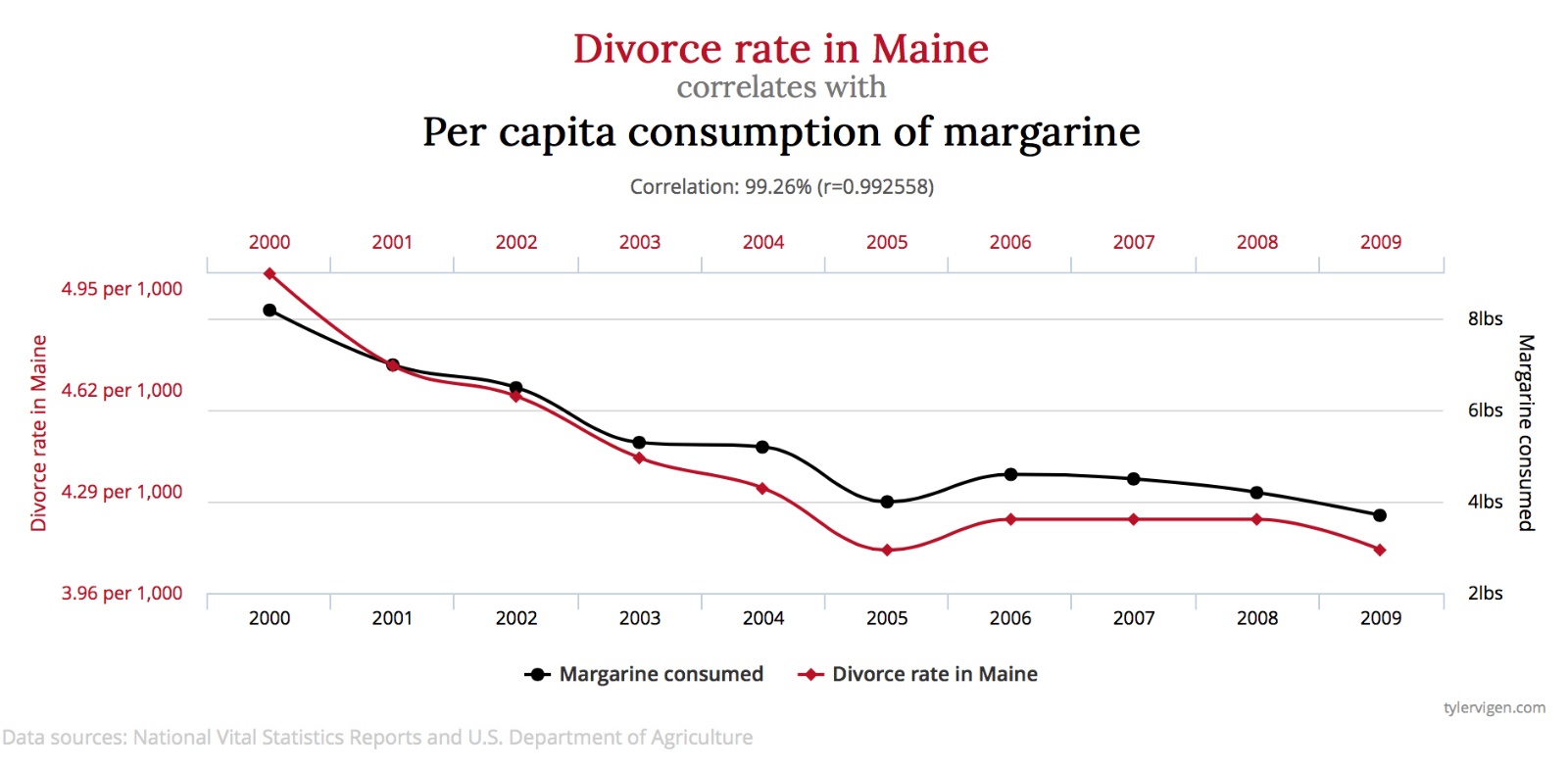
Dans le chapitre précédent, on étudiait déjà des corrélations, en se posant la question : sachant l'âge d'un individu, peut-on avoir une indication plus précise de la valeur de la variable « intérêt pour un album donné » ?

### La causalité

Dès que l’on parle de corrélation, il y a une erreur, très fréquente, à ne JAMAIS commettre : celle de dire qu’il y a un lien de cause à effet d’une variable sur l’autre.

S’il y a corrélation entre une variable **A** et une variable **B**, est-ce **A** qui est la cause de **B** , ou **B** qui est la cause de **A** ? Il est souvent impossible de le savoir sans effectuer une expérimentation. Le plus souvent, c’est un troisième (ou plusieurs) facteur **C** (qui n’est d’ailleurs pas toujours observé) qui est la cause de **A** et de **B**.

Il est aussi possible que 2 variables soient corrélées sans qu’il n’y ait aucun lien entre elles. On les appelle les corrélations fallacieuses (« spurious correlations » en anglais).



Pour en voir d’autres, c’est [**par ici**](http://www.tylervigen.com/spurious-correlations). Certaines sont amusantes !

En fait, pour avoir le droit d’établir un lien de cause à effet entre des variables, il faut construire une expérimentation qui respecte certaines conditions. Ces conditions ne sont en général pas vérifiées lorsque votre échantillon ne provient pas de cette expérimentation conçue spécialement.

Si vous aimez les paradoxes mathématiques, il y en a un qui traite du lien de cause à effet, c’est le paradoxe de Simpson. Tout bon Data Analyst devrait être conscient de ce paradoxe. Une vidéo très bien expliquée et très compréhensible a été réalisée dans la [série Science Étonnante](https://www.youtube.com/watch?v=vs_Zzf_vL2I).

A la minute 7:58 de cette vidéo, vous apprendrez par exemple que si 2 variables sont corrélées, agir sur la valeur de l'une des variables ne fait pas forcément changer la seconde. ;)

Vous verrez, ce paradoxe vous bluffera !

### Réfléchissons...

Comme souvent dans ce cours, je vais une fois de plus faire appel à votre imagination. Il est toujours bon d’avoir de l’imagination.

Aujourd’hui, il vous prend la soudaine envie de faire des statistiques sur les habitants de votre ville. Vous souhaitez connaître leurs boissons préférées parmi celles-ci : café, thé, eau, ou autre.

Vous réalisez donc votre enquête en vous rendant dans les cafés pour observer discrètement les clients et noter la boisson qu’ils ont commandé. Vous souhaitez rassembler un échantillon de 100 personnes. Pour chacune d’elle, vous avez noté la boisson commandée et le nom du café dans lequel vous l’avez observée. On appellera ces deux variables nom café et boisson préférée. Voici la distribution que vous obtenez pour la variable boisson préférée :

* café : 50 personnes sur 100, soit fcafefcafe = 50 %
* thé : 30 personnes sur 100, soit fthefthe = 30 %
* autres boissons : 20 personnes sur 100, soit fautrefautre = 20 %

Vous continuez votre enquête en vous rendant dans un café où il y a 10 clients. Combien de personnes vous attendez-vous à voir face à un thé ? Intuitivement, vous vous attendez à trouver 3 personnes qui ont commandé un thé car vous savez qu’en général, 30 % des personnes commandent un thé. Vous avez donc réalisé le calcul suivant : 30%\*10 = 3.  
De même, vous vous attentez à voir 5 personnes avec un café, et 2 personnes avec d’autres boissons.

A votre grande surprise, il y a en fait 9 personnes avec du thé, et seulement 1 avec un café ! Cela diffère beaucoup de ce à quoi vous vous attentiez : il y a 90 % de personnes qui boivent du thé. C'est peut-être un hasard, alors vous décidez de revenir régulièrement pour savoir si ce 90 % se confirme de jour en jour ou pas. Effectivement, ce pourcentage reste à peu près constant même après de nombreuses observations !

Mais vous comprenez vite pourquoi en regardant le nom du café : « Salon de thé Chez Luc ». Vous êtes dans un café un peu spécial : c’est un salon de thé ! Les clients qui fréquentent cet endroit sont donc principalement des amateurs de thé.

On dit alors que le fait d’aimer le thé et le fait de fréquenter le salon de thé Chez Luc ne sont pas **indépendants**. Si deux événements ne sont pas indépendants, alors on s’attend à trouver une corrélation entre ceux-ci. Vous souvenez de la question que l'on se pose pour les corrélations : Sachant que l'on connaît la valeur d'une variable, peut-on avoir une indication un peu plus précise sur la valeur d'une autre variable ?

Sachant qu’une personne fréquente le café Salon de thé Chez Luc, peut-on avoir une indication un peu plus précise sur sa boisson préférée ?

La réponse est oui ! Sans connaître la valeur de la variable nom café, alors on suppose que la variable boisson préférée suivra cette distribution : 50 % pour le café, 30 % pour le thé et 20 % pour les autres boissons. **MAIS**, si on connaît la valeur de la variable nom café (ici : Salon de thé Chez Luc), alors on peut avoir une meilleure indication sur la variable boisson préférée ; ici on s'attendra à trouver bien plus que 30%\*10=3 personnes devant une tasse de thé.

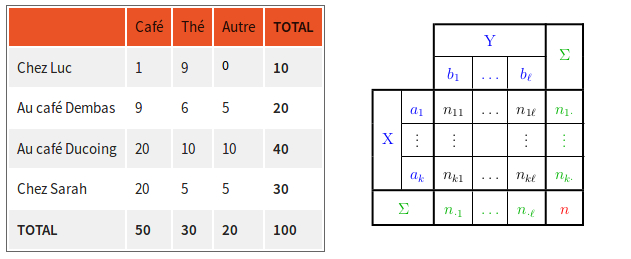
Appelons **I** l'événement "préférer le thé" et **J** l'événement "être au Salon de thé Chez Luc". Voici ce qu'il faut retenir :

Si deux événements **I** et **J** sont indépendants, alors on s'attend à ce que le nombre d'individus qui satisfont à la fois **I** et **J** (appelons ce nombre ) soit égal à  (c'est le calcul que vous aviez fait en entrant dans le bar : 30%\*10=3). Au contraire, plus  sera différent de , plus on aura de raison de penser que **I** et **J** ne sont pas indépendants.

Quand vous avez vu que ****était en fait égal à 9, vous vous êtes dit que le fait d'être au Salon de thé Chez Luc n'était pas indépendant du fait d'aimer le thé !

### Le tableau de contingence

On peut résumer tout cela dans un tableau appelé **tableau de contingence** (où X = nom café et Y = boisson préférée) :



On voit en première ligne les effectifs de Chez Luc : 1, 9 et 0. On retrouve aussi très facilement les fréquences sur la population globale en dernière ligne : 50 % (café), 30 % (thé), 20 % (autres).

Allez, un peu de vocabulaire pour finir ce chapitre :

* Chacune des valeurs du tableau de contingence (hors colonnes TOTAL) est appelée **effectif conjoint **.
* L'ensemble effectifs conjoints est appelé **distribution conjointe empirique** de (nom café, boisson préférée).
* La dernière ligne (TOTAL) est appelée **distribution marginale empirique** de boisson préférée, et la dernière colonne (TOTAL) est appelées **distribution marginale empirique** de nom café.
* L'ensemble des effectifs conjoints de la première ligne (Chez Luc) est appelée **distribution conditionnelle empirique de boisson préférée étant donné que nom café = Chez Luc**.

## Analysez la corrélation entre deux variables quantitatives

Jusqu'à maintenant, nous avons vu 2 manières de présenter des données en analyse bivariée : le [diagramme de dispersion](https://openclassrooms.com/courses/nettoyez-et-decrivez-votre-jeu-de-donnees/entrez-dans-le-monde-de-lanalyse-bivariee#r-4727902) (scatterplot), et le [tableau de contingence](https://openclassrooms.com/courses/nettoyez-et-decrivez-votre-jeu-de-donnees/recherchez-les-correlations#r-4773286).

Le premier est adapté quand les 2 variables sont quantitatives, et le second est adapté quand les 2 variables sont qualitatives.

Et quand il y a une variable quantitative et une variable qualitative ?

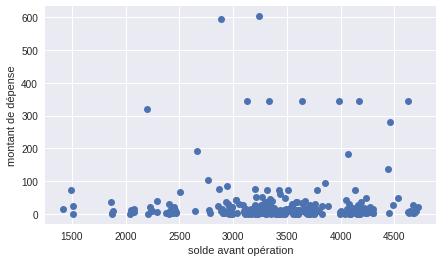
C'est bien, vous avez trouvé le 3e cas possible ! Il reste 4 chapitres à ce cours, durant lesquels nous parcourrons ces 3 cas : ce chapitre et le suivant seront dédiés à l'étude de 2 variables quantitatives, il y aura ensuite un chapitre pour le cas d'une variable quantitative et l'autre qualitative, et le tout dernier sera dédié à l'étude de deux variables qualitatives. Prêts ? C'est parti !

### 2 variables quantitatives : graphiques

Posons-nous la question suivante :

**Êtes-vous moins dépensier lorsque vous avez peu d'argent sur votre compte ?**

Vous l'aurez deviné, les 2 variables à étudier sont : montant et solde\_avt\_operation. Rechercher une corrélation entre ces variables revient à dire : "Sachant que le solde de votre compte est petit, peut-on s'attendre à ce que le montant de l'opération soit lui aussi petit ?" (Ou l'inverse).



A priori sur ce diagramme de dispersion, il ne semble pas que quand le solde est petit, les montants soient particulièrement petits. Il semble ne pas y avoir de corrélation. Mais vous en trouverez peut-être une dans vos propres relevés ! Cependant, les points sont assez dispersés et nombreux. Il est donc difficile d'y voir très clair. Pour remédier à cela, il existe une représentation qui peut être meilleure. Elle est donnée dans la section Aller plus loin, tout en bas ;).

### 2 variables quantitatives : indicateurs numériques

C'est bien beau les graphiques, mais je sens que vous êtes en manque de calcul ! Il nous faut un indicateur numérique qui puisse nous dire si les variables sont corrélées ou pas.

Ici, on veut savoir si quand on a un solde (**X**) petit, on a aussi un montant (**Y**) petit. Mais petit par rapport à quoi ? Ici, quand on dit "petit", c'est par rapport aux autres valeurs, donc on veut dire "plus petit que la moyenne". Prenons une opération bancaire (= un individu) au hasard, et notons **x** la valeur du solde avant opération, et **y** le montant de l'opération. Pour mesurer si x est plus petit que la moyenne , on peut calculer

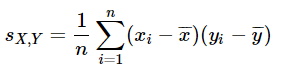


Cette quantité sera négative si x est inférieur à   , et positive dans le cas contraire. De même, on peut calculer pour comparer **y** à la moyenne . Maintenant, multiplions-les !



Si **x** est plus petit que la moyenne et que **y** est plus petit que la moyenne, alors les deux termes seront négatifs. Quand on multiplie deux nombres négatifs, on obtient un nombre positif. C'est aussi valable dans l'autre sens : si **x** est supérieur à la moyenne et **y** aussi, alors **a** sera aussi un nombre positif.

OK, avec cette multiplication, on obtient la quantité **a** pour une seule opération bancaire (un seul individu). Mais si les montants sont vraiment petits quand le solde est petit (et inversement), alors les **a** de toutes les opérations seront positifs ! Et si on fait la moyenne de tous ces **a**, alors on obtiendra encore un nombre positif. La moyenne de tous ces **a** s'écrit comme ceci :



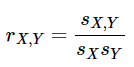
Au contraire, s'il n'y a pas de corrélation entre le solde et le montant, alors quand **x** sera petit, **y** ne sera pas forcément petit. **y** sera parfois petit, parfois grand, donc sera tantôt positif, tantôt négatif. Ainsi, **a** sera elle aussi tantôt positive, tantôt négative. Au final, la moyenne de tous les **a** sera proche de 0.

Résumons : Si **x** est petit quand **y** est petit (et inversement), alors sera positif. Si **X** et **Y** ne sont au contraire pas corrélés,  sera plutôt proche de 0. Pour les motivés, vous pouvez aussi déduire que si **x** est grand quand **y** est petit (et inversement), alors sera négatif. Dans ce dernier cas, il y a corrélation certes, mais on dit que c'est une corrélation négative.

#### La covariance empirique et le coefficient de corrélation

Devinez quoi ! L'indicateur que nous venons de construire est très utilisé en statistiques ; il s'appelle la **covariance empirique** de X et Y. Ce terme vous rappelle la variance empirique ? C'est normal : elles sont similaires. Effectivement, si vous calculez la covariance empirique de X et Y, vous retombez sur la formule de la variance empirique de X, qui s'écrit  . Magique !

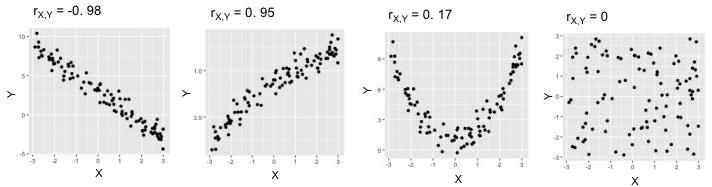
Pour ramener la covariance empirique à une valeur qui soit comprise entre -1 et 1, alors on peut la diviser par le produit des écarts-types. Cette normalisation nous permet de faire des comparaisons. Ce qui nous donne



Ce coefficient r est appelé **coefficient de corrélation**, ou **coefficient de corrélation linéaire**, ou encore **coefficient de corrélation de Pearson**.

Pourquoi "linéaire" ?

Parce que malheureusement, il ne détecte les relations que lorsqu'elles sont **linéaires**, c'est-à-dire lorsque les points sont plutôt bien alignés sur une ligne droite. Sur le graphique ci-dessous, les deux schémas du haut montrent des points bien alignés : leur **r** est donc proche de 1 ou de -1. Sur le 4e graphique en revanche, il n'y a pas vraiment de corrélation (connaître la valeur du **x** d'un point ne nous donne aucune indication sur la valeur de **y**) : **r** est donc proche de 0. Cependant sur le 3e graphique, il y a une forte corrélation, mais sa forme n'est pas linéaire, et **r** est donc malheureusement proche de 0.



### Du côté du code

Voici le code du diagramme de dispersion du haut de ce chapitre :

import matplotlib.pyplot as plt

depenses = data[data.montant < 0]

plt.plot(depenses["solde\_avt\_ope"],-depenses["montant"],'o',alpha=0.5)

plt.xlabel("solde avant opération")

plt.ylabel("montant de dépense")

plt.show()

Pour calculer le coefficient de Pearson et la covariance, 2 lignes suffisent !

import scipy.stats as st

import numpy as np

print(st.pearsonr(depenses["solde\_avt\_ope"],-depenses["montant"])[0])

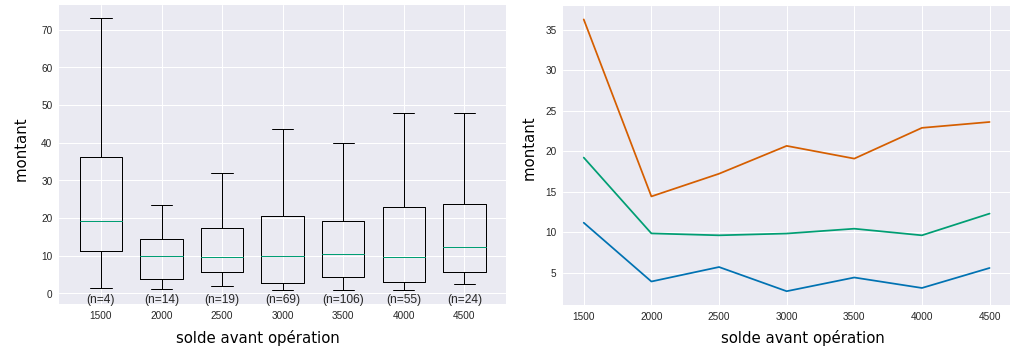
print(np.cov(depenses["solde\_avt\_ope"],-depenses["montant"],ddof=0)[1,0])

Le coefficient de corrélation linéaire se calcule grâce à la méthode st.pearsonr . On lui donne ensuite les 2 variables à étudier. Remarquez que dans ce chapitre, on préfère ramener les dépenses en montants positifs, d'où le signe - devant depenses["montant"]. Un couple de valeurs est renvoyé, le coefficient de corrélation est le premier élément de ce couple, d'où le [0] à la fin de la ligne 4.

La méthode np.cov renvoie la *matrice de covariance*, que vous n'avez pas à connaître à ce niveau. Cette matrice est en fait un tableau, et dans ce dernier, c'est la valeur située sur la 2e ligne à la 1e colonne, d'où le [1,0]  .

### Aller plus loin : Alternative au diagramme de dispersion

Pour avoir une représentation plus efficace que le scatter plot, il est possible d'agréger la variable X en abscisses (axe horizontal) en différentes classes. Cela équivaut à "découper" au couteau le graphique précédent en tranches verticales. On représente ensuite pour chaque tranche une boîte à moustaches calculée à partir de tous les points présents dans la tranche. Voici donc le nouveau graphique obtenu :



Sur le graphique de gauche, on ne peut pas vraiment dire que plus le solde est petit, plus le montant est petit, même si les boîtes à moustaches des tranches [2000;2500[ et [2500;3000[ semblent légèrement moins dispersées vers le haut.

La boîte à moustaches la plus à gauche est très dispersée : cela peut sembler étonnant, mais elle n'est en fait pas très représentative car elle ne représente que 4 individus sur une population qui en contient presque 300. On remarque qu'il est donc important d'afficher les effectifs de chaque classe (n=4, n=14, n=19, etc.)

A droite, c'est un graphique équivalent qui représente les 1e, 2e et 3e quartiles pour chaque tranche (chaque classe). Il est superposable à celui des boîtes à moustaches, car les 3 quartiles sont aussi visibles sur les boîtes à moustaches.

taille\_classe = 500 # taille des classes pour la discrétisation

groupes = [] # va recevoir les données agrégées à afficher

# on calcule des tranches allant de 0 au solde maximum par paliers de taille taille\_classe

tranches = np.arange(0, max(depenses["solde\_avt\_ope"]), taille\_classe)

tranches += taille\_classe/2 # on décale les tranches d'une demi taille de classe

indices = np.digitize(depenses["solde\_avt\_ope"], tranches) # associe chaque solde à son numéro de classe

for ind, tr in enumerate(tranches): # pour chaque tranche, ind reçoit le numéro de tranche et tr la tranche en question

montants = -depenses.loc[indices==ind,"montant"] # sélection des individus de la tranche ind

if len(montants) > 0:

g = {

'valeurs': montants,

'centre\_classe': tr-(taille\_classe/2),

'taille': len(montants),

'quartiles': [np.percentile(montants,p) for p in [25,50,75]]

}

groupes.append(g)

# affichage des boxplots

plt.boxplot([g["valeurs"] for g in groupes],

positions= [g["centre\_classe"] for g in groupes], # abscisses des boxplots

showfliers= False, # on ne prend pas en compte les outliers

widths= taille\_classe\*0.7, # largeur graphique des boxplots

)

# affichage des effectifs de chaque classe

for g in groupes:

plt.text(g["centre\_classe"],0,"(n={})".format(g["taille"]),horizontalalignment='center',verticalalignment='top')

plt.show()

# affichage des quartiles

for n\_quartile in range(3):

plt.plot([g["centre\_classe"] for g in groupes],

[g["quartiles"][n\_quartile] for g in groupes])

plt.show()

### Aller plus loin : Propriétés de la covariance empirique

Très rapidement, voici deux propriétés de la covariance empirique :

* . C'est la propriété de **symétrie**.
* Si on crée une nouvelle variable Z à partir de 2 variables U et V dont on connaît la covariance empirique, et que  , alors . C'est la propriété de **bi-linéarité**

## Analysez deux variables quantitatives par régression linéaire

Un très bon exemple introductif d'une régression linéaire est donné  dans [**cette vidéo**](https://www.youtube.com/watch?v=trWrEWfhTVg), entre 1:28 et 3:00

Nous avons étudié au chapitre précédent la corrélation entre 2 variables quantitatives. Continuons ici avec 2 autres variables également quantitatives : attente et montant.

La variable attente est renseignée seulement pour les opérations bancaires de la catégorie COURSES. Lors d’un précédent chapitre, vous avez peut-être renseigné une catégorie COURSES sur certaines de vos opérations bancaires. Si ce n’est pas le cas, je vous invite à y refaire [un tour](https://openclassrooms.com/courses/nettoyez-et-decrivez-votre-jeu-de-donnees/telechargez-les-donnees) pour y télécharger l’échantillon enrichi operations\_enrichies.csv, ou bien si vous êtes motivé, à reprendre vos 20 derniers tickets de caisse pour recréer un échantillon à la main !

La variable attente d’une opération donne le nombre de jours écoulés entre celle-ci et la précédente opération de catégorie COURSES. Si vous faites vos courses tous les 7 jours en moyenne, alors la moyenne de attente sera de 7.

Que s’attend-t-on à trouver ?

En théorie, plus vous attendez pour aller faire les courses, plus vous aurez besoin d’acheter de provisions. On s’attend donc à ce que plus la valeur de attente est grande, plus la valeur du montant soit grande.



### Étape préliminaire

Tout d'abord, il faut calculer la variable attente ! Je vous donne le code, vous n'avez pas forcément besoin de le comprendre :

import datetime as dt

# Selection du sous-échantillon

courses = data[data.categ == "COURSES"]

# On trie les opérations par date

courses = courses.sort\_values("date\_operation")

# On ramène les montants en positif

courses["montant"] = -courses["montant"]

# calcul de la variable attente

r = []

last\_date = dt.datetime.now()

for i,row in courses.iterrows():

days = (row["date\_operation"]-last\_date).days

if days == 0:

r.append(r[-1])

else:

r.append(days)

last\_date = row["date\_operation"]

courses["attente"] = r

courses = courses.iloc[1:,]

# on regroupe les opérations qui ont été effectués à la même date

# (courses réalisées le même jour mais dans 2 magasins différents)

a = courses.groupby("date\_operation")["montant"].sum()

b = courses.groupby("date\_operation")["attente"].first()

courses = pd.DataFrame({"montant":a, "attente":b})

On crée ici un sous échantillon qui ne contient que les opérations de catégorie courses, et que l'on appelle... courses !

### Modélisons !

Mais nous allons faire mieux que cela : calculer le prix moyen des produits que vous consommez en 1 jour, ainsi que la vitesse à laquelle vous accumulez du stock dans vos placards ! Pour cela, nous allons utiliser un modèle. Vous allez voir, c’est très puissant.

Pour le modèle que nous allons créer, nous allons faire plusieurs suppositions. Tout d’abord, nous supposons qu’à chaque fois que vous faites les courses, vous achetez 3 types de produits :

1. Les produits que vous consommerez avant la prochaine fois que vous irez faire les courses (produits alimentaires, d’hygiène, etc.)
2. Les produits qui ne seront pas consommés durant la durée de l’étude (la durée de l’étude étant la période entre votre 1er ticket de caisse enregistré dans l’échantillon et le dernier) : ce sont vos stocks de long terme (boîtes de conserves, produits surgelés, etc.)
3. Les produits qui ne sont pas des consommables (ex : une fourchette, une serpillière, etc.), que vous n’achetez que très rarement.

Ensuite, nous supposons que vous consommez chaque jour des produits, et que le prix des produits que vous consommez en 1 jour est à peu près constant.

Appelons **a** le prix moyen des produits consommés en un jour (ceux de type 1), et **b** le prix moyen des produits de type 2 et 3 rassemblés que vous achetez à chaque course. Enfin, appelons **x** le nombre de jours que vous avez attendu depuis vos dernières courses, et **y** le montant du ticket de caisse.

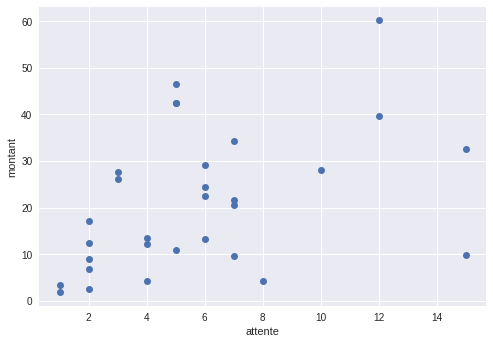
Quel sera le prix de votre prochain ticket de caisse ?

Il sera égal au nombre de jours d’attente multiplié par le prix moyen ce que vous consommez en 1 jour. Mais en plus, il faut ajouter le prix moyen des produits de type 2 et 3. Ceci donne cette formule :



C'est un peu simpliste ton truc, mon prochain ticket ne vaudra pas exactement ce montant. Je ne consomme pas tous les jours exactement la même somme d'argent, et je n'achète pas à chaque fois la même quantité de stock ! Et imagine que j'aie envie de me faire plaisir en m'achetant des produits plus chers !

C'est vrai, c'est simpliste ! Cette équation n'est pas exacte. D'ailleurs, vous aurez peut-être remarqué qu'il s'agit d'une équation d'une droite (remémorez-vous les [fonctions affines](https://fr.wikipedia.org/wiki/Fonction_affine)). Équation de droite signifie que si je prends tous les **x** possibles compris entre (par exemple) 0 et 5, puis que je calcule tous leurs **y** associés, avant de les placer sur un graphique avec les **x** sur l'axe horizontal et les **y** sur l'axe vertical, alors tous les points seront parfaitement alignés ! Essayons donc d'afficher le diagramme de dispersion avec X = attente et Y = montant, et regardons si tous les points sont alignés :



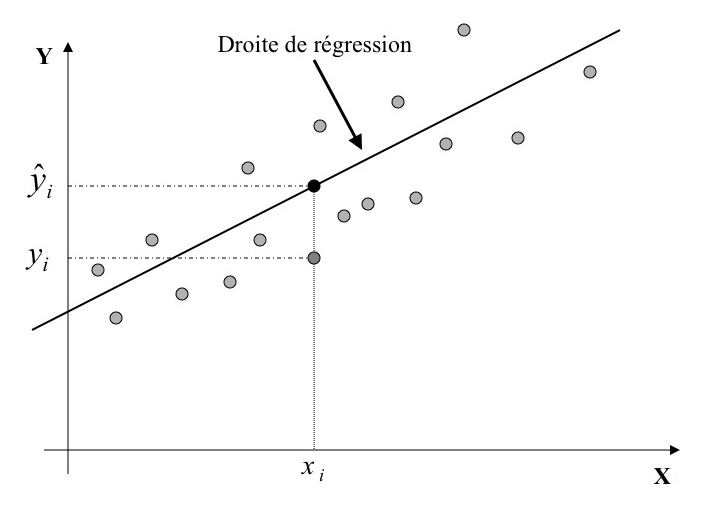
Ils sont loin d'être alignés ! Cela signifie que l'équation n'est pas tout-à-fait exacte : elle est simpliste. En écrivant cette équation, j’admets que je commettrai une certaine erreur entre la valeur que j'aurai prédite et la vraie valeur du prochain ticket. Mais je peux intégrer cette erreur à l'équation, en l'appelant ϵϵ (epsilon) :



Ce modèle est l'un des plus utilisés en statistiques. C'est la **régression linéaire**.

Pour calculer **a** et **b**  je pourrais très bien les prendre au hasard. Mais dans ce cas, l'erreur **ϵ** serait souvent très grande. Ce que je souhaite, c'est me tromper le moins possible. On dit que l'on cherche à minimiser l'erreur.

Graphiquement, voici comment on peut se représenter les choses. Si je fais varier **a** et **b**, alors je déplace la droite sur le graphique. Minimiser l'erreur revient en fait à placer la droite dans l'alignement général des points. Voici une illustration très pédagogique, car les points sont presque dans le même alignement :

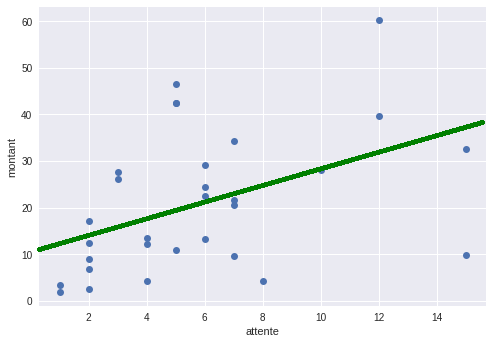


On y voit que pour un point **i**, on cherche à ce que la différence entre le **yi** (qui est la vraie valeur) et le  (qui est la valeur prédite par mon équation inexacte ) soit minimale.

Il existe plusieurs manières de minimiser une erreur. La plus utilisée est celle qui minimise la somme des carrés de l'erreur . On l'appelle **méthode des moindres carrés ordinaire** (MCO).

Pour estimer **a** et **b**, l'ordinateur peut s'en charger. Pour en savoir plus, rendez-vous à la section Aller plus loin. On obtient les estimations suivantes :

* =1.74
* =10.94



Droite de régression d'équation y = 1.74x + 10.94

En début de chapitre, nous avons fait des suppositions. En gros, on a supposé qu'il existait un lien linéaire entre attente et montant, c'est-à-dire un lien de type y=ax+b . Mais cette supposition est-elle réaliste ? Après avoir appliqué un modèle, il faut toujours analyser sa qualité. Je vous invite donc vivement à lire la section Aller plus loin : Analyser la qualité du modèle.

### Critiquons ce résultat !

Ces résultats signifient que je ne consomme que 1.74 € par jour, cela me parait peu ! De plus, 10.94€ de stock à chaque course, c'est énorme !

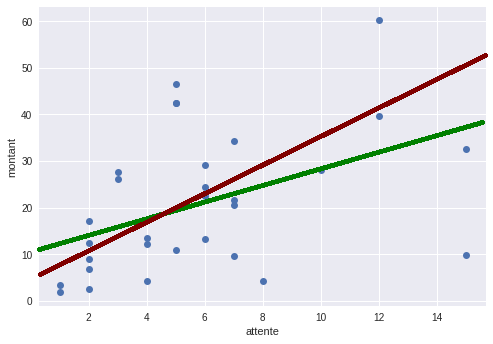
C'est vrai ... à y regarder de plus près on voit qu'il a 2 points qui "sortent du lot", on les appelle des outliers. En connaissant mes propres habitudes de consommation, je sais que je ne fais jamais les courses à plus de 15 jours d'intervalle. Ces deux points, pour lesquels attente = 15 jours, correspondent en fait à des retours de vacances (durant lesquels je n'ai pas fait de courses). Comme je ne souhaite pas que ceux-ci interfèrent dans mon calcul, je les écarte.

Une fois écartés, j'obtiens ces nouvelles estimations :

* =3.03



* =5.41



Les 2 droites de régression (une pour chaque estimation) d'équation y=ax+b

Ce résultat est bien différent du précédent. Avec seulement 2 individus écartés, les résultats changent beaucoup. On dit donc que le traitement statistique que nous venons d'appliquer (la régression linéaire avec estimation par la méthode des moindres carrés) est **peu robuste** aux outliers.

C'est d'ailleurs le cas également du coefficient de corrélation linéaire du chapitre précédent : il est peu robuste. En fait, ce n'est pas un hasard s'il l'est lui aussi. En effet, le coefficient de corrélation linéaire et la régression linéaire sont très liées ! Pour découvrir pourquoi, continuez votre lecture par la section ci-dessous ! ;)

### Aller plus loin : Analyser la qualité du modèle

Imaginons que j'aie effacé par erreur le montant d'une opération bancaire de catégorie COURSES.

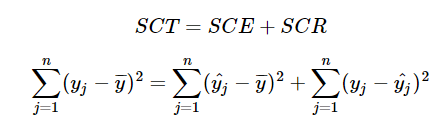
Je pourrais combler cette valeur manquante par la moyenne des montants des opérations. C'est la solution la plus basique qui soit, et vous vous imaginez qu'elle n'est pas très bonne ! Elle n'est pas très bonne car autour de la moyenne, les valeurs des montants varient, parfois de beaucoup.

Je peux alors faire mieux : je peux regarder la valeur de la variable attente de cette opération. Avec le modèle de régression linéaire que j'ai construit, je peux estimer la valeur du montant (grâce à l'équation y = ax+b). Vous vous en doutez, cette estimation sera meilleure que la précédente. En effet, quand nous avons cherché à minimiser l'erreur de modèle, nous avons en fait cherché à minimiser les variations des valeurs de montant autour de la droite de régression.

Les variations autour de la moyenne sont donc plus grandes que les variations autour de la droite de régression.

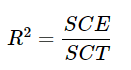
Si on avait trouvé un modèle parfait, alors il n'y aurait plus d'erreur, et donc plus de variations entre les valeurs prédites et les valeurs réelles. Dans ce cas, on dirait que le modèle a réussi à **expliquer** la totalité des variations. Les variations autour de la moyenne sont mesurées par la **variance**. Un modèle parfait aurait expliqué 100 % de la variation.

Ce pourcentage est calculé grâce à la formule de décomposition de la variance (analysis of variance, en anglais : ANOVA).



SCT (Somme des Carrés Totale) traduit la variation totale de Y , SCE (Somme des Carrés Expliquée) traduit la variation expliquée par le modèle et SCR (Somme des Carrés Résiduelle) traduit la variation inexpliquée par le modèle.

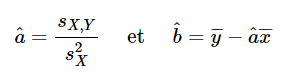
Pour la régression linéaire, le pourcentage de variation expliquée est donné par le **coefficient de détermination** noté  :



Surprise ! Par le calcul, on s'aperçoit que  est en fait le carré du **coefficient de corrélation linéaire** vu au chapitre précédent ! On a .

### Aller plus loin : Estimation de **a** et **b**

 Voici les formules qui permettent d'estimer **a** et **b** :



Pourquoi les chapeaux sur a et b ?

C'est une histoire d'estimation. On considère que l'on ne peut pas avoir accès directement à vos comportements de consommation caractérisés par **a** et **b**, mais que l'on peut tout de même les estimer grâce à vos tickets de caisse. Ces estimations de **a** et de **b** sont notées .

Si on rajoute à l'échantillon un nouveau ticket de caisse, celui-ci fera varier un peu   et , même si votre comportement de consommation ne bouge pas (c'est-à-dire "même si **a** et **b** ne bougent pas").

Pour la démonstration, c'est [**par ici**](https://fr.wikipedia.org/wiki/R%C3%A9gression_lin%C3%A9aire#Application_dans_le_cas_d.27un_mod.C3.A8le_de_r.C3.A9gression_lin.C3.A9aire_simple).

 Comment estimer a et b avec du code ?

Voici comment faire. Le code est un peu complexe, mais retenez que la ligne 6 crée les variables a et b contenant les estimations.

import statsmodels.api as sm

Y = courses['montant']

X = courses[['attente']]

X = X.copy() # On modifiera X, on en crée donc une copie

X['intercept'] = 1.

result = sm.OLS(Y, X).fit() # OLS = Ordinary Least Square (Moindres Carrés Ordinaire)

a,b = result.params['attente'],result.params['intercept']

Pourquoi y a t'il un simple crochet ligne 2 e un double crochet ligne 3 ?

Une régression linéaire prédit une variable en fonction d'une **ou plusieurs variables**.  sm.OLS s'attend donc à trouver une unique colonne (c-à-d un pd.Series  ) en premier argument (ici Y), mais s'attend à trouver potentiellement plusieurs colonnes en 2nd argument (ici X, qui est un  pd.DataFrame  ). Pour sélectionner plusieurs colonnes d'un dataframe, on passe une liste de noms de colonnes. Et comme une liste s'écrit entre crochets, ceux-ci viennent s'ajouter aux crochets déjà présents !

Pour afficher la droite, il faut faire comme ceci :

plt.plot(courses.attente,courses.montant, "o")

plt.plot(np.arange(15),[a\*x+b for x in np.arange(15)])

plt.xlabel("attente")

plt.ylabel("montant")

plt.show()

La ligne 1 affiche le graphique de dispersion.

En ligne 2, np.arange  crée une liste de nombres entiers allant de 0 à 14 :  [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14]  .

On place cette liste en abscisse. Pour chacune de ces 15 valeurs, on calcule les ordonnées grâce à la formule y=ax+b comme ceci : [a\*x+b for x in np.arange(15)]  . On vient donc de créer une série de points tous alignés sur la droite d'équation y=ax+b. La ligne 2 affiche tous ces points, en les reliant entre eux, ce qui nous donne une belle ligne !

## Analysez une variable quantitative et une qualitative par ANOVA

Dans les 2 chapitres précédents, nous avons étudié les corrélations entre 2 variables quantitatives. Maintenant, passons à l'étude de deux variables dont l'une est qualitative, l'autre quantitative.

### Quelles questions se poser ?

En fonction des couples de variables que nous utiliserons, la méthode d'analyse sera la même, mais nous pourrons répondre à différentes questions intéressantes :

* Les dépenses que vous faites le week-end sont-elles plus grosses qu'en semaine ? (variables montant et weekend)
* Les dépenses que vous faites en début de mois sont-elles plus grosses qu'en fin de mois ? (montant et quart\_mois)
* Le montant d'une opération est-il différent d'une catégorie de dépense à l'autre ? (montant et categ)
* Vos paiements en carte bancaire sont-ils toujours petits et vos virements importants ? (type et montant)
* Le solde de votre compte est-il plus petit en fin de mois qu'en début de mois ? (solde\_avt\_operation et quart\_mois)

### Des graphiques !

Voici le code qui permet de représenter une variable quantitative et une variable qualitative. Tout d'abord, créez le sous-échantillon sur lequel vous souhaitez travailler en adaptant ce code, notamment les variables X et Y selon la question que vous aurez choisie parmi celles ci-dessus.

X = "categ" # qualitative

Y = "montant" # quantitative

# On ne garde que les dépenses

sous\_echantillon = data[data["montant"] < 0].copy()

# On remet les dépenses en positif

sous\_echantillon["montant"] = -sous\_echantillon["montant"]

# On n'étudie pas les loyers car trop gros:

sous\_echantillon = sous\_echantillon[sous\_echantillon["categ"] != "LOYER"]

Ensuite, ces 6 lignes de code affichent votre graphique !

modalites = sous\_echantillon[X].unique()

groupes = []

for m in modalites:

groupes.append(sous\_echantillon[sous\_echantillon[X]==m][Y])

# Propriétés graphiques (pas très importantes)

medianprops = {'color':"black"}

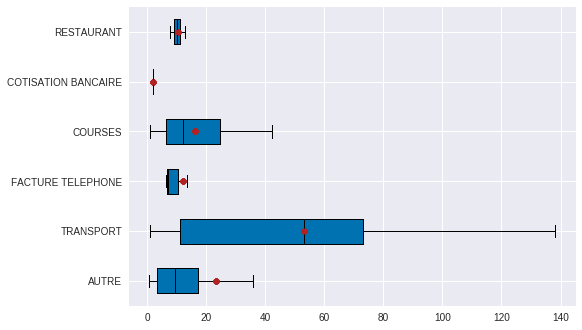
meanprops = {'marker':'o', 'markeredgecolor':'black',

'markerfacecolor':'firebrick'}

plt.boxplot(groupes, labels=modalites, showfliers=False, medianprops=medianprops,

vert=False, patch\_artist=True, showmeans=True, meanprops=meanprops)

plt.show()



Les points rouges au milieu de chaque boîte à moustache représentent la moyenne des valeurs.\*

On voit ici que les montants sont très différents d'une catégorie à l'autre. Par exemple, les montants des dépenses de transport sont plus élevés et plus dispersés que ceux des factures téléphoniques. Mais vérifions maintenant cette affirmation par les chiffres, grâce à une modélisation.

### Modélisons

Revenons sur la démarche que nous avons eue au chapitre précédent. Pour voir s'il existait une corrélation (linéaire) entre 2 variables, nous avons supposé que cette corrélation existait, puis nous avons appliqué un modèle sur cette supposition. Nous avons ensuite estimé les paramètres **a** et **b** . Enfin, nous avons vérifié la supposition de départ en évaluant la qualité du modèle. Si ce modèle est de bonne qualité, alors c'est qu'il y a une forte corrélation entre **X** et **Y**. En bonus, nous avons profité de la formule de la régression linéaire **Y=aX+b+ϵ** pour interpréter **a** comme la somme d'argent consommée en 1 jour, et **b** comme la somme d'argent de vos stocks et produits non consommables.

Ici, nous utiliserons la même démarche.

On pourrait reprendre la formule de la régression linéaire ci-dessus, sauf qu'elle implique de multiplier **X** par **a**. Or cette fois-ci, **X** est qualitative, comme notre variable categ. Multiplier une variable qualitative par un nombre n'a aucun sens (ex : "TRANSPORT" \* 3 n'a aucun sens !).

Nous allons faire autrement. Nous allons donc faire la supposition que vos opérations bancaires ont un montant de référence en commun appelé **μ**. Ensuite, on considère que le montant de l'opération s'ajuste en fonction de la catégorie **i** de dépense (loyer, transport, courses, etc.). Si une catégorie a des montants qui sont en général inférieurs à **μ**, alors cet ajustement **αi** sera négatif. Dans le cas contraire, il sera positif. On ajoute la contrainte que la somme de tous les **αi** soit égale à 0.

Par exemple, un loyer est en général assez onéreux : son **αloyer** sera donc positif.

Comme au chapitre précédent, tu commettras toujours une erreur de prédiction, car au sein d'une même catégorie, les montants ne sont pas tous les mêmes !

C'est bien vrai. Comme pour le modèle de la régression linéaire, on aura ici aussi un terme d'erreur **ϵ** :

**Y=αi+μ+ϵ**

Comme dans le chapitre précédent, on peut laisser l'ordinateur estimer tous les **αi** et **μ**, sauf qu'ici, les calculs mathématiques qui nous disent quels sont les **αi** et **μ** qui minimisent l'erreur **ϵ** donnent des résultats très intuitifs :

* Le montant de référence **μ** est estimé par la moyenne de tous les montants. On appelle cette estimation .
* Pour une catégorie **i**, **αi** est estimé en calculant l'écart entre  et la moyenne  des montants de la catégorie **i**, c'est-à-dire 

Ce modèle est très utilisé en statistiques inférentielles et est appelé **analyse de la variance**, en anglais ANalysis Of VAriance (**ANOVA**).

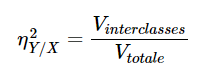
### Évaluation du modèle : les variables sont-elles corrélées ?

Notre modèle est-il de qualité ? Prévoit-on correctement les montants des opérations uniquement à partir de leur catégorie ?

Comme au chapitre précédent, on espère que notre modèle parvienne à expliquer un gros pourcentage des variations des données. Si c'est le cas, cela signifie que les variables categ et montant sont fortement corrélées.

Pour évaluer cela, la formule utilisée est exactement la même que celle du chapitre précédent : **SCT=SCE+SCR**. Mais ici, comme la variable X est qualitative, on peut donner à SCT, SCE et SCR des expressions équivalentes à celles du chapitre précédent. Elles sont données dans la section Aller plus loin. De plus, cela permet une meilleure interprétation de ces 3 sigles, qui peuvent être renommés respectivement en **variation totale**, **variation interclasse** et **variation intraclasse** (les classes sont les modalités de X).

De la même manière qu'on avait au chapitre précédent , l'équivalent ici s'appelle le **rapport de corrélation**, compris entre 0 et 1, donné par :



Si , cela signifie que les moyennes par classes sont toutes égales. Il n’y a donc pas a priori de relation entre les variables Y et X. Au contraire, si , cela signifie que les moyennes par classes sont très différentes, chacune des classes étant constituée de valeurs identiques : il existe donc a priori une relation entre les variables Y et X.

Voici le code permettant de calculer (eta carré ou eta squared en anglais). Je vous propose ici de faire le calcul à la main ;) :

X = "categ" # qualitative

Y = "montant" # quantitative

sous\_echantillon = data[data["montant"] < 0] # On ne garde que les dépenses

def eta\_squared(x,y):

moyenne\_y = y.mean()

classes = []

for classe in x.unique():

yi\_classe = y[x==classe]

classes.append({'ni': len(yi\_classe),

'moyenne\_classe': yi\_classe.mean()})

SCT = sum([(yj-moyenne\_y)\*\*2 for yj in y])

SCE = sum([c['ni']\*(c['moyenne\_classe']-moyenne\_y)\*\*2 for c in classes])

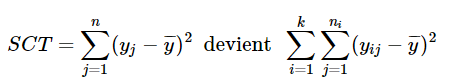
return SCE/SCT

eta\_squared(sous\_echantillon[X],sous\_echantillon[Y])

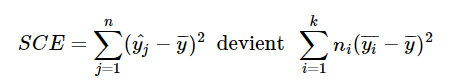
On obtient un résultat proche de 0.4, ce qui laisse penser qu'il y a effectivement une corrélation entre le montant des dépenses et leur catégorie. C'est ce que nous avions observé sur le graphique en haut du chapitre !

### Aller plus loin : Les expressions de SCT, SCE et SCR

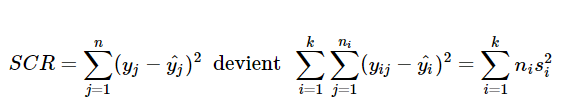
Les expressions de SCT, SCE et SCR introduisent les effectifs de chacune des classes **i**, qui sont au nombre de **k**. Ainsi,



**SCT** est ici appelé **variation totale** (les anglophones utilisent plutôt Total Sum of Squares).



**SCE** est ici appelé **variation interclasse** (Sum of Squares of the Model).



**SCR** est ici appelé **variation intraclasse** (Sum of Squares of the Error) car  est la variance au sein de la classe **i**.

## Analysez deux variables qualitatives avec le Chi-2

Bon... il ne nous reste plus qu'à étudier le cas de 2 variables qualitatives. Je vous rassure, vous avez déjà fait la moitié du travail dans ce chapitre sur le [tableau de contingence](https://openclassrooms.com/courses/nettoyez-et-decrivez-votre-jeu-de-donnees/recherchez-les-correlations#r-4773286) : si vous avez compris ce principe, c'est déjà très bien.

### Quelles questions nous posons-nous ?

La méthode d'analyse sera la même pour répondre à toutes les questions suivantes. La seule chose qui change, ce sont les 2 variables étudiées :

* Avez-vous les mêmes catégories de dépenses le weekend et en semaine ? (variables categ et weekend)
* Avez-vous plus d'entrées d'argent en début de mois ou en fin de mois ? (sens et quart\_mois)
* Vos dépenses sont-elles plus grandes en début de mois qu'en fin de mois ? (tranche\_depense et quart\_mois)
* Le montant d'une opération est-il différent d'une catégorie de dépense à l'autre ? (tranche\_depense et categ)
* Vos paiements en carte bancaire sont-ils toujours petits et vos virements importants ? (type et tranche\_depense)
* Y a-t-il des catégories d'opérations qui arrivent toujours au même moment du mois, comme votre loyer par exemple ? (categ et quart\_mois)
* Y a-t-il certaines catégories d'opérations qui s'effectuent toujours selon le même mode de paiement, par exemple par virement bancaire ? (type et categ)

Certaines questions sont identiques à celles du chapitre précédent. Dans ce dernier, on utilisait la variable quantitative montant, mais ici, on utilise la variable agrégée tranche\_depense, qui représente la même grandeur mais qui est qualitative.

### Représentation

Pour répondre à ces questions, vous pouvez afficher le tableau de contingence comme ceci :

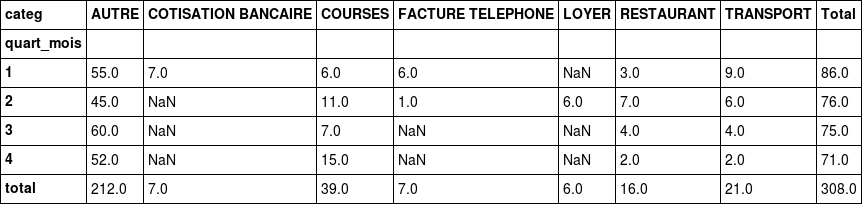
X = "quart\_mois"

Y = "categ"

cont = data[[X,Y]].pivot\_table(index=X,columns=Y,aggfunc=len,margins=True,margins\_name="Total")

cont

Adaptez les 2 variables qualitatives que vous souhaitez étudier en lignes 1 et 2. Le tableau de contingence se calcule grâce à la méthode pivot\_table. Chaque case du tableau de contingence compte un nombre d'individus. Ce comptage se fait grâce à la fonction len.



### Résultat du code ci-dessus : le tableau de contingence

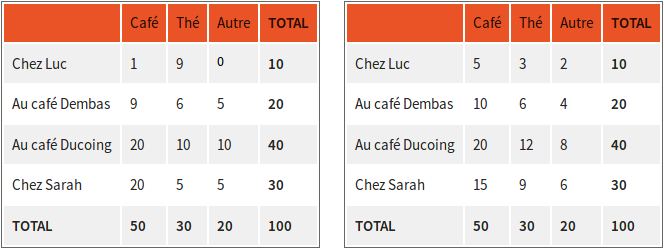
### Des statistiques !

Malheureusement, il ne nous sera ici pas possible de proposer un modèle comme dans les 2 derniers chapitres. Mais ce n'est pas grave ! On se rattrapera grâce à une mesure statistique.

Reprenez le chapitre sur le tableau de contingence, et remettez-vous en tête ce petit encadré que nous avions écrit :

Si deux événements **I** et **J** sont indépendants, alors on s'attend à ce que le nombre d'individus qui satisfont à la foisI et **J** (appelons ce nombre ) soit égal à  (c'est le calcul que vous aviez fait en entrant dans le bar : 30%\*10=3). Au contraire, plus sera différent de , plus on aura de raison de penser que **I** et **J** ne sont pas indépendants.

Etudier une corrélation entre deux variables qualitatives revient donc à comparer les   avec les . Les , ce sont les nombres qui sont dans le tableau de contingence (en dehors des 2 lignes et colonnes TOTAL). On pourrait donc créer un autre tableau qui aurait la même forme que le tableau de contingence, mais qui contiendrait plutôt les . Voici donc à gauche le tableau de contingence que nous avions, et à droite le tableau des  :



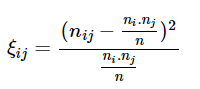
Le tableau de droite est ce à quoi on s'attend en cas d'indépendance des 2 variables. Il nous faudrait donc une statistique qui puisse comparer les valeurs de ces 2 tableaux deux-à-deux, et qui nous permettrait aussi de trouver les cases pour lesquelles les valeurs sont très différentes. Ces cases seront des valeurs dignes d'intérêt, et qui seront source de non-indépendance des 2 variables.

Allez hop ! Si vous voulez comparer deux nombres, je vous conseille de faire leur différence ! Et pour avoir des différences toujours positives (pour éviter qu'elles ne s'annulent en les sommant), passons-les au carré. Ce n'est pas la première fois que l'on utilise ce petit stratagème :



Oui mais là, si j'ai un =2 et un =4 , l'écart au carré sera de 4. Si j'ai un =1000 et un   =1002, l'écart au carré sera aussi de 4. Or une erreur de 4 quand  vaut 4, c'est une erreur bien plus importante que si =1002.

C'est vrai. On peut alors normaliser cet écart au carré en le divisant par . On obtient donc la formule suivante, en sachant que  :



Cette mesure est calculable pour chacune des cases du tableau de contingence. Il peut être intéressant de colorer ce dernier en fonction de cette mesure : foncé quand la mesure est grande, clair quand elle est proche de 0. Ainsi, on détecte facilement les cases qui sont source de non-indépendance :

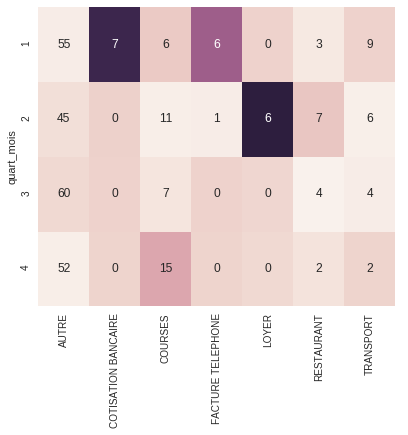
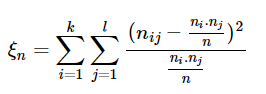


Tableau de contingence coloré

En regardant ces cases foncées, on apprend que les cotisations bancaires et factures téléphoniques sont souvent payées en tout début de mois, que les loyers sont souvent payés en 2e quartier de mois, etc.

Enfin, si on somme toutes ces mesures pour chaque case du tableau (de la colonne j=1 à la colonne j=l, puis de la ligne i=1 à la ligne i=k), on obtient la statistique  ( se prononce "xi") :



Normalement, on applique à cette mesure un seuil au-delà duquel on dira que les 2 variables sont corrélées. C'est un peu compliqué, mais pour en savoir plus, vous pouvez regarder cet article [Wikipedia](https://fr.wikipedia.org/wiki/Test_du_%CF%87%C2%B2" \l "Exemple" \t "_blank). Retenons juste ici que plus est grand, moins l'hypothèse d'indépendance est valide.

Le test avec le seuil s'appelle le test du d'indépendance (prononcer "ki-2"), parfois écrit chi-2 ou khi-2, rendu célèbre par un fameux chanteur français grâce à sa chanson [**Khi-2 nous deux**](https://www.youtube.com/watch?v=XVW5qA7QLmw).

Dans ces 4 derniers chapitres précédents, nous avons vu les mesures ,  et . Ce sont tous des nombres qui nous donnent une indication sur la corrélation (ou au contraire sur l'indépendance) de deux variables.

Mais quand vous avez un  ou un  égal à 0.4 (ce qui n'est ni proche de 0 ni de 1), peut-être avez-vous ressenti la frustration de dire "Bon oui certes, c'est un peu corrélé, mais pas trop non plus...". N'écrivez jamais cette phrase dans un rapport d'analyse de données ! Pour remédier à cela, rendez-vous dans la section Aller plus loin.

Comme nous l'avons vu, on peut associer à chaque case du tableau un . On peut ensuite normaliser chaque  en le divisant par . On obtient ainsi pour chaque case une valeur comprise entre 0 et 1.

On peut considérer cette valeur comme une **contribution** à la non indépendance. Elle est optionnellement exprimable en pourcentage si on la multiplie par 100. Plus cette contribution sera proche de 100%, plus la case en question sera source de non-indépendance. La somme de toutes les contributions vaut 100%.

Voici le code affichant le tableau de contingence coloré :

import seaborn as sns

tx = cont.loc[:,["Total"]]

ty = cont.loc[["Total"],:]

n = len(data)

indep = tx.dot(ty) / n

c = cont.fillna(0) # On remplace les valeurs nulles par 0

measure = (c-indep)\*\*2/indep

xi\_n = measure.sum().sum()

table = measure/xi\_n

sns.heatmap(table.iloc[:-1,:-1],annot=c.iloc[:-1,:-1])

plt.show()

Les lignes 1 à 6 calculent le tableau indep, qui est le tableau représentant le cas d'indépendance. Il fait intervenir le produit matriciel (avec .dot()), que vous n'avez pas à connaître à ce niveau.

En ligne 9,  measure contient tous les  pour chaque case du tableau. On peut ensuite calculer les contributions (que nous avons définies plus haut) en divisant chaque par  (placé dans la variable xi\_n). On fait ceci en ligne 11 par measure/xi\_n. On obtient ainsi pour chaque case une valeur comprise entre 0 et 1.

### Aller plus loin : "C'est un peu corrélé mais pas trop...!"

Dans ces 4 derniers chapitres précédents, nous avons vu les mesures ,  et . Ce sont tous des nombres qui nous donnent une indication sur la corrélation (ou au contraire sur l'indépendance) de deux variables.

Mais quand vous avez un  ou un  égal à 0.4 (qui n'est ni proche de 0 ni de 1), peut-être avez-vous ressenti la frustration de dire "Bon oui certes, c'est un peu corrélé, mais pas trop non plus...". N'écrivez jamais cette phrase dans un rapport d'analyse de données !

En fait, il existe des seuils que l'on peut calculer. En dessous du seuil, on dira "Les variables ne sont pas corrélées", et au-dessus, on dira l'inverse.

C'est bien plus pratique ! Mais pour être tout à fait précis, il faut donner une précision supplémentaire, qui est une sorte de "niveau de certitude", que l'on appelle **degré de significativité**. On utilise pour cela la **p-value**, exprimée en pourcentage.

C'est grâce à elle que l'on peut dire si un test statistique est significatif ou pas.

Ces notions de test statistique, de degré de significativité et de p-value font partie du domaine des [**statistiques inférentielles**](https://openclassrooms.com/courses/initiez-vous-a-la-statistique-inferentielle).

## Entraînez-vous à réaliser des analyses bivariées

### À vous de jouer

Pour vous entraîner, réalisez cet exercice étape par étape. Une fois terminé, vous pouvez comparer votre travail avec les pistes que je vous propose.

L'un des jeux de données les plus utilisés dans les cours et tutoriels de statistiques est appelé "iris". C'est un échantillon contenant des mesures prises sur des fleurs de la variété iris. Chaque individu représente une fleur dont on a mesuré :

* la longueur du pétale
* la largeur du pétale
* la longueur du sépale
* la largeur du sépale

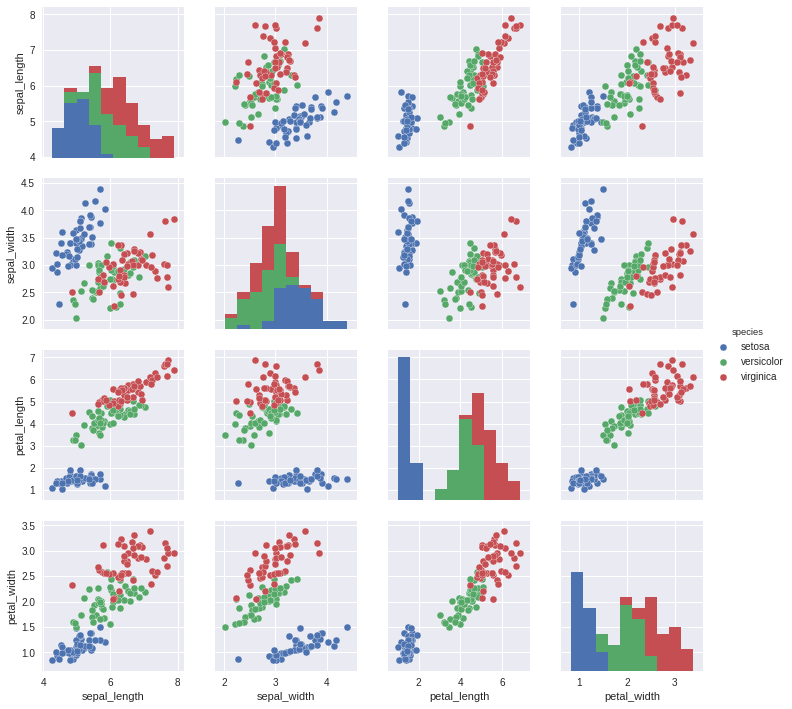


La 5e variable disponible est l'espèce de la fleur, il y en a 3 : Setosa, Versicolor et Virginica.

Le [jeu de données](https://en.wikipedia.org/wiki/Iris_flower_data_set) original a été modifié : quelques valeurs ont été supprimées. Il y a donc des valeurs manquantes. Certaines valeurs ont également été modifiées.  
Le but de cette activité ne sera pas de retrouver ces valeurs exactes, mais de les approcher en utilisant les méthodes d'analyse bivariée que vous avez vues lors des derniers chapitres.

Le jeu de données est à télécharger à [**cette adresse**](https://s3-eu-west-1.amazonaws.com/static.oc-static.com/prod/courses/files/parcours-data-analyst/Cours_nettoyez-et-decrivez-votre-jeu-de-donnees/iris_dataset.csv).

Voici les diagrammes de dispersion des variables qualitatives prises 2 à 2. Sur ces graphiques, les points ont été colorés en fonction de la variable qualitative  Espèce.



La traduction du code en R des 3 blocs suivants est donnée en bas de cette page

Voici le code Python destiné à charger le jeu de données :

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Chargement

iris = pd.read\_csv("iris\_dataset.csv")

# On renomme les colonnes

iris.columns = ["id","sepal\_length","sepal\_width","petal\_length","petal\_width","species"]

# On supprime l'identifiant des iris

del iris["id"]

# On supprime les individus contenant au moins une valeur manquante

iris\_dna = iris.dropna(axis=0, how='any')

print("iris : {} individus, iris\_dna : {} individus".format(len(iris),len(iris\_dna)))

# Affichage des diagrammes de dispersion

sns.pairplot(iris\_dna,hue="species")

plt.show()

On sépare également iris\_dna en 3 échantillons ; un pour chaque espèce :

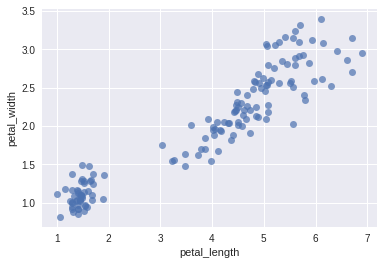
iris\_setosa = iris\_dna[iris\_dna["species"] == "setosa"]

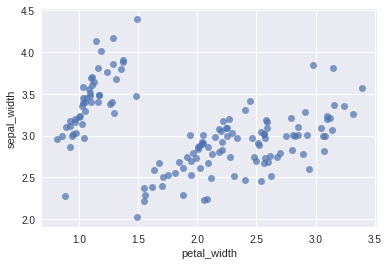
iris\_virginica = iris\_dna[iris\_dna["species"] == "virginica"]

iris\_versicolor = iris\_dna[iris\_dna["species"] == "versicolor"]

Voici 2 diagrammes de dispersion :

* celui de petal\_width en fonction de petal\_length
* celui de sepal\_width en fonction de petal\_width





### Questions

##### Question 1 :

Sur le dataframe iris\_dna, calculez les coefficients de corrélation linéaires suivants :  
- petal\_width en fonction de petal\_length  
- sepal\_width en fonction de petal\_width

##### Question 2 :

Commentez les résultats de la question 1 grâce aux 2 diagrammes de dispersion donnés dans l'énoncé.

Ensuite, gardez en tête le coefficient de corrélation linéaire de sepal\_width en fonction de petal\_width. Celui-ci est calculé sur l'ensemble des iris. Quand on calcule les coefficients de corrélation linéaire de ces mêmes variables, mais en les séparant par espèce, on obtient les résultats suivants :

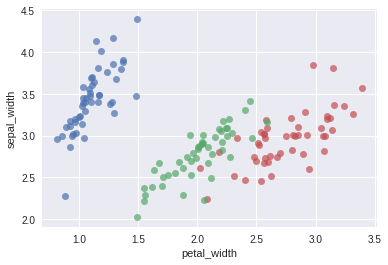
* sepal\_width en fonction de petal\_width sur iris\_setosa : 0,753
* sepal\_width en fonction de petal\_width sur iris\_virginica : 0,685
* sepal\_width en fonction de petal\_width sur iris\_versicolor : 0,825

**NB : Vous n'avez pas besoin de refaire ce calcul.**

Comparez (dans votre tête) ces 3 résultats à celui obtenu sur l'ensemble des iris (question 1).

**NB: Vous n'avez pas à répondre à cette question dans votre rendu.**

Ces résultats sont mieux interprétables quand on colore les points selon l'espèce d'iris :



##### Question 3 :

Réalisez 4 régressions linéaires (avec la méthode des moindres carrés). Nous utiliserons la notation suivante : Y = aX + b + epsilon .

a et b sont les réels que vous devez estimer, epsilon le terme d'erreur (vous n'avez pas à vous en soucier), X et Y sont 2 variables.

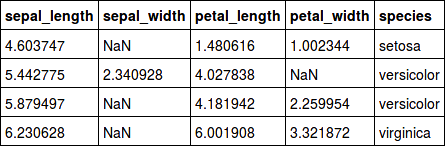
Les 4 régressions linéaires correspondent à chacun de ces 4 cas :

**cas 1/ X est la variable petal\_length et Y la variable petal\_width, sur le dataframe iris\_dna**  
**cas 2/ X est la variable petal\_width et Y la variable sepal\_width, sur le dataframe iris\_setosa**  
**cas 3/ X est la variable petal\_width et Y la variable sepal\_width, sur le dataframe iris\_virginica**  
**cas 4/ X est la variable petal\_width et Y la variable sepal\_width, sur le dataframe iris\_versicolor**  
Remarque : ce sont les 4 cas que nous étudions depuis le début.

Donnez pour chacun de ces 4 cas les estimations de a et de b.

##### Question 4 :

Voici les 4 lignes contenant des données manquantes :



Pour chaque individu, la valeur manquante se trouve soit dans la variable petal\_width soit dans la variable sepal\_width. Dans ces 2 cas, on peut imputer (remplir) ces valeurs grâce aux régressions linéaires que nous avons étudiées précédemment. Ces valeurs seront approchées mais fausses.  
En supposant qu'un individu n'a jamais plus d'une valeur manquante, voici le code qui remplace les valeurs manquantes grâce aux régressions linéaires. Des bouts de code ont été supprimés et remplacés par "[...]" : à vous de les retrouver.

coeffs = {

"cas 1" : {'a': [...] , 'b':[...]},

"cas 2" : {'a': [...] , 'b':[...]},

"cas 3" : {'a': [...] , 'b':[...]},

"cas 4" : {'a': [...] , 'b':[...]},

}

lignes\_modifiees = []

for (i,individu) in iris.iterrows(): # pour chaque individu de iris,...

if pd.isnull(individu["petal\_width"]): #... on test si individu["petal\_width"] est nul.

a = coeffs["cas 1"]['a']

b = coeffs["cas 1"]['b']

X = individu["petal\_length"]

Y = a\*X + b

iris.loc[i,"petal\_width"] = Y # on remplace la valeur manquante par Y

lignes\_modifiees.append(i)

print("On a complété petal\_width par {} a partir de petal\_length={}".format(Y,X))

if pd.isnull(individu["sepal\_width"]):

espece = individu["species"]

X = individu["petal\_width"]

[...]

lignes\_modifiees.append(i)

print("On a complété sepal\_width par {} a partir de l'espece:{} et de petal\_width={}".format(Y,espece,X))

print(iris.loc[lignes\_modifiees])

### Traduction du code en R

# Chargement

iris = read.table("iris\_dataset.csv",sep=",",header=1)

# On supprime l'identifiant des iris

iris["id"] = NULL

# On supprime les individus contenant au moins une valeur manquante

iris\_dna = na.omit(iris)

sprintf("iris : %i individus, iris\_dna : %i individus",nrow(iris),nrow(iris\_dna))

# Affichage des diagrammes de dispersion

pairs(iris\_dna[1:4], pch = 21,bg = c("red", "green3", "blue")[unclass(iris$species)])

iris\_setosa = iris\_dna[iris\_dna["species"] == "setosa",]

iris\_virginica = iris\_dna[iris\_dna["species"] == "virginica",]

iris\_versicolor = iris\_dna[iris\_dna["species"] == "versicolor",]

coeffs = list(

"cas 1" = list('a'=[...] , 'b'=[...]),

"cas 2" = list('a'=[...] , 'b'=[...]),

"cas 3" = list('a'=[...] , 'b'=[...]),

"cas 4" = list('a'=[...] , 'b'=[...])

)

lignes\_modifiees = c()

for(i in 1:nrow(iris)){ # pour chaque individu de iris,...

individu = iris[i,]

if(is.na(individu["petal\_width"])){ #... on test si individu["petal\_width"] est nul.

a = coeffs[["cas 1"]][['a']]

b = coeffs[["cas 1"]][['b']]

X = individu["petal\_length"]

Y = a\*X + b

iris[i,"petal\_width"] = Y # on remplace la valeur manquante par Y

lignes\_modifiees = c(lignes\_modifiees,i)

print(sprintf("On a complété petal\_width par %f a partir de petal\_length=%f",Y,X))

}

if(is.na(individu["sepal\_width"])){

espece = individu["species"]

X = individu["petal\_width"]

[...]

lignes\_modifiees = c(lignes\_modifiees,i)

print(sprintf("On a complété sepal\_width par %f a partir de l'espece %s et de petal\_width=%f",Y,espece,X))

}

}

print("Lignes modifiées:")

print(iris[lignes\_modifiees,])

## Repérez les différents types d'erreurs

Bravo, vous avez réussi à passer l'épreuve de la partie 3 de ce cours ! Maintenant, repos. Cette dernière partie est très simple et courte. Elle porte sur le nettoyage des données.

En vous lançant dans l'analyse de vos données, il vous faudra les nettoyer de toute erreur. Sinon, le code que vous écrirez pour faire de beaux graphiques (ou autres) plantera. Pire, vos analyses pourront contenir des erreurs si votre échantillon contient des erreurs.

Tous les data analysts ou data scientists [vous le diront](https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/) : on passe malheureusement la majeure partie du temps à nettoyer les données plus qu'à les analyser... c'est frustrant, car le nettoyage n'est vraiment pas la [partie la plus captivante](http://www.kdnuggets.com/2015/05/data-science-inconvenient-truth.html) ! Mais j'ai une bonne nouvelle : le nettoyage des données ne nous prendra que 3 chapitres, alors courage. ;)

 Il serait faux de dire que le nettoyage des données intervient avant de les analyser. Dans la plupart des cas, on est obligé de faire des allers-retours entre la phase de nettoyage et la phase de description (analyse). En phase d'analyse, on trouve souvent de nouvelles erreurs, et il faut revenir au nettoyage. De plus, le nettoyage nécessaire à l'analyse différera d'un traitement à un autre : d'où les allers-retours !

### D'où proviennent les erreurs ?

En fait, tout dépend de la source de vos données. Prenons deux exemples de sources parmi d'autres : les saisies "à la main" effectuées par des humains, et les capteurs.

Si les données ont été saisies par un humain, alors il y a de fortes chances pour que des erreurs se soient glissées dans la saisie, par exemple lorsque quelqu'un tape dans un tableur les résultats d'un sondage rempli sur papier, ou encore lorsqu'un site web contient un formulaire dans lequel l'internaute saisit de fausses données.

Si les données proviennent de capteurs (exemple : le système de géolocalisation de votre téléphone, le capteur de vitesse de votre véhicule, la machine qui valide votre billet à l'entrée du bus, etc. ), alors il se peut que le capteur se dégrade au cours du temps et ne soit plus étalonné (ex: un thermomètre qui indique 23°C alors que la température réelle est de 25°C) ou bien qu'il ne fonctionne plus (il n'envoie plus de données).

### Les différents types d'erreurs

Nous allons ici voir quelques types d'erreurs. Pas besoin de les apprendre par cœur, ni de retenir leur nom : cela n'a aucun intérêt !

Prenons l'exemple d'un échantillon de personnes, décrites par plusieurs variables :

| **Prénom** | **Email** | **Date de naissance** | **Pays** | **Taille** |
| --- | --- | --- | --- | --- |
| Leila | leila@example.com | 23/01/1990 | France | 1,49 m |
| Samuel | samuel\_329@example.com | 20/09/2001 |  | 1,67 m |
| Radia | choupipoune@supermail.eu | 12 sept. 1984 | Côte d'ivoire | 153 cm |
| Marc | marco23@example.com, mc23@supermail.eu | 10/02/1978 | France | 1,65 m |
| Heri | helloworld@supermail.eu | 05/03/2008 | Madagascar | 1,34 m |
| Hanna | hanna2019@supermail.eu | 01/01/1970 | 24 | 3,45 m |
| samuël | samuel\_329@example.com |  | Bénin | 1,45 m |

Bon... vous voyez que cet échantillon n'est pas vraiment vraiment propre, n'est-ce pas ?

1. Tout d'abord, il y a des cases vides pour les variables Pays et Date de naissance. On appelle cela les valeurs manquantes.
2. Si vous regardez dans la colonne Pays, il y a une case qui contient 24. Or 24 n'est absolument pas un pays ! Il s'agit ici d'une erreur lexicale.
3. Ensuite, vous avez peut-être vu qu'un 153 cm  s'est glissé dans la colonne Taille. C'est un problème car toutes les autres valeurs sont données en mètres, et pas en centimètres ! C'est une erreur d'irrégularité, car la variable Taille n'est pas représentée de manière régulière.
4. Marc a 2 adresses email. Ce n'est pas forcément problématique, mais si vous oubliez cela et que vous codez un programme d'analyse en faisant la supposition qu'une personne n'a qu'un seul email, votre programme plantera probablement ! Si vous faites effectivement cette supposition, alors il y aura une erreur de formatage, car marco23@example.com, mc23@supermail.eu ne respecte pas le format voulu.
5. Regardez la variable Date de naissance. Il y a également une erreur de formatage : la date de naissance de Radia n'est pas du même format que les autres dates.
6. Samuel est présent sur 2 lignes. Comment être sûr qu'il s'agit bien du même Samuel ? Par son adresse email bien sûr ! Il s'agit d'un doublon. De plus, sur les 2 lignes de Samuel, les tailles sont différentes : 1,67 m et 1,45 m, ça c'est une erreur de contradiction.
7. Hanna mesure 3,45 m. Cette taille est très différente des tailles usuelles des êtres humains : c'est une valeur qualifiée d'outlier. Même si nous n'en sommes pas sûrs, il s'agit probablement d'une erreur.

Le terme anglophone d'**outlier** peut désigner deux choses en français : une **valeur atypique** ou une **valeur aberrante**.

### Que faire de toutes ces erreurs ?

Je préfère vous le dire tout de suite, dès que vous devrez nettoyer un jeu de données, il n'y a pas de règle toute faite. Tous les traitements que vous ferez seront en fonction de l'utilisation que vous aurez de vos données. Deux data analysts ne nettoieront pas un même jeu de données de la même manière s'ils ont des objectifs différents !

Pas de règle donc, mais je peux vous donner quelques pistes :

1. Concernant les valeurs manquantes, c'est l'objet du chapitre suivant. ;)
2. Pour le pays invalide, il est possible de fixer à l'avance une liste des pays autorisés, puis de supprimer les valeurs qui ne sont pas dans cette liste (ici 24 n'y sera pas). Une telle liste est souvent appelée dictionnaire.
3. Pour les erreurs d'irrégularité, c'est plus compliqué ! On peut par exemple fixer un format fixe (ici : un nombre décimal suivi du caractère "m") et supprimer les valeurs qui ne suivent pas ce format. Mais on peut faire mieux, et détecter d'abord dans quelle unité est exprimée la valeur (mètres ou centimètres) puis tout convertir en une même unité.
4. Pour l'erreur de formatage de la double adresse email, tout dépend de ce que vous souhaitez faire. Si vous n'analyserez pas les emails dans votre analyse future, alors pas besoin de corriger l'erreur. Si par contre vous souhaitez connaître la proportion du nombre de personnes dont l'adresse finit par @example.com, par @supermail.eu, etc., alors vous pouvez choisir entre :
   1. prendre la première adresse email, et oublier la seconde ;
   2. garder l'ensemble des adresses email.
5. Passons à la variable Date de naissance. Aaaaaaah les dates ! :'( Croyez-moi, elles vous donnerons toujours du fil à retordre ! Eh oui, il existe d'innombrables formats de date, et chaque pays a sa propre habitude quand il s'agit d'écrire une date (ex : les Français et les Nord-Américains n'utilisent pas les mêmes formats). En plus de cela, il faut ajouter les problèmes des fuseaux horaires ! Dans notre cas, la plus simple des solutions consiste à supprimer les dates qui ne sont pas au format jour/mois/année.
6. Pour le doublon, vous verrez cela dans le chapitre suivant.
7. Pour l'outlier, c'est également dans le chapitre suivant ! ;)

En règle générale, si une variable contient peu d'erreurs et que cette variable n'est pas d'une importance cruciale pour votre analyse, on peut se permettre de supprimer les valeurs erronées. On se retrouvera alors avec des valeurs manquantes. Vous verrez que faire des valeurs manquantes dans le chapitre suivant.

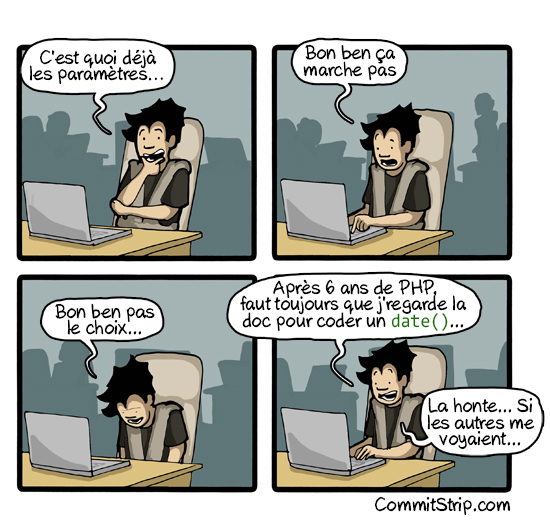
Cependant, si les erreurs sont nombreuses et de même nature, autant créer un programme informatique qui corrigera les erreurs.

Par exemple, si 60 % des tailles sont données en mètres, 35 % en centimètres et 5 % dans d'autres unités, alors il y a 35 % d'erreurs qui sont de même nature (35 % des valeurs sont en centimètres au lieu de mètres). Autant donc coder quelques lignes de code qui convertiront les centimètres en mètres. Si vous êtes motivés et que le jeu en vaut la chandelle, attaquez-vous aussi aux 5 % restants, mais cela vous prendra beaucoup de temps !

### Aller plus loin : Les dates

Sachez qu'entre différents pays, nous avons quand même réussi à nous mettre d'accord sur un format de date normalisé : c'est le format [ISO 8601](https://fr.wikipedia.org/wiki/ISO_8601). Il est de cette forme :  1977-04-22T06:00:00Z.

De toute manière, sachez que les dates vous feront toujours souffrir, comme en témoigne ce [CommitStrip](https://www.commitstrip.com/fr/2013/06/20/inavouable/" \t "_blank) :



### Aller plus loin : Ressources externes

Dans ce cours, nous nettoierons nos données grâce à Python ou R.

Mais sachez qu'il existe un très bon outil destiné au nettoyage de données, accessible à ceux qui ne maîtrisent pas la programmation : c'est [OpenRefine](http://openrefine.org/" \t "_blank).

## Traitez les valeurs manquantes, les outliers et les doublons

Nous l'avons vu dans le chapitre précédent, un échantillon peut contenir des valeurs manquantes, des outliers et des doublons. Alors que faire ?

Certaines des méthodes suivantes suppriment de l'information dans votre échantillon. Faites attention à toujours conserver une copie de votre échantillon. Si vous souhaitez supprimer de l'information, alors créez une copie de l'échantillon, puis supprimez ce que vous voulez : le nouvel échantillon que vous obtenez sera appelé **sous-échantillon**.

### Les valeurs manquantes

Lorsque l'échantillon contient des valeurs manquantes, on ne peut malheureusement pas faire de miracle pour les retrouver ! Cependant, plusieurs attitudes sont possibles.

#### Ne rien faire et travailler avec un gruyère

Pour une variable donnée (par exemple date de naissance dans l'exemple du chapitre précédent), si la proportion de valeurs manquantes est faible, alors on peut les oublier et ne rien faire : on laisse l'échantillon intact. On travaillera alors avec un jeu de données qui contiendra des "trous", comme dans un gruyère. :honte: Selon le traitement statistique que vous appliquerez, cette solution sera ou non acceptable.

#### Oublier une variable

Cependant, si pour cette même variable, la proportion de valeurs manquantes est beaucoup trop importante, mieux vaut l'oublier, à condition que la variable ne soit pas trop importante pour l'analyse. Cela équivaut à ne pas considérer une colonne dans le tableau du chapitre précédent.

#### Oublier des individus

Si la variable qui contient des données manquantes est cruciale dans l'analyse, alors mieux vaut créer un sous-échantillon et y supprimer les individus pour lesquels cette variable est manquante. Par exemple, si vous analysez vos relevés de comptes bancaires en vous intéressant aux sommes d'argent que vous gagnez/dépensez, la variable "montant de l'opération" sera très importante. S'il arrive que le montant de l'opération soit inconnu pour certaines lignes de votre relevé, alors mieux vaut créer un sous-échantillon et y supprimer la totalité de ces lignes.

Cette dernière méthode contient cependant des risques. En effet, vous pouvez vous retrouver avec un nombre d'individus (un nombre de lignes) trop petit pour que votre analyse ait encore du sens. De plus, il se peut que votre échantillon ne soit plus représentatif de la population globale. Pour savoir pourquoi, rendez-vous à la section Aller plus loin au bas de ce chapitre.

#### Essayer de deviner quand même !

Une méthode un peu plus aventurière consiste à combler les trous par des valeurs à deviner. C'est un peu la méthode des aventuriers ! :zorro: Bien sûr, ces valeurs ne correspondront pas à la valeur réelle, mais certaines méthodes permettent de ne pas se tromper de beaucoup. Deviner une valeur manquante s'appelle l'**imputation**.

Par exemple, on peut remplacer les valeurs manquantes de la variable taille par la taille moyenne des individus de notre échantillon. Dans notre exemple, pour corriger la taille de Hanna (que nous supposons être erronée), on la remplace par la moyenne des autres individus, soit 1,52 m. C'est l'**imputation par la moyenne**.

##### Deviner à partir d'autres variables

Mais on peut faire mieux ! Pour remplacer une variable donnée, on peut regarder les autres variables aux alentours. Il y a plusieurs méthodes qui utilisent ce principe.

Imaginons un nouvel individu : Luc, né en 1991, dont la taille est inconnue. Plutôt que de lui attribuer la moyenne de tout l'échantillon (1,52 m), on peut lui attribuer la moyenne des personnes qui ont à peu près son âge. Attribuons-lui donc la moyenne des tailles des personnes nées entre 1990 et 2000, soit 1,49 m. Ici, on a regardé la valeur de la variable date\_de\_naissance pour déduire la valeur de la variable taille.

D'autres méthodes sont également basées sur le fait de déduire une variable à partir d'autres. On peut citer les méthodes de Hot-deck, ou les méthodes basées sur des régressions linéaires. Pour trouver un exemple d'imputation grâce à une régression linéaire, réalisez l'[activité de fin de la partie précédente](https://openclassrooms.com/courses/4525266/exercises/1957). ;)

Deviner (imputer) des valeurs modifie votre échantillon, car les valeurs imputées sont fausses. En particulier, vos calculs de variances ou de corrélations seront faussées. Il faut donc les utiliser avec des pincettes, comme en témoigne [**ce document**](http://www.i3s.unice.fr/~crescenz/publications/Florence/valeurs-manquantes-ou-aberrantes.pdf) (pages 4 et 5).

Une bonne pratique peut être d'observer la distribution de la variable (ou tout au moins ses quantiles) avant et après l'imputation, pour voir si sa forme n'a pas trop été impactée.

Dans tous les cas, il est nécessaire de toujours préciser quelle méthode vous avez utilisé dans chacun des résultats d'analyse que vous présenterez. C'est une question d'honnêteté intellectuelle. ;)

### Les outliers

Hanna mesure 3,45 m. Vous ne trouvez pas cela très grand ? Si. C'est très grand comparé aux tailles des autres êtres humains.

Mais attention, un outlier n'est pas forcément une valeur fausse ! En effet, Hanna mesure peut-être réellement 3,45 m. Ok, c'est difficile à concevoir, mais c'est possible.

Un **outlier** peut être :

* une valeur **aberrante** : c'est une valeur qui est manifestement fausse
* une valeur **atypique** : c'est une valeur qui "sort du lot", mais pas forcément fausse.

En français, il arrive très souvent que le terme valeur aberrante soit employé à tort pour désigner une valeur atypique.

Idéalement, il faudrait vérifier si les outliers sont erronées ou pas. Par exemple, un thermomètre qui mesure les températures en France peut indiquer 40°C, mais il peut s'agir soit d'une défaillance du capteur de température, soit d'une valeur réelle.

Alors que faire avec les outliers ? Si nous sommes sûrs que la valeur est erronée (erreur de saisie ou défaut d’un capteur par exemple), alors il faut la supprimer s’il n’est pas possible de connaître la vraie valeur. Dans les autres cas, nous avons le choix entre :

* Supprimer la valeur. On se retrouve alors avec une valeur manquante, à laquelle on peut imputer une valeur comme nous l’avons vu précédemment. L’imputation n’est pas obligatoire.
* Conserver la valeur.

Comment choisir entre ces deux options ? Tout dépend des traitements que vous appliquerez par la suite. Certaines méthodes sont dites « robustes », car elle ne sont pas déstabilisées par les outliers. Par exemple, nous verrons par la suite que la moyenne est très sensible aux outliers, alors que la médiane ne l’est pas. Si vous souhaitez faire une moyenne, créez un sous-échantillon dans lequel vous ne considérez pas les outliers. Mais si vous calculez aussi la médiane, travaillez sur l’échantillon de départ. ;)

N’hésitez pas, quand vous présentez votre analyse, à citer les outliers s'ils sont intéressants. Par exemple, « les relevés de température contiennent 2 outliers à 42°C, dus à 2 jours de canicule extrême ».

### Et les doublons ?

Dans notre exemple, Samuel est présent 2 fois. C’est problématique, car ce doublon (aussi appelé « donnée dupliquée ») fausse les analyses : notamment la taille moyenne de l'échantillon.

Il faut éliminer les doublons. Cependant, il n’y a pas de règle précise pour les détecter : vous seuls pouvez les détecter, à partir de la structure de vos données et en sachant comment elles ont été collectées. Mais parfois, ce sera impossible. :(

Un petit exemple : si votre échantillon contient une variable « identifiant », alors il est aisé de détecter des doublons. Ce sont ceux qui auront le même identifiant ;) . Dans notre exemple, on peut considérer que l’adresse email est l’identifiant d’une personne. Dans notre exemple, les 2 lignes qui ont pour email  samuel\_329@example.com constituent un doublon.

Si vous êtes familier avec les bases de données, vous connaissez probablement la notion de [**clé**](https://openclassrooms.com/courses/initiez-vous-a-lalgebre-relationnelle-avec-le-langage-sql/comprenez-limportance-des-cles) (clé primaire ou clé candidate). Deux individus avec les mêmes valeurs pour une clé sont un doublon !

Autre exemple : vous analysez des relevés de température pris dans un village. Il y a 2 stations météo dans ce village : la station 1, qui a fonctionné de nombreuses années jusqu’au 15 janvier 2019, puis qui s’est arrêtée à cause de son ancienneté. Cette panne ayant été prévue, une station 2 avait été installée (au même endroit) pour la remplacer : elle a été mise en service le 2 janvier 2019. Votre échantillon est donc constitué de relevés provenant des 2 stations. Cependant, les relevés compris entre le 2 janvier et le 15 janvier 2019 sont en double, car les 2 stations fonctionnaient en parallèle. Il vous faut donc supprimer, pour chaque date comprise dans cette période, l’un des 2 relevés.

Oui mais de nos deux lignes contenant  samuel\_329@example.com, faut-il en supprimer une au hasard ?

En fait, il faut faire un peu plus attention. Mieux vaut les regrouper en une ligne. En effet, parmi ces 2 lignes, la 1ère nous informe que Samuel est né le 20/09/2001, et la seconde ligne nous informe que Samuel habite au Bénin (information qui est manquante dans la 1ère ligne). Le problème, c’est pour la taille : la première ligne nous dit que Samuel mesure 1,67 m, alors que la seconde nous affirme qu’il n’en mesure que 1,45 m. Il y a contradiction. S’il n’y a pas d’autre moyen de vérification, on peut par exemple choisir de prendre la moyenne de ces 2 valeurs.

### Aller plus loin : Conséquence de la suppression d'individus

Imaginez un échantillon de personnes de la même forme que celui du chapitre précédent :

| Prénom | Pays | Date de naissance | Taille |
| --- | --- | --- | --- |
| Albert | France | 23/09/1930 | 1,45 m |
| Sophia | USA | 01/20/1959 | 1,68 m |
| Donald | USA | 02/16/2002 | 1,65 m |
| Ali | France | 16/02/2000 | 1,57 m |
| Doriane | Togo | 17/08/1978 | 1,58 m |

Vous décidez de supprimer les dates de naissances pour lesquelles le format n'est pas jour/mois/année, ce qui créera des valeurs manquantes dans la variable date de naissance. Puis, vous décidez de supprimer toutes les lignes (tous les individus) qui ont une date de naissance manquante. Vous aurez probablement supprimé toutes les personnes habitant aux USA, car ceux-ci ont l'habitude d'écrire les dates [différemment des francophones](https://frenchmorning.com/pourquoi-les-americains-mettent-le-mois-en-tete-de-la-date/). Si vous réalisez ensuite une analyse sur les tailles, votre échantillon ne sera plus représentatif, car les personnes des USA ont sûrement une taille moyenne différente de celle des autres pays.

## TP : Nettoyez votre jeu de données

Nous allons nettoyer le jeu de données du chapitre précédent. Nous illustrerons ceci en Python. Si vous préférez R, un code équivalent est donné en bas de chapitre ;).

Nous commencerons par charger l'échantillon à partir de [ce fichier CSV](https://s3-eu-west-1.amazonaws.com/course.oc-static.com/courses/4525266/personnes.csv) (qu'il vous faut télécharger) dans une variable que nous appellerons  data. Cette variable sera donc un dataframe.

Ensuite, nous allons parcourir chacune des colonnes pour détecter les erreurs, les corriger, puis actualiser les colonnes en conséquence. Que ce soit en Python ou en R, actualiser une colonne d'un dataframe se fait de cette manière :

data["nom\_colonne"] = nouvelle\_colonne

Ici, on cherche à remplacer les valeurs de la colonne (ou variable)  nom\_colonne. Si le dataframe a 7 lignes, alors la colonne nom\_colonne contient 7 valeurs. Pour les remplacer, nouvelle\_colonne doit ainsi être une liste de 7 valeurs.

### Les méthodes apply et map

Il reste encore à savoir comment remplir nouvelle\_colonne. En fait, elle sera calculée à partir de nom\_colonne. Ce qu'il nous faut, c'est parcourir chaque valeur de nom\_colonne, vérifier si elle est correcte ou pas, et la corriger si besoin.

Pour cela, nous utilisons la méthode apply. C'est une méthode qui est appelée sur une colonne du dataframe, et qui permet de parcourir toutes ses valeurs. Alternativement, on peut utiliser la méthode  map, qui lui est ([à peu près](https://stackoverflow.com/questions/19798153/difference-between-map-applymap-and-apply-methods-in-pandas)) équivalente. Pour chaque valeur, elle applique une fonction destinée à la vérification/correction :

import pandas as pd # On importe la librairie Pandas, que l'on surnomme 'pd'

def lower\_case(value):

print('Voici la valeur que je traite:', value)

return value.lower()

data = pd.DataFrame([['A',1],

['B',2],

['C',3]], columns = ['lettre','position'])

nouvelle\_colonne = data['lettre'].apply(lower\_case)

nouvelle\_colonne = nouvelle\_colonne.values

print(nouvelle\_colonne)

data['lettre'] = nouvelle\_colonne

print(data)

Ce code est disponible dans l'archive téléchargeable au chapitre [**Téléchargez les données**](https://openclassrooms.com/courses/nettoyez-et-decrivez-votre-jeu-de-donnees/telechargez-les-donnees), dans le dossier nettoyage, dans ces différents fichiers : R\_nettoyage et python\_nettoyage.

En ligne 7, on crée notre dataframe. Celui-ci est un tableau avec 2 colonnes ('lettre' et 'position') et 3 lignes. En ligne 3, on crée une fonction nommée lower\_case, qui prend en paramètre une valeur value, qui l'affiche (ligne 4), la transforme en minuscules (ligne 5), puis la retourne.

Ensuite, on sélectionne la colonne lettre de data, on appelle la méthode apply, et on spécifie que chacune des valeurs doit être envoyée une à une à la fonction lower\_case (ligne 13).

En ligne 11, nouvelle\_colonne est de type "colonne" (car la méthode apply renvoie une colonne). Dans la librairie Pandas, le type exact d'une colonne est appelé Series. Pour obtenir les valeurs de cette colonne sous forme de liste, on appelle nouvelle\_colonne.values  (ligne 12). Voici ce que ce programme affichera :

Voici la valeur que je traite: A

Voici la valeur que je traite: B

Voici la valeur que je traite: C

['a' 'b' 'c']

lettre position

0 a 1

1 b 2

2 c 3

Les lignes 1 à 3 affichent ce que fait la fonction lower\_case, la ligne 4 affiche le résultat du traitement, c'est-à-dire les 3 lettres en minuscule, et les autres lignes affichent le dataframe dont la colonne lettre a été actualisée avec les minuscules !

La ligne 12convertit  nouvelle\_colonne  d'un objet  Series  en une liste. Cette ligne est en fait optionnelle, car la syntaxe de la ligne 13 fonctionne aussi bien si nouvelle\_colonne est une liste ou si nouvelle\_colonne est un objet Series !

### Attaquons !

#### Charger les données

Commencez par télécharger le fichier CSV qui correspond à l'exemple des chapitres précédents (donné en haut de chapitre), puis chargez-le grâce à ces lignes de code :

# import des librairies dont nous aurons besoin

import pandas as pd

import numpy as np

import re

# chargement et affichage des données

data = pd.read\_csv('personnes.csv')

print(data)

#### Traiter les pays

Vous l'avez compris, il nous faudra une fonction par traitement. Oublions lower\_case, et écrivons une fonction qui vérifie si les pays présents dans la colonne pays sont corrects. Pour ceci, il nous faut une liste des pays acceptés :

VALID\_COUNTRIES = ['France', 'Côte d\'ivoire', 'Madagascar', 'Bénin', 'Allemagne'

, 'USA']

def check\_country(country):

if country not in VALID\_COUNTRIES:

print(' - "{}" n\'est pas un pays valide, nous le supprimons.' \

.format(country))

return np.NaN

return country

Ici, si le pays présent dans la variable country n'est pas présent dans la liste VALID\_COUNTRIES, on affiche le message des lignes 6 et 7. Ensuite, on retourne np.NaN, qui est la valeur utilisée par les librairies  Numpy  et  Pandas  pour spécifier qu'une valeur est inconnue. C'est en quelque sorte un équivalent de None.

Plus précisément, "NaN" signifie "Not a Number" ("pas un nombre" en français). Vous rencontrerez également pd.NaT, "not a time" (pour les colonnes de dates).

Sinon, si le pays est valide, on renvoie simplement ce pays (ligne 9) !

Un petit doute sur la méthode format de la ligne 6 ? Allez faire un tour [**ici**](https://openclassrooms.com/courses/demarrez-votre-projet-avec-python/modifiez-des-chaines-de-caracteres) ;)

#### Traiter les emails

Au tour des emails maintenant ! Le problème avec cette colonne, c'est qu'il y a parfois 2 adresses email par ligne. Nous ne souhaitons prendre que la première. Nous créons donc la fonction first :

def first(string):

parts = string.split(',')

first\_part = parts[0]

if len(parts) >= 2:

print(' - Il y a plusieurs parties dans "{}", ne gardons que {}.'\

.format(parts,first\_part))

return first\_part

Lorsqu'il y a plusieurs emails par ligne, ceux-ci sont séparés par des virgules. Nous séparons donc la chaîne de caractère de la variable  string  selon les virgules grâce à la méthode [split](https://openclassrooms.com/courses/demarrez-votre-projet-avec-python/modifiez-des-chaines-de-caracteres) (ligne 2). Le résultat est donc une liste avec autant d'éléments que d'adresses email: cette liste est placée dans la variable  parts.

Comme parts contient au moins 1 élément, on place celui-ci dans la variable first\_part. Ensuite, on compte le nombre d'éléments que contient parts grâce à la fonction len. S'il y en a au moins 2, alors on affiche le message des lignes 5 et 6. Finalement, on retourne first\_part, qui contient le premier email !

#### Traiter les tailles

Nous aurons ici 2 fonctions :  convert\_height, qui convertira les chaînes de caractères de type "1,34 m" en nombre décimal, ainsi que fill\_height, qui remplacera les valeurs manquantes par la moyenne des tailles de l'échantillon.

def convert\_height(height):

found = re.search('\d\.\d{2}m', height)

if found is None:

print('{} n\'est pas au bon format. Il sera ignoré.'.format(height))

return np.NaN

else:

value = height[:-1] # on enlève le dernier caractère, qui est 'm'

return float(value)

def fill\_height(height, replacement):

if pd.isnull(height):

print('Imputation par la moyenne : {}'.format(replacement))

return replacement

return height

La première fonction est un peu plus élaborée. Vous pouvez soit lui accorder une confiance aveugle, soit tenter de percer son mystère dans la section Aller plus loin en bas de ce chapitre ;).

Passons à la seconde fonction. Ah ! elle prend 2 paramètres :  height et replacement. Le premier est la hauteur, comme d'habitude. Le second est la valeur que nous devrons renvoyer en cas de valeur manquante. La ligne 11 vérifie si la valeur de height est manquante (None, NaN ou Nat). Si c'est le cas, on retourne la valeur de remplacement (ligne 13), sinon, on retourne height.

### Appliquons toutes ces fonctions

Maintenant que ces fonctions sont définies, il faut les exécuter ! A la fin de votre programme, ajoutez ceci :

data['email'] = data['email'].apply(first)

data['pays'] = data['pays'].apply(check\_country)

data['taille'] = [convert\_height(t) for t in data['taille']]

data['taille'] = [t if t<3 else np.NaN for t in data['taille']]

mean\_height = data['taille'].mean()

data['taille'] = [fill\_height(t, mean\_height) for t in data['taille']]

data['date\_naissance'] = pd.to\_datetime(data['date\_naissance'],

format='%d/%m/%Y', errors='coerce')

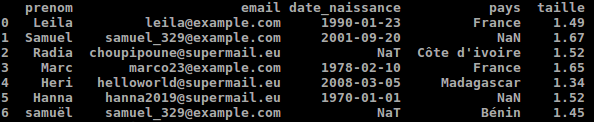
print(data)

Vous vous souvenez de la syntaxe que nous avons vu tout en haut de ce chapitre, pour actualiser une colonne ? Nous l'employons ici dans les lignes 1 à 4, 6 et 7. Vous connaissez la syntaxe utilisée dans les lignes 1 et 2. Par contre, pour les lignes 3, 4 et 6 il faudra peut-être vous rafraîchir la mémoire sur les [compréhensions de listes](https://openclassrooms.com/courses/pratiques-avancees-et-meconnues-en-python#r-1174227). Si cela ne vous dit rien, rendez-vous un peu plus bas, dans la section Aller plus loin.

Que vous dire d'autre... Vous savez l'essentiel. Bon quelques petites précisions mineures :

* t if t<3 else np.NaN  renvoie  t  si  t  est inférieur à 3, ou renvoie  np.NaN  sinon. On l'utilise pour supprimer les tailles supérieures à 3 m, qui sont aberrantes.
* data['taille'].mean()  renvoie une unique valeur, qui est la moyenne des tailles.
* La colonne date\_naissance  contient des chaînes de caractères. On les [convertit en dates](https://pandas.pydata.org/pandas-docs/stable/generated/pandas.to_datetime.html), en spécifiant le format d'écriture des dates. Les chaînes de caractères qui ne respectent pas ce format seront transformées en  pd.NaT  (c'est le cas de la date de naissance de Radia).

La ligne 9 affiche ici le résultat final :



### Aller plus loin : Les compréhensions de listes

Les compréhensions de listes, c'est une syntaxe très pratique, car elle permet en une ligne d'écrire une boucle for qui construit une liste. Par exemple, la ligne 3 du tout dernier bout de code ci-dessus est équivalente à ces 4 lignes :

data = pd.read\_csv('personnes.csv')

nouvelle\_colonne = []

for t in data['taille']:

nouvelle\_colonne.append(convert\_height(t))

data['taille'] = nouvelle\_colonne

On aurait très bien pu utiliser apply :

data = pd.read\_csv('personnes.csv')

data['taille'] = data['taille'].apply(convert\_height)

### Aller plus loin : le traitement des tailles

Reprenons la fonction que nous avions laissée de côté :

def convert\_height(height):

found = re.search('\d\.\d{2}m', height)

if found is None:

print('{} n\'est pas au bon format. Il sera ignoré.'.format(height))

return np.NaN

else:

value = height[:-1] # on enlève le dernier caractère, qui est 'm'

return float(value)

Normalement, Pandas détecte automatiquement si une colonne provenant d'un fichier CSV contient des chaînes de caractères ou des nombres. Mais ici, la colonne "taille" contient des "m". Comme c'est une lettre, Pandas considère que "1,34 m" est une chaîne de caractères, pas un nombre ! Il faudra donc le convertir nous-même.

Alors... c'est vrai, la ligne 2 est difficile à comprendre. Elle vérifie si la taille est bien formatée : un chiffre, suivi d'un point, puis 2 chiffres, puis un "m". Ainsi, "1,34 m" est correct, alors que "153 cm" n'est pas correct.

Il s'agit d'une expression régulière. Pas besoin de les maîtriser si vous voulez suivre ce cours, mais sachez quand même que les expressions régulières sont très très très pratiques, et indispensables si vous souhaitez devenir Data Analyst. Heureusement, elles ne sont pas difficiles à comprendre : Céline les a très bien expliquées [**ici**](https://openclassrooms.com/courses/manipulez-des-donnees-avec-python-1/utilisez-des-expressions-regulieres).

Vous avez normalement toutes les clés en main pour comprendre le reste de cette fonction. Remarquez que float(value)permet de convertir une chaîne de caractères qui représente un nombre en... un vrai nombre (dont le type est "float") !

# Nettoyez votre échantillon

Bravo ! Vous avez réussi cet exercice !

### Compétences évaluées

* Nettoyer un jeu de données

### Question 1

**Que faire face à un ou plusieurs outliers ?**

* + 

Les garder s'ils ne sont pas erronés, puis, en fonction des méthodes statistiques utilisées, les garder si la méthode est robuste aux outliers, les supprimer sinon

* + 

Les supprimer systématiquement

*Il faut les garder si on a de bonnes raisons de penser qu'ils ne sont pas erronés (erreur de mesure, mauvaise saisie humaine, erreur dans l'acheminement de la donnée, etc.).*

### Question 2

**Mis à part le cas dans lequel les individus possèdent une (ou plusieurs) variable(s) qui les identifie (exemple: un email, un numéro d'identifiant, un numéro de client, etc.), supprimer les doublons d'un échantillon est :**

* + 

Généralement difficile

* + 

Généralement facile

*C'est plutôt difficile de les repérer. Relisez le chapitre sur les doublons si besoin ;).*

### Question 3

**Ces deux morceaux de code effectuent la même opération. L'un est écrit en Python, l'autre en R. Que font-ils ?**

import pandas as pd

STATUS\_VALUES = ["INVITE","COLLABORATEUR","EMPLOYE"]

df = pd.read\_csv("mapetiteentreprise.csv")

def process(value):

if value not in STATUS\_VALUES:

return "INVITE"

else:

return value

df["status"] = df["status"].map(process)

STATUS\_VALUES = c("INVITE","COLLABORATEUR","EMPLOYE")

df = read.table("mapetiteentreprise.csv", header=1)

process <- function(value){

if(!value %in% STATUS\_VALUES)

return("INVITE")

else

return(value)

}

df["status"] = apply(df["status"],1,process)

**Pas besoin de connaître les 2 langages pour répondre. Seul l'un des deux suffit.**

* + 

Ils attribuent la valeur "INVITE" pour les individus dont la valeur de la variable status n'est ni "EMPLOYE", ni "COLLABORATEUR".

* + 

Ils attribuent la valeur "INVITE" pour les seuls individus dont la valeur de la variable status est manquante.

*Tout se joue sur la ligne "value not in STATUS\_VALUES" (ou "!value %in% STATUS\_VALUES" en R), qui vérifie si value est dans la liste STATUS\_VALUES.*

### Question 4

**L'imputation c'est :**

* + 

Couper un membre à un individu

* + 

Attribuer une valeur lorsqu'une valeur est manquante dans un échantillon

* + 

Supprimer un individu d'un échantillon

### Question 5

**Quelle colonne de ce tableau contient une erreur d'irrégularité ?**

| identifiant | age | prénom | score |
| --- | --- | --- | --- |
| 2873 | 27 ans | Leila | 39 points |
| 1028 | 999 ans |  | 45 points |
| 3892 | 78 ans | Samir | 89% |
| 8273 | 12 ans | Cindy | 24 points |

* + 

identifiant

* + 

âge

* + 

prénom

* + 

score

*La colonne score contient des résultats dans des unités différentes (points et %) : c'est une erreur d'irrégularité.*