

O'REILLY®

Machine Learning avec Scikit-Learn

Mise en œuvre et cas concrets

3^e édition

Aurélien Géron

Traduit de l'anglais par Anne Bohy

DUNOD

Authorized French translation of material from the English edition of
Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow, 3E

ISBN 9781098125974

© 2023 Aurélien Geron.

This translation is published and sold by permission of O'Reilly Media, Inc.,
which owns or controls all rights to publish and sell the same.

Conception de la couverture : Karen Montgomery

Illustratrice : Kate Dullea

© Dunod, 2017, 2019, 2023

11 rue Paul Bert, 92240 Malakoff

www.dunod.com

ISBN 978-2-10-086537-6

Table des matières

Avant-propos	VII
Chapitre 1 – Vue d’ensemble du Machine Learning	1
1.1 Qu’est-ce que l’apprentissage automatique ?	2
1.2 Pourquoi utiliser l’apprentissage automatique ?	3
1.3 Exemples d’applications	6
1.4 Types de systèmes d’apprentissage automatique	7
1.5 Principales difficultés de l’apprentissage automatique	25
1.6 Test et validation	32
1.7 Exercices	36
Chapitre 2 – Un projet de Machine Learning de bout en bout	39
2.1 Travailler avec des données réelles	39
2.2 Prendre du recul pour une vision d’ensemble	41
2.3 Récupérer les données	46
2.4 Explorer et visualiser les données pour mieux les comprendre	61
2.5 Préparer les données pour les algorithmes d’apprentissage automatique	67
2.6 Sélectionner et entraîner un modèle	87
2.7 Régler avec précision votre modèle	90
2.8 Lancer, surveiller et maintenir votre système	96
2.9 Essayez !	99
2.10 Exercices	100

Chapitre 3 – Classification	103
3.1 MNIST	103
3.2 Entraînement d'un classificateur binaire	106
3.3 Mesures de performance	107
3.4 Classification multi-classes	119
3.5 Analyse des erreurs	121
3.6 Classification multi-étiquettes	125
3.7 Classification multi-sorties	127
3.8 Exercices	129
Chapitre 4 – Entraînement de modèles	131
4.1 Régression linéaire	132
4.2 Descente de gradient	138
4.3 Régression polynomiale	149
4.4 Courbes d'apprentissage	150
4.5 Modèles linéaires régularisés	154
4.6 Régression logistique	162
4.7 Exercices	171
Chapitre 5 – Machines à vecteurs de support	173
5.1 Classification SVM linéaire	173
5.2 Classification SVM non linéaire	176
5.3 Régression SVM	182
5.4 Sous le capot des classificateurs SVM linéaires	184
5.5 Exercices	191
Chapitre 6 – Arbres de décision	193
6.1 Entraîner et visualiser un arbre de décision	193
6.2 Effectuer des prédictions	195
6.3 Estimation des probabilités des classes	197
6.4 Algorithme d'entraînement CART	197
6.5 Complexité algorithmique	198
6.6 Impureté de Gini ou entropie ?	199
6.7 Hyperparamètres de régularisation	200

6.8 Régression	201
6.9 Sensibilité à l'orientation des axes	204
6.10 Arbres de décision ayant une variance élevée.	205
6.11 Exercices	206
Chapitre 7 – Apprentissage d'ensemble et forêts aléatoires	209
7.1 Classificateurs par vote	210
7.2 Bagging et pasting	213
7.3 Parcelles aléatoires et sous-espaces aléatoires	217
7.4 Forêts aléatoires	217
7.5 Boosting	220
7.6 Stacking	229
7.7 Exercices	231
Chapitre 8 – Réduction de dimension	233
8.1 Le fléau de la dimension	234
8.2 Principales approches de la réduction de dimension.	235
8.3 PCA	239
8.4 Projection aléatoire	248
8.5 LLE.	250
8.6 Autres techniques de réduction de dimension	252
8.7 Exercices	254
Chapitre 9 – Techniques d'apprentissage non supervisé.	255
9.1 Partitionnement.	256
9.2 Mélanges gaussiens.	279
9.3 Exercices	290
Le mot de la fin	293
Annexe A – Solutions des exercices.	295
Annexe B – Liste de contrôle de projet de Machine Learning.	309
Annexe C – SVM : le problème dual	315
Index.	319

Avant-propos

La révolution du Machine Learning

Seuls ceux qui ont vécu dans une caverne depuis une quinzaine d'années ont pu ignorer l'incroyable révolution de l'apprentissage automatique, ou *Machine Learning* (ML). Il ne se passe plus une semaine sans qu'il ne fasse parler de lui : cela a commencé par de formidables progrès en reconnaissance d'images, puis en analyse de la voix, le programme Watson d'IBM est ensuite devenu champion du jeu de Jeopardy, on a vu les premières voitures autonomes de Google sillonner les routes, puis le programme AlphaGo de DeepMind a vaincu le champion du monde du jeu de go, le logiciel Libratus de l'université Carnegie Mellon a écrasé des champions de poker, des patients paralytiques ont pu contrôler le mouvement de leurs membres par la pensée, grâce à un programme qui avait appris à déchiffrer certaines de leurs ondes cérébrales... bref, les succès s'enchaînent et ne se ressemblent pas. Il y a une quinzaine d'années, de telles intelligences artificielles n'existaient que dans les romans de science-fiction.

Le Machine Learning en entreprise

Au-delà de ces exemples qui font la une des journaux, le Machine Learning envahit plus discrètement les systèmes informatiques de toutes les entreprises. D'ores et déjà, lorsque vous parcourez Internet ne serait-ce que quelques minutes, une horde de systèmes d'apprentissage automatique s'activent : certains analysent votre personnalité pour vous proposer les produits qui vous correspondent le mieux, d'autres sélectionnent les publicités qui attireront votre attention, et d'autres encore analysent votre comportement pour s'assurer que vous n'êtes pas un fraudeur. Mais le Machine Learning n'est pas réservé aux géants du web : qu'il s'agisse de prédire des séries temporelles (comme les cours de la Bourse), de détecter des anomalies de production, d'optimiser des centres d'appel, d'analyser les profils des clients, ou encore de classer automatiquement des documents, l'apprentissage automatique s'est révélé d'une grande utilité dans une multitude de domaines. Si votre entreprise n'utilise pas encore le Machine Learning, elle risque fort de se faire devancer rapidement par ses concurrents.

Data Scientist, une espèce rare

Le marché du Machine Learning croît si rapidement que le nombre d'experts en analyse de données (*data scientist*) a bien de la peine à suivre. Malgré un nombre d'étudiants en forte hausse en sciences des données (*data science*), il est aujourd'hui difficile pour une entreprise de trouver suffisamment de profils compétents. Face à cette pénurie d'expertise, la meilleure solution est probablement de former certains employés au Machine Learning. Les mieux placés pour cela sont naturellement les ingénieurs en informatique, car ils maîtrisent déjà la programmation et bien souvent aussi les bases mathématiques que requiert le Machine Learning. En outre, ils sont souvent très demandeurs, à la fois car ils savent que la compétence est rare, mais aussi et surtout car le sujet est absolument passionnant !

Objectif et approche

Ce livre a donc pour objectif de vous former au Machine Learning. Vous apprendrez les concepts fondamentaux et les outils nécessaires pour créer des systèmes capables d'apprendre à partir d'exemples, qu'il s'agisse de classer des images, d'estimer la valeur d'un bien, etc. Nous aborderons un grand nombre de techniques et d'algorithmes, depuis de simples systèmes linéaires, aux arbres de décision et aux forêts aléatoires en passant par les machines à vecteurs de support, et bien d'autres encore. Vous apprendrez aussi à utiliser Scikit-Learn¹, l'une des bibliothèques open source les plus simples et néanmoins les plus puissantes disponibles aujourd'hui, que vous pourrez directement utiliser dans vos systèmes en production.

Hormis le premier chapitre, qui offre une vision d'ensemble du Machine Learning et des principaux concepts, le reste de cet ouvrage est résolument axé sur la pratique : le but n'est pas juste de vous enseigner la théorie, mais aussi que vous sachiez concrètement développer vos propres systèmes de ML. Bien que vous puissiez lire ce livre sans allumer votre ordinateur, vous êtes fortement encouragé(e) à mettre la main à la pâte au cours de la lecture, en particulier en faisant les exercices proposés à la fin de chaque chapitre, et dont les solutions sont disponibles à la fin de l'ouvrage.

Exemples de code

Tous les exemples figurant dans ce livre sont en open source et disponibles sous <https://github.com/ageron/handson-ml3> en tant que notebooks Jupyter : il s'agit de documents interactifs comportant du texte, des images et des fragments de code exécutable (en Python dans notre cas). La façon la plus simple et la plus rapide de commencer est d'exécuter ces notebooks en utilisant Google Colab, un service gratuit vous permettant d'exécuter directement en ligne n'importe quel notebook Jupyter, sans avoir à installer quoi que ce soit sur votre machine. Vous aurez seulement besoin d'un navigateur internet et d'un compte Google.

Dans ce livre, je supposerai que vous utilisez Google Colab, mais j'ai aussi testé les notebooks sur d'autres plateformes en ligne comme Kaggle ou Binder, que vous

1. Cette bibliothèque a été créée par David Cournapeau en 2007, et le projet est maintenant dirigé par une équipe de chercheurs à l'Institut national de recherche en informatique et en automatique (Inria).

pouvez donc utiliser si vous préférez. Sinon, vous pouvez aussi installer les bibliothèques et outils requis (ou l'image Docker de ce livre) et exécuter les notebooks directement sur votre propre ordinateur : reportez-vous aux instructions figurant sur la page <https://homl.info/install>.

Prérequis

Bien que ce livre ait été écrit plus particulièrement pour les ingénieurs en informatique, il peut aussi intéresser toute personne sachant programmer et ayant quelques bases mathématiques. Il ne requiert aucune connaissance préalable sur le Machine Learning mais il suppose les prérequis suivants :

- vous devez avoir un minimum d'expérience de programmation ;
- sans forcément être un expert, vous devez connaître le langage Python, et si possible également ses bibliothèques scientifiques, en particulier NumPy, Pandas et Matplotlib ;
- enfin, si vous voulez comprendre comment les algorithmes fonctionnent (ce qui n'est pas forcément indispensable, mais est tout de même très recommandé), vous devez avoir certaines bases en mathématiques dans les domaines suivants :
 - l'algèbre linéaire, notamment comprendre les vecteurs et les matrices (par exemple comment multiplier deux matrices, transposer ou inverser une matrice),
 - le calcul différentiel, notamment comprendre la notion de dérivée, de dérivée partielle, et savoir comment calculer la dérivée d'une fonction.

Si vous ne connaissez pas encore Python, il existe de nombreux tutoriels sur Internet, que nous vous encourageons à suivre : ce langage est très simple et s'apprend vite. En ce qui concerne les bibliothèques scientifiques de Python et les bases mathématiques requises, le site github.com/ageron/handson-ml3 propose quelques tutoriels (en anglais) sous la forme de notebooks Jupyter. De nombreux tutoriels en français sont disponibles sur Internet. Le site fr.khanacademy.org est particulièrement recommandé pour les mathématiques.

Plan du livre

- Le premier chapitre présente une vue d'ensemble du Machine Learning ainsi que les concepts fondamentaux : qu'entend-on exactement par *apprentissage automatique* ? Quels problèmes le Machine Learning peut-il résoudre ? Quelles sont les principales catégories d'algorithmes et les principales difficultés que l'on peut rencontrer ? Qu'est-ce que le surajustement et le sous-ajustement ?
- Le deuxième chapitre attaque le vif du sujet en présentant un projet de Machine Learning de A à Z (en introduisant Scikit-Learn au passage) : d'abord analyser le problème (en l'occurrence estimer le prix de l'immobilier), puis obtenir les données, les nettoyer, choisir une fonction d'évaluation, sélectionner un modèle, l'entraîner sur les données d'entraînement, évaluer le système sur un jeu de données préalablement mis de côté, rechercher les meilleurs hyperparamètres grâce à la validation croisée, etc.

- Le chapitre 3 analyse les difficultés particulières liées aux problèmes de classification, en particulier les diverses façons d'évaluer un classificateur, et comment le régler au mieux selon ses besoins.
- Le chapitre 4 est le premier à ouvrir les boîtes noires, pour expliquer comment fonctionnent les algorithmes. Il explique en particulier les principaux modèles linéaires et montre comment les entraîner en ajustant progressivement leurs paramètres à l'aide de l'algorithme de descente de gradient. Il montre aussi diverses techniques de régularisation qui permettent d'éviter le piège du surajustement.
- Le chapitre 5 se concentre sur les machines à vecteur de support (SVM), des modèles très puissants, particulièrement prisés pour les problèmes complexes pour lesquels on ne dispose que d'assez peu de données.
- Le chapitre 6 détaille les arbres de décision. Il s'agit de modèle simples et polyvalents, particulièrement faciles à interpréter.
- Le chapitre 7 présente les forêts aléatoires, qui reposent sur une multitude d'arbres de décision, et il explore plus généralement diverses méthodes ensemblistes, qui permettent de combiner plusieurs modèles de Machine Learning.
- Le chapitre 8 détaille plusieurs algorithmes de réduction de dimensionnalité, permettant de réduire le volume d'un jeu de données pour échapper au fléau de la dimensionnalité et ainsi accélérer l'apprentissage, ou encore à des fins de visualisation des données.
- Enfin, le chapitre 9, ayant pour objet l'apprentissage non supervisé, traite notamment du partitionnement avec la méthode des k -moyennes ou l'algorithme DBSCAN, des modèles de mélange gaussien, de l'inférence bayésienne et de l'utilisation des modèles de mélange pour le partitionnement, l'estimation de densité, la détection d'anomalies ou de nouveautés. Il fait également un tour d'horizon des algorithmes correspondants.

Le Deep Learning

Pour traiter des problèmes particulièrement complexes, tels que la reconnaissance de la parole ou encore les traductions automatiques, on utilise généralement des réseaux de neurones, qui requièrent souvent de gros volumes de données et une immense puissance de calcul. Presque toutes les applications du Machine Learning mises en avant par la presse reposent sur des réseaux de neurones profonds (c'est-à-dire organisés en de nombreuses couches successives): on parle alors de *Deep Learning* (apprentissage profond).

Ce livre n'aborde pas le Deep Learning, mais il vous donne les bases nécessaires pour le faire. Si à l'issue de ce livre vous souhaitez poursuivre vers le Deep Learning, nous vous recommandons le livre *Deep Learning avec Keras et TensorFlow*, des mêmes auteur et éditeur. Vous y apprendrez comment utiliser la bibliothèque TensorFlow pour créer diverses architectures de réseaux de neurones profonds – des réseaux de neurones convolutionnels (très utilisés pour l'analyse d'images), des réseaux de neurones

récurrents (utiles pour l'analyse de séquences de tailles arbitraires, telles que des séries temporelles ou la langue naturelle), des auto-encodeurs (capables de reconnaître des motifs et de les imiter) – et comment entraîner et déployer ces réseaux de neurones sur de nombreux serveurs grâce à TensorFlow. Enfin, ce deuxième livre présente le Deep Reinforcement Learning: il s'agit des techniques développées par DeepMind pour créer un système capable d'apprendre tout seul à jouer à des jeux Atari sans en connaître les règles à l'avance.

Les différents types de textes en exercice



Ce symbole indique une astuce ou une suggestion.



Ce symbole indique une précision ou une remarque générale.



Ce symbole indique une difficulté particulière ou un piège à éviter.

Remerciements

Jamais, dans mes rêves les plus fous, je n'aurais imaginé que la deuxième édition de ce livre rencontrerait un public aussi vaste. J'ai reçu de nombreux messages de lecteurs, avec beaucoup de questions, certains signalant gentiment des erreurs et la plupart m'envoyant des mots encourageants. Je suis extrêmement reconnaissant envers tous ces lecteurs pour leur formidable soutien. Merci beaucoup à vous tous ! N'hésitez pas à me contacter si vous voyez des erreurs dans les exemples de code ou simplement pour poser des questions (<https://homl.info/issues3>) ! Certains lecteurs ont également expliqué en quoi ce livre les avait aidés à obtenir leur premier emploi ou à résoudre un problème concret sur lequel ils travaillaient. Ces retours sont incroyablement motivants. Si vous trouvez ce livre utile, j'aimerais beaucoup que vous puissiez partager votre histoire avec moi, que ce soit en privé (par exemple, via <https://linkedin.com/in/aurelien-geron>) ou en public (par exemple dans un tweet via @aureliengeron ou par le biais d'un commentaire Amazon).

Grand merci aussi à toutes les personnes merveilleuses qui ont offert de leur temps et leur expertise pour réviser cette troisième édition, en corrigeant des erreurs et en faisant d'innombrables suggestions. Cette édition est tellement meilleure grâce à cela : Olzhas Akpambetov, George Bonner, Francois Chollet, Siddha Ganju, Sam Goodman, Matt Harrison, Sasha Sobran, Lewis Tunstall, Leandro von Werra et mon cher frère Sylvain. Vous êtes tous incroyables !

Je suis également très reconnaissant envers toutes les personnes qui m'ont soutenu tout au long du chemin, en répondant à mes questions, en suggérant des améliorations et en contribuant au code sur GitHub : en particulier, Yannick Assogba, Ian Beauregard, Ulf Bissbort, Rick Chao, Peretz Cohen, Kyle Gallatin, Hannes Hapke, Victor Khaustov, Soonson Kwon, Éric Lebigot, Jason Mayes, Laurence Moroney, Sara Robinson, Joaquin Ruales et Yuefeng Zhou.

Ce livre n'existerait pas sans le personnel fantastique d'O'Reilly, en particulier Nicole Taché, qui m'a fait des commentaires perspicaces, toujours encourageants et utiles : je ne pouvais pas rêver d'un meilleur éditeur. Merci également à Michele Cronin, qui m'a encouragé et m'a permis d'arriver au bout. Merci à toute l'équipe de la production, en particulier Elizabeth Kelly et Kristen Brown. Merci également à Kim Cofer pour la révision minutieuse, et à Johnny O'Toole, qui a géré la relation avec Amazon et répondu à beaucoup de mes questions. Merci à Kate Dullea pour l'amélioration importante de mes illustrations. Merci à Marie Beaugureau, Ben Lorica, Mike Loukides et Laurel Ruma d'avoir cru en ce projet et de m'avoir aidé à le définir. Merci à Matt Hacker et à toute l'équipe d'Atlas pour avoir répondu à toutes mes questions techniques concernant AsciiDoc, MathML et LaTeX, ainsi qu'à Nick Adams, Rebecca Demarest, Rachel Head, Judith McConville, Helen Monroe, Karen Montgomery, Rachel Roumeliotis et tous les autres membres d'O'Reilly qui ont contribué à ce livre.

Je n'oublierai jamais les personnes formidables qui m'ont aidé sur les deux premières éditions du livre : amis, collègues, experts, dont de nombreux membres de l'équipe de TensorFlow. La liste est longue : Olzhas Akpambetov, Karmel Allison, Martin Andrews, David Andrzejewski, Paige Bailey, Lukas Biewald, Eugene Brevdo, William Chargin, François Chollet, Clément Courbet, Robert Crowe, Mark Daoust, Daniel « Wolff » Dobson, Julien Dubois, Mathias Kende, Daniel Kitachewsky, Nick Felt, Bruce Fontaine, Justin Francis, Goldie Gadde, Irene Giannoumis, Ingrid von Glehn, Vincent Guilbeau, Sandeep Gupta, Priya Gupta, Kevin Haas, Eddy Hung, Konstantinos Katsiapis, Viacheslav Kovalevskyi, Jon Krohn, Allen Lavoie, Karim Matrah, Grégoire Mesnil, Clemens Mewald, Dan Moldovan, Dominic Monn, Sean Morgan, Tom O'Malley, James Pack, Alexander Pak, Haesun Park, Alexandre Passos, Ankur Patel, Josh Patterson, André Susano Pinto, Anthony Platanios, Anosh Raj, Oscar Ramirez, Anna Revinskaya, Saurabh Saxena, Salim Sémaoune, Ryan Sepassi, Vitor Sessak, Jiri Simsa, Iain Smears, Xiaodan Song, Christina Sorokin, Michel Tessier, Wiktor Tomczak, Dustin Tran, Todd Wang, Pete Warden, Rich Washington, Martin Wicke, Edd Wilder-James, Sam Witteveen, Jason Zaman, Yuefeng Zhou, et mon frère Sylvain.

Je souhaite également remercier Jean-Luc Blanc, des éditions Dunod, pour avoir soutenu ce projet, et pour la gestion et les relectures attentives des deux premières éditions. Merci également à Matthieu Daniel, qui a pris la suite de Jean-Luc Blanc pour cette troisième édition. Je tiens aussi à remercier vivement Anne Bohy qui a traduit cet ouvrage en français bien mieux que je n'aurais su le faire. Dotée d'une grande expérience en développement informatique et d'un doctorat en mathématiques, elle était sans doute la traductrice idéale pour ce livre. Enfin, je remercie chaleureusement Brice Martin, des éditions Dunod, pour sa relecture extrêmement rigoureuse, ses excellentes suggestions et ses nombreuses corrections.

Pour finir, je suis infiniment reconnaissant à ma merveilleuse épouse, Emmanuelle, et à nos trois enfants, Alexandre, Rémi et Gabrielle, de m'avoir encouragé à travailler dur pour ce livre. Leur curiosité insatiable fut inestimable : expliquer certains des concepts les plus difficiles de ce livre à ma femme et à mes enfants m'a aidé à clarifier mes pensées et à améliorer directement de nombreuses parties de ce livre. J'ai même eu droit à des biscuits et du café, qui pourrait en demander davantage ?

1

Vue d'ensemble du Machine Learning

Il n'y a pas si longtemps, si vous aviez demandé à votre téléphone de vous indiquer comment rentrer chez vous, ce dernier aurait ignoré votre demande et les gens se seraient interrogés quant à votre état mental. Mais l'apprentissage automatique, qu'on désigne volontiers par son appellation anglaise de « Machine Learning » ou ML, n'appartient plus au domaine de la science-fiction : des milliards de personnes l'utilisent chaque jour. À dire vrai, cela fait déjà quelques décennies qu'on le met en œuvre dans certaines applications spécialisées telles que la reconnaissance optique de caractères. Mais la première utilisation à grande échelle qui a amélioré la vie de centaines de millions de personnes date des années 1990 : c'est le filtre de spam. S'il ne s'agit pas exactement d'un robot intelligent, techniquement cela relève bien néanmoins de l'apprentissage automatique : le logiciel a si bien appris qu'il est rare désormais que vous ayez besoin de marquer un message comme indésirable. Par la suite, des centaines d'applications ML ont été peu à peu introduites dans les produits et fonctionnalités que vous utilisez régulièrement : commandes vocales, traduction automatique, recherche d'images, recommandations de produits et bien d'autres encore.

Où commence l'apprentissage automatique, et où s'arrête-t-il ? Lorsqu'on dit qu'une machine *apprend* quelque chose, qu'est-ce que cela signifie ? Si je télécharge l'ensemble des articles de Wikipédia sur mon ordinateur, ce dernier a-t-il vraiment « appris » quelque chose ? Est-il soudainement devenu plus intelligent ? Dans ce chapitre, nous allons commencer par clarifier ce qu'est l'apprentissage automatique et les raisons qui peuvent vous pousser à l'utiliser.

Puis, avant d'entreprendre d'explorer ce vaste territoire, nous en examinerons une vue d'ensemble, en découvrirons les principaux domaines et en étudierons les spécificités les plus notables : apprentissage supervisé / apprentissage non supervisé et leurs variantes, apprentissage en ligne / apprentissage groupé, apprentissage à partir

d'observations / apprentissage à partir d'un modèle. Après quoi, nous nous intéresserons au déroulement d'un projet ML type, nous évoquerons les principales difficultés que vous pourriez rencontrer et expliquerons comment évaluer et régler finement votre système.

Ce chapitre introduit un grand nombre de concepts fondamentaux (et de jargon) que chaque spécialiste du traitement des mégadonnées doit connaître par cœur. Il s'agira d'une présentation très générale (c'est le seul chapitre qui ne comporte pratiquement pas de code) ; l'ensemble est assez simple, mais je tenais à ce que tout soit parfaitement clair pour vous avant de passer à la suite. Préparez-vous un café, et commençons !



Si vous avez déjà des connaissances de base en apprentissage automatique, vous pouvez passer directement au chapitre 2. En cas de doute, essayez de répondre à toutes les questions figurant à la fin de ce chapitre.

1.1 QU'EST-CE QUE L'APPRENTISSAGE AUTOMATIQUE ?

L'apprentissage automatique est la science (et l'art) de programmer les ordinateurs de sorte qu'ils puissent *apprendre à partir de données*.

Voici une définition un peu plus générale :

« [L'apprentissage automatique est la] discipline donnant aux ordinateurs la capacité d'apprendre sans qu'ils soient explicitement programmés. »

Arthur Samuel, 1959

En voici une autre plus technique :

« Étant donné une tâche T et une mesure de performance P , on dit qu'un programme informatique apprend à partir d'une expérience E si les résultats obtenus sur T , mesurés par P , s'améliorent avec l'expérience E . »

Tom Mitchell, 1997

Votre filtre anti-spam est un programme d'apprentissage automatique qui peut apprendre à identifier les e-mails frauduleux à partir d'exemples de pourriels ou *spam* (par exemple, ceux signalés par les utilisateurs) et de messages normaux (parfois appelés *ham*). Les exemples utilisés par le système pour son apprentissage constituent le *jeu d'entraînement* (en anglais, *training set*). Chacun de ces exemples constitue ce qu'on appelle en statistique une *observation* (en anglais, *instance*). On parle parfois aussi d'*échantillon* (en anglais, *sample*). On appelle *modèle* la partie du système d'apprentissage automatique qui apprend et effectue des prédictions. Les réseaux de neurones et les forêts aléatoires sont des exemples de modèles. Dans le cas présent, la tâche T consiste à identifier parmi les nouveaux e-mails ceux qui sont frauduleux, l'expérience E est constituée par les *données d'entraînement*, et la mesure de

performance P doit être définie (vous pourrez prendre par exemple le pourcentage de courriels correctement classés). Cette mesure de performance particulière, appelée *exactitude* (en anglais, *accuracy*), est souvent utilisée dans les tâches de classification.

Si vous téléchargez l'ensemble des articles de Wikipédia, votre ordinateur dispose de beaucoup plus de données, mais cela ne le rend pas soudainement meilleur pour quelque tâche que ce soit. Par conséquent, il ne s'agit pas d'apprentissage automatique.

1.2 POURQUOI UTILISER L'APPRENTISSAGE AUTOMATIQUE?

Voyons comment vous écririez un filtre de spam en utilisant des techniques de programmation traditionnelles (voir figure 1.1) :

1. Tout d'abord, vous examinerez à quoi ressemble en général un courrier frauduleux. Vous remarqueriez peut-être que certains mots (tels que « pour vous », « carte bancaire », « gratuit », « bravo »...) semblent apparaître souvent dans l'intitulé du sujet. Vous remarqueriez peut-être aussi quelques autres détails caractéristiques dans le nom de l'expéditeur, le corps du message et d'autres parties du message.
2. Après quoi, vous écririez un algorithme de détection de chacune des caractéristiques repérées, et votre programme marquerait comme indésirables tous les messages dans lesquels un certain nombre de ces caractéristiques ont été détectées.
3. Vous testeriez alors le programme et répéteriez les étapes 1 et 2 jusqu'à obtenir un résultat satisfaisant.

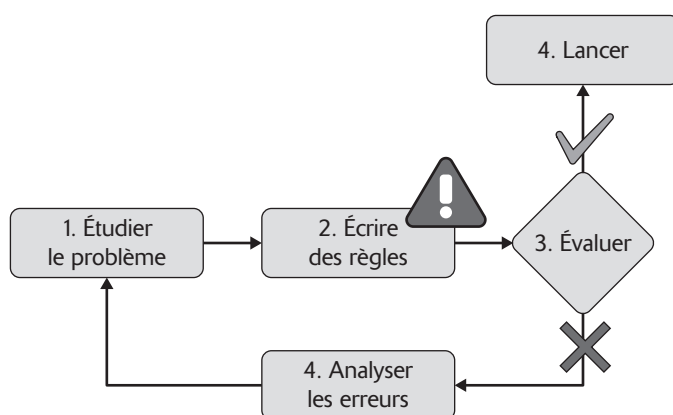


Figure 1.1 – Approche traditionnelle

Le problème étant compliqué, votre programme deviendra vraisemblablement une longue liste de règles complexes assez difficiles à maintenir.

Par contraste, un filtre anti-spam basé sur des techniques de Machine Learning apprend automatiquement quels sont les mots et les phrases qui constituent de bons prédicteurs de courriels frauduleux en détectant des associations de mots aux fréquences inhabituelles dans des exemples de courriels frauduleux comparés à des exemples de courriels licites (voir figure 1.2). Le programme est beaucoup plus court, plus facile à maintenir et a plus de chances de fournir de bons résultats.

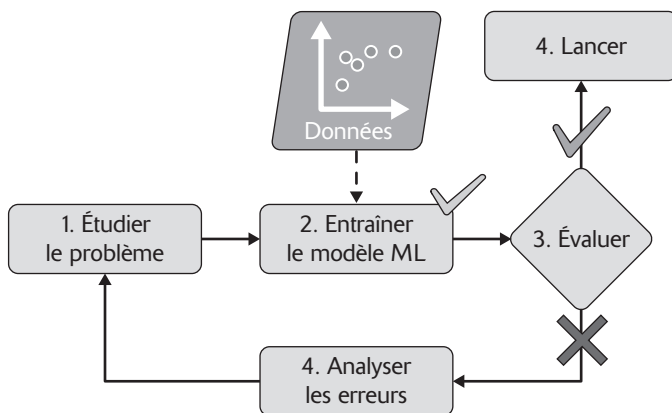


Figure 1.2 – L'approche du Machine Learning

Si les pirates remarquent que leurs e-mails contenant le mot « Pour vous » sont bloqués, ils peuvent décider d'écrire « For you » à la place. Un filtre anti-spam utilisant des techniques de programmation traditionnelles nécessiterait une mise à jour pour pouvoir intercepter les messages « For you ». Si les pirates continuent à tenter de contourner votre filtre anti-spam, vous devrez sans arrêt écrire de nouvelles règles.

Par contre, un filtre anti-spam basé sur des techniques de Machine Learning repère automatiquement que « For you » est devenu inhabituellement fréquent dans les messages indésirables signalés par les utilisateurs et il commence à en tenir compte sans votre intervention (voir figure 1.3).

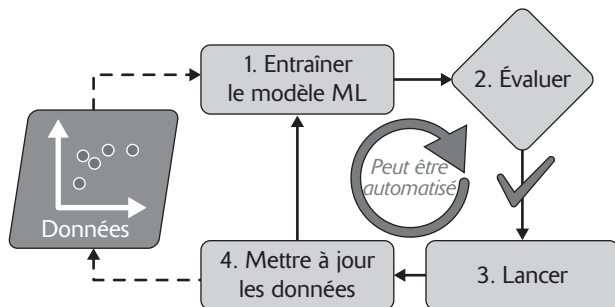


Figure 1.3 – Adaptation automatique aux évolutions