

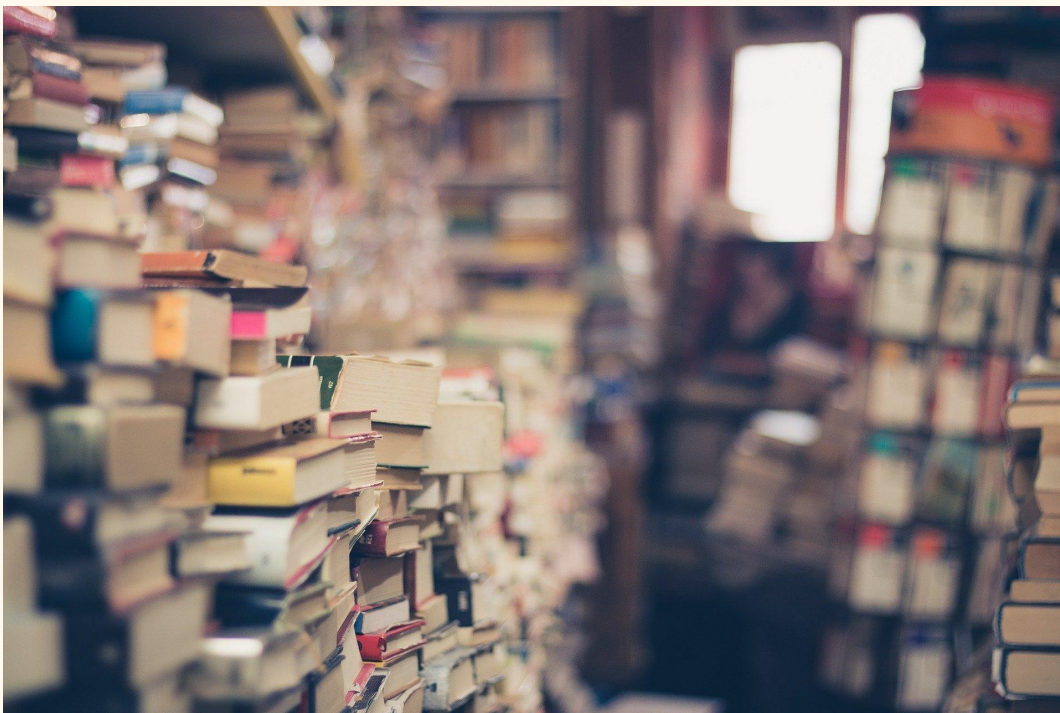
OPENCLASSROOMS

PROJET 7 : Développez une preuve de concept

# SYSTÈMES DE RECOMMANDATIONS : APPLICATIONS AUX LIVRES, APPORT DES TRANSFORMERS ET IMPACT ECOLOGIQUE

---

Par Cécile Guillot

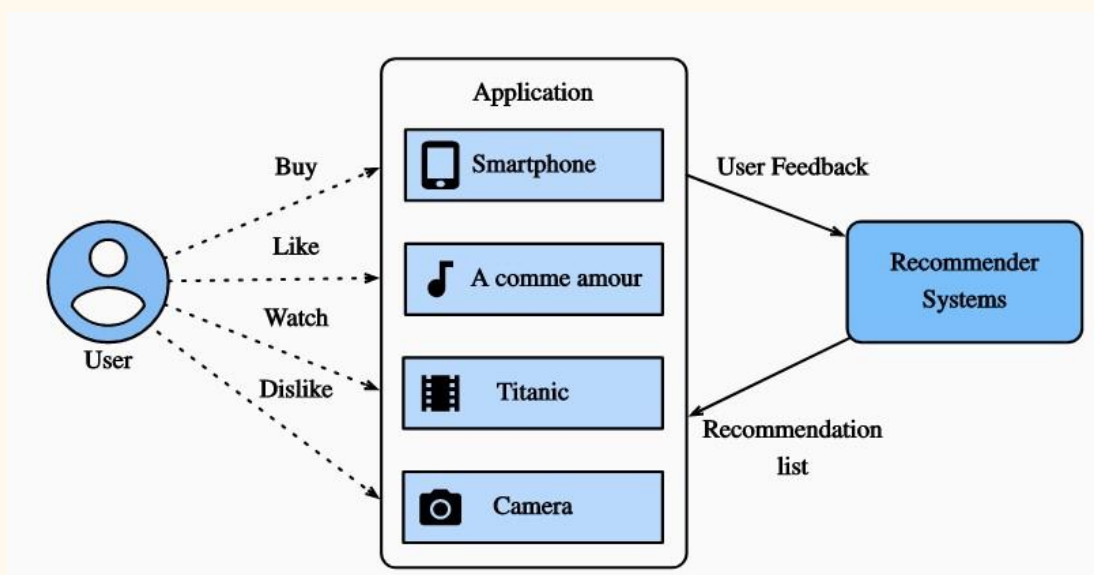


*Parcours Ingénieur Machine Learning*

<b>INTRODUCTION</b>	<b>3</b>
Typologie des algorithmes de recommandations	3
Travaux de Bhowmick et al.	4
Objectifs de ce projet	5
Un système de recommandation spécialisé dans les livres	5
L'apport des avancées dans le Deep Learning : L'exemple des transformers	5
L'impact écologique de l'entraînement des algorithmes de recommandation	5
<b>MÉTHODOLOGIE</b>	<b>6</b>
Les données et les outils	6
Les données	6
Les outils	6
Les algorithmes	7
L'algorithme basé sur la popularité	7
Les algorithmes basées sur le filtrage collaboratif	8
Les algorithmes basées sur le contenu	9
La métrique d'évaluation	10
<b>RÉSULTATS</b>	<b>10</b>
Description des données	10
Informations sur les livres	10
Informations sur les utilisateurs	10
Evaluations des modèles	11
Performance des algorithmes de popularité et de filtrage collaboratif	11
Émissions de CO2 de l'entraînement	11
<b>CONCLUSION</b>	<b>12</b>
Retour sur les objectifs	12
Perspectives et limites	12
<b>RÉFÉRENCES</b>	<b>13</b>
<b>ANNEXE I : Schéma explicatif du transformer BERT</b>	<b>16</b>

## INTRODUCTION

Les algorithmes spécialisés dans la recommandation sont omniprésents dans notre quotidien qu'on le souhaite ou non. Ils font la renommée de certaines entreprises comme Netflix, YouTube ou encore Spotify. On les retrouve aussi dans les sites d'e-commerce pour suggérer des produits à acheter en fonction de nos préférences.



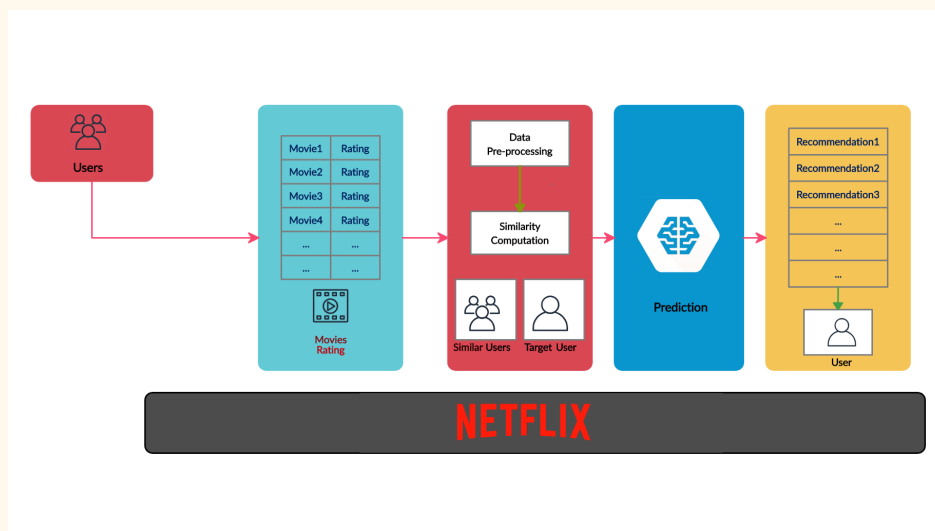
**Figure 1.** Illustration du processus de fonctionnement d'un type d'algorithmes de recommandation

En revanche, il existe peu de systèmes de recommandations basés uniquement sur les livres. Certains sites comme Babelio ou Esprit Critique suggèrent du contenu selon les notations d'œuvres que l'on a lu mais ce sont de rares exceptions qui se basent sur un seul type d'algorithmes de recommandation. Les livres ne bénéficient pas forcément des avancées technologiques des algorithmes de recommandation comme cela peut être le cas des autres médias.

## Typologie des algorithmes de recommandations

Les algorithmes de recommandations se divisent en plusieurs parties. On va retrouver trois grandes familles d'algorithmes de recommandations : les algorithmes de filtrage collaboratif, les algorithmes basés sur le contenu et les algorithmes hybrides qui vont mixer les deux premières approches.

Historiquement, les algorithmes les plus anciens sont les algorithmes basés sur le contenu. Ils sont apparus lors de l'émergence du web 1.0. Les algorithmes de filtrage collaboratif sont beaucoup plus récents. Ce sont ceux qui sont utilisés par des plateformes tels que Netflix ou Spotify.



**Figure 2.** Exemple du fonctionnement des algorithmes de prédiction de films selon Netflix

Plus récemment, des méthodes hybrides ont émergé et se basent sur les découvertes récentes des algorithmes de recommandation.

Enfin, on va retrouver des algorithmes de recommandations plus simples se basant sur la popularité. Ce sera le cas de la plateforme IMDb qui proposera à ses utilisateurs les films les mieux notés.

## Travaux de Bhowmick et al.

Dans la preuve de concept qui va suivre, plusieurs algorithmes vont être testés sur la base d'un article datant d'un article de Bhowmick et al. (2021). Dans cet article, les auteurs se sont intéressés à la comparaison de plusieurs algorithmes de recommandation dans le cas de la prédiction de films. Pour cela, ils ont utilisé la célèbre base de données Movielens en prenant des avis d'utilisateurs entre Mars 1996 et Septembre 2018.

Les auteurs ont fait le choix de tester 8 méthodes différentes : un algorithme basé sur la popularité du genre, un algorithme utilisant la corrélation de Pearson, des algorithmes basés sur le contenu des films, des algorithmes basés sur les interactions des utilisateurs et des algorithmes de Machine Learning utilisant des méthodes de clustering.

Pour évaluer la pertinence de leur modèle, ils se sont basés sur le jeu de données en lui-même qui a déjà des catégories.

Le but de cette étude était aussi de relever les avantages et les inconvénients de chacun de ces algorithmes en prenant en compte les dernières avancées en ce qui concerne les algorithmes de recommandation.

## Objectifs de ce projet

On va retrouver trois objectifs que cette preuve de concept va essayer de couvrir.

### 1. Un système de recommandation spécialisé dans les livres

La littérature montre une prépondérance d'articles et de tutoriels sur l'utilisation des algorithmes de recommandations sur des bases de données de films ou de musiques mais on trouve très peu de choses sur les livres. Les exemples disponibles en open source utilisent des bases de données anciennes et ne font que peu de comparaisons sur les différents types d'algorithmes. De plus, les algorithmes ne semblent pas optimisés et ne vont pas chercher à utiliser de nouvelles technologies.

### 2. L'apport des avancées dans le Deep Learning : L'exemple des transformers

Les algorithmes se basant sur le contenu sont les premiers à avoir été utilisés. Dans la partie contenu, il est possible de traiter des données non-tabulaires comme des images ou du texte. L'augmentation de la puissance des ordinateurs a permis de considérables avancées dans le traitement de ces données non-tabulaires. Dans le cas des données textuelles, de récentes méthodes comme les transformers permettent d'obtenir des vectorisations textuelles plus intéressantes que les méthodes classiques comme le Bag-Of-Word ou le Tf-Idf. L'idée ici sera donc de tester les résultats obtenus avec une vectorisation par l'intermédiaire d'un transformer. Les transformers sont des modèles pré-entraînés sur de grandes quantités de données et qui peuvent être réutilisés pour faire de l'extraction de features, de l'analyse de sentiments ou tout autres tâches en lien avec le traitement automatisé du langage.

### 3. L'impact écologique de l'entraînement des algorithmes de recommandation

Les algorithmes de recommandations sont omniprésents dans notre environnement. De plus, les utilisateurs commencent à se poser des questions sur l'impact écologique de leur activité numérique. Lors de cette preuve de concept, la mesure de l'émission de CO2 a été mesurée. Ces mesures sont surtout présentes à titre de sensibilisation sur nos usages numériques quotidien et leur impact sur l'environnement.

# METHODOLOGIE

## Les données et les outils

### 1. Les données

Contrairement aux films et, dans une moindre mesure, la musique, les jeux de données comportant des informations nécessaires à l'élaboration d'algorithmes de recommandation pour les livres sont peu nombreux. Il existe deux importants datasets.

Le premier a été obtenu à l'aide de la communauté Bookcrossing. Ce dataset comporte des informations sur les utilisateurs (âge, localisation) mais aussi sur les livres. Cependant, il date de 2005.

Le choix s'est donc porté sur le second dataset existant et obtenu grâce à l'API du site GoodReads. L'API a cessé de fonctionner en décembre 2020. Les données sont visibles et téléchargeables sur Kaggle à cette adresse : <https://www.kaggle.com/bahramjannesarr/goodreads-book-datasets-10m>.

Ce dataset contient une vingtaine de fichiers que l'on peut diviser en deux catégories : une partie des fichiers contenant les informations sur les livres et une partie contenant des informations sur les évaluations des utilisateurs.

### 2. Les outils

Les différentes parties de ce projet ont été rédigées sous la forme de notebook Jupyter. Le langage de programmation choisi est *Python*. Pour les étapes de nettoyage de données, d'exploration et de modélisation, plusieurs packages ont été utilisés. Parmi ces outils, on retrouvera l'utilisation de la library *Surprise* spécialisée dans les algorithmes de recommandation ainsi que *sentence\_transformers* qui regroupe plusieurs transformers dérivés de BERT. Enfin, la library *CodeCarbon* a été utilisée dans le but d'obtenir des informations sur les émissions de gaz à effets de serre de notre code.

*GitHub* a été utilisé comme logiciel de versionning lors de la réalisation de ce projet.

Le code écrit pour ce projet est disponible sur GitHub<sup>1</sup>.

---


<sup>1</sup> <https://github.com/Sylviane/Recommender System Books>

## Les algorithmes

Plusieurs algorithmes vont être testés. Ils se basent sur différentes hypothèses de base. On va s'intéresser à un algorithme basé sur la popularité, des algorithmes basés sur du filtrage collaboratif (*collaborative filtering*) et des algorithmes basés sur le contenu (*content-based filtering*).

### 1. L'algorithme basé sur la popularité

Il s'agit d'un algorithme assez basique. Il part du principe que les utilisateurs d'une plateforme vont avoir les mêmes goûts et donc qu'un nouvel utilisateur va aimer le contenu le plus populaire de la plateforme. On est ici sur une méthode se basant sur des estimations bayésiennes. La formule utilisée se base sur celle de la plateforme IMDb ([https://help.imdb.com/article/imdb/track-movies-tv/weighted-average-ratings/GWT2DSBYVT2F25SK?ref=helpsect\\_pro\\_2\\_8#](https://help.imdb.com/article/imdb/track-movies-tv/weighted-average-ratings/GWT2DSBYVT2F25SK?ref=helpsect_pro_2_8#)).



Ranking System

$$W = \frac{Rv + Cm}{v + m}$$

W= Weighted Rating  
 R = average rating for the movie  
 v = number of votes for the movie  
 m = minimum votes required to be listed in the Top 250  
 (currently 3000)  
 C = the mean vote across the whole report (currently 6.9)

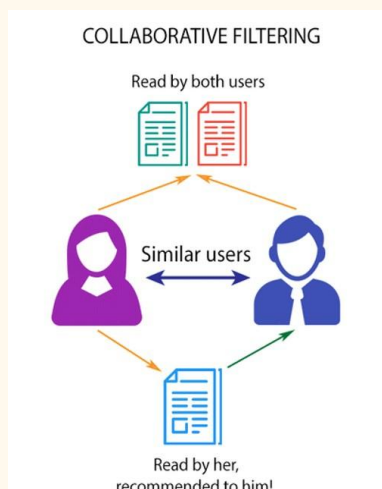
**Figure 3.** Formule du ranking system utilisée par IMDb

Les valeurs m et C ont été calculées au préalable pour adapter cette formule à notre jeu de données. Cette formule sera implémentée sous forme de fonction.

### 2. Les algorithmes basées sur le filtrage collaboratif

Ces algorithmes sont les plus récents mais aussi ceux qui sont le plus présents dans les applications de recommandation de contenu. Ces algorithmes ont pu se développer avec l'augmentation des interactions entre utilisateurs. Ces interactions prennent plusieurs formes : un avis explicite, un avis implicite, un nombre de clics, le temps de visionnage d'une vidéo, etc.

Des utilisateurs similaires vont avoir des goûts similaires. On va donc conclure que si un contenu a été consulté par un utilisateur, on va pouvoir le proposer à des utilisateurs ayant les mêmes caractéristiques.

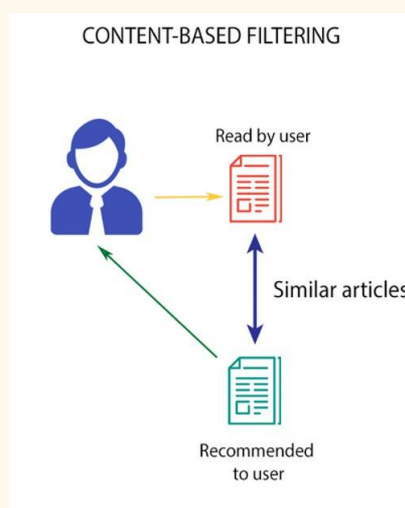


**Figure 4.** Principe de base des algorithmes de filtrage collaboratif

Les algorithmes de filtrage collaboratif se divisent en plusieurs familles. On va retrouver des algorithmes se basant sur de la factorisation de matrices comme le NMF, des algorithmes de (co)clustering, des algorithmes de type Slope One et des algorithmes de décomposition en valeurs singulières comme le célèbre SVD. Dans notre projet, on a choisi de tester les algorithmes suivants : NMF, KNN, SVD, Co Clustering et Slope One.

### 3. Les algorithmes basées sur le contenu

La dernière famille d'algorithmes qui sera étudiée dans ce projet porte sur le contenu de nos livres. Ce sont les premiers algorithmes qui ont été développés pour la recommandation de contenu. Ils partent du principe que si un utilisateur a consommé un contenu, il devrait apprécier de consommer un contenu aux caractéristiques similaires.



**Figure 5.** Principe de base des algorithmes basé sur le contenu



Dans ce projet, le contenu que l'on va utiliser correspond aux descriptions des livres que l'on trouve sur les quatrièmes de couverture. Habituellement, les algorithmes de recommandations utilisent uniquement le genre du contenu.

L'utilisation de données textuelles va nécessiter un preprocessing particulier. Une fois les données textuelles nettoyées, il faut passer par une étape de vectorisation. Ici le choix a été fait de comparer deux méthodes de vectorisation : une méthode plus ancienne qu'est le TF-Idf et une méthode plus récente qui se base sur les transformers et notamment l'un des plus complets : BERT. Le schéma explicatif de ce transformer est disponible en annexe.

## La métrique d'évaluation

Tous les algorithmes utilisés ne peuvent être évalués. Seuls les algorithmes basés sur la popularité et de filtrage collaboratif sont évalués à l'aide du Root Mean Squared Error (RMSE).

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (y_i - \hat{y})^2}{n}}$$

**Figure 6.** Formule du Root Mean Squared Error

Tous les algorithmes sont soumis à une évaluation de leur émission de CO2 par l'utilisation du tracker présent dans la librairie CodeCarbon.

# RÉSULTATS

## Description des données

Après nettoyage de nos données, on dispose d'informations sur 140.000 ouvrages et de 360.000 interactions d'utilisateurs.

### 1. Informations sur les livres

Les livres ont été publiés entre 1829 et 2020. On observe une augmentation du nombre de publications aux alentours des années 2000.

Les évaluations sont comprises entre 1 et 5 avec une note moyenne aux alentours de 4. Parmi les auteurs les plus présents dans notre jeu de données, on retrouve Nora Roberts, Agatha Christie et Stephen King.

Les livres les plus notés sont *The Catcher in the Rye*, *The Great Gatsby* et *The Da Vinci Code*. Enfin, on observe que la plupart des ouvrages sont en anglais.

## 2. Informations sur les utilisateurs

On dispose de peu d'informations concernant les utilisateurs. On ne possède que leurs notations et les ouvrages qu'ils ont notés. On remarque que l'utilisateur le plus actif a évalué plus de 3500 ouvrages.

## Evaluation des modèles

L'évaluation des modèles porte sur deux aspects. Le premier aspect est la performance de l'algorithme à réussir une recommandation pour un utilisateur. On va donc regarder le RMSE des algorithmes de filtrage collaboratif ainsi que celui de popularité.

Les modèles se basant sur le contenu ne peuvent pas être évalués de manière fiable car il s'agit de modèles se basant sur de l'apprentissage non supervisé. On peut donc juste comparer les résultats des prédictions.

Le deuxième aspect évalué est l'émission de CO2 lors de l'entraînement de chacun de nos modèles. On va pouvoir utiliser cette évaluation sur tous les modèles sauf celui basé sur l'algorithme de popularité.

### 1. Performance des algorithmes de popularité et de filtrage collaboratif

Pour la performance, la métrique choisie est le Root Mean Squared Error qui calcule l'erreur moyenne de la prédiction. Ici, ce qui est prédit est la note qu'un utilisateur va mettre à un livre qu'on lui suggère.

Modèles	Popularité	SVD	NMF	KNN	SlopeOne	Coclust.
RMSE	0.108	0.765	0.875	0.770	0.891	0.891

**Tableau 1.** Résumé des performances obtenus pour chacun des algorithmes

On remarque que les algorithmes de type SlopeOne et de Co-clustering ne sont pas efficaces pour effectuer cette prédiction. En revanche, l'algorithme de popularité obtient une faible erreur sur la notation. Cependant, il faut garder à l'esprit que ce modèle se base sur une estimation bayésienne et a donc une hypothèse de départ des autres modèles. Si on regarde les algorithmes ayant la même hypothèse de départ, on remarque que l'algorithme SVD est plus performant.

### 2. Émissions de CO2 de l'entraînement

Lorsque l'on s'intéresse à l'impact écologique des utilisations numériques, on va utiliser l'équivalent CO2. Dans notre cas, les données sont exprimées en g équivalent CO2.

Modèles	SVD	NMF	KNN	SlopeOne	Coclust.	Tf-Idf	BERT
GHG	0.027	0.029	0.015	0.034	0.015	0.2398	0.3380

**Tableau 2.** Résumé des émissions de CO2 pour chacun des algorithmes

Le modèle le plus écologique est le KNN. Il semblerait que ça soit aussi celui qui obtient de meilleure performance après le SVD. En revanche, les modèles se basant sur le contenu des livres sont les plus polluants avec un maximum de 0.34 g éq. CO2 pour l'algorithme utilisant le transformers BERT.

## CONCLUSION

### Retour sur les objectifs

### Perspectives et limites

## REFERENCES

- Bhowmick, H., Chatterjee, A. & Sen, J. (2021). Comprehensive Movie Recommendation System. 8th International Conference on Business Analytics and Intelligence, 20-22/12/2021. <https://arxiv.org/abs/2112.12463>
- Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc.
- CodeCarbon (2021). <https://codecarbon.io/>
- Delvin, J., Chang, M.-W., Lee, K. & Toutona, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, Google AI Language, <https://arxiv.org/abs/1810.04805>
- Garodia, S. (2020). Content-based Recommender Systems in Python, <https://medium.com/analytics-vidhya/content-based-recommender-systems-in-python-2b330e01eb80>
- Github. (2020). *GitHub*. Retrieved from <https://github.com/>
- Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585, 357–362. <https://doi.org/10.1038/s41586-020-2649-2>
- Honnibal, M., & Montani, I. (2017). *spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing*.
- Hug, N. (2020). Surprise: A Python library for recommender systems, Journal of Open Source Software. <http://surpriselib.com/>
- Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90–95.
- Kluyver, T., Ragan-Kelley, B., Fernando Perez, Granger, B., Bussonnier, M., Frederic, J., ... Willing, C. (2016). Jupyter Notebooks – a publishing format for reproducible computational workflows. In F. Loizides & B. Schmidt (Eds.), *Positioning and Power in Academic Publishing: Players, Agents and Agendas* (pp. 87–90)
- McKinney, W., & others. (2010). Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference* (Vol. 445, pp. 51–56).
- Myhill, C. (2020). The Paradox of choice. Why more isn't always better, Pixelfridge, <https://pixelfridge.digital/the-paradox-of-choice-why-more-isnt-always-better/>

- Paialunga, P. (2022). Hands-On Content Based Recommender System using Python, <https://towardsdatascience.com/hands-on-content-based-recommender-system-using-python-1d643bf314e4>
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... others. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12(Oct), 2825–2830.
- Oesper, L., Merico, D., Isserlin, R., & Bader, G. D. (2011). WordCloud: a Cytoscape plugin to create a visual semantic summary of networks. *Source Code for Biology and Medicine*, 6(1), 7.
- Reimers, N. & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, <https://arxiv.org/abs/1908.10084>
- Time For the Planet. (2021). Pourquoi parle-t-on en équivalent CO2 ?, <https://www.time-planet.com/fr/tout-savoir/dereglement-climatique/pourquoi-parle-t-on-en-equivalent-co2>
- Van Rossum, G., & Drake, F. L. (2009). *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace.
- Waskom, M. et al., 2017. *mwaskom/seaborn: v0.8.1 (September 2017)*, Zenodo. Available at: <https://doi.org/10.5281/zenodo.883859>.
- Zhang, A., Lipton, C. Z., Li, M. & Smola, A. J. (2021). Dive into Deep Learning, Chapitre 16: Recommender System, [https://d2l.ai/chapter\\_recommender-systems/recsys-intro.html](https://d2l.ai/chapter_recommender-systems/recsys-intro.html)
- Ziegler, C.-N., McNee, S. N., Konstan, J. S. & Lausen, G. (2005). Improving Recommendation Lists Through Topic Diversification. Proceedings of the 14th International World Wide Web Conference, <http://www2.informatik.uni-freiburg.de/~ciegler/BX/>

# ANNEXES

## ANNEXE I : Schéma explicatif du transformer BERT

