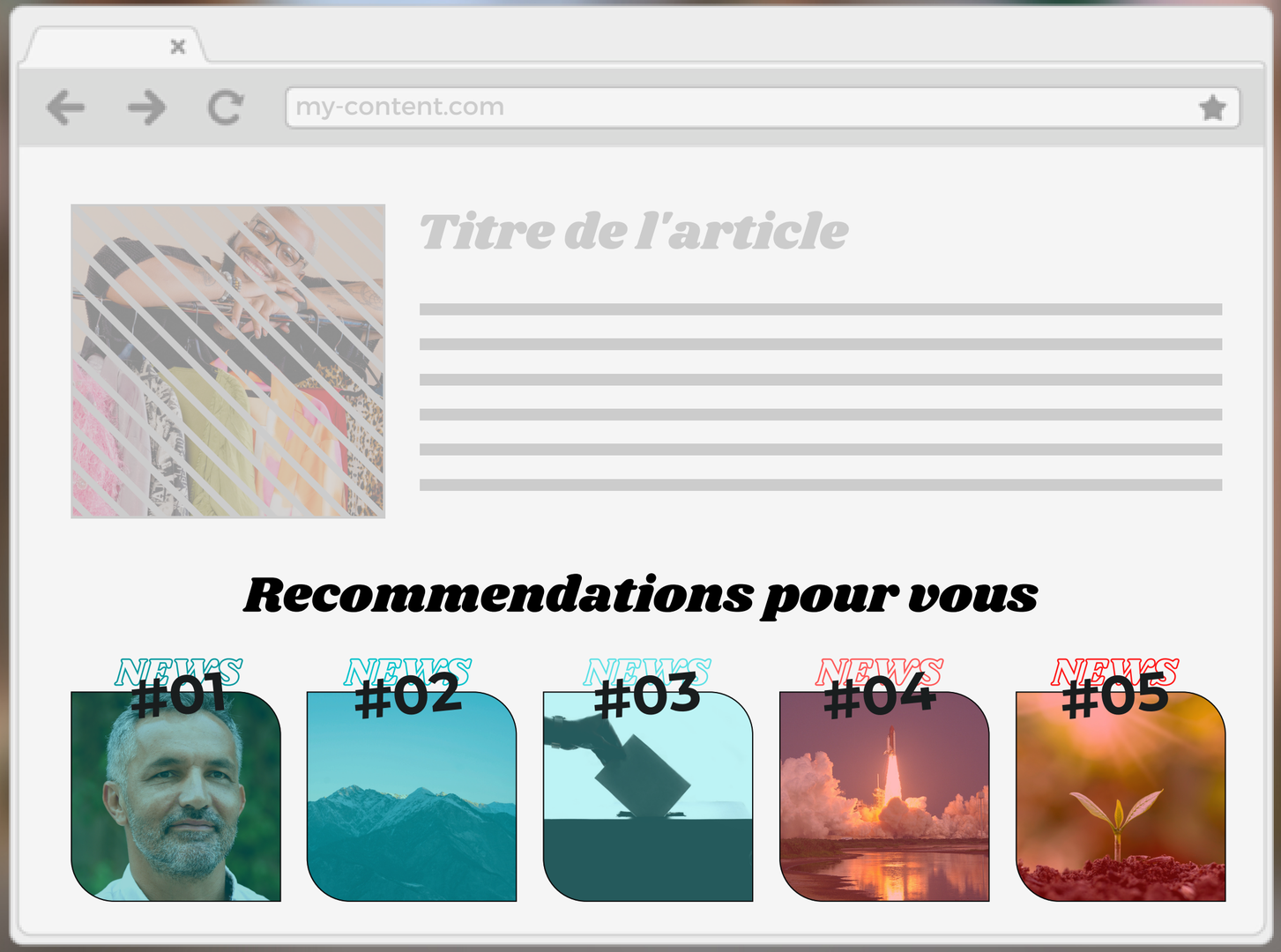
Recommandation de contenu

My Content



Contexte

My Content

My Content est une start-up qui veut encourager la lecture en **recommandant des contenus** pertinents pour ses utilisateurs.

Les lecteurs sont submergés par un tsnunami d'articles ou de livres toujours plus nombreux, et cette abondance peut diminuer l'envie de lire.

Nous aimerions **pouvoir proposer des lectures correspondant aux sujets d'intérêt de nos utilisateurs, sans les isoler dans des bulles**.

Objectif du projet

My Content

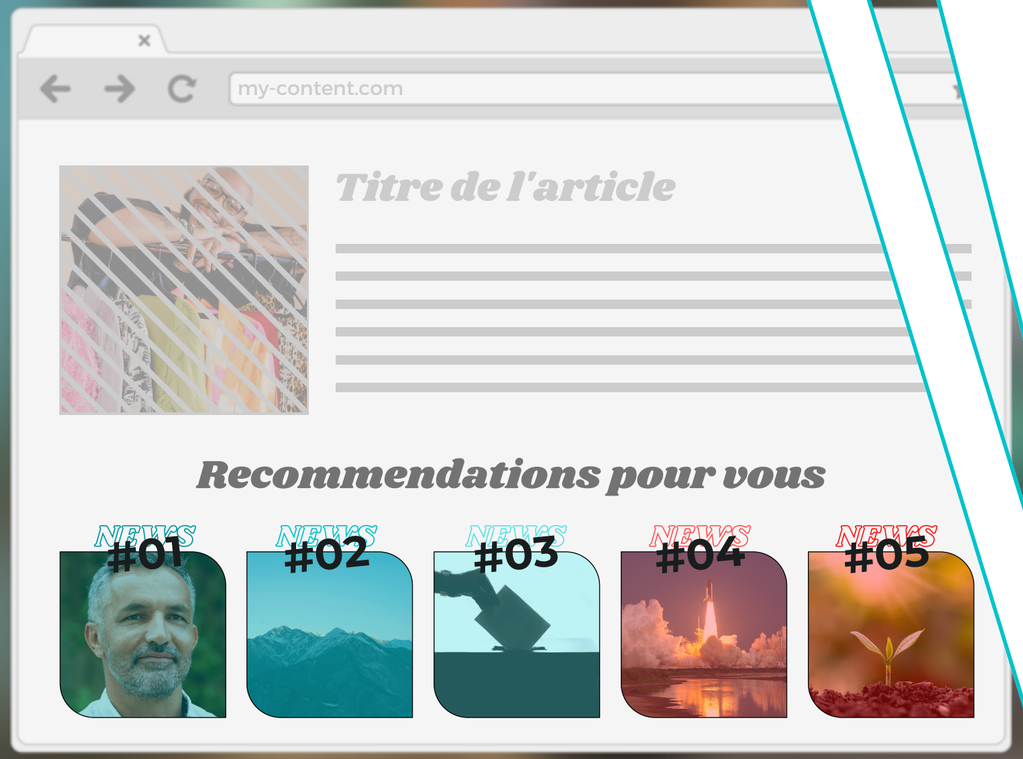
Notre premier objectif est donc de réaliser un MVP permettant d'interagir via une app. simple avec un modèle de recommandation développé sur la base d'un jeu de données représentatif.

Pour ce faire, nous avons suivi ces étapes :

- **EDA - analyses exploratoires des données**
- **Spot Checking - recherche de modèles**
- **Packaging - création d'un modèle Hybrid**
- **Déploiement serverless - azure function**
- **Création d'un client - streamlit**

Systèmes de recommandation

My Content



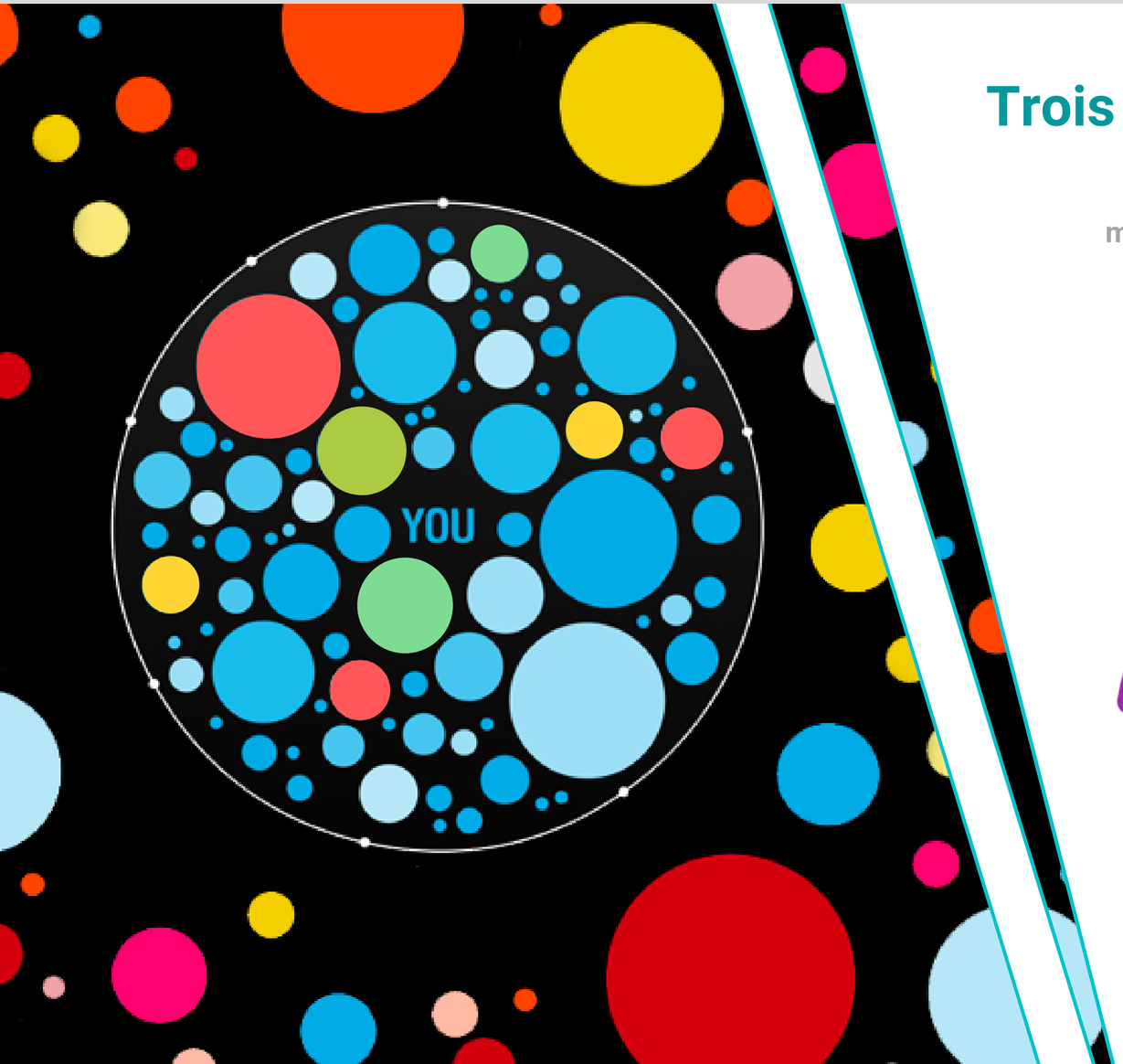
Les **systèmes de recommandation** recherchent des **similarités entre utilisateurs ou éléments** (livres, articles ...) sur la base d'**indicateurs implicites** ou **explicites** suggérant le degré de satisfaction (user-item interaction matrix)

Ces similarités permettent les recommandations.

Users	User-item interactions matrix	Items
suscribers	rating given by a user to a movie (integer)	movies
readers	time spent by a reader on an article (float)	articles
buyers	product clicked or not when suggested (boolean)	products
...		

Systèmes de recommandation

My Content

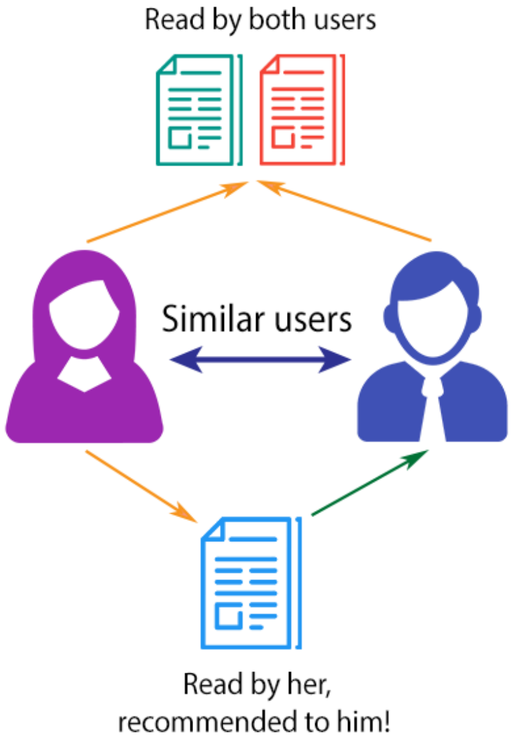


Trois approches principales

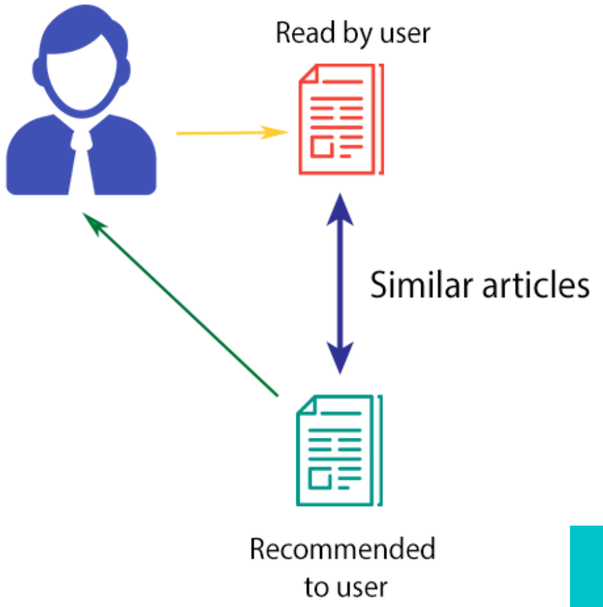
Collaborative filtering

memory based : user-user / item-item

model based : matrix factorization



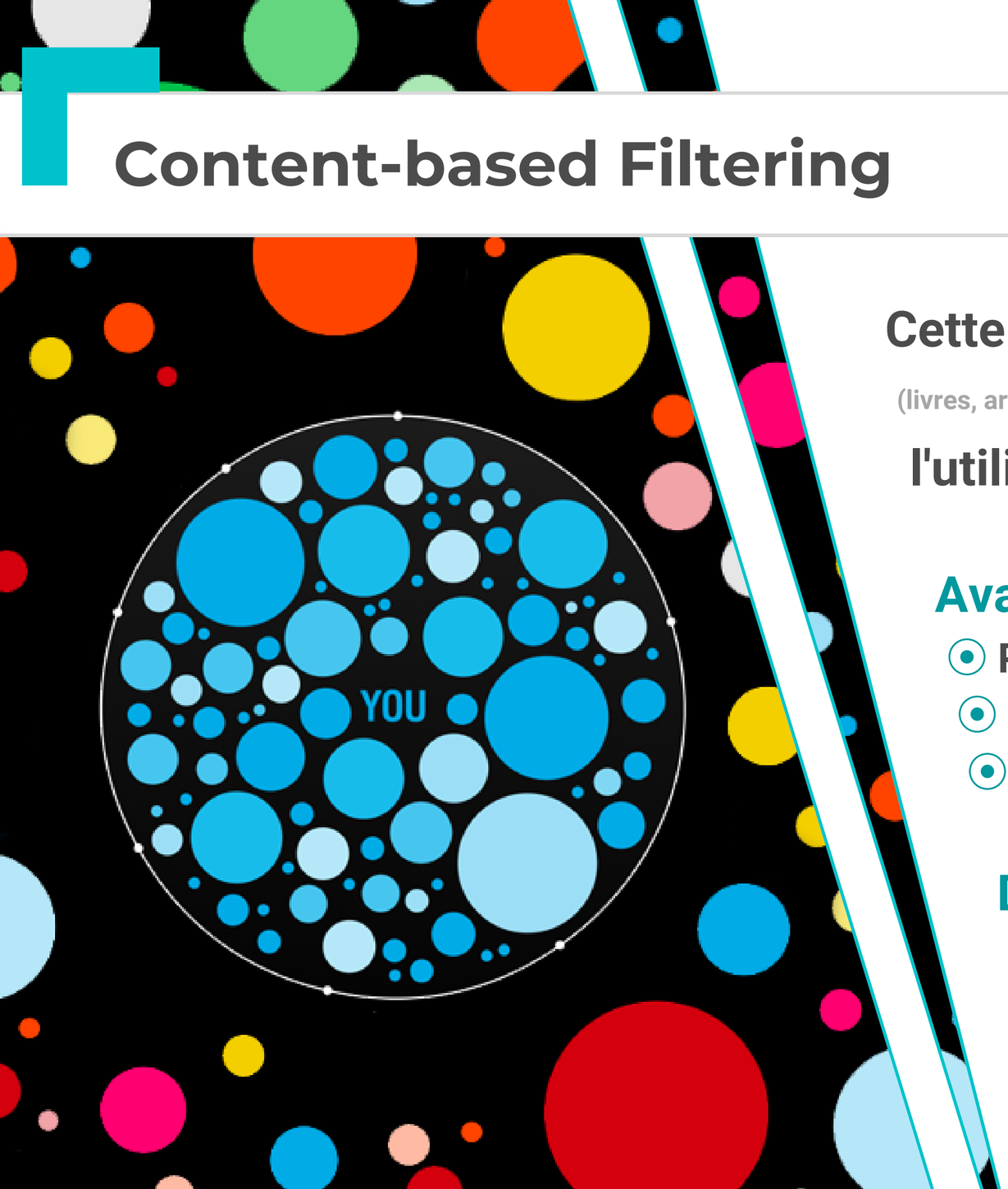
Content-based filtering



Hybrid filtering

Content-based Filtering

My Content



Cette approche utilise les similarités entre les items, (livres, articles ...) **ainsi que les informations accumulées sur l'utilisateur pour faire des recommandations.**

Avantages

- Pas besoin d'infos sur les autres utilisateurs
- Permet des recommandations de niche
- Non-supervisé donc pas de problème de cold-start

Désavantages

- Nécessite des connaissances sur l'item (pour les embeddings)
- Se base uniquement sur les centres intérêt connus de l'utilisateur cible (pas de nouveaux sujets / bulles de filtres)

Collaborative Filtering

My Content

Cette approche fais des recommandations en utilisant les préférences des utilisateurs similaires.

Avantages

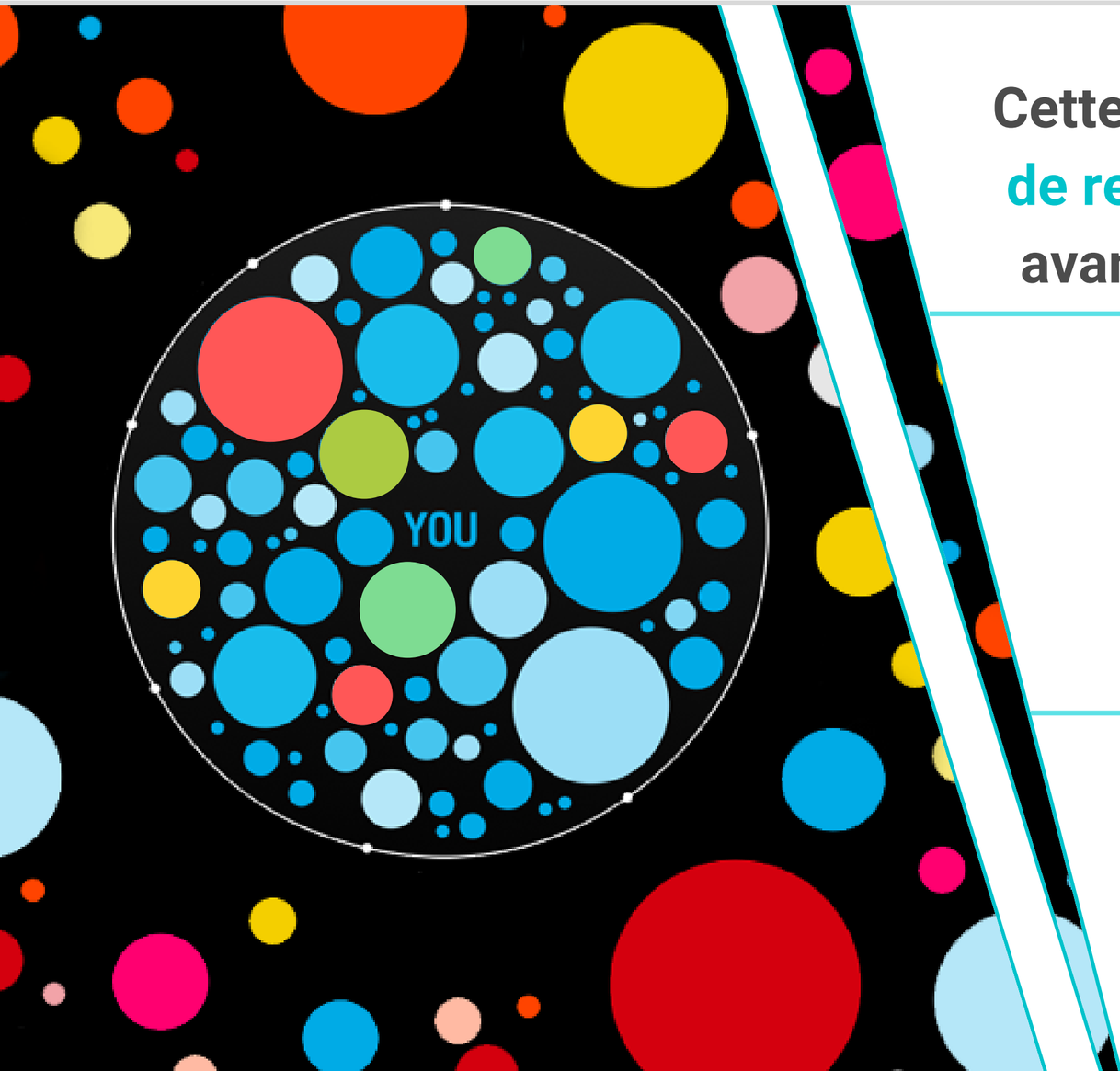
- Ne nécessite pas de connaissance de l'item (embeddings appris)
- Permet de proposer de nouveaux sujets d'intérêt
- Ne nécessite que la matrice d'interaction user-item
- Peut faire du user-user ou du item-item

Désavantages

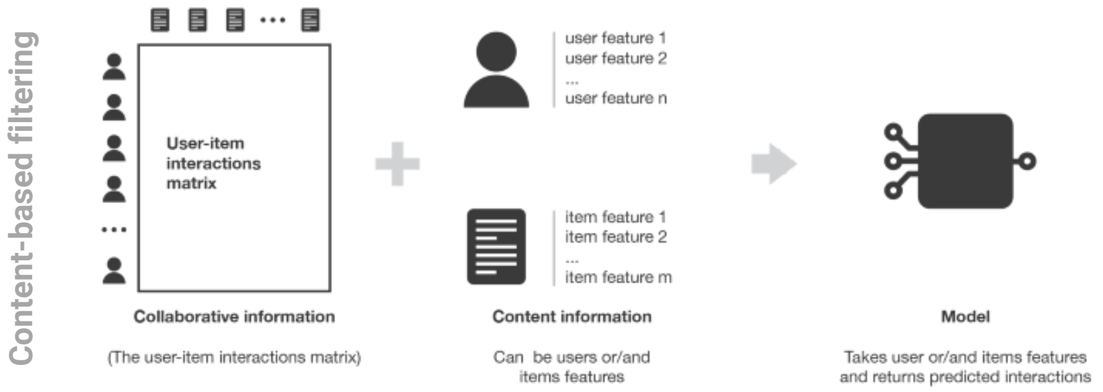
- Difficile d'inclure des features annexes (age, sex, article_category...)
- Doit prendre en compte le problème du cold-start

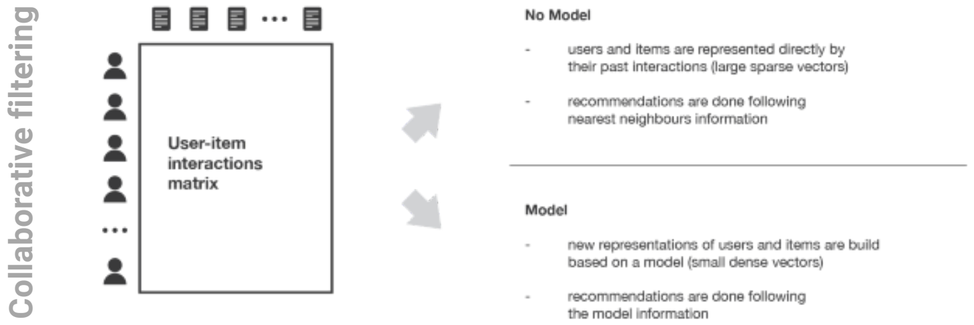
Hybrid Filtering

My Content



Cette approche combine plusieurs méthodes de recommandation pour essayer d'utiliser les avantages de chacune d'entre elles.





Le jeu de données

My Content

Présentation du jeu de données

Nous avons travaillé avec un jeu de données venant de Globo.com et réparti sur 388 fichiers qu'il a fallu rassembler.

Cet ensemble contient:

- **2 988 181 interactions utilisateurs / articles**
- **364 047 méta-données d'articles**
- **364 047 embeddings d'articles**

Le jeu de données

My Content

EDA - Vérifications de base

Les vérifications des erreurs les plus fréquentes
(valeurs manquantes, doublons, outliers, erreurs de format, erreurs lexicales, contenus multiples)
puis les analyses univariées et multivariées ont menées aux actions suivantes :

- Suppression des colonnes inutiles au projet
- Traitement des outliers (durée des sessions, nombre de mots ...)
- Suppression des utilisateurs trop peu actifs
- Suppression des articles trop peu lus
- Création d'un score implicite
- Normalisation des données (scores)

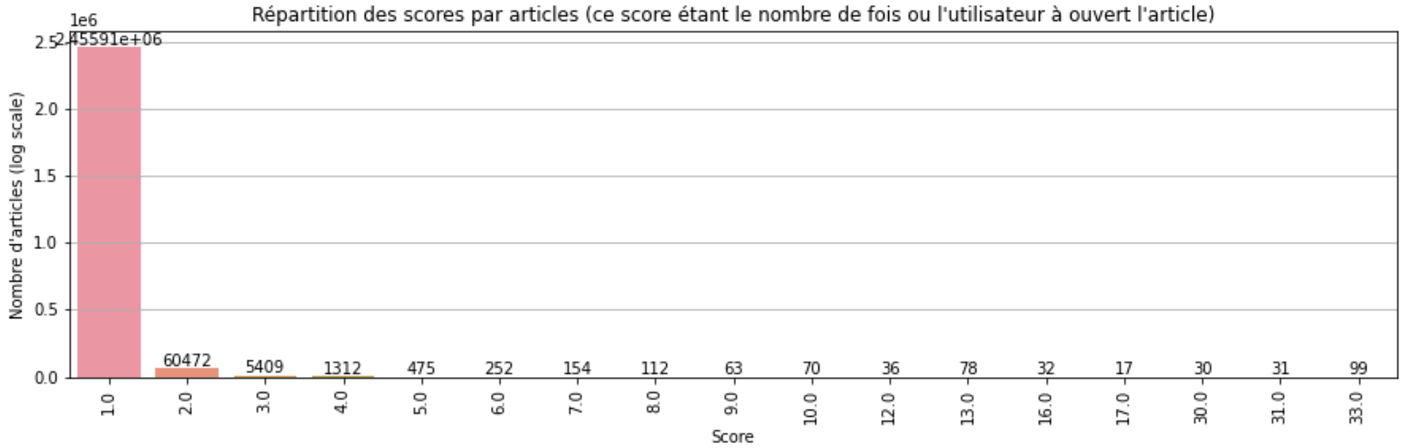
Le jeu de données

My Content

EDA - Création d'un score implicite

Score basé sur le nombre de re-lectures

(nombre de clicks sur un même article par un même utilisateur)



	1.0%	25%	50%	75%	97.0%	98.0%	99.0%	count	max	mean	min	std
score	1.00	1.00	1.00	1.00	1.00	2.00	2.00	2524552.00	33.00	1.03	1.00	0.35

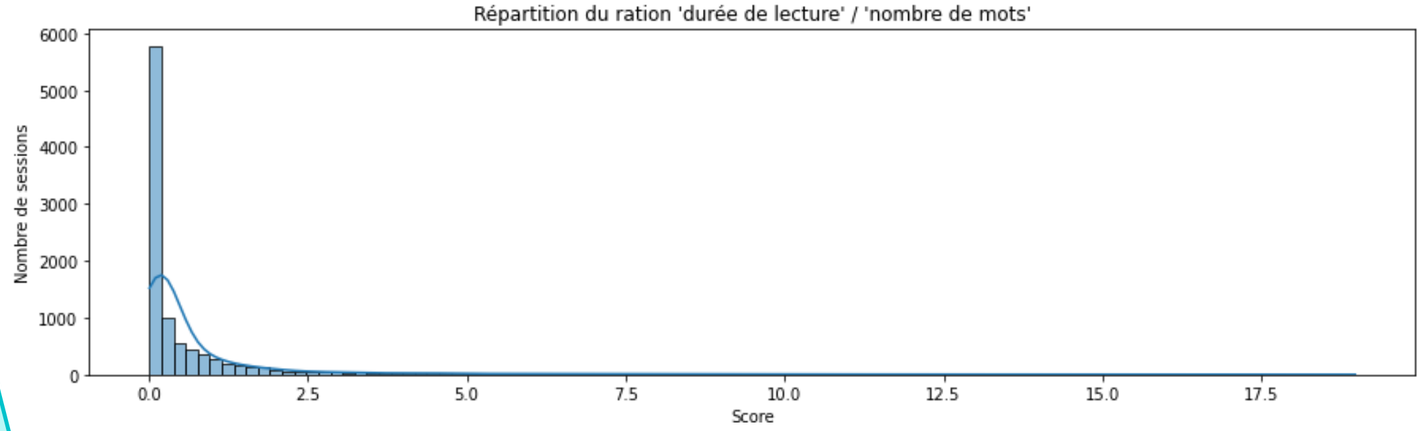
Le jeu de données

My Content

EDA - Création d'un score implicite

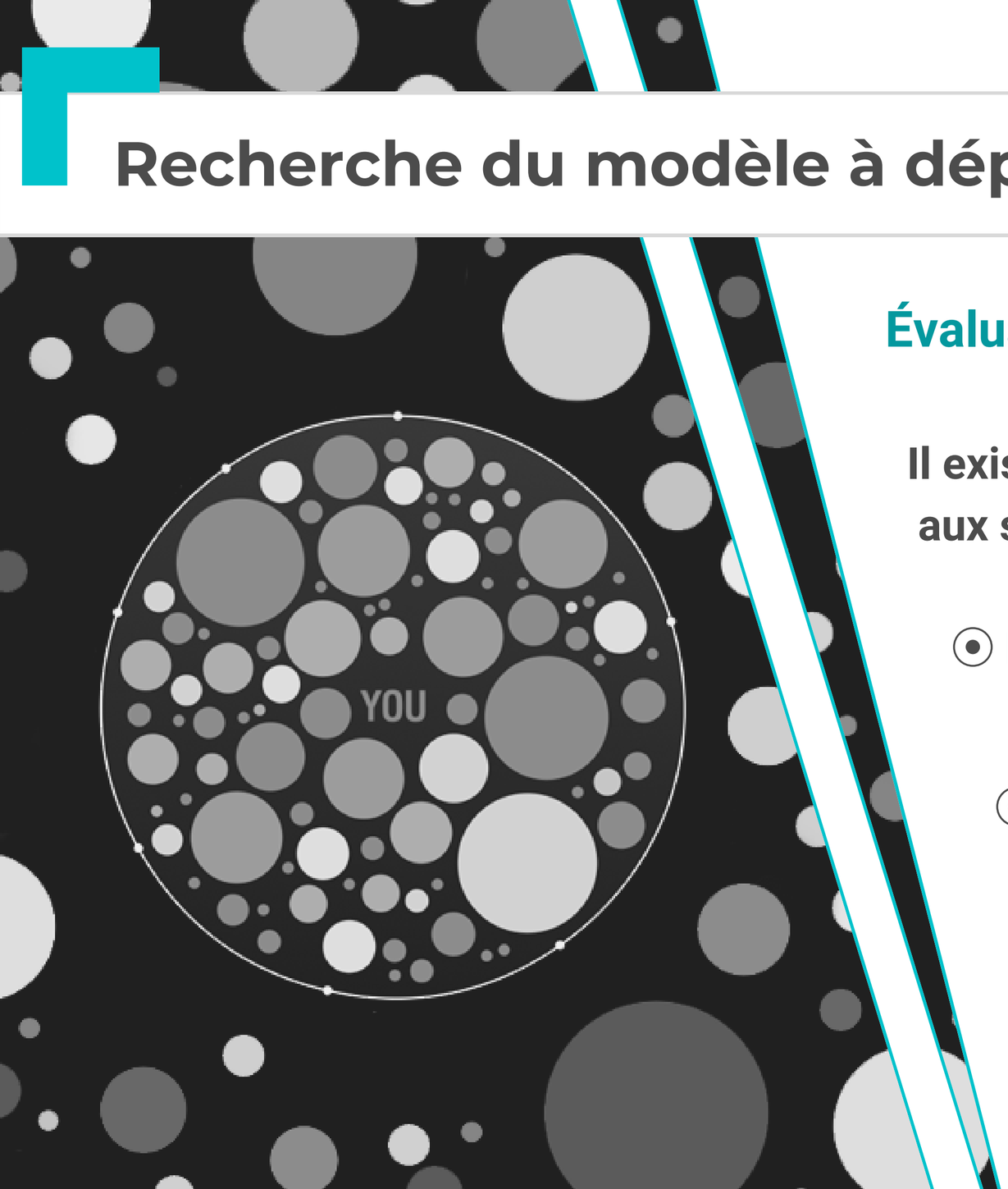
Score basé un ratio temps de lecture / nombre de mots

(le temps de lecture étant l'écart entre le timestamp de 2 articles d'une même session)



	0.5%	1.0%	25%	50%	75%	99.0%	99.5%	count	max	mean	min	std
duration_ratio	0.03	0.04	0.14	0.18	0.67	10.81	13.27	1855641.00	42.78	0.85	0.01	1.95

Recherche du modèle à déployer



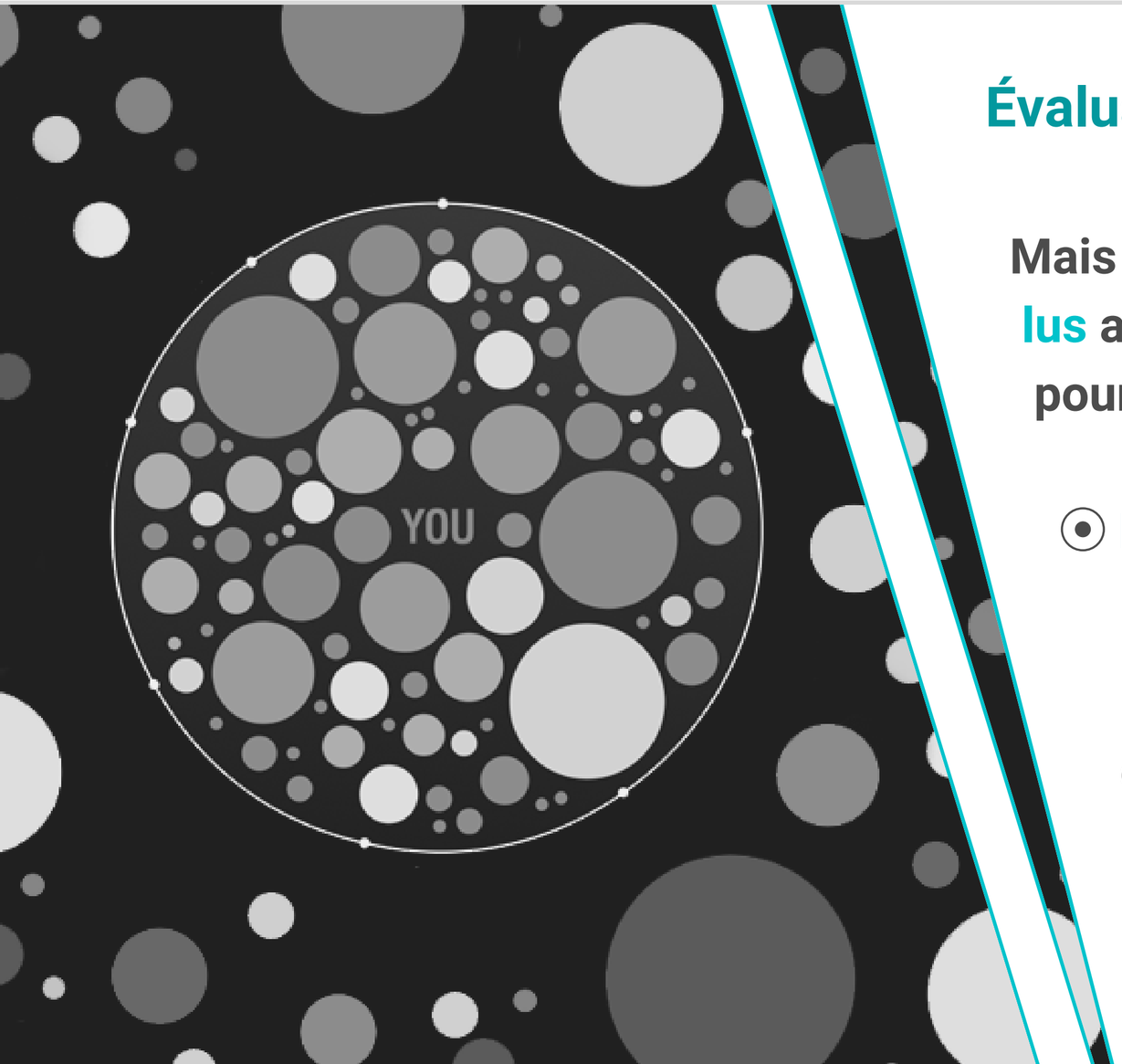
Évaluations

Il existe un certain nombre de métriques adaptées aux systèmes de recommendations...

- **RMSE** ou **MAE** pour évaluer des modèles basés sur des **scores**
- **Recall@k**, **Precision@k** et **Ap@k** pour évaluer les recommendations faites à **un utilisateur précis**.
- **Map@k** ou **nDCG@k** pour évaluer l'ensemble du jeu de données (le second étant plus adapté pour les modèles avec plus de 2 niveaux de pertinence)

Recherche du modèle à déployer

My Content

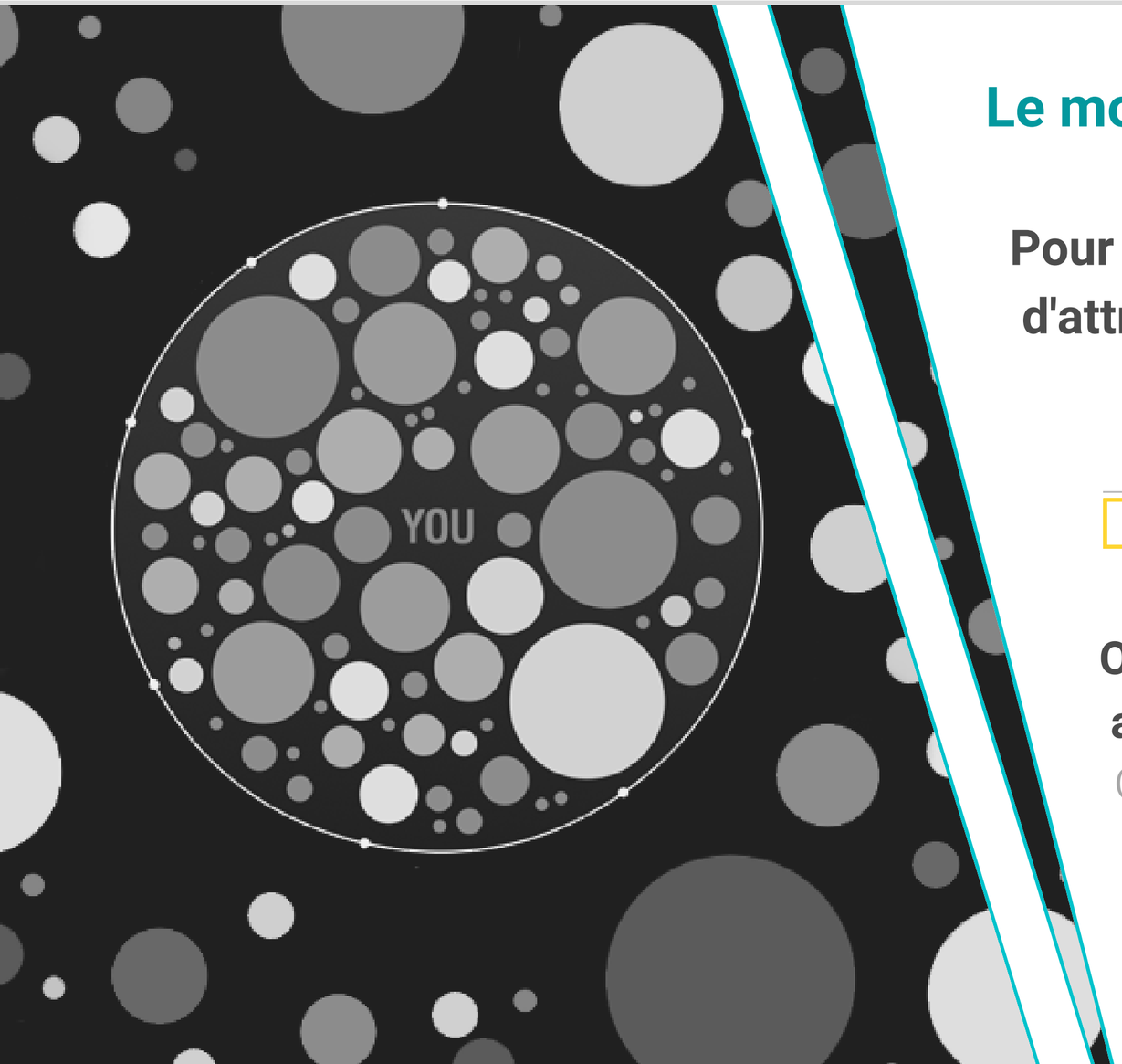


Évaluations

Mais nous voulons évaluer la proximité des sujets / articles **lus** avec les sujets / articles **recommandés** similairement pour le **collaborative filtering** et le **content-based filtering**.

- **Mean cosine similarity** pour évaluer la **similarité** entre l'embedding moyen des **articles recommandés** et l'embedding moyens des **articles effectivement lus**.
- **Map@K** pour évaluer le nombre d'éléments pertinents présents dans les top-k recos...
- À terme, de l'**A / B testing** serait préferable

Recherche du modèle à déployer



Le modèle de référence

Pour établir ce modèle de référence, il a été décidé d'attribuer 5 articles au hasard à chaque utilisateur.

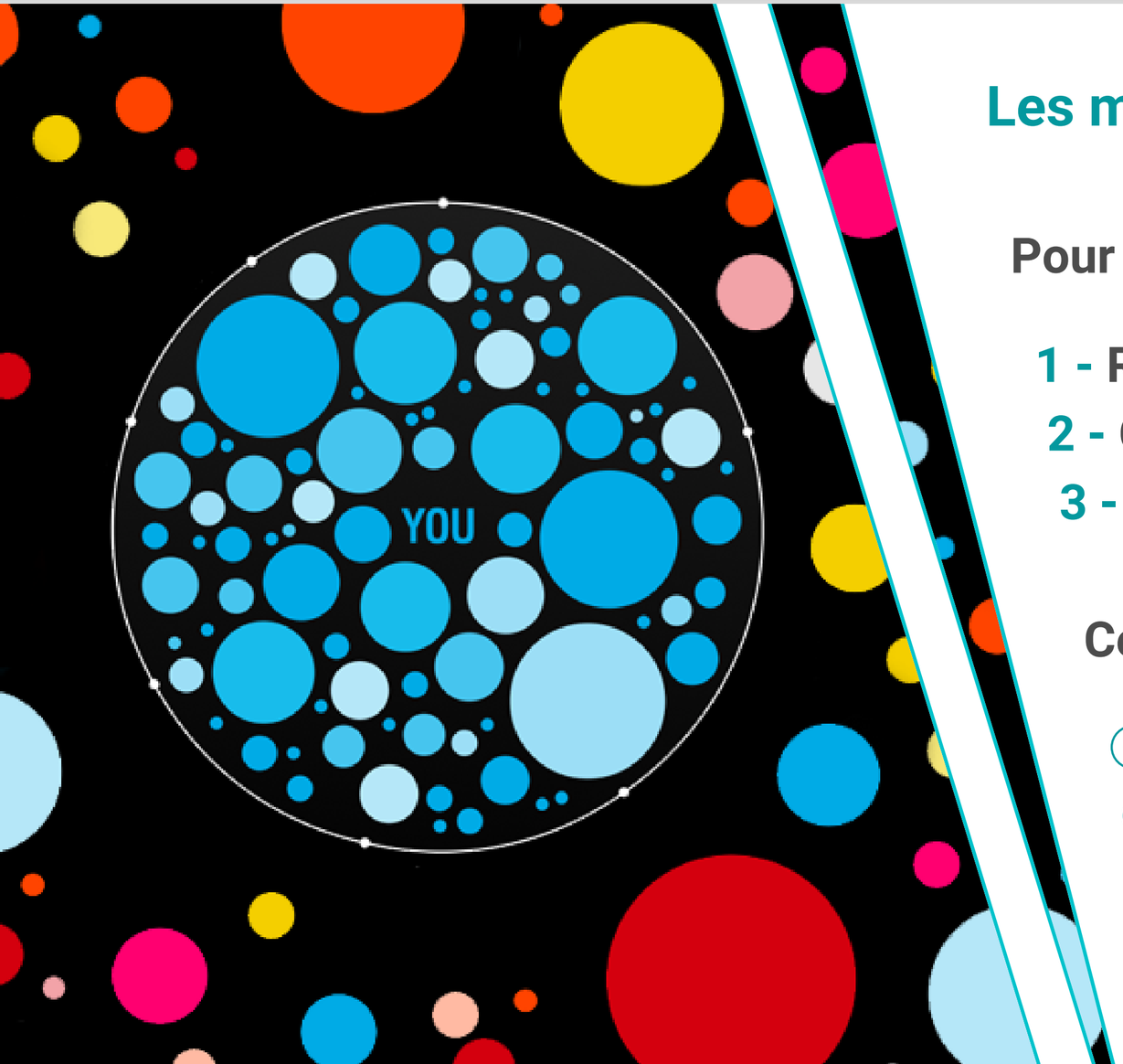
	mean_cosine_similarity	map@k	training_time
Baseline model - Random	0.482212	0.0	0

On constate que la mean_cosine_similarity n'est pas autour de 0 comme on pourrait s'y attendre.

(les articles ne couvrent probablement pas tout le spectre de l'embedding choisi)

Recherche du modèle à déployer

My Content



Les modèles de content-based filtering

Pour ces modèles, nous avons procédé en 3 temps :

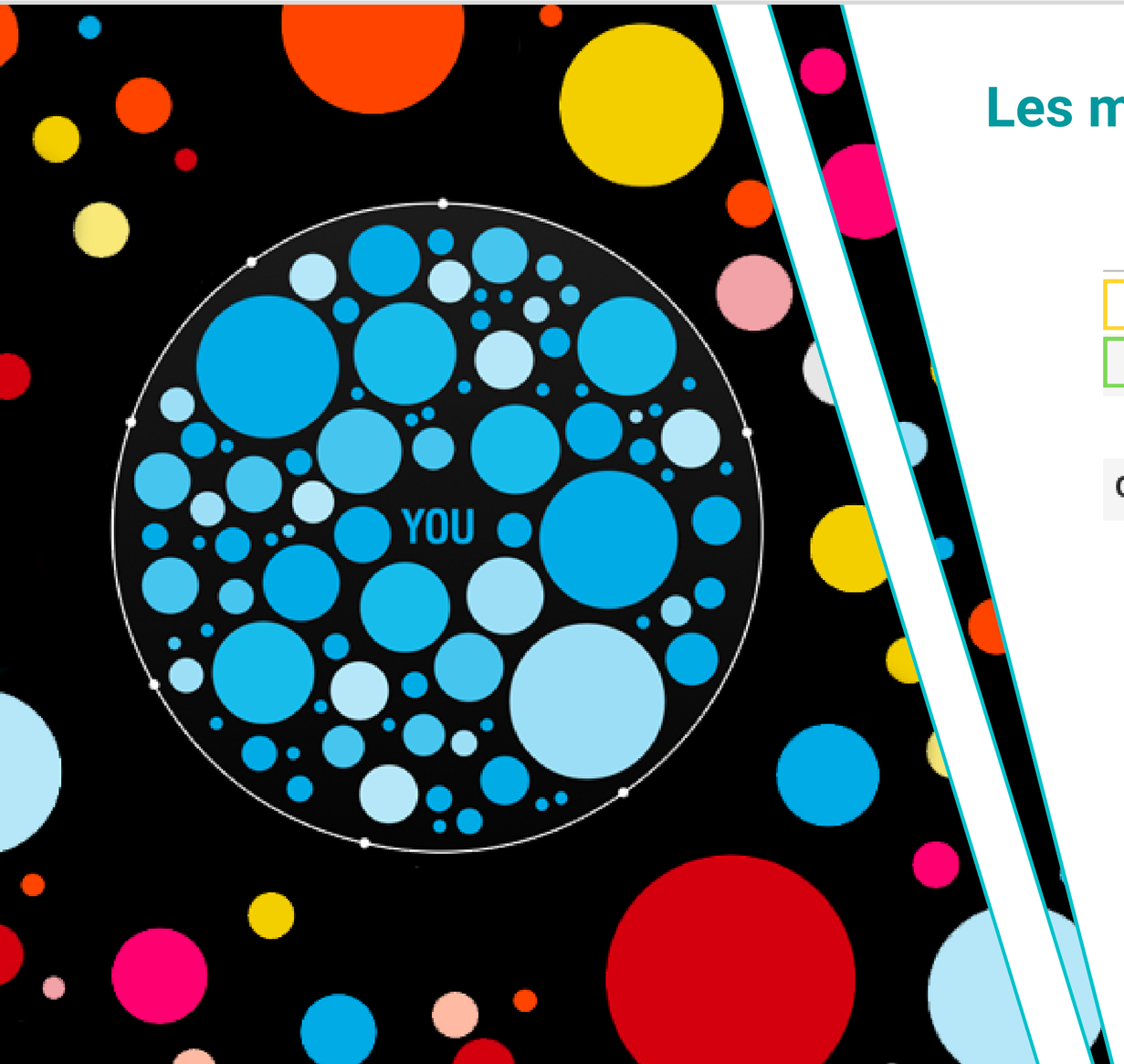
1 - Recherche des X derniers articles lus par l'utilisateur
2 - Calcul de la moyenne des embedding des articles lus
3 - Recherche par similarité cosinus des 5 plus proches

Ce processus a été décliné pour produire 3 modèles :

- un modèle utilisant tout l'historique
- un modèle utilisant les 5 derniers articles
- un modèle utilisant le dernier article lu

Recherche du modèle à déployer

My Content

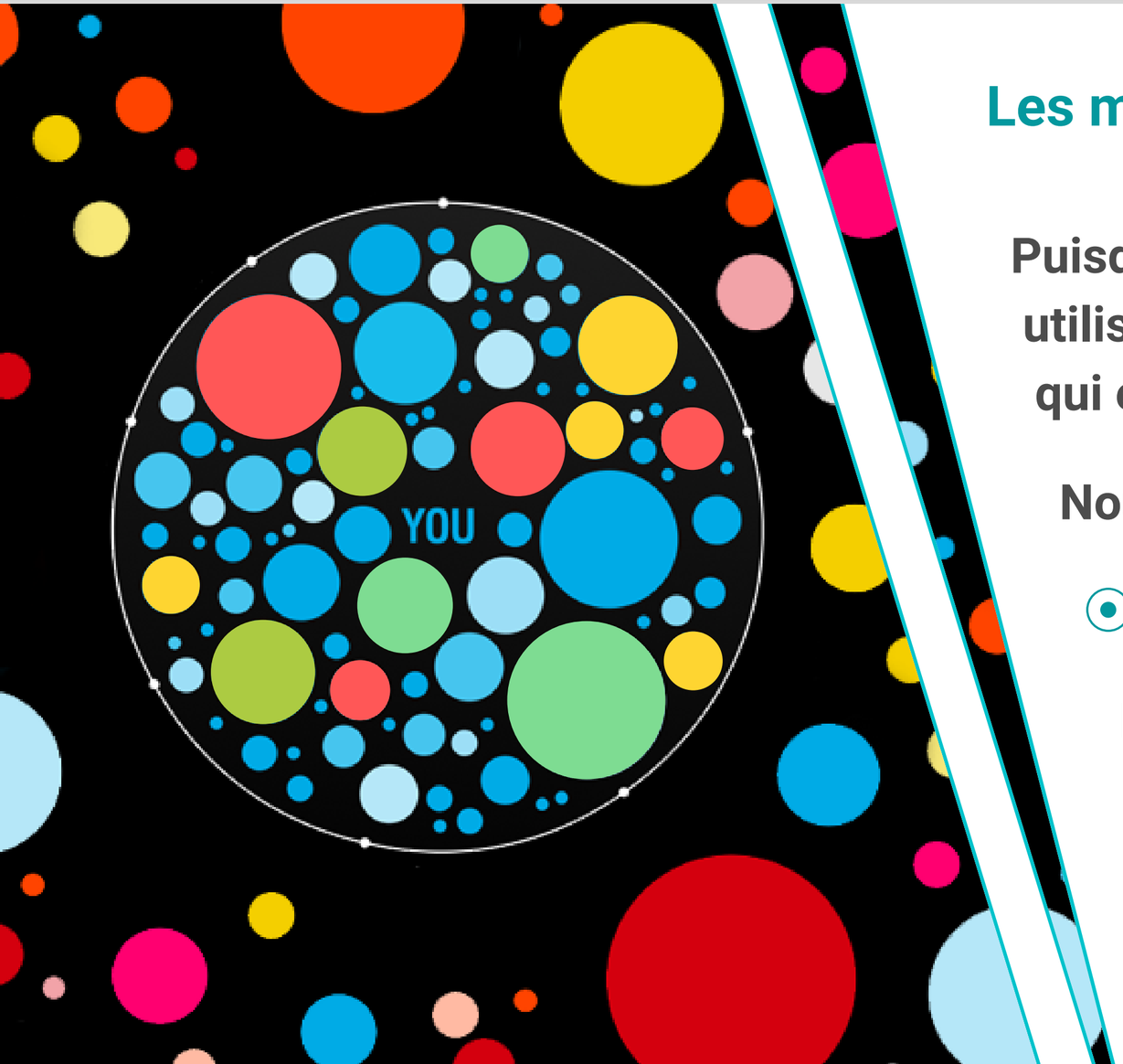


Les modèles de content-based filtering > résultats

	mean_cosine_similarity	map@k	training_time
Baseline model - Random	0.482212	0.0	0
Content Based Filtering [full history]	0.576889	0.000207	0
Content Based Filtering [1 last article]	0.412783	0.0006	0
Content Based Filtering [5 last articles]	0.518529	0.0	0

Recherche du modèle à déployer

My Content



Les modèles de collaborative filtering

Puisque nous n'avons qu'un **score implicite**, nous avons utilisé divers modèles proposés par la **librairie Implicit** qui est spécialisée dans ce type de Recommenders.

Nous avons donc testé un **memory based model** :

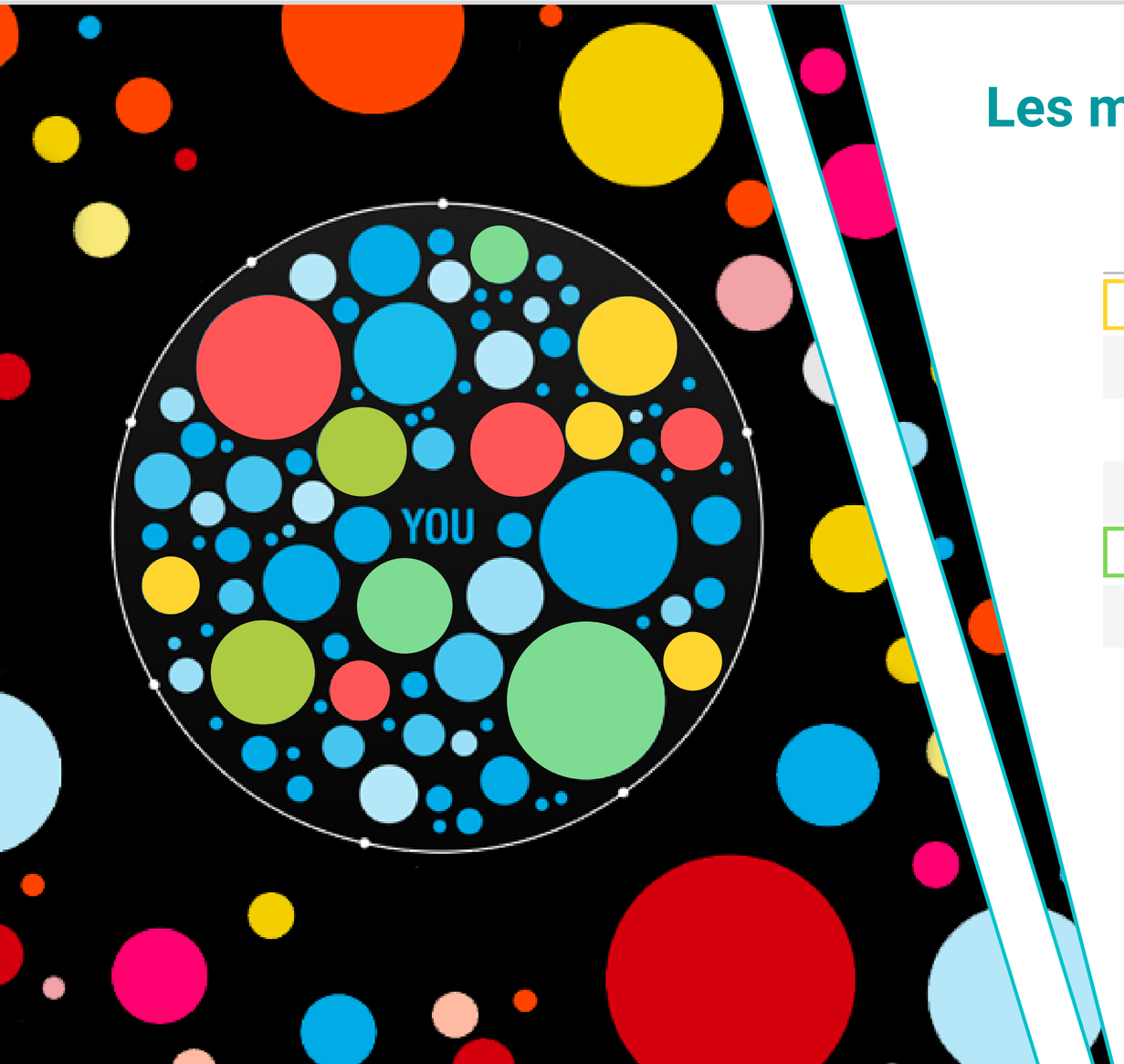
- **Item-Item Nearest Neighbour**

Et des **model-based / matrix factorization** models :

- **Alternating Least Square** avec ou sans **BM25**
- **Bayesian Personalized Ranking**
- **Logistic Matrix Factorization**

Recherche du modèle à déployer

My Content

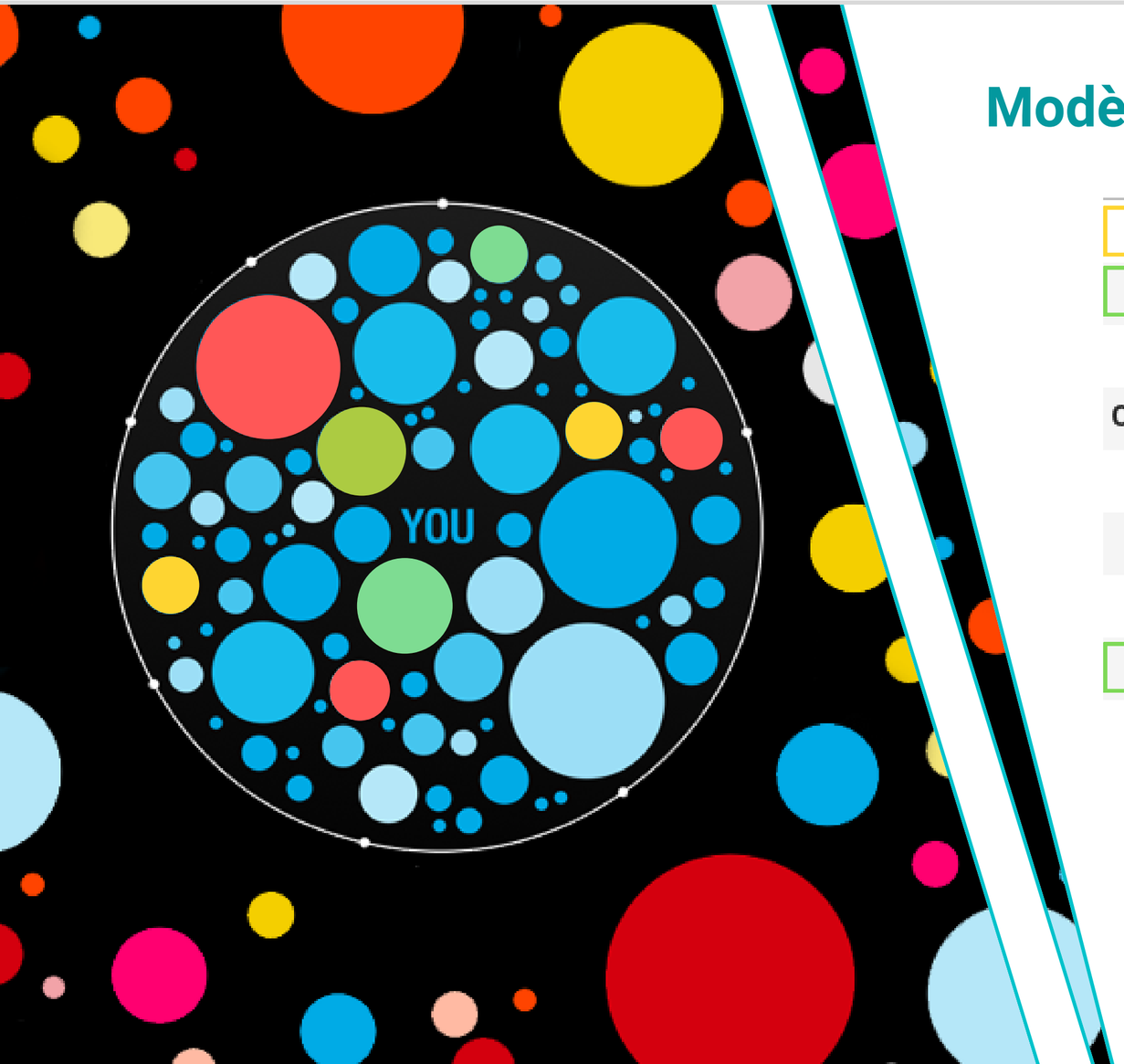


Les modèles de collaborative filtering > résultats

	mean_cosine_similarity	map@k	training_time
Baseline model - Random	0.482212	0.0	0
AlternatingLeastSquares_with_BM25	0.505538	0.0	85.557782
AlternatingLeastSquares	0.575309	0.0003	138.599869
LogisticMatrixFactorization	0.563861	0.0001	31.612296
BayesianPersonalizedRanking	0.589526	0.00034	9.405733
ItemItemRecommender	0.573461	0.001775	0.393621

Recherche du modèle à déployer

My Content



Modèle hybride

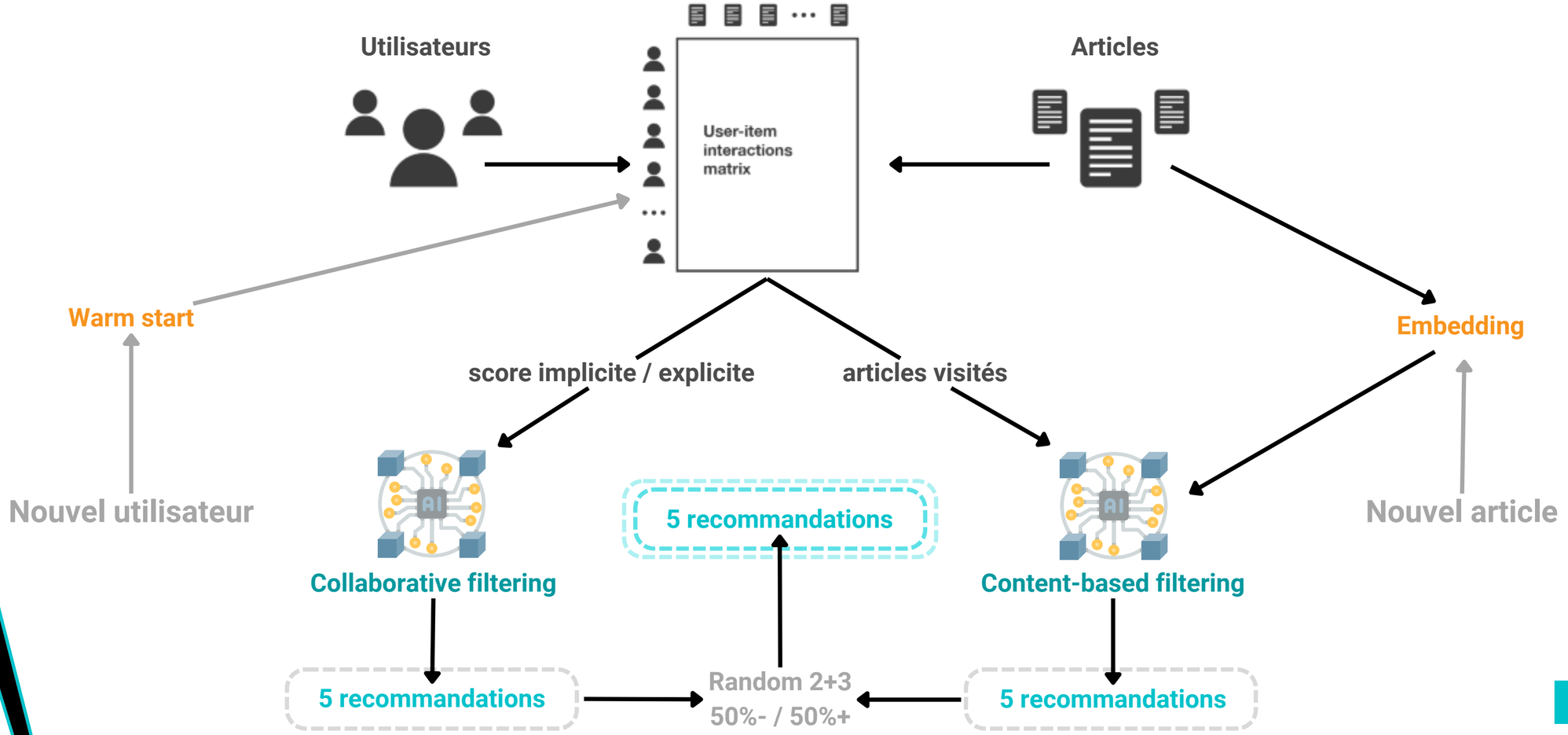
	mean_cosine_similarity	map@k	training_time
Baseline model - Random	0.482212	0.0	0
Content Based Filtering [full history]	0.576889	0.000207	0
Content Based Filtering [1 last article]	0.412783	0.0006	0
Content Based Filtering [5 last articles]	0.518529	0.0	0
AlternatingLeastSquares_with_BM25	0.505538	0.0	85.557782
AlternatingLeastSquares	0.575309	0.0003	138.599869
LogisticMatrixFactorization	0.563861	0.0001	31.612296
BayesianPersonalizedRanking	0.589526	0.00034	9.405733
ItemItemRecommender	0.573461	0.001775	0.393621

Pour bénéficier des avantages des deux familles de modèles, il a été décidé de créer un modèle avec le meilleur de chaque famille.

Recherche du modèle à déployer

My Content

Modèle hybride > schéma



Recherche du modèle à déployer

My Content

Ajouter un nouvel utilisateur ou article

Les **collaborative filtering models** sont confrontés à un **problème de cold start lorsqu'il y a très peu de données** disponibles sur un utilisateur ou un article.

L'une des solutions possible est l'approche **warm-start**

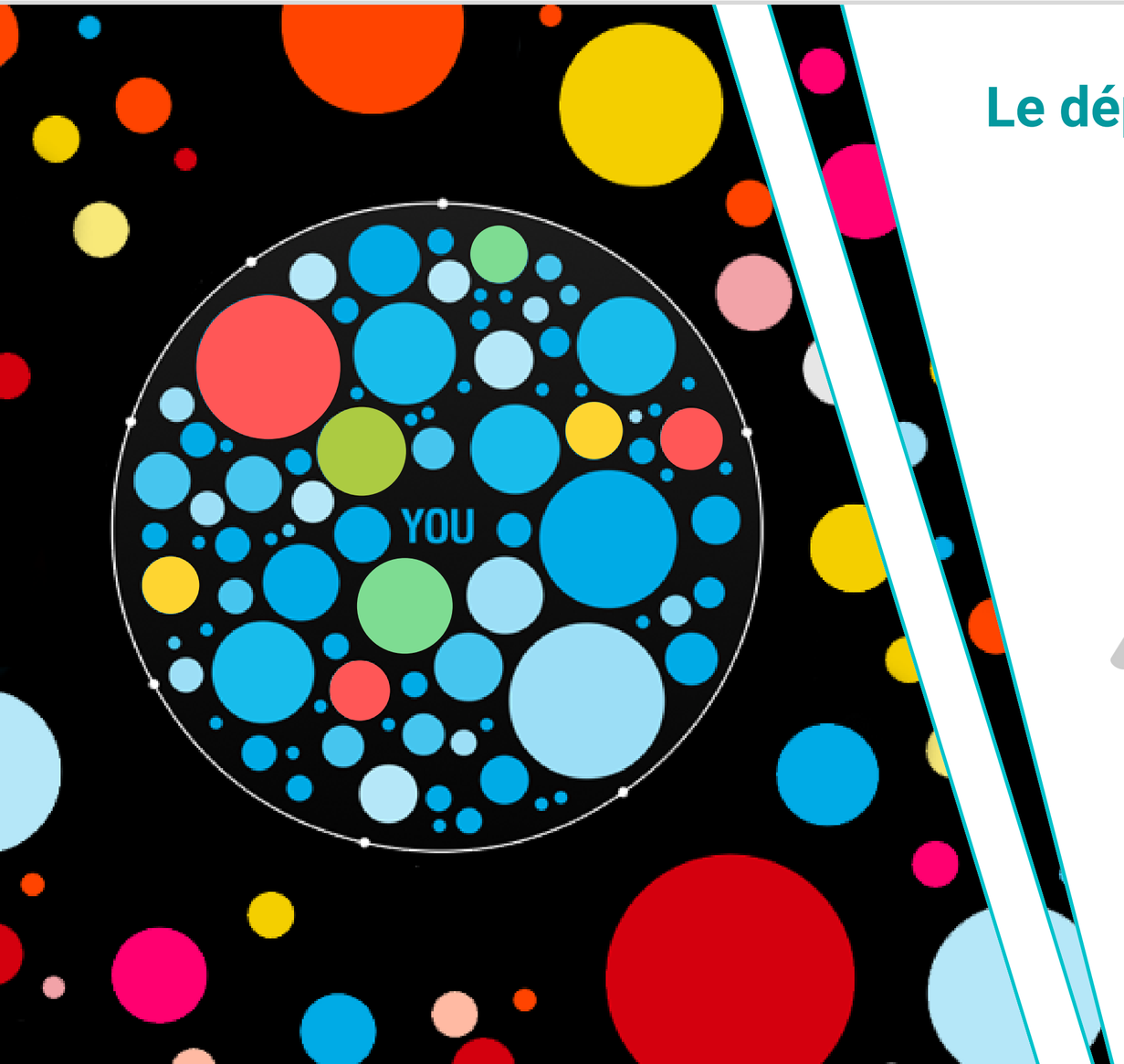
(globale ou par groupes représentatifs)

	Alice	Louis	Zoé	Bob
Article 1	3	5	1	3
Article 2	5	0	?	2.5
Article 3	4	2.5	1	2.5

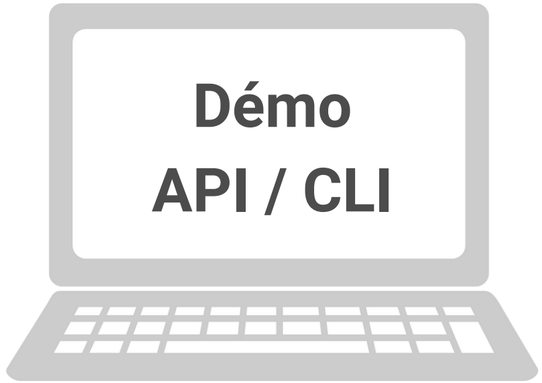
Les **content-based models** n'ont **pas ce problème de cold-start**, mais les nouveaux articles doivent être **préparés avec l'embedding** déjà utilisé.

Recherche du modèle à déployer

My Content



Le déploiement serverless



GitHub DVC

Azure Functions

Streamlit

Axes d'amélioration

My Content

- **Évaluer les recommandation avec les utilisateurs**
  (A/B testing sur une KPI à définir)
- **Essayer d'autres de modèles**
  (DNN recommenders, qui acceptent plus de features)
- **Ajouter une approche warm-start au MVP**
  (pour initializer les nouveaux utilisateurs / articles)
- **Collecter notre propre matrice user-item**
- **Surveiller l'évolution des recommandations**
  (pour éviter les problèmes de data-drift / concept-drift)

Merci de m'avoir écouté, évalué et conseillé

My Content