MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE DE JIJEL - MOHAMMED SEDDIK BENYAHIA
FACULTE DES SCIENCES EXACTES ET INFORMATIQUE

MEMOIRE

Présenté en vue de l'obtention du diplôme de

Master en Informatique

Option : **Intelligence Artificielle**

Vers un nouveau cadre d'évaluation des systèmes d'apprentissage automatique : Application aux systèmes de recommandation

---

Par

Imane BOUCHELIF et **Zineb HAMMOUDI**

Soutenue publiquement, devant le jury composé de :

M.	Mokhtar Taffar	Maître de conférences	Président
M.	Hemza Ficel	Maître assistant	Encadrant
M.	Mouad Benkiniouar	Maître de conférences	Examinateur

- Promotion 2022 -

Remerciements

Nous remercions Dieu le tout puissant de nous avoir donné la santé et la volonté d'entamer et de terminer ce mémoire.

Tout d'abord, ce travail ne serait pas aussi riche et n'aurait pas pu avoir le jour sans l'aide de l'encadrement de ***Mr Hemza Ficel****, on le remercie pour la qualité de son encadrement exceptionnel, pour sa patience, sa rigueur et sa disponibilité durant notre préparation de ce mémoire.*

Merci aux membres du jury, qui ont bien voulu assister à la soutenance de ce mémoire.

Nous remercions aussi les professeurs de l'université de Jijel-Mohammed Seddik BenYahia qui nous ont fourni les outils nécessaires à la réussite de nos études universitaires.

Dédicace

Je tiens à dédier cet humble travail à :

À ma très chère mère ***Zahira***
Quoi que je fasse ou que je dise, je ne saurai point te remercier comme il se doit. Ton affection me couvre, ta bienveillance me guide et ta présence à mes côtés a toujours été ma source de force pour affronter les différents obstacles.

A mon très cher père ***Abdeallah***
Tu as toujours été à mes côtés pour me soutenir et m'encourager. Que ce travail traduit ma gratitude et mon affection.

A mes très chers soeurs ***Soraya, Ilhem et Selma***
A mes très chers frères ***Mohammed, Hamza et Yahia***

Puisse Dieu vous donne la santé, le bonheur,le courage et surtout La réussite.

Je remercie ma chère binôme ***Imane*** *pour sa sympathie et sa patience*
Et mes chères amies ***Loubna,Imene,Niama*** *qui ont toujours été là pour moi.*

Enfin, je remercie tous mes amis, collègues et tous les étudiants de la promotion
A tous ceux qui, par un mot, m'ont donné la force de continuer ...

Zineb

Dédicace

Je tiens à dédier cet humble travail à :

À ma très chère mère ***Hajira***
Quoi que je fasse ou que je dise, je ne saurai point te remercier Comme il se doit.Ton affection me couvre, ta bienveillance me quide et ta présence à mes côtés a toujours été ma source de Force pour affronter les différents obstacles.

A mon très cher père ***Bachir***
Tu as toujours été à mes côtés pour me soutenir et m'encourager. Que ce travail traduit ma gratitude et mon affection.

A mes très chers soeurs ***Selma, Amina et Sarah***
A mes très chers frères ***Mahmoud, Atmane et Hamza***

A mes très chers petites nièces **Maissa, Ritel et Rouaa**

Puisse Dieu vous donne la santé, le bonheur,le courage et surtout La réussite.

je remercie ma chère binôme ***Zineb*** *pour sa sympathie et sa patience*
Et mes chères amies ***Hana, Ikram, Marwa, Amani*** *qui ont toujours été là pour moi.*

Enfin, je remercie tous mes amis, collègues et tous les étudiants de la promotion
A tous ceux qui, par un mot, m'ont donné la force de continuer ...

Imen

Résumé

Dans le contexte du Web actuel, les systèmes de recommandation sont devenus un moyen incontournable pour améliorer l'engagement des clients et augmenter les revenus des entreprises. Selon AQOA et McKinsey & Co, 35% des ventes d'Amazon et 95% des titres recommandés par Netflix proviens de leurs systèmes de recommandation. L'émergence rapide de ces systèmes de recommandation a créé le besoin de développer des cadres appropriés pour évaluer leurs performances et pertinences. Le contexte statique qui a entouré l'avènement de ces systèmes a favorisé la large utilisation d'un cadre hors-ligne qui omettent les propriétés réelles des données en flux sur les plateformes en production telle que Amazon et Netflix. Ceci mène à une évaluation artificielle et illogique qui rend les résultats d'évaluation indéterministes. Ainsi, les algorithmes les plus pertinents dans ce cadre ne sont pas toujours les meilleurs algorithmes à adopter dans les plateformes réelles. Dans ce mémoire, nous proposons un cadre d'évaluation des systèmes de recommandation qui conservent l'aspect réaliste des données en flux. Notre proposition est basée sur des techniques d'échantillonnage par fenêtre temporelle pour modéliser l'aspect dynamique des données. Par la suite, notre cadre suit une démarche de distribution de messages en flux pour simuler le plus fidèlement possible le scénario de recommandation en ligne. Nos expérimentations menées sur un jeu de données réel ont montré la différence entre les résultats d'évaluation sur un cadre statique hors ligne et en suivant les directives de notre proposition dynamique.

Mots clés : Système de recommandation, cadres d'évaluation, évaluation hors ligne, évaluation en ligne, fenêtre temporelle.

Abstract

In today's web environment, recommendation systems have become a key way to improve customer engagement and increase company revenues. According to AQOA and McKinsey & Co, 35% of Amazon's sales and 95% of Netflix's recommended titles come from their recommendation systems. The rapid emergence of these recommender systems has created the need to develop appropriate frameworks to evaluate their performance and relevance. The static context surrounding the advent of these systems has favored the widespread use of an offline framework that omits the actual properties of streaming data on production platforms such as Amazon and Netflix. This leads to an artificial and illogical evaluation that makes the evaluation results indeterminate. Thus, the most relevant algorithms in this framework are not always the best algorithms to adopt in real platforms. In this dissertation, we propose a framework for evaluating recommender systems that retains the realistic aspect of streaming data. Our proposal is based on time window sampling techniques to model the dynamic aspect of the data. Subsequently, our framework follows a stream message distribution approach to simulate the online recommendation scenario as closely as possible. Our experiments conducted on a real dataset showed the difference between the evaluation results on a static offline framework and following the guidelines of our dynamic proposal.

Keywords : Recommendation system, evaluation frameworks, offline evaluation, online evaluation, sliding time window.

Table des matières

Liste des figures

Liste des tableaux

Introduction générale

Contexte et enjeux

Chaque jour, nous sommes amenés à prendre des dizaines de décisions, des plus simples aux plus complexes : Quelle actualité à consulter ? Quel livre à acheter ? Quel voyage à organiser ? ou quel film à voir ?

Nos décisions sont souvent basées sur les conseils de nos proches, nos expériences, les avis des internautes ou simplement prises à l'instinct. La capacité de prendre les « bons choix » qui satisferont nos besoins et nos préférences est un défi de taille qui n'est pas simple à relever. En effet, dans nos expériences quotidiennes, nous sommes souvent confortés à des situations de « surcharge de choix », où le processus de prise de décision devient un véritable cauchemar.

L'avènement d'internet, avec ses plateformes multiservices et ses moteurs de recherche, a rendu nos décisions moins difficiles. Nous n'avons plus besoin de nous casser la tête pour trouver ce que nous voulons. Par exemple, une simple recherche sur « Google », avec des termes spécifiques, permet d'obtenir des résultats précis et fiables. Or, formuler de telles requêtes explicites peut s'avérer peu pratique, car une simple recherche peut générer des milliers de résultats. Ce processus, même s'il génère des résultats qui s'avèrent très pertinents, il nous fait retourner à la case de départ : « situation de surcharge de choix qui tue le choix ».

Des systèmes proactifs, basés sur l'intelligence artificielle, ont été donc proposés pour débloquer la situation précédente. Par exemple, quand une personne achète un article sur la plateforme d'Amazon [1], elle lui propose des rubriques de types « Les clients qui ont acheté ce produit ont également acheté » contenant des recommandations adaptées au besoin courant du client. De même, la plateforme de streaming Netflix [2] nous propose des films qui s'adaptent parfaitement à nos goûts et nos préférences cinématographiques. Sans dévoiler un secret, ce sont les systèmes de recommandation qui leur permettent d'apprendre nos préférences et personnaliser les choix selon nos intérêts et préférences. Ces systèmes n'attendent pas donc la requête explicite de l'utilisateur, et tentent d'apprendre ses intérêts et besoins à partir de ses données pour lui proposer des recommandations pertinentes répondant à ses attentes.

Dans le contexte du Web actuel, les systèmes de recommandation sont devenus un moyen incontournable pour améliorer l'engagement des clients et augmenter les revenus des entreprises. Selon AQOA [3] et McKinsey & Co [4], 35% des ventes d'Amazon et 95% des titres recommandés par Netflix proviens de leurs systèmes de recommandation. L'émergence rapide de ces systèmes de recommandation a créé le besoin de développer des cadres appropriés pour évaluer leurs performances et pertinences. Dans ce contexte, la plupart des systèmes de recommandation proposés sont évalués dans un cadre hors-ligne sur des collections de données statiques. Cette phase d'évaluation hors-ligne est censée indiquer les performances en ligne, et constitue ainsi une étape importante pour la sélection d'algorithmes pouvant être déployés dans un environnement réel. Or, plusieurs travaux ont montré que les algorithmes les plus précis dans ce cadre hors-ligne et statique n'étaient pas toujours les meilleures solutions à adopter dans des plateformes en production telles que Amazon et Netflix.

Problématique et objectif

Le problème principal des méthodes d'évaluation courantes vient du fait que le protocole hors ligne met à la disposition des systèmes à évaluer tout l'ensemble d'apprentissages dès le démarrage du processus d'évaluation. Ainsi, pour prédire des recommandations à

1. https://www.amazon.com/
2. https://www.netflix.com/
3. https://www.bfmtv.com/tech/sur-netflix-95-des-contenus-mis-en-valeur-sont-selectionnes-par-un-algorithme_AN-202107010002.html
4. https://www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers

un instant spécifique, un système à évaluer disposera non seulement des interactions déroulées avant ce moment, mais aussi des interactions futures qu'il n'est pas censé avoir dans la plateforme d'origine (évènements non déjà arrivés). Ceci mène à une évaluation artificielle et illogique qui contredit l'objectif principal d'un système de recommandation, à savoir l'utilisation des données historiques pour prédire les données futures. De plus, ce cas rend difficile l'évaluation de la capacité d'un système candidat de proposer des recommandations pertinentes pour les nouveaux utilisateurs ainsi que pour les nouveaux items (le problème du démarrage à froid).

Le travail conçu et réalisé dans ce projet de fin d'études a pour but de combler les lacunes existantes en matière d'évaluation des systèmes de recommandation. En effet, au vu du contexte très dynamique dans les plateformes de recommandation opérationnelles dans le Web actuel, nous avons essayé de proposer les fondements pour un nouveau cadre d'évaluation plus réaliste, qui tient compte des propriétés des données sur ces plateformes.

Plan du manuscrit

Ce mémoire est organisé en 3 chapitres :

Le chapitre I présente un bref état de l'art sur les systèmes de recommandation. Le processus de recommandation, les approches de recommandation les plus utilisées ainsi que leur classification sont donc détaillés. De plus, ce chapitre identifie et discute les lacunes actuelles des cadres d'évaluation de ces systèmes.

Le chapitre II est consacré à la présentation de notre proposition. Tout d'abord, nous nous intéressons à la modélisation de la problématique d'évaluation d'un système de recommandation en nous appuyant sur notre discussion des travaux de l'état de l'art. Ensuite, nous détaillons le cadre d'évaluation que nous avons proposé sous ses différents angles, à savoir la préparation, le partitionnement et la distribution des données aux systèmes de recommandation candidats, et enfin l'évaluation des recommandations générées par ces systèmes.

Le chapitre III détaille les expérimentations menées pour juger l'intérêt de notre proposition. Les résultats de ces expérimentations sont discutés dans ce chapitre afin de mieux valider les hypothèses et les idées de cette proposition.

Chapitre I

Etat de l'art sur les systèmes de recommandation

Sommaire

Introduction

Les systèmes de recommandation sont partout de nos jours. En fait, certaines des plus grandes marques avec lesquelles nous interagissons chaque jour (p. ex. Netflix, Amazon et Google, etc.) sont construites autour de ces systèmes. En effet, 35% des achats sur Amazon proviennent des algorithmes de recommandation de produits déployés dans cette plateforme [1]. Alors, qu'est-ce qu'un système de recommandation, comment ça marche et comment évaluer sa pertinence ? Ce chapitre a pour but de répondre à ces questions et de discuter les lacunes actuelles des cadres d'évaluation de ces systèmes.

I.1 Définition et processus général de recommandation

Le système de recommandation est un programme informatique qui utilise des algorithmes d'apprentissage automatique pour recommander les items les plus pertinents à un utilisateur ou à un client particulier. Ces items peuvent être de différentes natures : un produit à acheter, un morceau de musique, un film à regarder, un livre à lire, une page Web à consulter, ou bien autre chose. Afin de pouvoir fournir des recommandations personnalisées aux utilisateurs, le système de recommandation doit connaître les préférences de chaque utilisateur.Il tente alors d'acquérir les informations nécessaires pour construire des profils pour chaque utilisateur. La figure I.1 représente le processus général de recommandation.

1. https://www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers

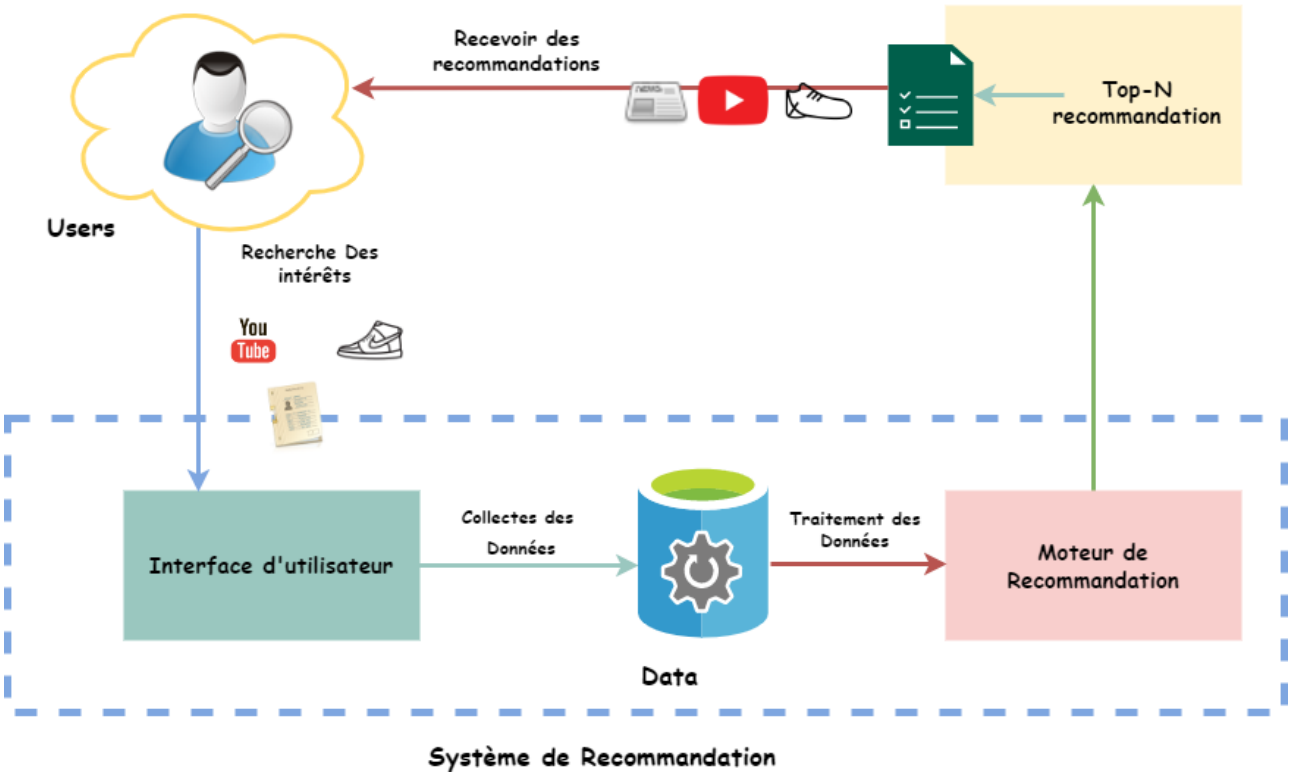

Figure I.1 – Processus générale de recommandation

L'utilisateur interagit avec les systèmes de recommandation en cliquant sur des publicités, consultant des articles d'actualités, procurant des produits, postant des commentaires, etc. Cette série d'actions détermine implicitement ou explicitement ce qui a retenu l'attention de l'utilisateur [1]. En effet, dans le contexte des systèmes de recommandation, deux types de données se distinguent, à savoir les données explicites et implicites.

I.1.1 Données explicites

Les données explicites sont fournies volontairement par les utilisateurs. Ces données sont collectées via des mécanismes fournis par des systèmes de recommandation pour permettre aux utilisateurs de répondre et d'exprimer leurs préférences [2]. La Figure I.2 montre un exemple sur le site Amazon ou les préférences des utilisateurs sont collectées explicitement via un système de notations.

Figure I.2 – Exemples de récolte des données sur Amazon.fr

I.1.2 Données implicites

Les données implicites sont collectées en traçant les actions spontanées de l'utilisateur pendant la navigation (p. ex. historique de vues, ressources ignorées) [1]. La récolte de ces données se fait en analysant les logs de l'utilisateur que l'on trouve sur le serveur, ses sites favoris, ses historiques de navigation et de recherche, etc [2]. Par exemple, quand un utilisateur achète un produit, cela crée un historique d'achat pour cet utilisateur. L'avantage de ce type de données est qu'elles sont très abondantes et ne nécessitent pas l'implication directe de l'utilisateur. Par contre, leur inconvénient majeur par rapport aux données explicites est le caractère incertain de ces données, vu qu'elles ne sont pas indiquées directement par l'utilisateur [2].

I.2 Approches de recommandation

Les algorithmes de recommandation ont infiltré notre quotidien numérique sur le Web. En effet, les recommandations sont omniprésentes dans tous les grands domaines du Web : e-commerce, actualité en ligne, services de streaming vidéo, réseaux sociaux, etc. Plusieurs approches se distinguent derrière ces recommandations.

I.2.1 Approches basées sur le filtrage à base de contenu

L'approche basée sur le contenu, également connue sous le nom de filtrage à base de contenu, consiste à recommander à l'utilisateur des items similaires du point de vue de leur contenu. Plus précisément, ce type de filtrage essaye de construire un profil pour

chaque utilisateur en examinant les centres d'intérêt de ce dernier vis-à-vis le contenu des items. Les approches adoptant ce type de filtrage nécessitent donc de disposer des descripteurs des items à recommander (p. ex. genre, type de produit, couleur, longueur du mot) et des connaissances sur les préférences de l'utilisateur sur ces descripteurs [2]. La figure I.3 schématise le processus général de ce type de filtrage.

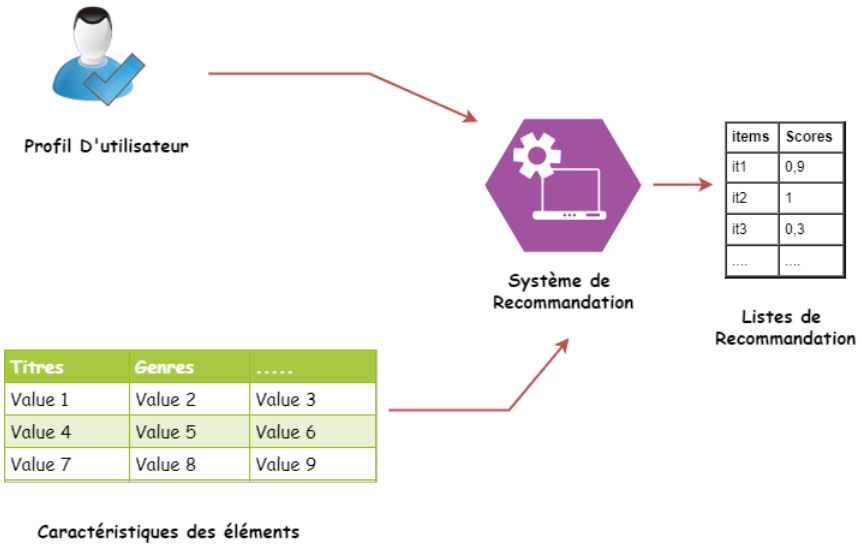

Figure I.3 – Processus de filtrage à base de contenu

I.2.2 Approches basées sur le filtrage collaboratif

Le filtrage collaboratif est considéré comme la technique la plus populaire dans le cadre des systèmes des recommandations. L'idée générale de ce type de filtrage est de recommander à l'utilisateur actif des items appréciés par les utilisateurs ayant un goût similaire à cet utilisateur. Les approches adoptant ce type de filtrage se basent donc sur des matrices de comportement qui permettent de représenter le voisinage et les préférences de chaque utilisateur. La matrice de comportement est de taille $\boldsymbol{n*m}$ ou $\boldsymbol{n}$ est le nombre total des utilisateurs et $\boldsymbol{m}$ est le nombre total des items. Les éléments de la matrice V_{ij} représente la note attribuée par l'utilisateur U_i à l'item A_j. Proposer des recommandations de ce contexte consiste à compléter les valeurs manquantes (V_{ij}=**nul**) de la matrice de comportement grâce à plusieurs méthodes de voisinage (p. ex. similarité cosinus, coefficient de Pearson, etc.) [3–5]. La figure I.4 schématise le processus général de ce type de filtrage.

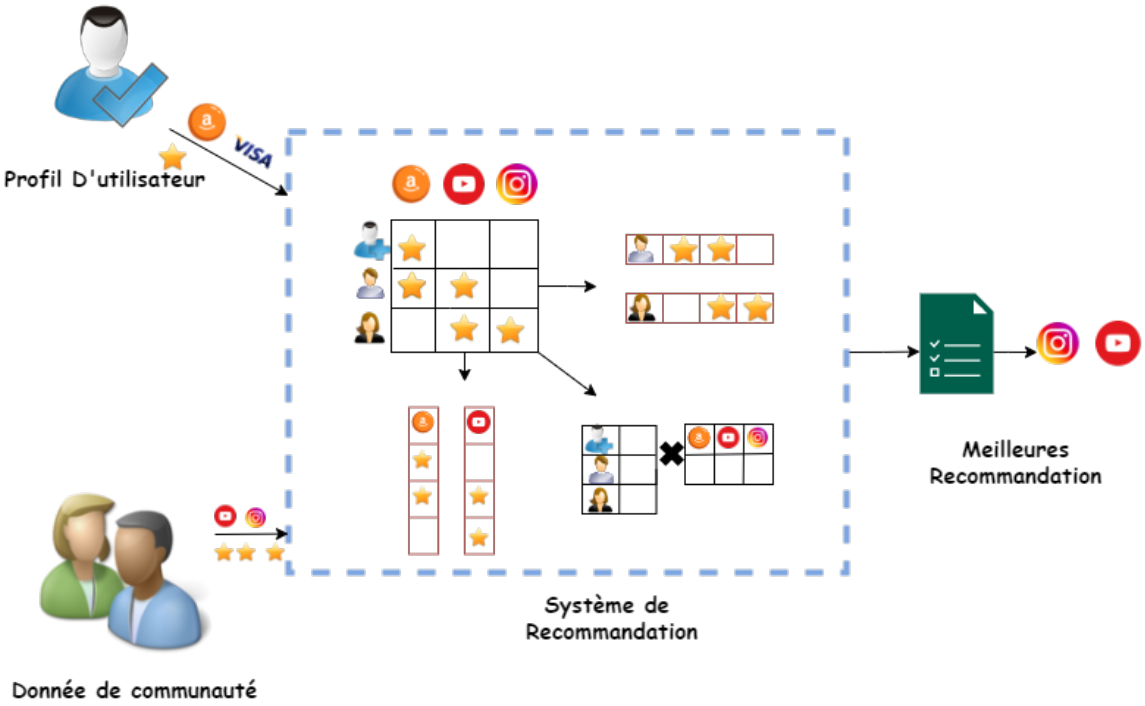

Figure I.4 – Processus de filtrage collaboratif

I.2.3 Approches basées sur le filtrage démographique

Les approches à base de filtrage démographique se basent sur les caractéristiques démographiques des utilisateurs (p. ex. sexe, âge, position géographique, etc.) et sur la logique du filtrage collaboratif pour proposer des recommandations. En effet, ces approches essayent de regrouper les utilisateurs dans des classes qui partagent les mêmes caractéristiques démographiques. Par exemple, une classe sera créée pour les hommes et une classe pour les femmes (cf. figure I.5). Par la suite, elles appliquent la logique du filtrage collaboratif pour identifier les items à recommander (cf. la figure I.5). Ainsi, les utilisateurs partageant les mêmes profils démographiques recevront les mêmes recommandations [4, 6].

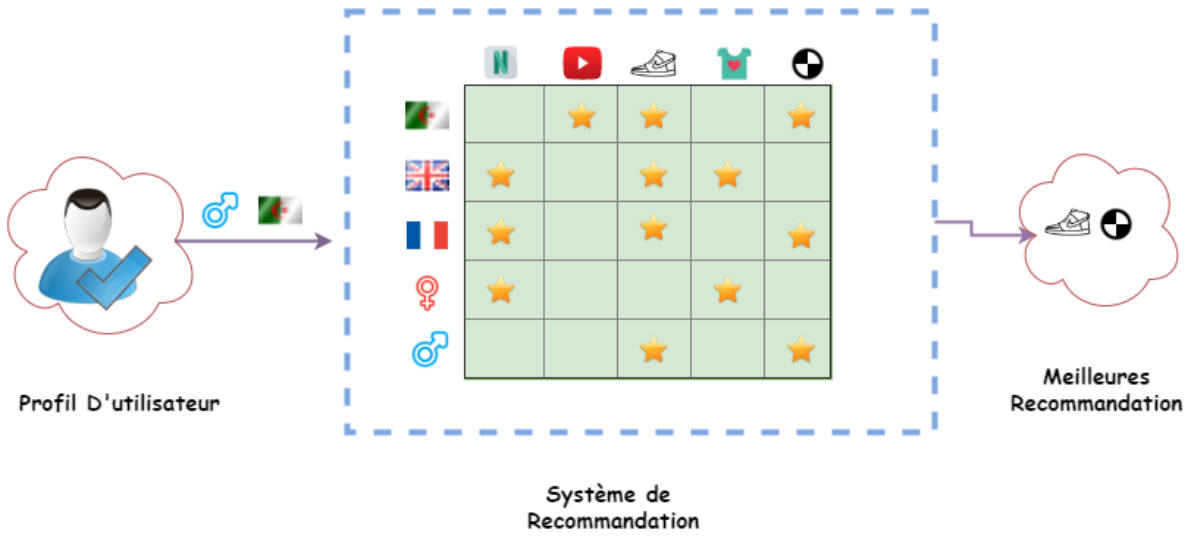

FIGURE I.5 – Processus de filtrage démographique

I.2.4 Approches basées sur le filtrage contextuel

Les approches à base de filtrage contextuel sont axées sur la notion du contexte. Cette notion représente les informations qui peuvent être utilisées pour caractériser une situation d'une entité donnée [7]. Dans le cadre des systèmes de recommandation, une entité peut faire référence à un utilisateur ou un item. Un exemple des informations contextuelles est la position géographique (p. ex. Jijel, Alger, Constantine, etc.) ou le support d'accès de l'utilisateur ciblé (p. ex. PC. Smartphone, Tablet, etc.). Il est à noter qu'il est possible de déduire ces données d'une façon implicite (p. ex. suivre la position géographique de l'utilisateur) ou bien explicite (p. ex. demander à l'utilisateur de spécifier les variables contextuelles) [4, 8].

I.2.5 Approches basées sur le filtrage à base de connaissances

Les approches à bases de connaissances s'appuient sur un processus d'inférence logique et un ensemble de connaissances liées à un domaine spécifique pour identifier les items à recommander. Le mode opératoire dans ce genre d'approches est conversationnel : l'utilisateur est amené à définir à chaque fois un certain nombre de critères pouvant mieux répondre à ses attentes(cf. figure I.6). En effet, ces approches n'essayent pas de construire des profils pour les utilisateurs. En revanche, elles demandent à l'utilisateur ciblé de déterminer ses besoins explicitement appris à l'aide des techniques de fouille de données ou

introduits par des experts qui filtrent les items selon les critères définis par l'utilisateur. Le processus de recommandation consiste donc à chercher les items remplissant au mieux les conditions définies par l'utilisateur [2].

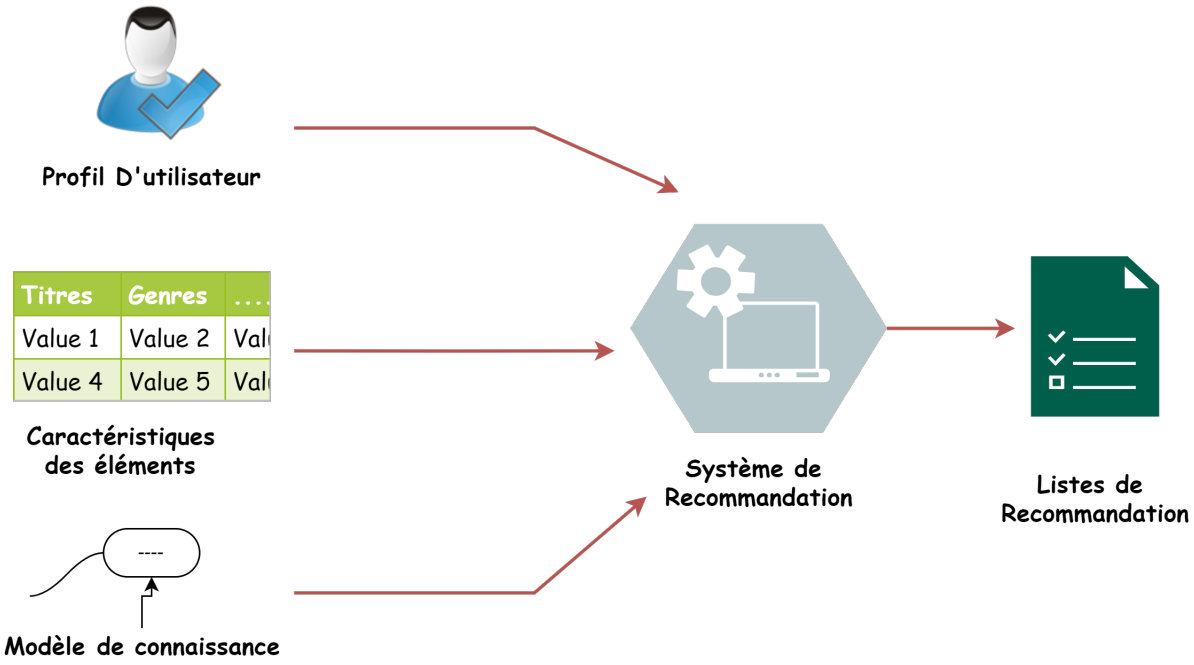

Figure I.6 – Processus de filtrage à base de connaissances

I.2.6 Approches hybrides

L'objectif des approches hybrides est de surmonter les inconvénients des approches de recommandation en s'appuyant sur la combinaison de plusieurs approches à la fois (cf. la figure I.7).

Ainsi, ces combinaisons permettent de proposer des recommandations plus pertinentes et personnalisées selon les intérêts des utilisateurs ciblés [2, 4]. Robin Burke [9] a cité plusieurs manières d'hybrider les approches de recommandation. Ces techniques sont décrites dans le tableau I.1.

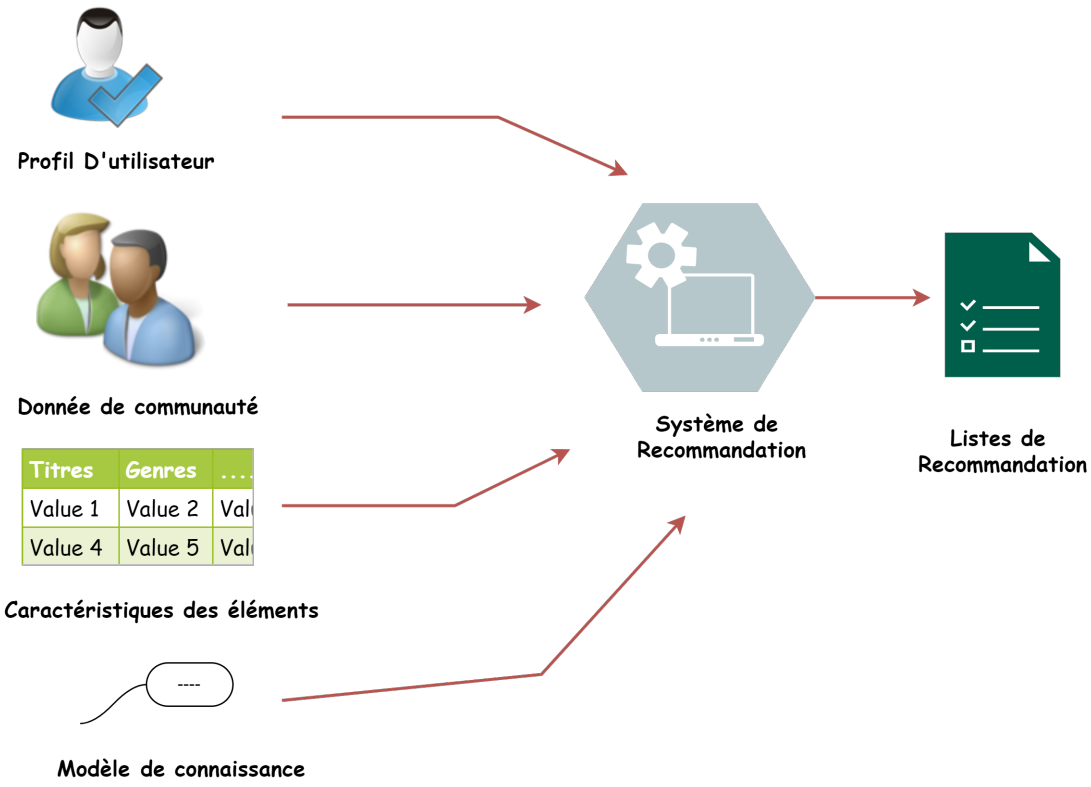

Figure I.7 – Processus d'approche hybride

Méthode d'hybridation	Description
Pondérée	Le score ou la prédiction obtenus par chacune des deux techniques est combiné en un seul résultat.
Commutation	Le système permute entre les différentes techniques de recommandation selon le résultat de la recommandation.
Mixte	Les listes des recommandations issues des différentes techniques sont fusionnées en une seule liste.
Combinaison	Différentes techniques de recommandation sont combinées en un unique algorithme de recommandation.
En Cascade	Une technique de recommandation est utilisée pour produire un premier classement des items candidats et une deuxième technique affine ensuite la liste des recommandations.
Augmentation	Le résultat d'une technique de recommandation est utilisé comme données en entrée pour l'autre technique.
Métaniveau	Cette méthode est analogue à la méthode par augmentation de propriétés, mais c'est le modèle appris qui est utilisé en entrée de la deuxième technique et non la liste résultat des recommandations.

Tableau I.1 – les différents types d'hybridation

I.3 Classification des approches de recommandation

Les systèmes de recommandation adoptent diverses hypothèses et approches pour identifier les items à recommander. Ainsi, ces systèmes peuvent être classés selon divers critères.

I.3.1 Approches personnalisées ou non personnalisées

Les systèmes de recommandation personnalisés adoptent l'une des approches précédentes pour optimiser les recommandations aux préférences de chaque utilisateur [10]. Contrairement à ces systèmes, les systèmes de recommandation non personnalisés ne tiennent pas compte des intérêts personnels des utilisateurs. Autrement dit, les recommandations générées par ces systèmes sont les mêmes pour tous les utilisateurs [11]. Par exemple, si nous visitons le site de commerce amazon.com comme un utilisateur anonyme, le site nous affichera comme recommandations les articles les plus populaires sur ce site.

I.3.2 Approches à base de modèle ou à base de mémoire

Les approches de recommandation les plus utilisées dans les systèmes de recommandation sont divisées en deux types : les approches à base de modèle ou bien les approches à base de mémoire. Les approches à base de modèle construisent des modèles résumant les intérêts des utilisateurs. Par la suite, elles interrogent ces modèles en ligne pour identifier les items à recommander sans avoir besoin d'accéder à la l'ensemble des comportements des utilisateurs. Contrairement à ces approches, les approches à base de mémoire se basent sur des méthodes heuristiques qui utilisent toutes les données disponibles pour générer les recommandations. Ces méthodes peuvent donc s'adapter rapidement aux changements de données [4].

I.3.3 Approches par lots ou par flux (environnement dynamique/statique)

Les approches de recommandation par lots traitent des blocs de données statistiques déjà collectées pour construire des modèles d'apprentissage. Par la suite, elles appliquent ces modèles sur les nouvelles données. Ainsi, l'actualisation des modèles d'apprentissage se fait périodiquement pour intégrer les nouvelles observations. Contrairement à ces ap-

proches, les approches par flux essayent de s'adapter au problème de flux de données dynamiques diffusées en croissance continue. Ces approches adoptent donc des solutions dynamiques qui s'adaptent rapidement aux changements de données [4].

I.3.4 Approche par domaine ou multidomaines

Les approches de recommandation par domaine sont optimisées pour fonctionner sur un domaine d'application spécifique. Par exemple, le système de recommandation PVR (Personalized Video Ranker) [12] est optimisé uniquement pour recommander des films sur la plateforme de Netflix. À l'opposé de ces approches, les approches de recommandation multidomaines peuvent proposer des recommandations destinées à divers domaines application [13].

I.4 Evaluation des systèmes de recommandation

Les travaux existants suivent généralement deux types de protocoles d'évaluation pour mesurer la pertinence des systèmes de recommandation. Dans ce qui suit, nous allons présenter les principes de ces protocoles ainsi que leurs lacunes actuelles.

I.4.1 Cadre d'évaluation hors ligne

L'évaluation hors ligne est le moyen le plus simple et le moins coûteux pour évaluer la pertinence des systèmes de recommandation. Elle opère sur des données statiques, pré-collectées auprès des utilisateurs. Ces données sont divisées en deux parties : un ensemble d'apprentissages et un ensemble de tests [3]. Le premier ensemble est utilisé pour entraîner les modèles prédictifs des systèmes à évaluer alors que le second ensemble est exploité pour valider les prédictions de ces systèmes [4]. Dans cette étape, plusieurs métriques de la qualité des prédictions peuvent être appliquées (p. ex., précision, rappel, erreurs, etc.). La figure I.8 représente le protocole d'évaluation hors ligne.

Le partitionnement des données dans un cadre d'évaluation hors ligne se base sur diverses stratégies d'échantillonnage. La section suivante résume le principe des stratégies d'échantillonnage les plus couramment utilisées dans le cadre des systèmes de recommandation [14, 15]

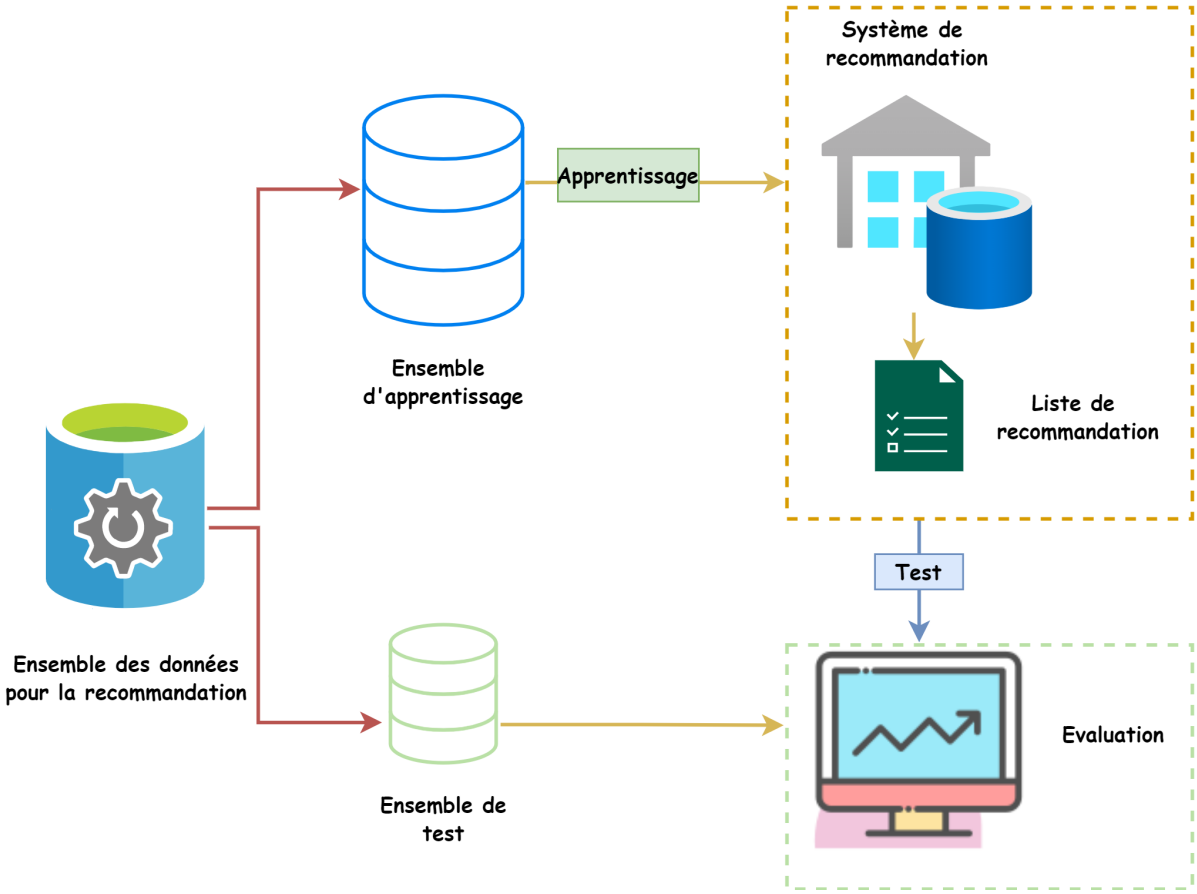

FIGURE I.8 – Le protocole d'évaluation hors ligne

— **Échantillonnage aléatoire par proportion :** est une technique d'échantillonnage qui consiste à sélectionner au hasard un pourcentage d'interactions des utilisateurs sur les items en tant qu'ensemble de tests. Toutes les autres interactions restantes seront considérées comme un ensemble d'apprentissages.

— **Échantillonnage aléatoire par utilisateur :** est une technique d'échantillonnage moins courante qui divise l'ensemble de données par utilisateur plutôt que par interactions. Elle consiste à sélectionner au hasard un pourcentage d'utilisateurs et prenez toutes leurs interactions comme ensemble de tests. Les instances restantes seront considérées comme un ensemble d'apprentissages.

— **Échantillonnage « leave one out » :** est une technique d'échantillonnage qui considère la dernière interaction de chaque utilisateur comme ensemble de tests. Toutes ses autres interactions seront considérées comme un ensemble d'apprentissages.

— **Échantillonnage temporel :** cette stratégie d'échantillonnage considère toutes les interactions avant un instant **t comme un ensemble d'apprentissages. Toutes les interactions après ce moment seront considérées comme un ensemble de tests.**

I.4.2 Cadre d'évaluation en ligne

La figure I.9 schématise le principe du protocole d'évaluation en ligne. En effet, l'objectif du cadre d'évaluation en ligne est d'évaluer les systèmes de recommandation sur des utilisateurs réels dans des plateformes en production. Ce cadre se base généralement sur des techniques d'évaluation de type « test A/B » pour comparer les performances de plusieurs systèmes à la fois. Le principe de ces techniques consiste à exposer, de manière aléatoire, en temps réel, deux variantes (ou plus) d'un système à différents utilisateurs. Par la suite, les performances des deux systèmes sont comparées pour optimiser plusieurs objectifs (clics, vues, achats, transactions terminées, retour sur investissement, etc.). Nous pouvons donc résumer l'idée principale du cadre d'évaluation en ligne comme suit :

— Diviser les utilisateurs en deux groupes A et B.

— Utiliser un algorithme de recommandation différent pour chaque groupe d'utilisateur.

— Maintenir les conditions d'évaluation des systèmes à évaluer aussi similaires que possible (p. ex. le contexte, les catégories des utilisateurs, le type des items, etc.).

— Comparer les performances des systèmes candidats à la fin de la procédure.

I.4.3 Défis actuels de l'évaluation des systèmes de recommandation

L'évaluation des systèmes de recommandation sur des environnements en production semble être la solution la plus réaliste pour mesurer la pertinence de ces systèmes. Cependant, l'accès limité à de telles plateformes en production en ligne a fait que les universitaires se limitent à des évaluations hors ligne sur des ensembles de données statiques [15]. Ceci nous a menés à poser plusieurs questions : « est-ce que les standards actuels de l'évaluation des systèmes de recommandation répondent réellement aux exigences de domaine ? », « est-ce que les résultats du protocole hors ligne indiquent les performances réelles d'un

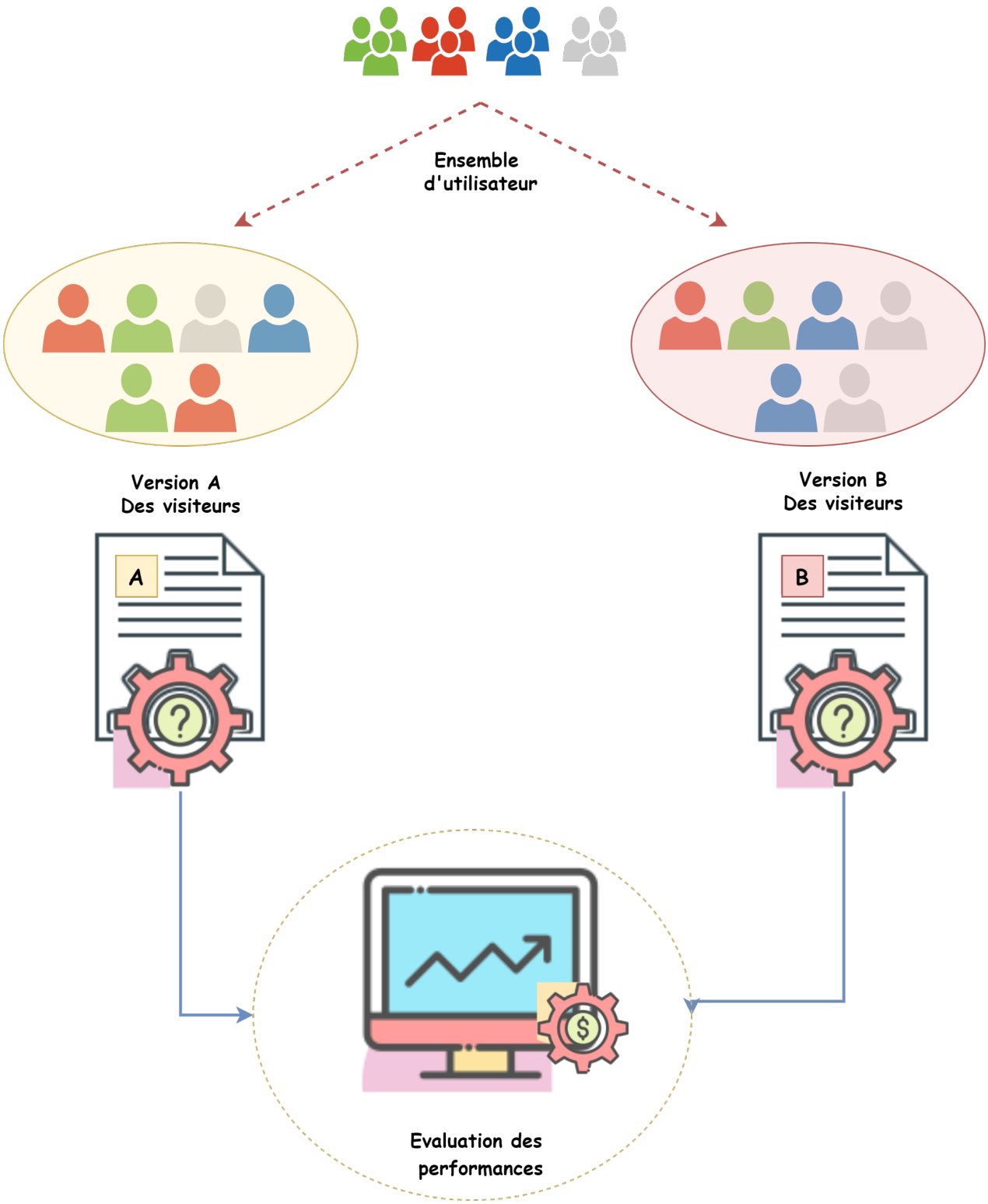

FIGURE I.9 – Le protocole d'évaluation en ligne

système de recommandation ? », « Le protocole hors-ligne est-il adapté pour l'évaluation des systèmes de recommandation déployés sur des environnements dynamiques ? » , « comment anticiper les performances réelles d'un système de recommandation ? ».

Pour répondre à ces questions, nous devons analyser le problème depuis sa source. Ainsi, plusieurs points sont à discuter.

I.4.3.1 Des environnements statiques vers des environnements très dynamiques

Le contexte statique qui a entouré l'avènement des systèmes de recommandation a favorisé l'utilisation du protocole hors ligne pour évaluer la pertinence de ces systèmes. En effet, ce contexte suppose que les données demeurent inchangées pendant une période relativement longue. Par exemple, le taux d'ajout de produits dans une plateforme de commerce électronique est relativement stable. Or, si nous examinons les données alimentant les plateformes de recommandation opérationnelles dans le Web actuel, nous pouvons apercevoir que ces données sont considérées comme un flux continu d'observations. Ce flux est soumis à un taux d'obsolescence et d'ajout très élevé et imprédictible. Par exemple, dans un portail d'actualité, une nouvelle de tendance actuelle peut perdre de l'intérêt dans les quelques heures qui suivent sa création. Les standards du cadre d'évaluation hors ligne omettent ces propriétés réelles des données, ce qui rend les résultats d'évaluation de ce protocole indéterministes,[16]. En effet, les résultats d'évaluation sous un cadre hors ligne et sur des données statiques peuvent nous inciter à choisir des systèmes de recommandation qui ne sont pas appropriés à la situation actuelle des plateformes en ligne.

I.4.3.2 Des méthodes d'échantillonnage qui omettent les propriétés réelles des données

Les travaux existants évaluent les systèmes de recommandation en se basant sur des techniques de partitionnement qui entraînent la perte de plusieurs propriétés importantes des données. En effet, la plupart des stratégies d'échantillonnage ne respectent pas la séquence et l'ordre chronologique des données. Par exemple dans un échantillonnage aléatoire par proportion et par utilisateur, le partitionnement des données se fait aléatoirement et ne respecte ni la chronologie locale ni globale des données [14]. De même, l'échantillonnage « Leave one out », malgré sa popularité, il ne peut pas refléter l'efficacité globale du système à évaluer vu que seule la dernière transaction par utilisateur est utilisée pour le test [14]. Enfin, le souci de l'échantillonnage temporel réside dans le choix de la durée de

la période de l'ensemble de tests. Autrement dit, pour une courte période de test, nous ne nous aurons pas assez de données pour faire les tests et de même pour une longue période de test, il y a un risque de changement du contexte temporel des évènements observés [15].

I.4.3.3 Une distribution illogique et irréaliste des évènements

Le principal point faible des évaluations hors ligne est qu'elles ne permettent pas d'évaluer l'intérêt réel de l'utilisateur. En effet, le protocole hors ligne mit à la disposition des systèmes à évaluer tout l'ensemble d'apprentissages dès le démarrage du processus d'évaluation. Ainsi, pour prédire des recommandations à un instant t, un système à évaluer disposera non seulement des interactions déroulées avant ce moment, mais aussi des interactions futures qu'il n'est pas censé avoir dans la plateforme d'origine à l'instant t (évènements non déjà arrivés). Ceci mène à une évaluation artificielle et illogique qui contredit l'objectif principal d'un système de recommandation, à savoir l'utilisation des données historiques pour prédire les données futures. De plus, ce cas rend difficile l'évaluation de la capacité d'un système candidat de proposer des recommandations pertinentes pour les nouveaux utilisateurs ainsi que pour les nouveaux items (le problème du démarrage à froid).

Les circonstances précédentes rendent les expérimentations hors ligne non réaliste, car elles peuvent nous induire en erreur lors du choix des meilleures solutions à déployer en ligne. Plusieurs études dans la littérature supportent cette hypothèse et estiment que les algorithmes les plus pertinents sous un cadre hors ligne ne sont pas toujours les solutions appropriées à déployer dans des environnements dynamiques en production [4, 17, 18].

I.4.3.4 Des tests A/B moins réalistes et moins accessibles

L'évaluation en ligne opère sur de vrais utilisateurs dans des systèmes commerciaux en ligne. Ce type d'évaluation est souvent moins sensible aux problèmes d'échantillonnage de données, car l'évaluation se fait dans le cours naturel des choses [19]. Néanmoins, de tels tests ne peuvent être exécutés que par les entreprises, et uniquement selon des scénarios contrôlés par ces dernières [20]. De plus, les tests A/B étant basés sur la distribution aléatoire des demandes de recommandation sur les systèmes candidats cause un effet de hasard dans l'évaluation. En effet, un test de deux instances identiques d'un même système

n'aboutit pas aux mêmes résultats, car les deux variantes ne ciblent pas les mêmes groupes d'utilisateurs. De plus, ce type d'évaluation ne peut être adoptée de manière réaliste que si un grand nombre d'utilisateurs sont déjà connectés.

Notre analyse précédente se résume dans le tableau I.2

Type d'évaluation / Critère de comparaison	**Évaluation hors-ligne**	**Évaluation en-ligne**
Accessible	**Oui**	**Non**
Permit l'évaluation du problème de démarrage à froid	**Non**	**Oui**
Respecte la séquence et l'ordre chronologique des données	**Non**	**Oui**
Garde les propriétés réelles des données	**Non**	**Oui**
Évalue les systèmes candidats sur les mêmes utilisateurs et les mêmes requêtes	**Oui**	**Non**
Nécessite l'implication des utilisateurs réels	**Non**	**Oui**

Tableau I.2 – Synthèse comparative entre l'évaluation hors ligne et en ligne

Conclusion

Les systèmes de recommandation sont devenus indispensables dans de nombreuses industries et ont attiré l'attention ces dernières années. Ceux-ci sont conçus pour aider les utilisateurs à trouver des ressources pertinentes parmi une large sélection pour leurs besoins. Dans ce chapitre, nous avons brièvement les principes fondamentaux ainsi que les procédures adoptées par les systèmes de recommandation. Par la suite, nous avons détaillé les cadres d'évaluation de ces systèmes. L'étude des lacunes de ces cadres nous a menés à poser plusieurs questions sur l'adéquation des standards actuels aux exigences des environnements dynamiques en ligne. Une discussion a été donc engagée pour identifier et analyser les causes de ce problème depuis sa source afin de proposer de nouveaux protocoles d'évaluation plus réaliste.

Chapitre II

Cadre d'évaluation proposé

Sommaire

Introduction

Ce chapitre est consacré à la présentation de notre proposition. Tout d'abord, nous nous intéressons à la modélisation de la problématique d'évaluation d'un système de recommandation en nous appuyant sur notre discussion des travaux de l'état de l'art. Ensuite, nous détaillons le cadre d'évaluation que nous avons proposé pour combler les lacunes existantes en matière d'évaluation des systèmes de recommandation. Ainsi, la proposition est détaillée sous ses différents angles, à savoir la préparation, le partitionnement et la distribution des données aux systèmes de recommandation candidats, et enfin l'évaluation des recommandations générées par ces systèmes.

II.1 Modélisation de la problématique d'évaluation sous un nouvel angle

Les entreprises optimisent en permanence la qualité des prédictions de leurs plateformes de recommandation afin d'offrir une meilleure expérience pour leurs clients. Dans le domaine académique, les universitaires manquent d'accès à de tels systèmes opérationnels à grande échelle pour évaluer leurs propositions. Comme alternative, la communauté académique collecte et standardise des jeux de données statiques, et adapte un cadre hors-ligne pour évaluer leurs systèmes de recommandation.

Le contexte statique qui a entouré l'avènement des systèmes de recommandation justifie la large utilisation de ces jeux de données. En effet, ce contexte suppose que les données demeurent inchangées pendant une période relativement longue. Or, si nous examinons les données alimentant les plateformes de recommandation opérationnelles dans le Web actuel, nous pouvons apercevoir que ces données sont considérées comme un flux continu d'observations. Ce flux est soumis à un taux d'obsolescence et d'ajout très élevé et imprédictible. La plupart des travaux existants omettent ces propriétés réelles des données et évaluent toujours les systèmes de recommandation selon les standards du cadre d'évaluation hors ligne.

Selon notre analyse des travaux existants (cf. chapitre I), le partitionnement des données et la manière de distribution de ces dernières constituent la lacune majeure du cadre d'évaluation hors ligne. En effet, cette étape se base souvent sur des techniques aléatoires qui fournissent un modèle imprécis qui omet plusieurs propriétés importantes telles que l'ordre chronologique des événements. Ceci mène donc à une évaluation irréaliste d'un système de recommandation. De ce fait, notre proposition doit adopter des stratégies qui conservent l'aspect réaliste des données en flux, ce qui va mener à reconcevoir les bases de toutes les phases suivantes :

— La phase de préparation des données par flux.

— La phase de partitionnement des données.

— La phase de distribution des flux de données.

— La phase de la comparaison et d'évaluation.

II.2 Cadre d'évaluation proposé

Au vu du contexte très dynamique dans les plateformes de recommandation opérationnelles dans le Web actuel, nous avons essayé de proposer les fondements pour un nouveau cadre d'évaluation plus réaliste, qui tient compte des propriétés des données sur ces plateformes. La figure II.1 présente un schéma représentatif des différents processus de notre proposition.

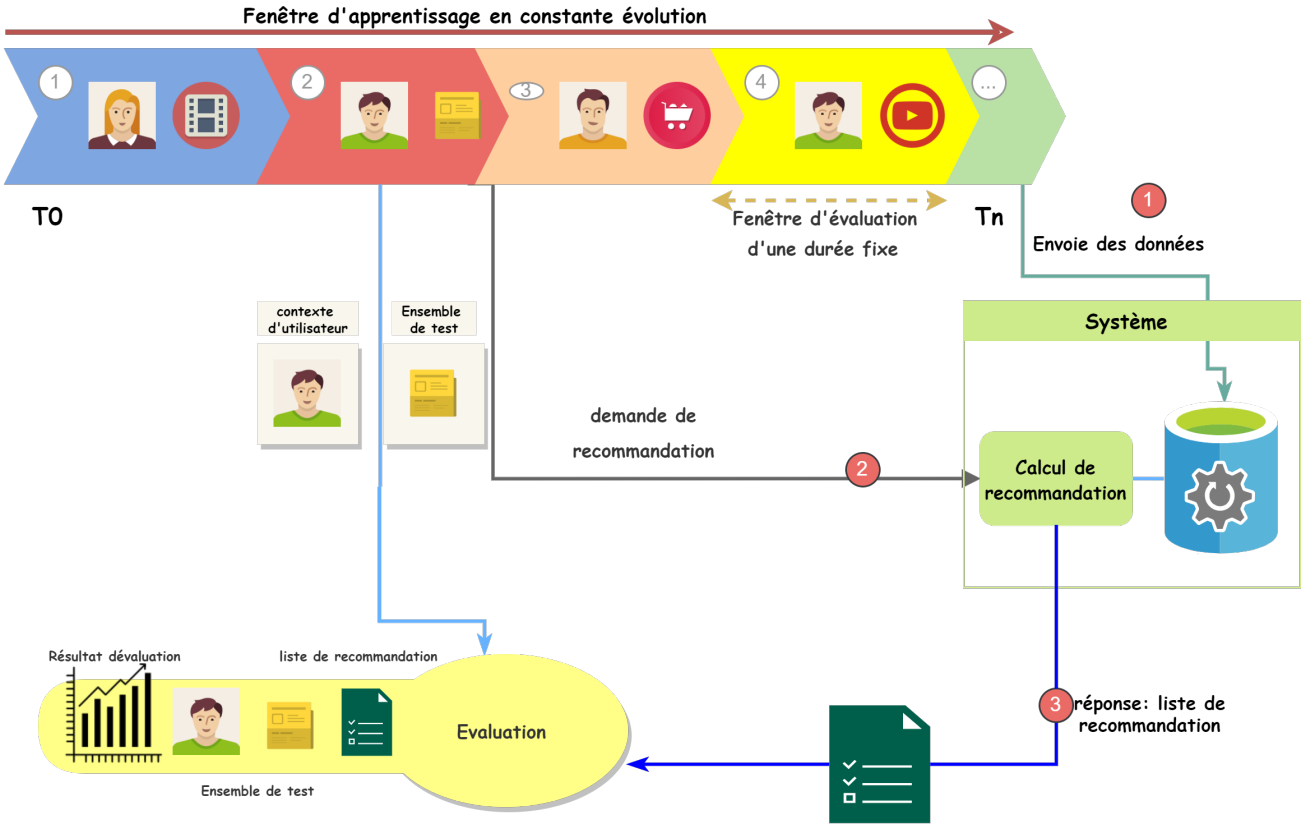

Figure II.1 – Cadre d'évaluation proposé

II.2.1 Préparation des données :

Dans le cas d'évaluation hors ligne, les données sont divisées en deux ensembles : un ensemble d'apprentissages et un ensemble de tests. Quand on se base sur un partitionnement aléatoire pour générer ces ensembles, cela entraîne la perte de plusieurs propriétés importantes des données (p. ex. la séquence et l'ordre chronologique). Pour pallier ce problème, nous proposons de suivre un partitionnement à caractère temporel[21]. L'objectif est de

se rapprocher le plus possible du flux de données tel qu'il est dans la plateforme d'origine. De cette façon, pour générer des recommandations à chaque laps de temps, un système à évaluer ne disposera que des observations qu'il est censé avoir dans la plateforme d'origine d'où les données sont collectées. Le concept des fenêtres temporelles est donc adopté dans notre proposition pour reproduire aussi fidèlement que possible les caractéristiques de ce flux de données[16].

II.2.1.1 Fenêtre d'apprentissage en constante évolution

Nous proposons de représenter l'ensemble d'apprentissages par une fenêtre temporelle en constante évolution (cf. la figure II.2). Cette fenêtre commence à l'instant **t0**, qui représente le début du flux de messages à distribuer (c.-à-d. le système n'a pas encore reçu de données). Par la suite, la fenêtre va s'agrandir au fil du temps jusqu'à l'instant **t-user**, qui représente le moment où l'utilisateur a accédé à la plateforme de recommandation d'origine. De cette façon, pour cibler l'utilisateur user avec une recommandation à l'instant **t-user**, un système n'opère que sur les messages qu'il est censé avoir dans la plateforme d'origine à cet instant. Par ailleurs, il est à noter que les propriétés des messages comme la séquence et l'ordre chronologique sont conservées telles qu'elles sont observées sur la plateforme d'origine.

Figure II.2 – Illustration de notre fenêtre d'apprentissage

II.2.1.2 Fenêtre de test glissante

L'objectif d'un système de recommandation est de satisfaire les besoins de l'utilisateur. Autrement dit, c'est de lui proposer des recommandations « pertinentes ». Dans notre contexte, la pertinence est considérée comme l'adéquation des items recommandés aux préférences/besoins/contextes de l'utilisateur concerné. La méthode la plus appropriée

pour évaluer cette pertinence est de faire juger les résultats du système à évaluer par l'utilisateur lui-même[22]. La question qui se pose donc est comment récupérer l'avis de l'utilisateur sur un item dont le système lui a recommandé ?

Si nous examinons le scénario réel de recommandation, nous pouvons nous apercevoir que l'avis de l'utilisateur **user** se trouve juste après l'instant **t-user** (c.-à-d. après le moment de la génération d'une recommandation). Ainsi pour répondre à la question précédente, nous proposons de représenter l'ensemble de tests par une fenêtre coulissante, d'une courte durée, qui commence à partir de l'instant t-user jusqu'à **t-user + la-durée-d'évaluation (c.-à-d. les M minute qui suivent la requête de recommandations)**. Le paramètre durée-d'évaluation représente la période durant laquelle l'utilisateur consulte/consomme et juge la pertinence de l'item recommandé (p. ex. à travers un clic, une notation, un commentaire, etc.). La figure II.3 montre un exemple de cette fenêtre d'évaluation.

Figure II.3 – Illustration de notre fenêtre de test

II.2.2 Distribution des Flux de données :

Dans cette phase, les flux de messages vont être distribués aux systèmes de recommandation à évaluer. Les messages sont envoyés dans l'ordre chronologique observé sur la plateforme d'origine. Par ailleurs, pour assurer le bon déroulement de cette phase, les messages sont regroupés en plusieurs catégories, à savoir les notifications de création de nouveaux items, les notifications d'observation d'une action de l'utilisateur et les demandes de recommandation.

— **Notifications de création de nouveaux items :** les messages de ce type servent à informer un système à évaluer des items ajoutés à la plateforme de recommandation. Ces messages incluent les informations relatives à l'item concerné (p. ex. le titre, l'URL et le résumé d'un article d'actualité).

— **Notifications d'observation d'une action de l'utilisateur :** les messages de ce type servent à informer un système à évaluer des événements observés sur la plateforme de recommandation d'origine (p. ex. l'utilisateur u1 a consulté l'article a150, l'utilisateur u15 a acheté le produit p20).

— **Demandes de recommandation :** ces messages représentent les requêtes de recommandations envoyées aux systèmes à évaluer pour leur demander de générer une liste de recommandations à un utilisateur bien déterminé. Ici le message décrit le contexte de la demande de recommandation en cours (p. ex. le nombre de recommandations à retourner, l'identifiant de l'utilisateur concerné, l'item que l'utilisateur est actuellement en train de consulter sur la plateforme, etc.).

Dans cette phase, les systèmes à évaluer ingèrent les messages envoyés et génèrent des listes de recommandation. En effet, à chaque arrivée d'une nouvelle notification (p. ex. la lecture d'un article de presse, l'achat d'un produit sur une plateforme de commerce électronique, la notation d'un film sur une plateforme de vidéo, etc.), le système mit à jour ses modèles de prédiction. Ainsi, lors de l'arrivée d'un message de type « **Demande de recommandation** », le système génère une liste des recommandations adaptée au contexte de cette demande. Un exemple concret de ce scénario est illustré dans la figure II.4 et II.5.

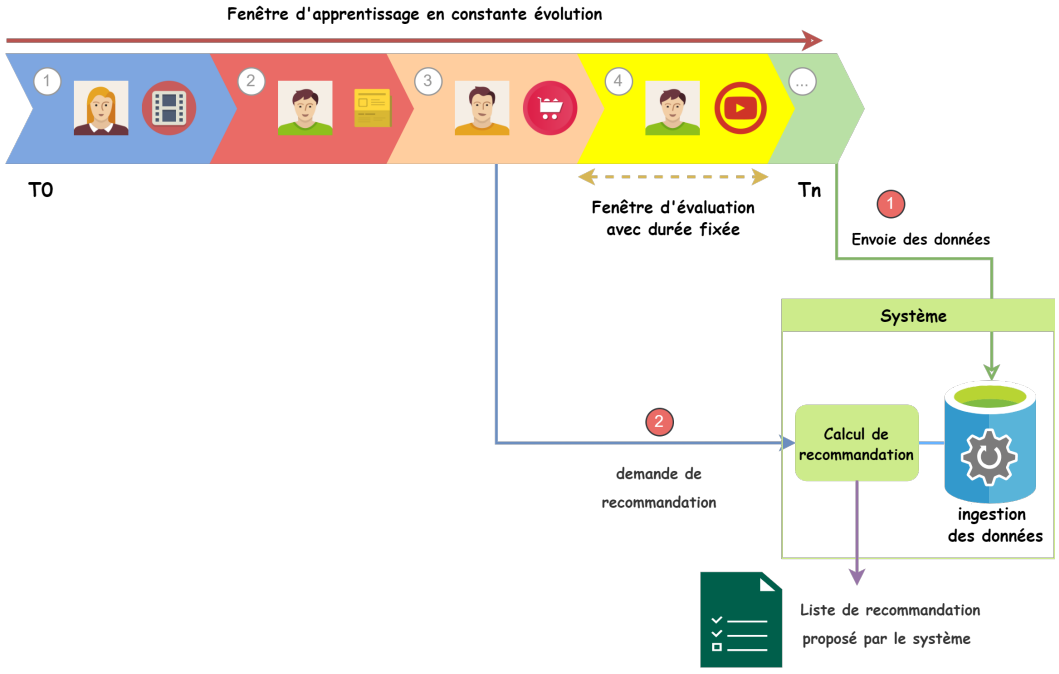

Figure II.4 – Phase de distribution des données

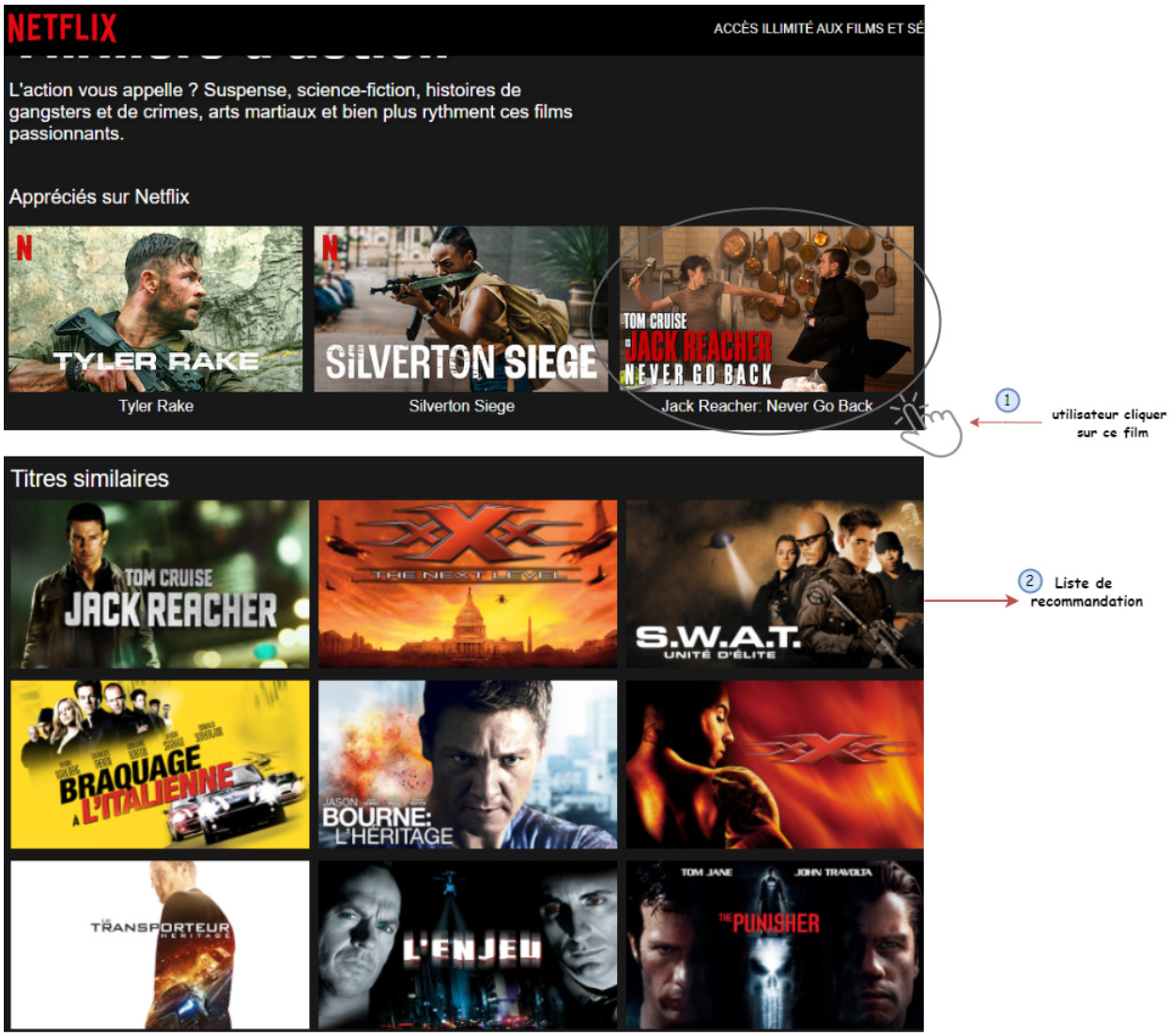

Figure II.5 – Exemple du contexte et du résultat d'une demande de recommandation

II.2.3 Évaluation de la pertinence des recommandations

Les recommandations retournées sont comparées vis-à-vis l'ensemble de tests pour évaluer la pertinence des prédictions de chaque système. Plusieurs métriques de la qualité des prédictions peuvent être donc appliquées. Le choix des métriques dépend des objectifs et des paramètres de l'évaluation. Un exemple d'évaluation en utilisant les métriques de précision de classification : « **précision** » et « **rappel** » est comme suit : pour une liste d'items disponibles :***<item 7, item 5, item 10, item 1, item 3>***, une liste de

recommandation, dont la limite k=3 :***<item 7, item 5, item 10 >,*** si on sait que l'utilisateur a consommé **l'item 5** et **l'item 10**, alors :
la précision = $\frac{2}{5}$. et le rappel = $\frac{2}{3}$.

Concrètement, le processus d'évaluation dans cette phase (cf. figure II.6) consiste principalement à répondre à la question : est-ce que la fenêtre de test contient les items recommandés ou non ? En effet, à ce stade, nous sommes dans l'impossibilité de reproduire à l'identique ce qui a été déjà observé sur la plateforme d'origine (mêmes utilisateurs, mêmes items, même contexte, etc.) pour diverses raisons (accès limité aux plateformes d'origine, politique de confidentialité des données personnelles, taille de données importante, etc.). Ainsi, pour juger la pertinence des recommandations générées, nous supposons qu'un item recommandé est pertinent pour l'utilisateur ciblé seulement si la fenêtre de test contient cet item[16]. Cela signifie que l'utilisateur ciblé par la recommandation générée a interagi avec cette recommandation dans les minutes qui suivent la demande de recommandations. Ceci nous permettra d'évaluer les systèmes candidat en se basant uniquement sur les le flux de messages enregistrés sur la plateforme d'origine, sans l'intervention des utilisateurs de cette plateforme. L'application de notre logique précédente nous mène à deux scénarios d'évaluation différents : une évaluation explicite ou implicite. La différence entre les deux scénarios réside dans l'estimation de la durée de la fenêtre de test (comment estimer la valeur du paramètre durée-d'évaluation).

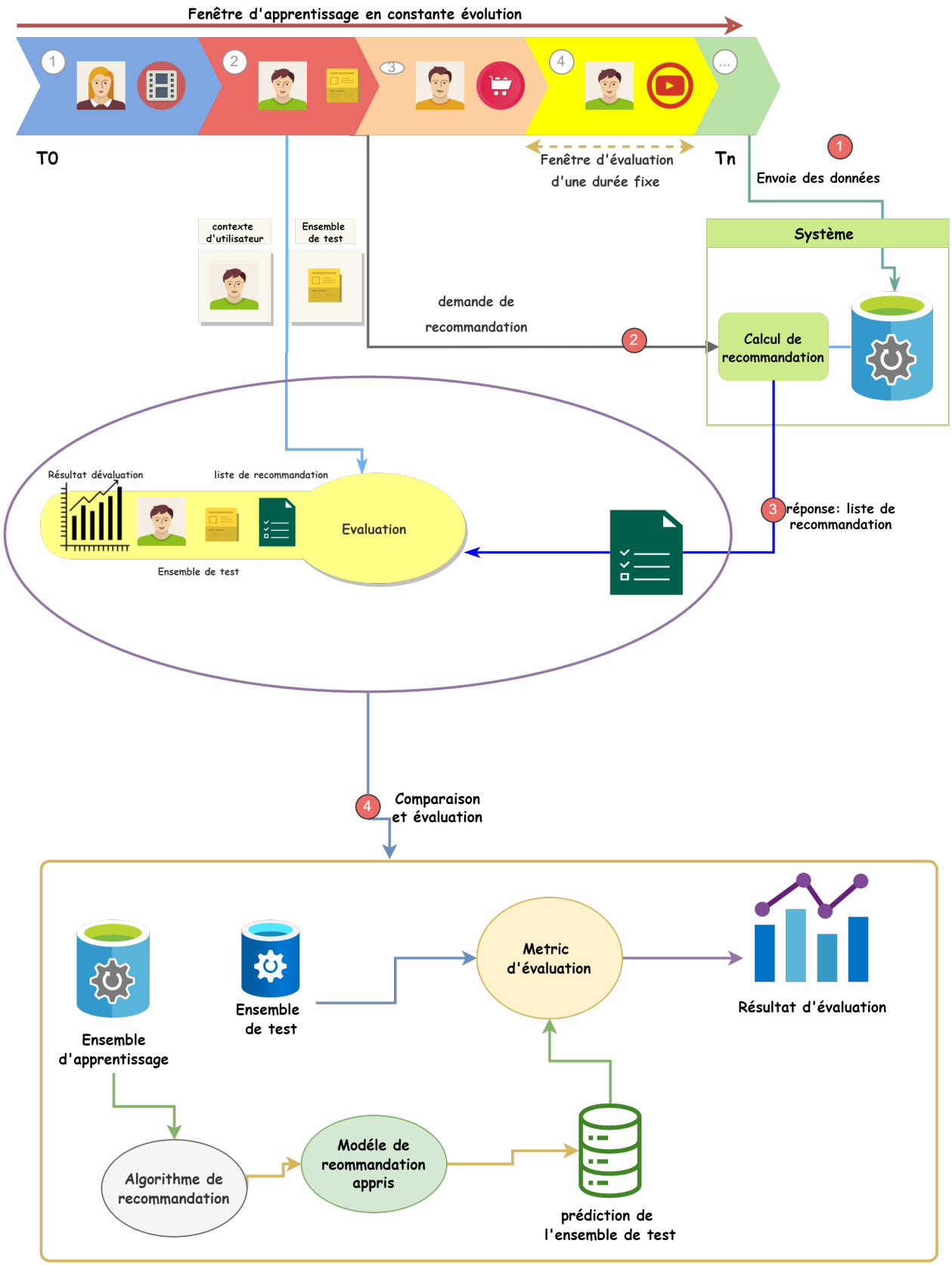

Figure II.6 – Phase d'évaluation d'un système candidat

II.2.3.1 Évaluation explicite

Comme le montre l'algorithme 1, l'idée de l'évaluation explicite est de proposer une durée fixe de la fenêtre de test pour toutes les requêtes de recommandation. Le choix et l'optimisation de la valeur de ce paramètre dépendent de la nature du domaine d'application ciblé.

Algorithm 1 Évaluation explicite

Entrée :Twt (en minute) : la durée de la fenêtre de test, **D :** Ensemble de données, **Tuser :** instant de l'affichage de la liste de recommandation générée.
Sortie : FT : fenêtre de test.
Lire D l'ensemble de données
Trier D l'ensemble de données par ordre chronologique
FT ← {}
for Tout élément de l'ensemble de données **D do**
    if D est un item **then**
        Envoyer un message de type « **notification de création d'un nouvel item** »
    else //D est un événement
        Envoyer un message de type « **demande de recommandation** »
        if D est un événement qui s'est produit après **Tuser** et avant **(Tuser + Twt) then**
            Ajouter **D** à **FT**
        end if
    end if
end for

À titre illustratif, pour un ensemble d'apprentissages **D = {item1, item2, item3, item4, event1 [user1, item-4, 15-06-2022-18h05], item5, event-2 [user2, item5, 15-06-2022-18h08], event3 [user1, item2, 15-06-2022-18h12], ...}** une demande de recommandation qui concerne l'utilisateur «**user1**» à l'instant **Tuser = 15-06-2022-18h04**, la fenêtre de test relatif à cette demande de recommandation utilisateur est :

— Si la taille de la fenêtre de test **Twt** = 5min, **FT = {event1 [user1, item-4, 15-06-2022-18h05]}**.

— Si la taille de la fenêtre de test **Twt** = 10min, **FT = {event1 [user1, item-4, 15-06-2022-18h05], event3 [user1, item2, 15-06-2022-18h12]}**.

II.2.3.2 Évaluation implicite

Les plateformes visant un domaine d'application spécifique proposent des catalogues d'items homogènes qui partagent généralement les mêmes spécificités. Ainsi, l'idée de fixer la taille de la fenêtre de test pour toutes les requêtes de recommandation semble très bien adaptée à cette plateforme. Cependant, cette idée est à remettre en cause dans le cas des plateformes multidomaines, où nous pouvons trouver une panoplie de produits très divergente. En effet, le degré de dynamisme de la plateforme ciblé peut être un facteur important pour la détermination de la taille de la fenêtre de test. Par exemple sur un blog ou un réseau social, un article annonçant une actualité sportive peut perdre de l'intérêt dans les quelques heures qui suivent sa création. En revanche, la durée de vie d'un post qui discute les idées d'un livre peut s'étendre sur plusieurs mois, voire plusieurs années. De plus, le temps de lecture de ces deux types items est très distinct.

Au vu de l'analyse précédente, nous proposons dans cette variante d'évaluation d'estimer la taille de la fenêtre de test pour chaque demande de recommandation. Comme le montre l'algorithme 2, l'idée est donc d'utiliser le contexte de la demande de recommandation en cours pour estimer ce paramètre.

Algorithm 2 Évaluation implicite

```
1: Entrée : D : Ensemble de données, Tuser : instant de l'affichage de la liste de recommandation générée, TwT :(en minute) la durée de la fenêtre de test à calculer.
2: Sortie : FT : fenêtre de test.
3: Fonction : EstimateTestWindowSize(i : item) : estimer la taille de la FT
4: Lire D l'ensemble de données
5: Trier D l'ensemble de données par ordre chronologique
6: FT ← {}
7: for Tout élément de l'ensemble de données D do
8:     if D est un item then
9:         Envoyer un message de type « notification de création d'un nouvel item »
10:    else //D est un événement
11:        Envoyer un message de type « demande de recommandation »
12:        TwT ← EstimateTestWindowSize(D)
13:        if D est un événement qui s'est produit après Tuser et avant (Tuser + Twt) then
14:            Ajouter D à FT
15:        end if
16:    end if
17: end for
```

Le choix de l'implémentation de la fonction EstimateTestWindowSize reste très large. Dans notre cas, nous supposons que la taille de la fenêtre de test est équivalente au temps que l'utilisateur prend pour passer du début de la page Web courante jusqu'à la rubrique affichant la liste de recommandation. Par exemple, dans le cadre d'un portail d'actualité, cette durée est la durée de lecture de l'item que l'utilisateur est actuellement en train de consulter (cf. la figure II.7,).

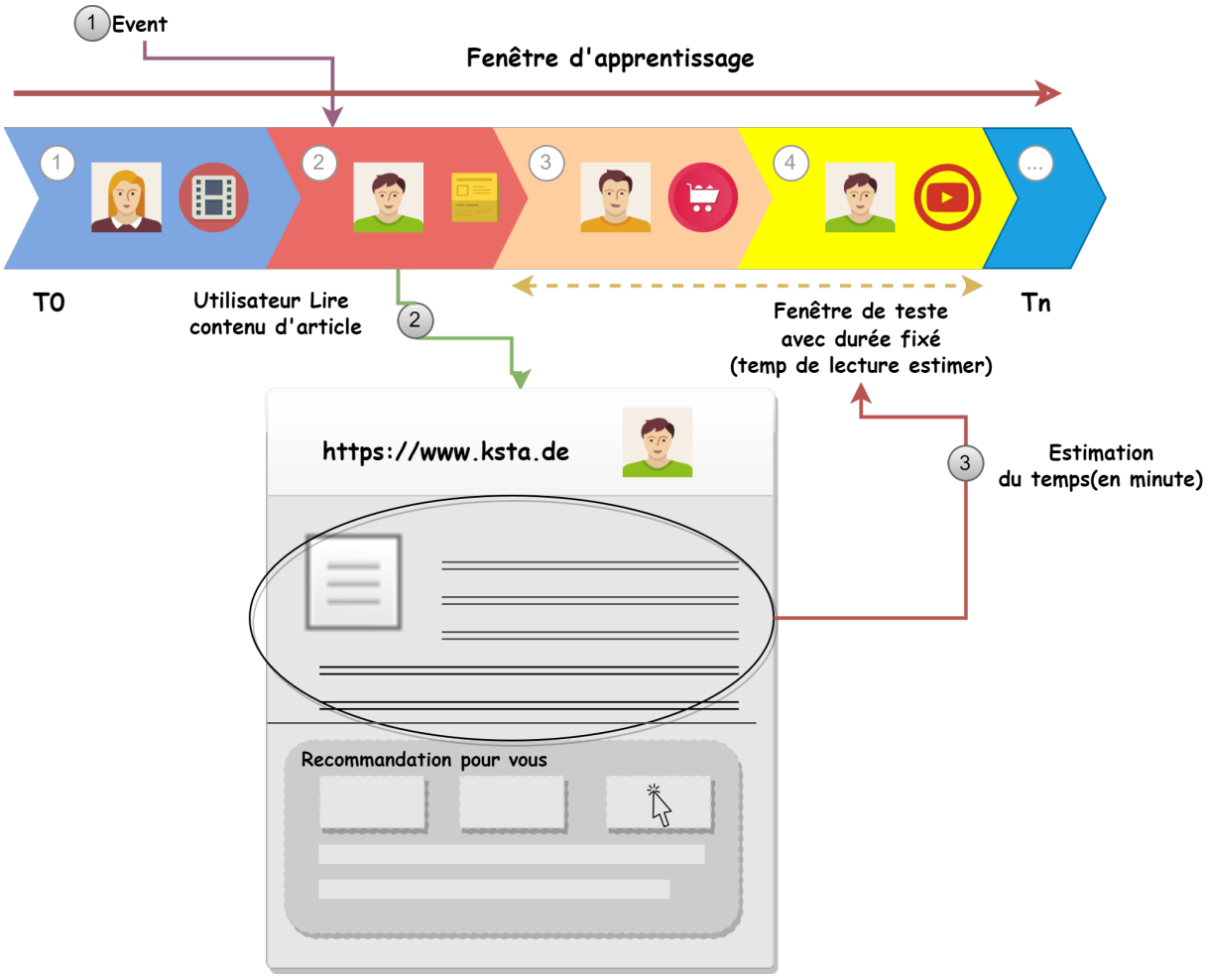

Figure II.7 – Illustration de la phase implicite

Conclusion

Ce chapitre est consacré à la présentation de nos directives vers un nouveau cadre d'évaluation plus réaliste. La problématique d'évaluation d'un système de recommandation a été modélisée sous un nouvel angle en s'inspirant des travaux de l'art. Dans le chapitre suivant, nous nous intéressons aux protocoles d'expérimentation adoptés pour la configuration et l'évaluation de cette proposition.

Chapitre III

Expérimentation et évaluation de la proposition

Sommaire

Introduction

Ce chapitre est consacré à l'évaluation de l'intérêt de notre cadre d'évaluation des systèmes de recommandation. Pour ce faire, nous présentons dans un premier temps le cadre expérimental adopté. Par la suite, nous détaillons les expérimentations menées pour juger l'intérêt de notre proposition. Les résultats obtenus sont alors discutés afin de mieux valider les hypothèses et les idées de cette proposition.

III.1 Cadre Expérimental

Pour évaluer l'intérêt du cadre d'évaluation proposé, nous avons mené plusieurs expérimentations en suivant le protocole hors ligne et en ligne. Le jeu de données, les algorithmes ainsi que les métriques d'évaluation adoptées dans ces expérimentations sont détaillés dans les sections suivantes.

III.1.1 Jeu de données

Pour évaluer l'intérêt du cadre d'évaluation proposé, nous avons utilisé le jeu de données de la plateforme de recommandation Plista. Cette collection de données regroupe de manière chronologique les événements observés sur le portail d'actualités http://www.ksta.de. Ce portail s'intéresse à toute l'actualité en Allemagne (p. ex. politique, sport, santé, culture, etc.)[23].

Le jeu de données de Plista contient **1088** items, **2066582** événements, et **857 906** utilisateurs. **30%** des utilisateurs de ce jeu de données sont des utilisateurs anonymes (qui activent le mode de navigation anonyme). Le choix de ce jeu de données est basé sur deux critères : la taille et l'aspect temps des données. En effet, nous étions obligés de choisir une collection de données dont la taille doit être adaptée avec les capacités de nos machines afin d'assurer un temps de traitement raisonnable. Par ailleurs, le facteur du temps représente un aspect important pour notre expérimentation, car l'ordre chronologique des évènements est une propriété importante du flux de données des plateformes de recommandation du Web actuel.

III.1.2 Algorithmes évalués

Dans le cadre de cette expérimentation, nous avons sélectionné quelques algorithmes de recommandation très répandus sur les plateformes de recommandation en ligne. Ces algorithmes se caractérisent par leur capacité de gérer des flux de données en permanence et générer des listes de recommandations dans les plus brefs délais. Les idées de ces algorithmes sont décrites dans ce qui suit :

— **Random :** c'est un algorithme qui propose des recommandations de manière aléatoire.

— **Recently Popular :** c'est un algorithme qui se base sur la popularité des items pendant un laps de temps spécifique pour proposer des recommandations. Ainsi, il recommande les items qui ont suscité le plus de réactions durant les dernières minutes/heurs qui précèdent une demande de recommandation.

— **Most Popular :** c'est un algorithme qui recommande les items les plus populaires, sans qu'il se limite à aucun facteur de temps.

— **Recently Clicked :** c'est un algorithme qui recommande les items qui ont été récemment consommés par les utilisateurs. C'est une sorte de tendances actuelles affichées généralement par les plateformes en ligne dans les rubriques intitulées « Nos utilisateurs consultent actuellement ... ».

— **CoOccurrence :** c'est un algorithme très répandu qui se base les corrélations possibles entre les items pour proposer des recommandations. Ainsi, il tente de déterminer combien de fois deux items sont apparus ensemble dans les données historiques de l'utilisateur pour générer les recommandations.

III.1.3 Les métriques d'évaluation

Dans ce cadre expérimental, nous avons adopté les principales métriques standards pour l'évaluation de la pertinence des recommandations. Ces métriques sont détallées dans ce qui suit :

— **La précision :** mesure la précision des prédictions. Autrement dit, il estime le pourcentage des prédictions qui sont correctes parmi les items recommandés. Ainsi, il est calculé en divisant le nombre des prédictions correctes **(vrais positifs, VP)**

sur la somme des prédictions correctes et incorrectes **(N)**.

$$\text{Précision } = \frac{VP}{N} \tag{III.1}$$

— **Le rappel :** mesure à quel point le système propose des recommandations valides. Autrement dit, il calcule la proportion des recommandations pertinentes **(vrais positifs, VP)** proposées à l'utilisateur ciblé parmi la liste de toutes les recommandations pertinentes pour cet utilisateur **(liste V)**.

$$\text{Rappel } = \frac{VP}{V} \tag{III.2}$$

— **La mesure F1 :** c'est une mesure qui estime la moyenne entre le rappel et la précision du modèle de recommandation à évaluer :

$$F1 = 2 * \frac{\text{rappel * précision}}{\text{rappel +précision}} \tag{III.3}$$

— **MAP (Mean Average precision) :** la moyenne de la précision moyenne est dérivée de la précision moyenne **(AP)** . Tout d'abord, nous devons calculer l'AP à un seuil arbitraire **N** de chaque ensemble de données. ou **rel@N** est juste un indicateur qui indique si i est un élément pertinent ou pas. Ensuite, nous résumons simplement et trouvons la moyenne de **AP@k** de chaque ensemble de données pour obtenir le **MAP@k**.

Les formules et calculs **AP@k** et **mAP@k** sont les suivants (cf. la formule III.4 et III.5) :

$$AP@N = \frac{1}{|V|} \sum_{i=1}^{N} \text{Precision@N} \times \text{rel@N} \tag{III.4}$$

$$MAP@N = \frac{1}{|Q|} \sum_{i=1}^{|Q|} AP@N(q) \tag{III.5}$$

— **MRR :** la mesure de la moyenne des réciproques des rangs (MRR) calcule la moyenne de l'inverse du rang auquel la première suggestion pertinente de chaque requête a été récupérée. Ainsi, elle permet d'évaluer la capacité des systèmes à proposer des recommandations à des utilisateurs ne souhaitons consommées qu'un seul

item à la fois. En notant $\boldsymbol{fc}R_i$ le rang de la première prédiction correcte pour la requête $i \in Q$, prenant pour valeur **0** si aucune prédiction correcte n'a été trouvée, la mesure MRR est définie de la manière suivante (cf. la formule III.6) :[24]

$$MRR@N = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{fcR_i} \tag{III.6}$$

— **NDCG (Normalized Discounted Cumulative Gain) :** Dans une liste de recommandation ciblant, certains items sont plus pertinents que d'autres pour l'utilisateur ciblé. Par conséquent, cette liste devra contenir les éléments les plus pertinents en premier, suivis des éléments moyennement pertinents, et ainsi de suite. Le gain cumulatif normalisé est une métrique qui permet de prendre en compte cet aspect en estimant à la fois la pertinence et le rang des items recommandés.En effet, étant donnée k le rang de l'item à recommander, et $r_{uk'}$, la note d'évaluation attribuée par l'utilisateur ciblé u à cet item, la mesure **NDCG@N** peut être définie par la formule (cf. la formule III.7) :[25]

$$NDCG@N = \frac{1}{IDCG@N}\sum_{1}^{N}\frac{2^{r_{uk}}-1}{\log_2(1+k)} \tag{III.7}$$

— **CTR(Click through rate) :** la mesure « **taux de clics (CTR)** » est utilisée pour estimer le rapport entre le nombre de clics sur les recommandations calculées par les systèmes participants et le nombre de requêtes de recommandations traitées par ces derniers (cf. la formule III.8). La prémisse est qu'en cliquant sur un item recommandé, l'utilisateur exprime implicitement une attitude positive indiquant sa possible satisfaction par l'item recommandé. D'un point de vue commercial, cette mesure montre à quel point le système de recommandation est efficace pour prédire les intérêts des utilisateurs ciblés[4].

$$CTR@N(\%) = \frac{\#\text{Clics}}{\#\text{Demandes de recommandation}} \times 100 \tag{III.8}$$

III.2 Évaluations comparatives hors ligne

Le protocole d'évaluation hors ligne est un protocole standard qui est très répandu dans la littérature. Ce protocole présente plusieurs points problématiques qui fait que les algorithmes les pertinents dans ce cadre statique ne sont pas toujours les meilleurs algorithmes à adopter dans les plateformes réelles. De ce fait, notre proposition essaie de redéfinir les principes de ce protocole pour simuler le plus fidèlement possible le scénario de recommandation en ligne. Ainsi, une analyse comparative avec les résultats de ce protocole est primordiale pour montrer l'intérêt de notre proposition.

III.2.1 Configuration hors ligne

Le principe du protocole hors ligne consiste à séparer les données en deux sous-ensembles : un sous-ensemble pour l'entraînement et un autre pour le test. Dans nos expérimentations, **80%** des données sont utilisées pour apprendre les modèles de prédiction des systèmes à évaluer et les **20 %** restants pour valider les recommandations générées par ses systèmes. Il est à noter que ce partitionnement respecte l'ordre chronologique des données.

III.2.2 Résultats comparatifs hors ligne et interprétations

La figure III.1 et le tableau III.2 présentent nos résultats comparatifs hors ligne issus de l'expérimentation sur le jeu de données de Plista. Deux variantes de la taille de la liste de recommandation sont présentées dans les résultats, à savoir N = 5 (Top-5) et N = 10 (Top-10)(avec N donner par la platforme d'origine)(cf. la figure III.2et le tableauIII.2).

Métrique / Algorithme	**F1**	**Map**	**NDCG**	**Click through rate**	**MRR**
Recently clicked	0,0022563	0,0016233	0,0031841	0,0220745	0,0063543
random	0,0015528	0,001067	0,0017159	0,0030494	0,0029992
Most Popular	0,0438595	0,0360238	0,0463707	0,0303508	0,0612915
Recently Popular	0,0067601	0,0044683	0,0073848	0,020029	0,0098081
CoOccurrence	0,8212946	0,8138202	0,8522261	0,134952	0,9420219

Tableau III.1 – Résultats comparatifs hors ligne avec N=5

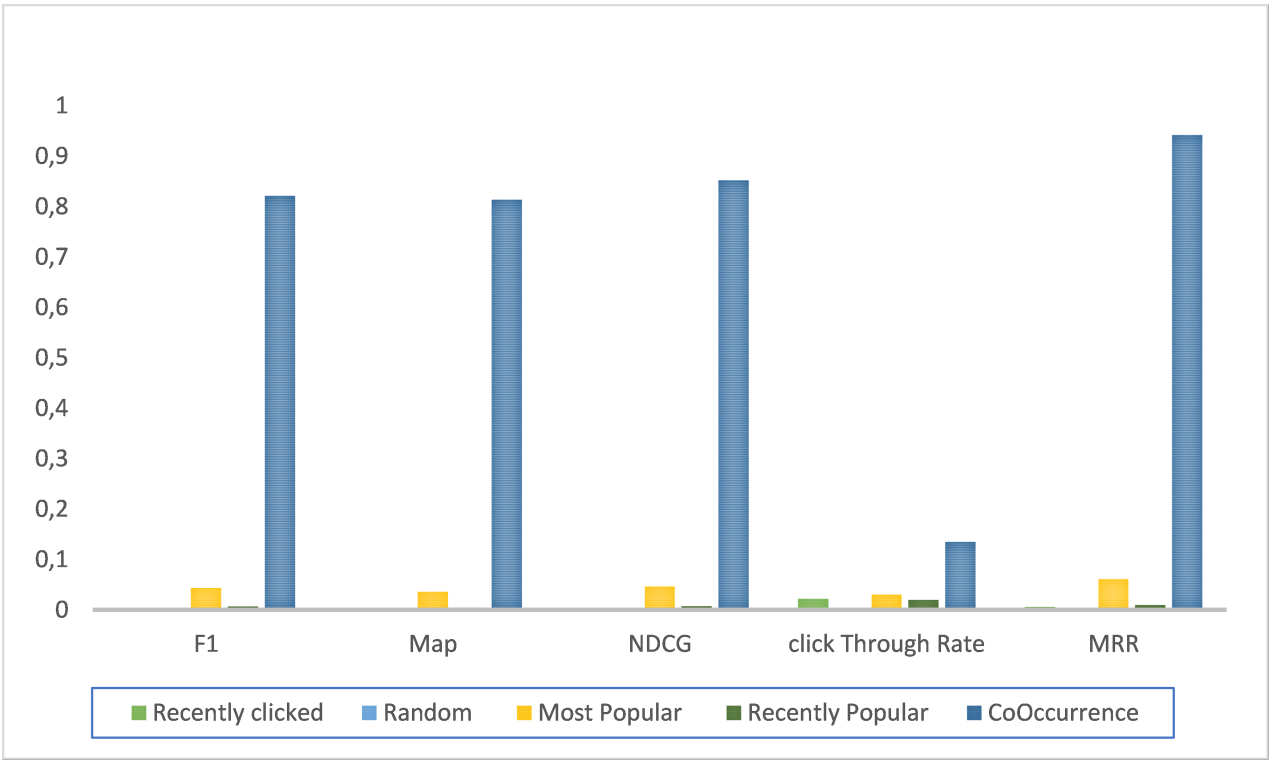

Figure III.1 – Résultats comparatifs hors ligne avec N=5

Métrique / Algorithme	F1	Map	NDCG	Click through rate	MRR
Recently clicked	**0,003081**	**0,0017699**	0,0036324	0,2901986	0,0071311
Random	**0,0016989**	**0,00112**	0,0017941	0,0045771	0,0032307
Most popular	**0,0438158**	**0,0363444**	0,0452667	0,0363238	0,0616961
Recently popular	**0,0065131**	**0,0045349**	0,0068029	0,0258039	0,0096784
CoOccurrence	**0,8196083**	**0,814291**	0,8474014	0,1451664	0,9388163

Tableau III.2 – Résultats comparatifs hors ligne Avec N=10

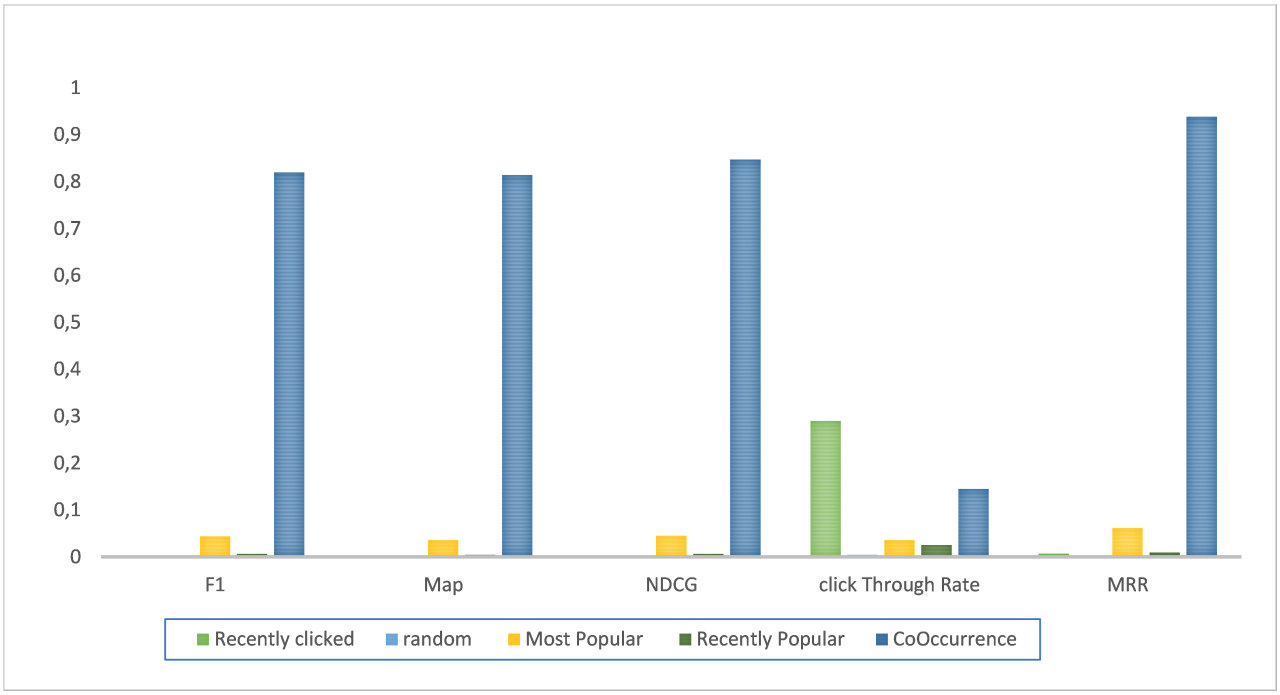

Figure III.2 – Résultats comparatifs hors ligne avec N=10

En observant les résultats, nous pouvons identifier trois tendances générales :

— Les résultats les plus faibles sont obtenus par l'algorithme aléatoire « **Random** » et les algorithmes dynamiques « **Recently Clicked** » et «**Recently Popular** » ne dépasse pas les **0,16%**, au point d'être comparable à des recommandations aléatoires **(0,10%)**. Ces résultats sont attendus, car le cadre d'évaluation hors ligne est un cadre statique, qui ne tient pas compte des propriétés dynamiques des données.

— L'algorithme « **Most Popular**» présente des résultats assez moyens par rapport aux autres systèmes candidats. Par exemple, la précision des recommandations de cet algorithme est dans les alentours des **4%**.

— Les résultats les plus élevés sont ceux de l'algorithme « **CoOccurrence** », où nous pouvons observer une précision de recommandation qui dépasse les **80%**. Ceci peut se justifier par le fait que cet algorithme s'adapte facilement à des contextes statiques, car il est basé sur des associations et des corrélations statiques entre les items à recommander.

III.3 Évaluations comparatives en ligne

Les évaluations en ligne sont basées sur le cadre d'évaluation que nous avons proposé. Notre proposition a pour but d'évaluer les systèmes de recommandation dans des conditions qui se rapprochent le plus possible du scénario de recommandation dans les plateformes réelles.

III.3.1 Configuration en ligne

Contrairement au protocole hors ligne qui se base généralement sur le partitionnement aléatoire et statique des données, notre configuration en ligne est basée sur une fenêtre d'apprentissage en croissance continue. Cette fenêtre temporelle commence à l'instant **t0**, qui représente le début du flux de messages à distribuer. Par la suite, elle continuera à s'agrandir au fil du temps jusqu'à l'instant **t-user**, qui représente le moment où l'utilisateur a accédé à la plateforme de recommandation d'origine.

L'ensemble de tests est représenté par une fenêtre coulissante, d'une courte durée, qui commence à partir de l'instant t-user jusqu'à **t-user + la-durée-d'évaluation (c.-à-d. les M minute qui suivent la requête de recommandations)**. Le paramètre durée-d'évaluation représente la période durant laquelle l'utilisateur consulte et juge la pertinence de l'item recommandé (dans notre contexte, à travers un clic). Dans les expérimentations qui suivent, ce paramètre est déterminé soit explicitement ou implicitement. Par ailleurs, une seule variante de la taille de la liste de recommandation est présentée dans les résultats, à savoir N = 10 (Top-10), car les résultats sont légèrement différents avec ceux des autres variantes.

III.3.2 Résultats comparatifs en ligne - Variante explicite

Dans cette variante, le paramètre du temps est fixé manuellement. Plusieurs valeurs de ce paramètre sont testées, à savoir 2 min, 5 min et 10 min. Ces choix sont basés sur le fait que le domaine de l'actualité est un domaine très actif où un événement qui est parmi les tendances à l'instant courant peut devenir sans aucune importance dans une courte durée.

Le tableau III.3 et la figure III.3 présentent les résultats de la variante explicite. À partir de ces résultats, nous pouvons identifier trois tendances :

Métrique / Algorithme	F1	Map	NDCG	Click through rate	MRR
Recently clicked	0,8861319	0,871976	0,9140701	0,6962722	0,9339299
Random	0,0027325	0,001636	0,0030295	0,0086963	0,0062484
Most Popular	0,0461061	0,0372907	0,0490906	0,0920998	0,0634177
Recently Popular	0,1563888	0,1355915	0,1678375	0,3039472	0,163581
CoOccurrence	0,7952857	0,7817323	0,8190246	0,6109718	0,8492552

Tableau III.3 – Résultats comparatifs en ligne -Variante explicite avec N=10 et twt= 2min

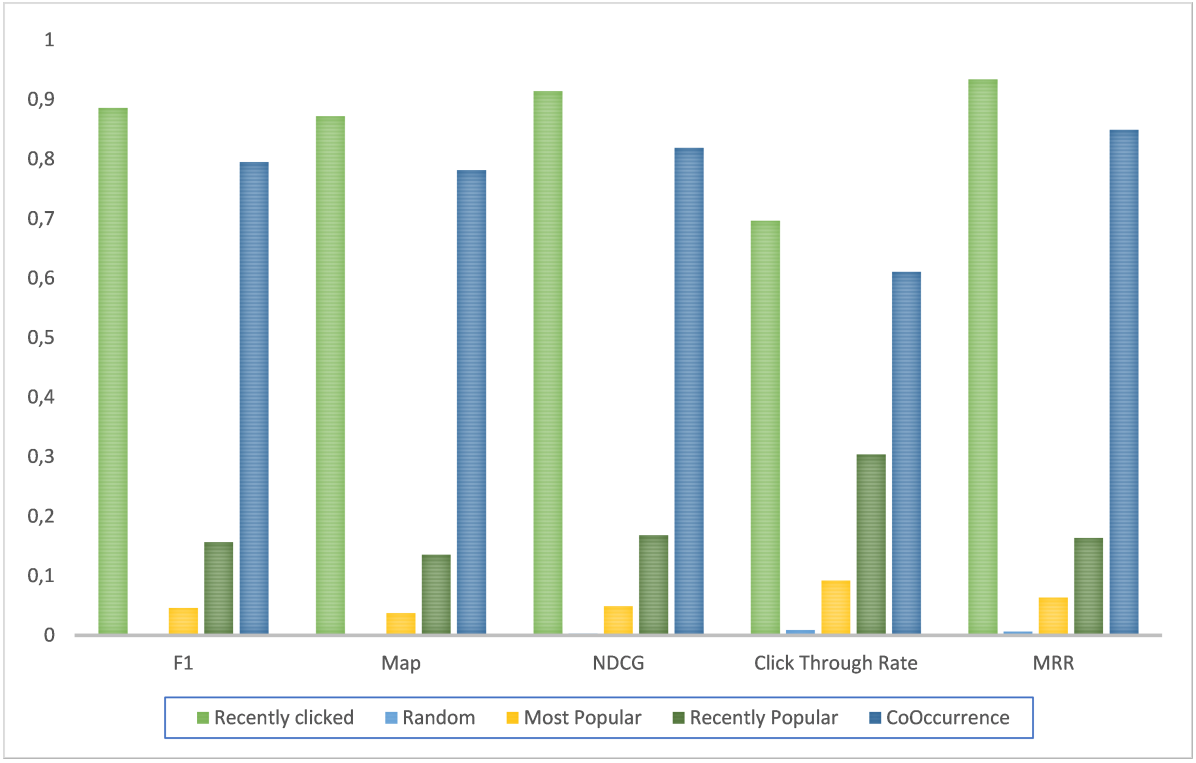

Figure III.3 – Résultats comparatifs en ligne -Variante explicite avec N=10 et twt=2min

— Tout d'abord nous pouvons clairement observer que l'algorithme de recommandation « **Recently Clicked** » présente les meilleurs résultats, avec un taux de clic qui dépasse les **71%**, et une précision dans les alentours des **90%**. Ceci peut se justifier par le fait que le cadre en ligne préserve les propriétés des données dynamiques telles qu'elles sont observées sur la plateforme d'origine. Similairement, l'algorithme « **CoOccurrence** » présente de bons résultats qui ne tombent pas en dessous des

75%, car les associations entre les items à recommander sont aussi possibles à identifier dans un cadre dynamique que dans un cadre statique. Il est à noter que les autres métriques présentent généralement les mêmes tendances.

— L'algorithme « **Recently Popular** » présente des résultats assez moyens par rapport aux autres systèmes candidats. Par exemple, la précision des recommandations de cet algorithme est dans les alentours des **14%**.

— L'algorithme « **Most Popular** » montre des résultats assez faibles, car il ignore le facteur de nouveauté alors que le domaine de l'actualité est un domaine très actif. Enfin, comme prévu, l'algorithme aléatoire présent les résultats les plus faibles.

III.3.3 Résultats comparatifs en ligne - Variante implicite

L'idée de fixer explicitement la taille de la fenêtre de test pour toutes les requêtes de recommandation semble très bien adaptée à des plateformes homogènes qui proposent les mêmes types d'items. Cependant, cette idée est à remettre en cause dans des plateformes dynamiques et multi domaines, où nous pouvons trouver une panoplie de produits très divergente. Par exemple, sur Facebook, un post annonçant une actualité sportive peut perdre de l'intérêt dans les quelques heures qui suivent sa création. En revanche, la durée de vie d'un post qui discute les idées d'un livre peut s'étendre sur plusieurs mois, voire plusieurs années. De plus, le temps de lecture de ces deux types items est très distinct.

L'approche implicite se base donc sur un apprentissage automatique, qui exploite le contexte de l'action en cours pour estimer la taille de la fenêtre de test pour chaque demande de recommandation (cf. algorithme2, chapitreII). Plusieurs choix s'offrent à nous pour implémenter de telle méthode. Cependant, vu les contraintes temporelles, nous supposons dans notre expérimentation que la taille de la fenêtre de test est équivalente à la durée de lecture de l'article que l'utilisateur est actuellement en train de consulter (cf. algorithme 3). Autrement dit, nous supposant que ce temps de lecture est le temps nécessaire pour un utilisateur pour qu'il passe du début de la page Web courante jusqu'à la rubrique affichant la liste de recommandation. Ce qui colle exactement avec notre définition de la taille de la fenêtre de test.

Le tableau III.4 et la figure III.4 présentent les résultats de la variante implicite. À partir de ces résultats, nous pouvons remarquer que les résultats d'évaluation se sont légèrement améliorés par rapport aux résultats de la variante explicite. Par exemple, l'algorithme « **Recently Clicked** » montre une pertinence de **89%** en ce qui concerne la

Algorithm 3 EstimateTestWindowSize

1: **Entrée : A :** l'article en cours de consultation, **WSM :** Vitesse de lecture moyenne d'un mot par minute
2: **Sortie : Twt :** la taille de la fenêtre de test en minutes
3: WordsCount ← calculer le nombre total des mots dans tout l'article A
4: Twt ← WordsCount/ WSM
5: **if** Twt==0 **then**
6: Twt ← 1
7: **end if**

métrique F1 sous la variante implicite, alors qu'il ne montre que **88%** de pertinence en ce qui concerne la même métrique sous la variante explicite. La raison de ce changement, c'est que dans l'approche explicite c'est à nous de fixer empiriquement le temps d'évaluation, ce qui reste toujours approximatif. En revanche, la variante implicite essaie d'optimiser ce paramètre au contexte de chaque demande de recommandation.

Métrique / Algorithme	**F1**	**Map**	**NDCG**	**Click through rate**	**MRR**
Recently clicked	0,8981774	0,8865803	0,921369	0,7074862	0,9448992
random	0,0025201	0,0016305	0,0027172	0,0027548	0,0077851
Most Popular	0,0433279	0,0358296	0,0456856	0,0840041	0,0588317
Recently Popular	0,1479606	0,1303051	0,1567856	0,268782	0,1627811
CoOccurrence	0,8020716	0,7914056	0,8223929	0,6219272	0,8573302

Tableau III.4 – Résultats comparatifs en ligne - Variante implicite avec N=10

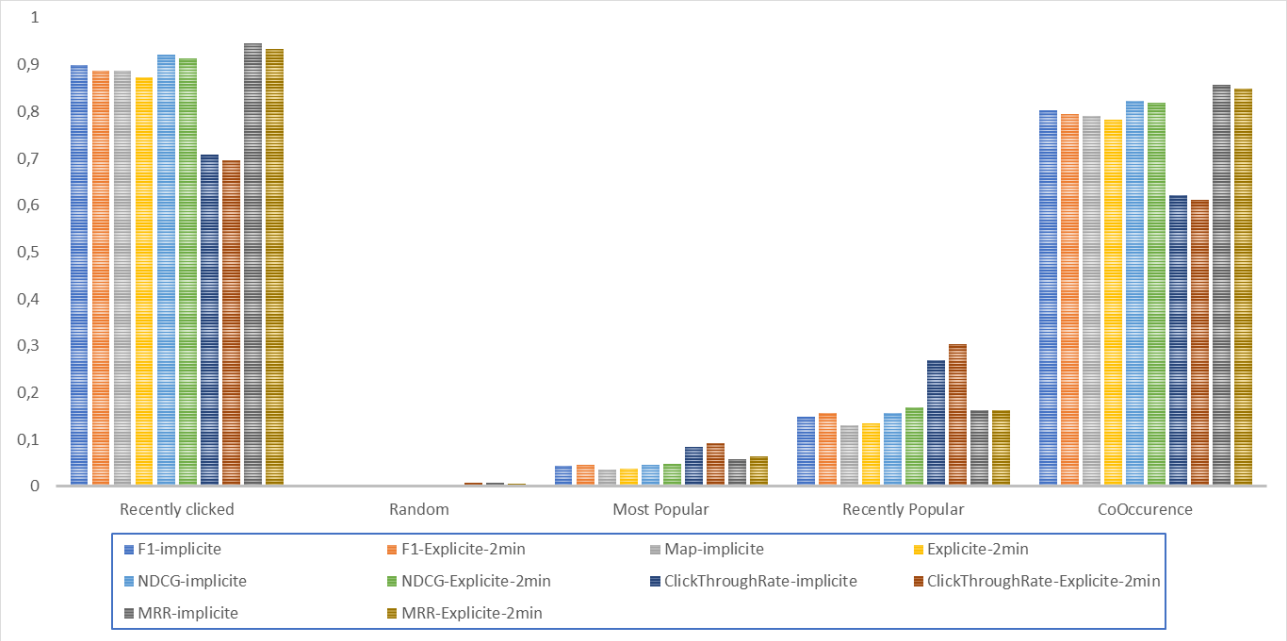

Figure III.4 – Résultats comparatifs en ligne-variante implicite vs variante explicite twt=2min avec N=10

III.4 Résultats comparatifs de l'évaluation hors ligne et en ligne

Les systèmes de recommandation sont souvent évalués dans des environnements statiques et hors ligne pour sélectionner les algorithmes les plus pertinents à déployer des environnements opérationnels en ligne. Notre proposition démarre du fait que ce protocole hors ligne mène à une évaluation irréaliste, ce qui rend les résultats d'évaluation indéterministes. Autrement dit, ces résultats des algorithmes candidats ne suffisent pas à prédire le comportement de ces algorithmes dans des environnements opérationnels en ligne. En revanche, notre proposition aborde l'évaluation d'un point de vue dynamique qui conserve l'aspect réaliste des données en flux. Ceci permet d'avoir une meilleure vision sur la pertinence des algorithmes à évaluer.

En comparant les résultats des expérimentations menées sur les deux cadres d'évaluation (cf. le tableau III.5 et III.6 et la figure III.5), nous pouvons clairement nous apercevoir que les algorithmes les plus précis dans le cadre hors-ligne ne sont pas toujours les meilleurs algorithmes à adopter dans un cadre en ligne. Les résultats de l'algorithme « **Recently clicked** » sont l'exemple idéal pour montrer cette grande différence. En effet,

selon le cadre d'évaluation hors ligne, l'algorithme « **Recently clicked** » est à écarter des solutions envisageables, car il génère des recommandations médiocres dont la précision ne dépasse pas les **0,16%**, au point d'être comparable à des recommandations aléatoires. En revanche, selon le cadre en ligne, l'algorithme « **Recently clicked** » est la meilleure solution à envisager, car il présente les meilleurs résultats, avec un taux de clic qui dépasse les **71%**, et une précision dans les alentours des **90%**. Ceci risque donc à induire en erreur les entreprises dans leur choix des meilleures solutions à déployer dans leurs plateformes de recommandation en ligne.

Protocole / Algorithme	**En ligne**			**Hors ligne**
	2min	**5min**	**10 min**	\
Recently clicked	0,8786909	0,850433	0,837638	0,0022563
Random	0,0022787	0,0026319	0,0029167	0,0015528
most popular	0,0437936	0,0470948	0,049405	0,0438595
Recently Popular	0,1510531	0,1594797	0,1627451	0,0067601
CoOccurrence	0,7961388	0,7800894	0,7730991	0,8212946

Tableau III.5 – Résultats comparatifs de l'évaluation hors ligne et en ligne avec métrique d'évaluation F1 et N=5

Protocole / Algorithme	**En ligne**			**Hors ligne**
	2min	**5min**	**10 min**	
Recently clicked	0,9555808	0,9474001	0,9440739	0,0063543
Random	0,0052405	0,0078527	0,0104511	0,0029992
most popular	0,0630383	0,0719201	0,0789575	0,0612915
Recently Popular	0,1786305	0,1859865	0,1891626	0,0098081
CoOccurrence	0,86403	0,8561218	0,8515364	0,9420219

Tableau III.6 – Résultats comparatifs de l'évaluation hors ligne et en ligne avec métrique d'évaluation MRR et N=5

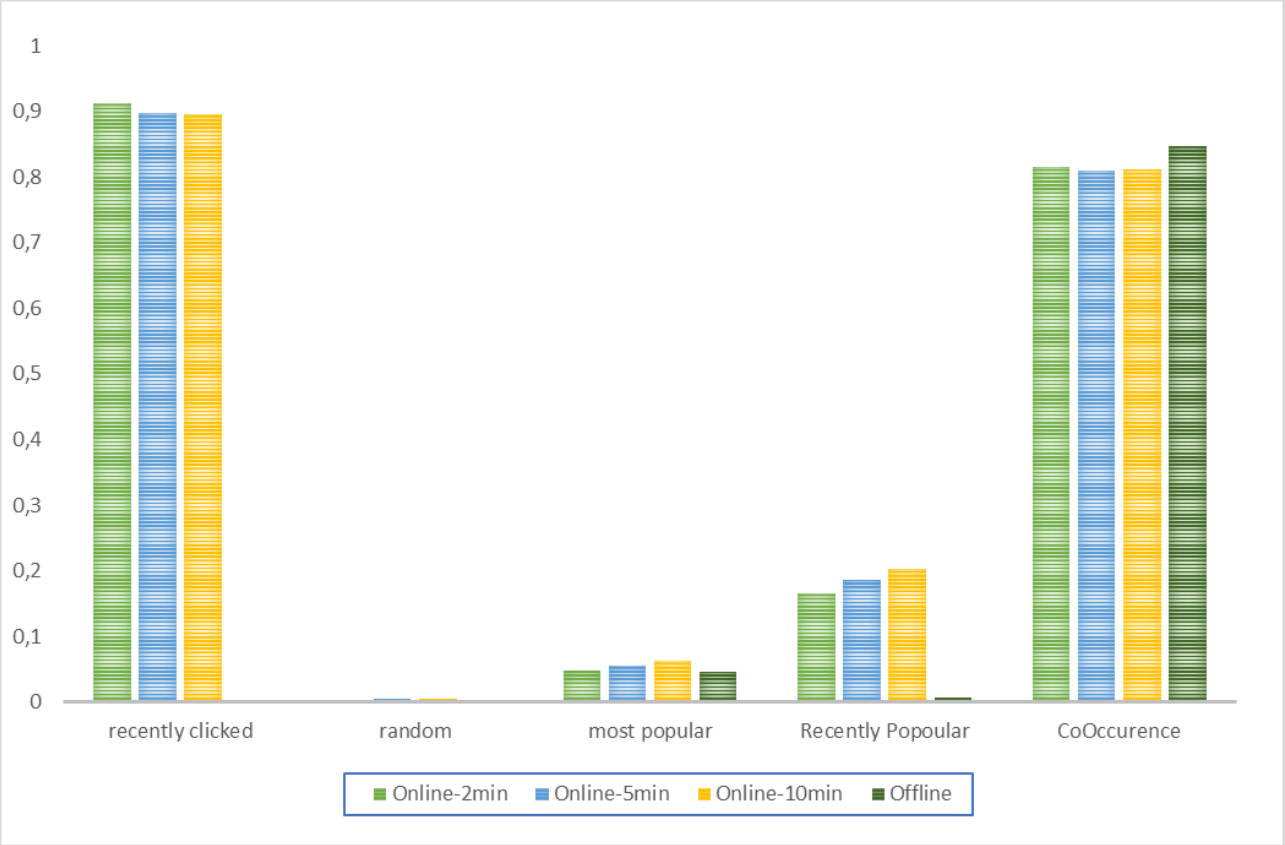

Figure III.5 – Résultats comparatifs de l'évaluation hors ligne et en ligne avec métrique d'évaluation NDCG et N=15

Conclusion

Dans ce chapitre, nous avons comparé les résultats issus des différentes expérimentations que nous avons menées pour valider notre proposition. Les résultats obtenus sur les cadres d'évaluation hors ligne et en ligne ont permis de conclure que les résultats hors-ligne ne permettent pas d'indiquer les résultats en ligne. Ceci risque donc à induire en erreur les entreprises dans leur choix des meilleures solutions à déployer dans leurs plateformes de recommandation en ligne. Les directives que nous avons adoptées dans notre proposition permettent d'assurer une évaluation réaliste et plus déterministe.

Conclusion générale et perspectives

Les systèmes de recommandation sont devenus indispensables dans de nombreuses plateformes du Web actuel. Ces systèmes sont conçus pour aider les utilisateurs à trouver des ressources pertinentes parmi une large sélection de ressources. Ainsi, ils ont attiré beaucoup d'attention ces dernières années, surtout de la part des chercheurs.

Pour s'assurer du bon fonctionnement des systèmes de recommandation, les travaux existants suivent généralement deux types de protocoles d'évaluation. En effet, les entreprises évaluent en permanence la qualité des prédictions de leurs systèmes sur des utilisateurs réels dans des plateformes en production. Le protocole adopté dans ce cadre d'évaluation consiste à exposer, de manière aléatoire, en temps réel, deux variantes (ou plus) d'un système à différents groupes d'utilisateurs. Par la suite, les performances des deux systèmes sont comparées pour optimiser plusieurs objectifs (précision, clics, vues, achats, etc.). Dans le domaine académique, les universitaires manquent d'accès à de tels systèmes opérationnels à grande échelle pour évaluer leurs propositions. Comme alternative, la communauté académique collecte et standardise des jeux de données statiques, et adapte un cadre hors-ligne pour évaluer leurs systèmes de recommandation.

L'évaluation des systèmes de recommandation sur des environnements en production semble être la solution la plus réaliste pour mesurer la pertinence de ces systèmes. Cependant, le caractère aléatoire de ce cadre et l'accès limité à de telles plateformes en production en ligne ont fait que les universitaires se limitent à des évaluations hors ligne sur des ensembles de données. En contrepartie, le protocole hors ligne fournit un modèle imprécis qui omet plusieurs propriétés importantes des données, ce qui fait que les algorithmes les plus pertinents dans ce cadre ne sont pas toujours les meilleurs algorithmes à adopter dans les plateformes réelles. Ceci nous a menés à la présentation de nos directives vers un nouveau cadre d'évaluation plus réaliste.

Notre proposition essaie de redéfinir les principes du protocole hors ligne pour simuler le plus fidèlement possible le scénario de recommandation en ligne. Ainsi, notre configuration basée sur une fenêtre d'apprentissage en croissance continue. Cette fenêtre temporelle commence au début du flux de messages à distribuer, et elle continuera à s'agrandir jusqu'à l'instant t-user, qui marque le début d'une requête de recommandation. En revanche, l'ensemble de tests est représenté par une fenêtre coulissante, d'une courte durée, qui commence à partir de l'instant t-user jusqu'au M minute qui suivent la requête de recommandations. La taille de cette fenêtre représente la période durant laquelle l'utilisateur consulte et juge la pertinence de l'item recommandé.

Nos expérimentations menées sur les deux protocoles et sur plusieurs algorithmes ont confirmé que le protocole hors ligne mène à une évaluation irréaliste, ce qui rend les résultats d'évaluation indéterministes. Autrement dit, ces résultats des algorithmes candidats ne suffisent pas à prédire le comportement de ces algorithmes dans des environnements opérationnels en ligne. En revanche, ces résultats ont montré que notre proposition aborde l'évaluation d'un point de vue dynamique qui conserve l'aspect réaliste des données en flux. Ceci permet d'avoir une meilleure vision sur la pertinence des algorithmes à évaluer.

Plusieurs améliorations peuvent être reflétées à l'avenir pour notre proposition. Premièrement, nous devons élargir la liste des algorithmes testés pour bien analyser les résultats d'évaluation. De plus, nous devons penser à une meilleure solution pour le choix de la taille de la fenêtre de test qui ne se limite pas à la durée de lecture de l'article en cours.

Bibliographie

[1] Douglas W Oard, Jinmook Kim, et al. Implicit feedback for recommender systems. In *Proceedings of the AAAI workshop on recommender systems*, pages 81–83. (1998). 6, 7

[2] Roza Lemdani. *Système hybride d'adaptation dans les systèmes de recommandation*. Theses Université Paris Saclay (COmUE) (2016). 6, 7, 8, 11

[3] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B Kantor. *Recommender Systems Handbook*. Springer New York, NY, 1 edition (2011). 8, 14

[4] Hemza Ficel. *Une approche de recommandation pour les plateformes de consommation en ligne hautement interactives*. Thèse de Doctorat, (2020). 8, 9, 10, 11, 13, 14, 19, 38

[5] Charif Alchiekh Haydar. *Les systèmes de recommandation à base de confiance*. Theses Université de Lorraine (2014). 8

[6] Michael J. Pazzani. *A framework for collaborative, content-based and demographic filtering*. ARTIFICIAL INTELLIGENCE REVIEW **13**, 393–408 (1999). 9

[7] Gregory D Abowd, Anind K Dey, Peter J Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a better understanding of context and context-awareness. In *International symposium on handheld and ubiquitous computing*, pages 304–307. Springer (1999). 10

[8] Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Recommender Systems Handbook*, pages 191–226. Springer US (2015). 10

[9] Robin Burke. Hybrid web recommender systems. In *The Adaptive Web*, pages 377–408. Springer Berlin Heidelberg (2007). 11

[10] Sonia Ben Ticha. *Recommandation personnalisée hybride.* Theses Université de Lorraine (2015). 13

[11] Manisha Hiralall and Wojtek Kowalczyk. *Recommender systems for e-shops.* Business Mathematics and Informatics paper (2011). 13

[12] Carlos A Gomez-Uribe and Neil Hunt. *The netflix recommender system : Algorithms, business value, and innovation.* ACM Transactions on Management Information Systems (TMIS) **6**(4), 1–19 (2015). 14

[13] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. Dkn : Deep knowledge-aware network for news recommendation. In *Proceedings of the 2018 world wide web conference*, pages 1835–1844 (2018). 14

[14] Zaiqiao Meng, Richard McCreadie, Craig Macdonald, and Iadh Ounis. Exploring data splitting strategies for the evaluation of recommendation models. In *Fourteenth ACM conference on recommender systems*, pages 681–686 (2020). 14, 18

[15] Yitong Ji, Aixin Sun, Jie Zhang, and Chenliang Li. *A critical study on data leakage in recommender system offline evaluation.* arXiv preprint arXiv :2010.11060 (2020). 14, 16, 19

[16] Hemza Ficel, Mohamed Ramzi Haddad, and Hajer Baazaoui Zghal. *A graph-based recommendation approach for highly interactive platforms.* Expert Syst. Appl. **185**(C) dec (2021). 18, 24, 28

[17] Andrii Maksai, Florent Garcin, and Boi Faltings. *Predicting online performance of news recommender systems through richer evaluation metrics.* Proceedings of the 9th ACM Conference on Recommender Systems (2015). 19

[18] Jeonghee Yi, Ye Chen, Jie Li, Swaraj Sett, and Tak W Yan. Predictive model performance : Offline and online evaluations. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1294–1302 (2013). 19

[19] Guy Shani and Asela Gunawardana. Evaluating recommendation systems. In *Recommender systems handbook*, pages 257–297. Springer (2011). 19

[20] Nima Taghipour, Ahmad Kardan, and Saeed Shiry Ghidary. Usage-based web recommendations : a reinforcement learning approach. In *Proceedings of the 2007 ACM conference on Recommender systems*, pages 113–120 (2007). 19

[21] Olivier Jeunen, Koen Verstrepen, and Bart Goethals. Fair offline evaluation methodologies for implicit-feedback recommender systems with mnar data. In *Proceedings of the REVEAL 18 Workshop on Offine Evaluation, October 2018, Vancouver, Canada* (2018). 23

[22] Tefko Saracevic. *The notion of relevance in information science : Everybody knows what relevance is. but, what is it really ?* Synthesis lectures on information concepts, retrieval, and services **8**(3), i–109 (2016). 25

[23] Benjamin Kille, Frank Hopfgartner, Torben Brodt, and Tobias Heintz. The plista dataset. In *Proceedings of the 2013 international news recommender systems workshop and challenge*, pages 16–23 (2013). 35

[24] Nick Craswell. Mean reciprocal rank. In *Encyclopedia of Database Systems*, pages 1703–1703. Springer U (2009). 38

[25] Kalervo Järvelin and Jaana Kekäläinen. *Cumulated gain-based evaluation of ir techniques.* ACM Transactions on Information Systems (TOIS) **20**(4), 422–446 (2002). 38

Annexe

Pour mener à bien nos expérimentations, nous avons développé une application de bureau qui implémente notre cadre d'évaluation ainsi que le cadre d'évaluation hors ligne.

.1 Outils logiciels et matériels

L'application développée est basée sur les technologies suivantes :

— **Java** est un langage de programmation orienté objet, basé sur des classes. Il est considéré comme l'un des langages de programmation les plus utilisés par la communauté.

— **Javafx** est un framework issu du projet OpenJFX, qui permet aux développeurs Java de créer des interfaces graphiques plus riches.

— **SceneBuilder** est un outil de conception visuelle qui facilite la création et la gestion des interfaces graphiques pour JavaFX.

— **IntelliJ IDEA** est un environnement de développement intégré (IDE) qui supporte le langage de programmation Java.

Notre application a été déployée sur une sur une machine dotée de la configuration suivante :

— **Processeur :** Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz 1.99GHz.

— **Mémoire installée (RAM) :** 8,00Go.

— **Système d'exploitation :** Windows 10 professionnel 64 bits

.2 Description de l'application

Notre application est composée des fenêtres suivantes :

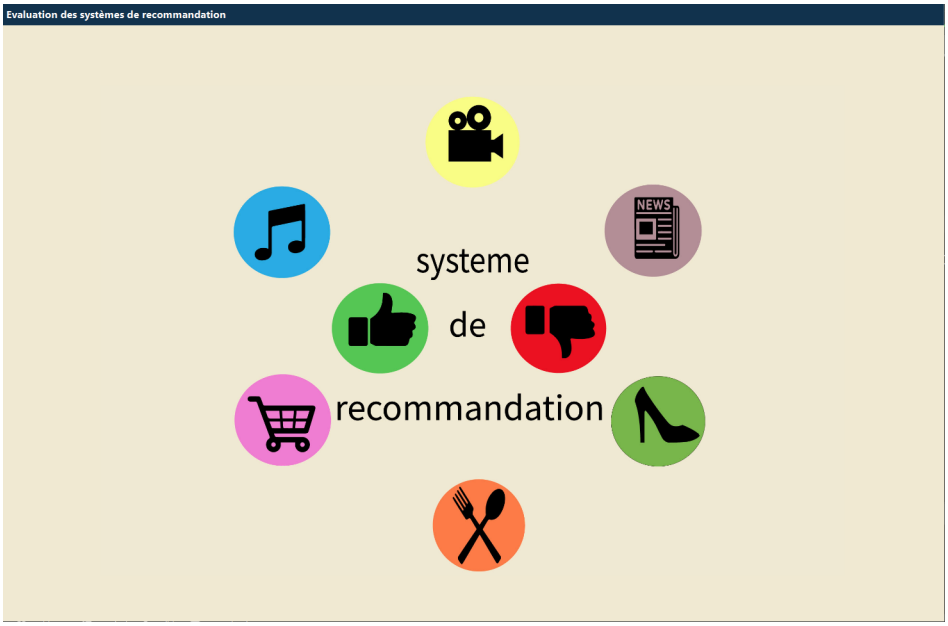

Figure A.1 – Page de chargement

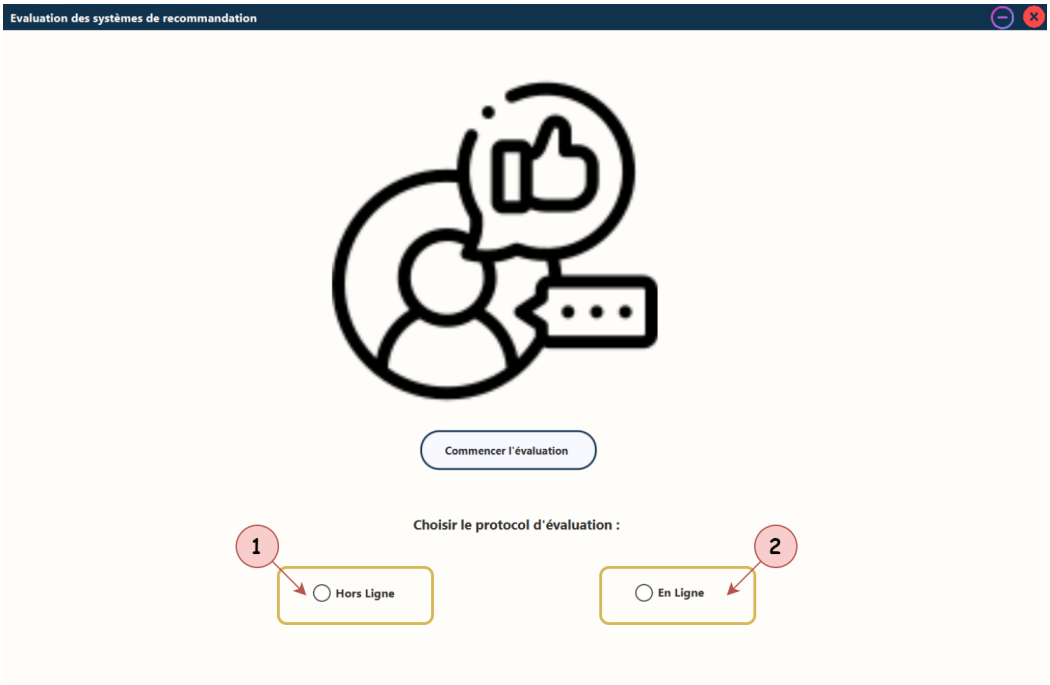

Figure A.2 – Page d'accueil

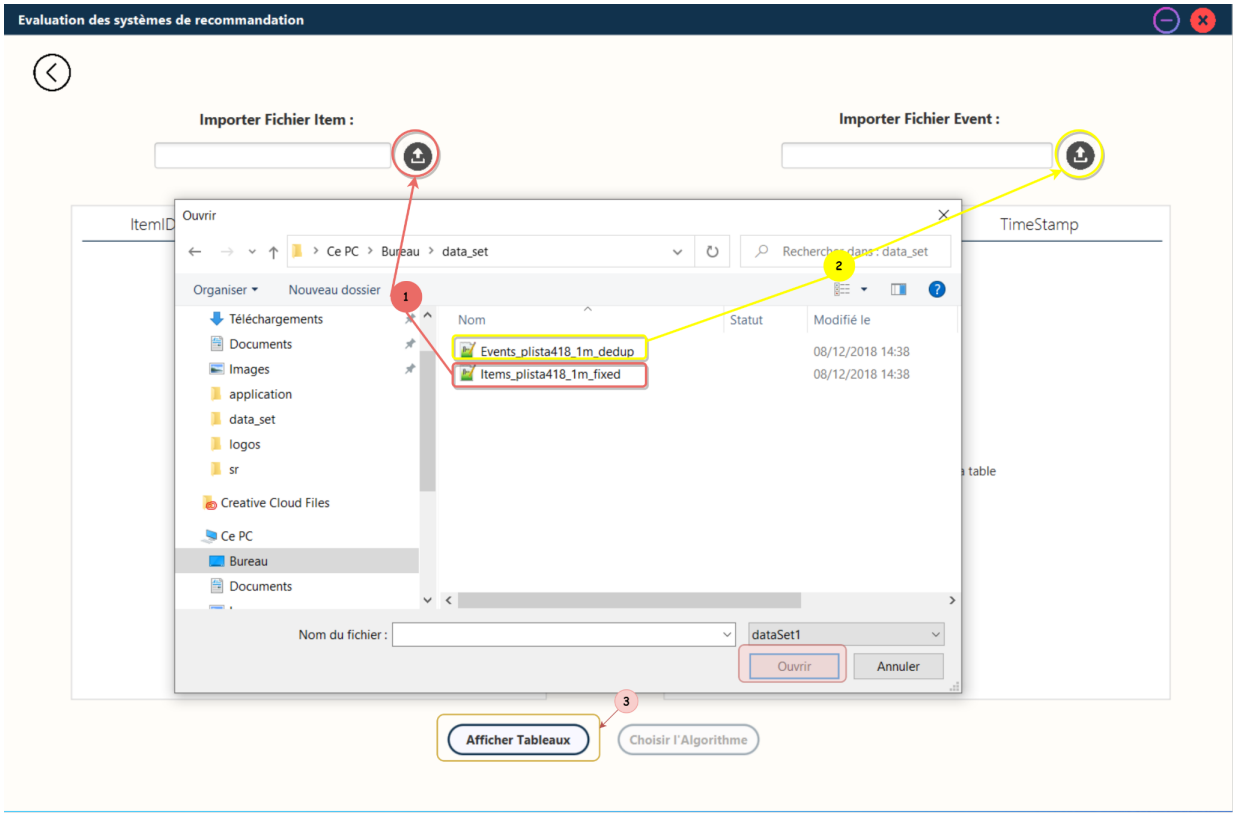

Figure A.3 – Page pour importer les jeux de données

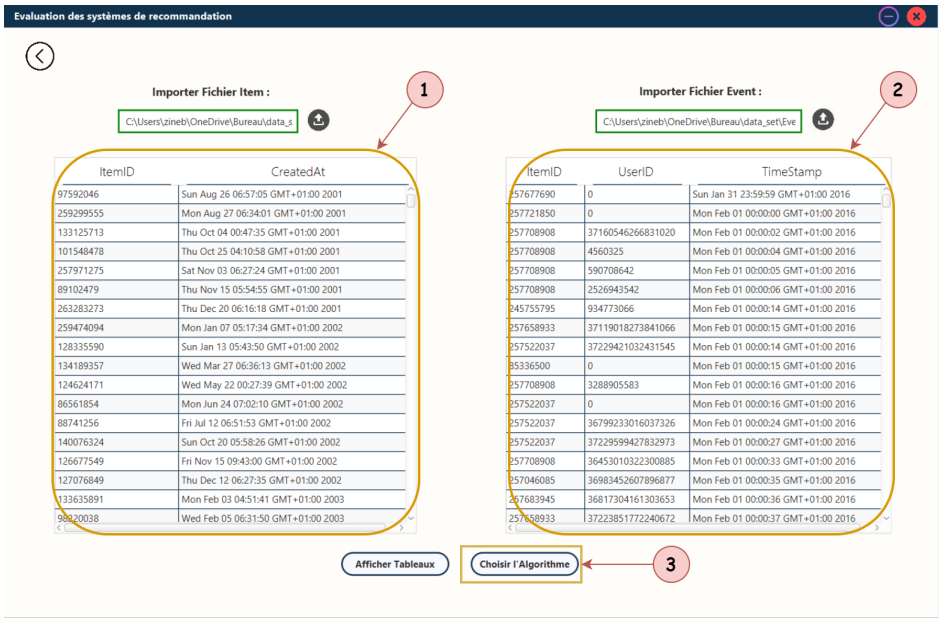

Figure A.4 – Page pour importer les jeux de données V2

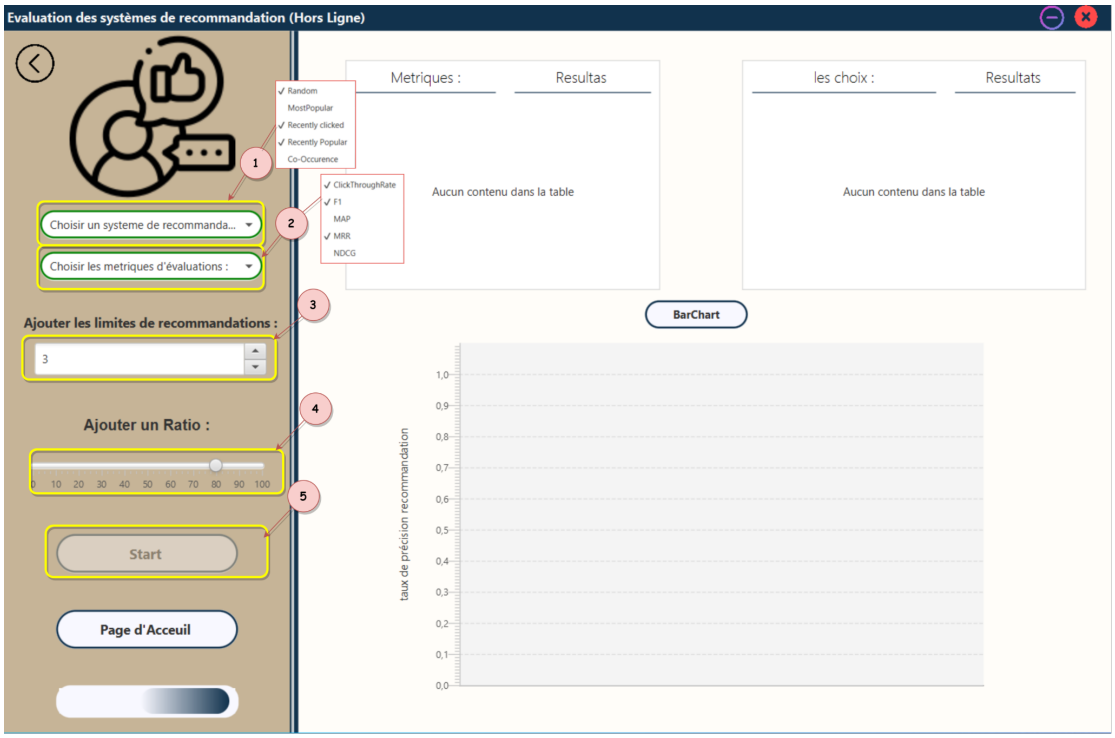

Figure A.5 – Page du protocole hors ligne

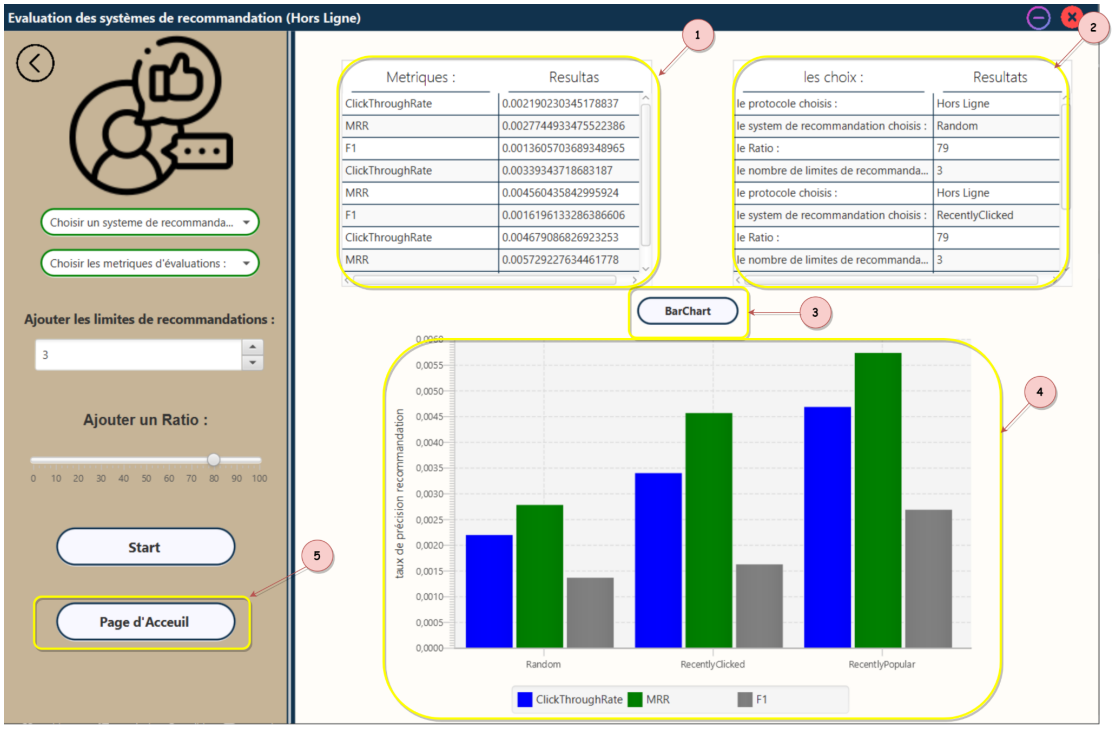

Figure A.6 – Résultats d'évaluation du protocole hors ligne

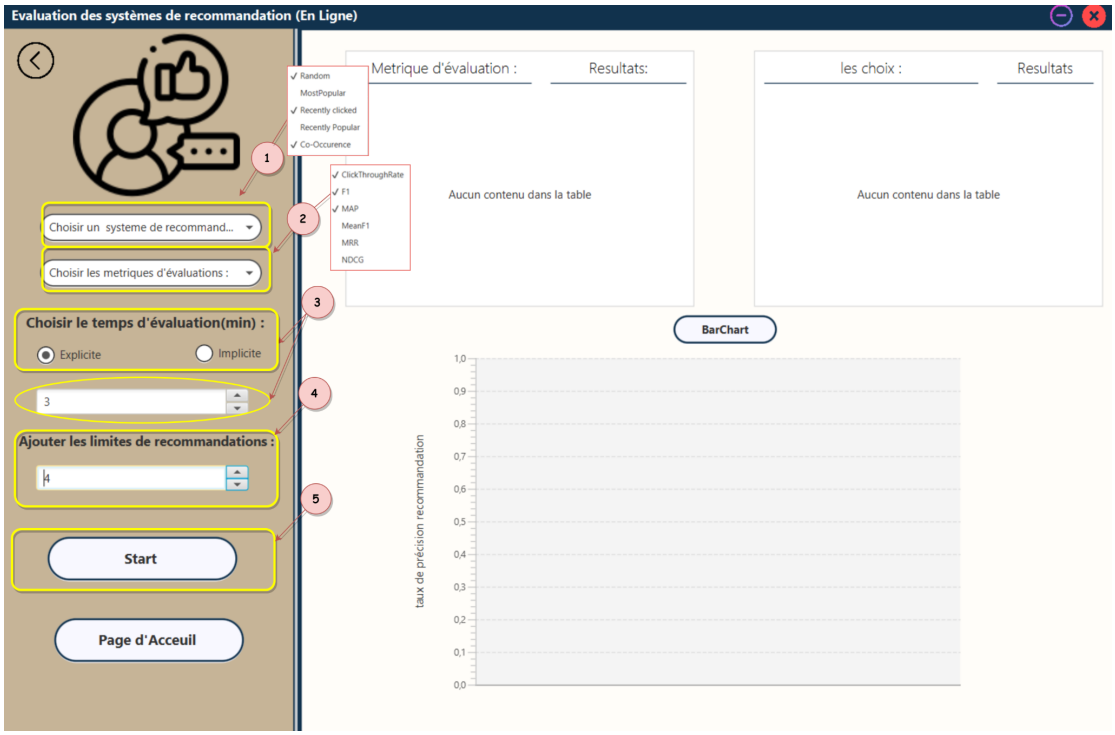

Figure A.7 – Page du protocole en ligne (méthode explicite)

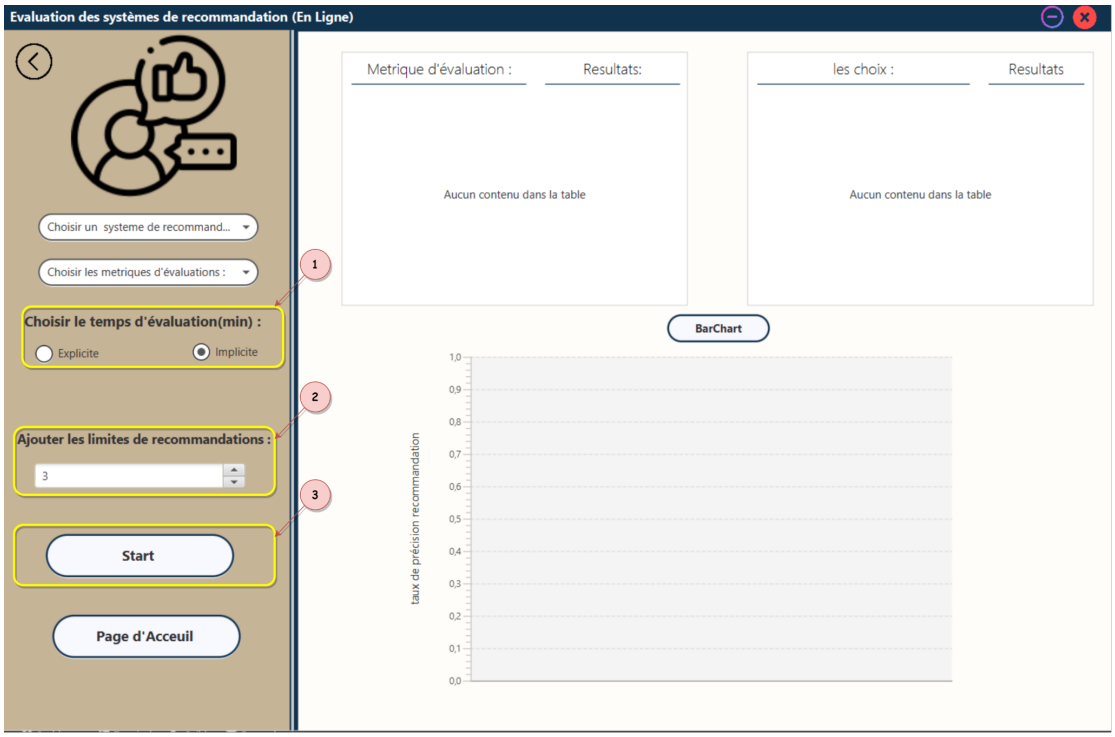

Figure A.8 – Page du protocole en ligne (méthode implicite)

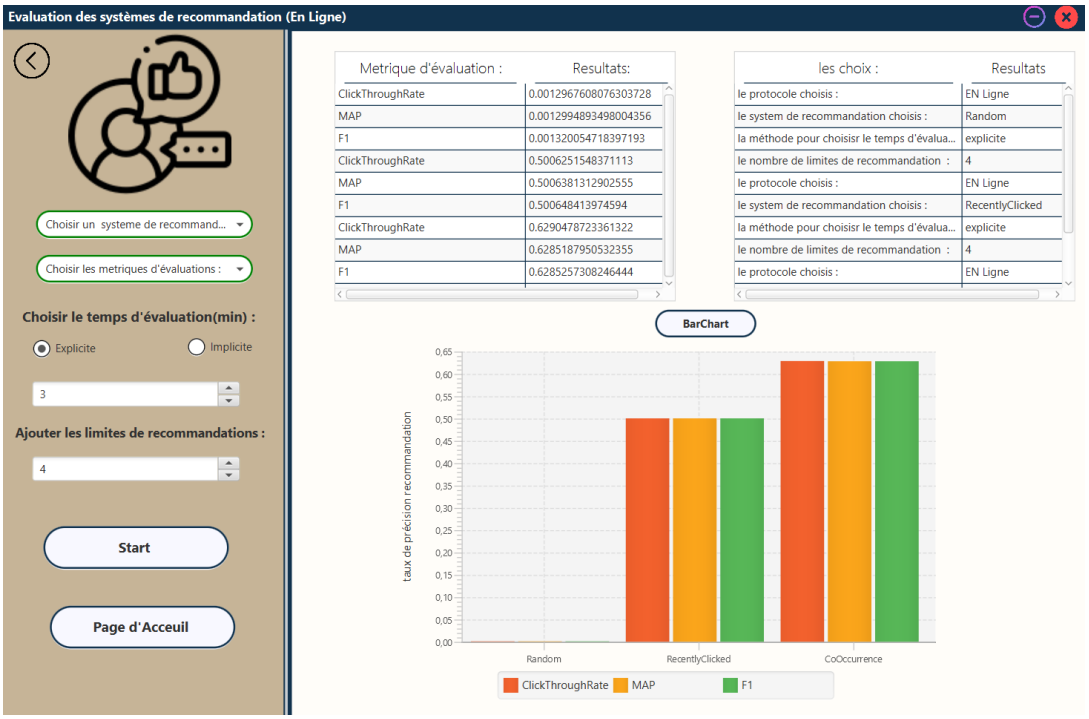

Figure A.9 – Résultats d'évaluation du protocole en ligne