République Algérienne Démocratique et Populaire
Ministère de l'enseignement Supérieur et de la Recherche Scientifique
Université de Mohamed El Bachir El Ibrahimi de Bordj Bou Arréridj
Faculté des Mathématiques et d'Informatique
Département d'informatique

MEMOIRE

Présenté en vue de l'obtention du diplôme

Master en informatique

Spécialité : Technologie de l'information et de la communication(TIC)

THEME

Fouille d'épisodes à partir de séquences d'événements : Application à l'analyse de l'historique des visites de pages web

Présenté par :

HEBBOUL AHLEM

MEKHOUKH LAMIS

Soutenu publiquement le : 19/06/2024

Devant le jury composé de :

Président : Dr. Attia Abdelouhab

Examinateur : Saha Adel

Encadrant : Pr. Nouioua Farid

Invité : Dr. Ouarem Oualid

2023/2024

Dédicace

Je dédie ce mémoire à ma famille, dont l'amour inconditionnel, le soutien constant et les encouragements ont été ma force motrice tout au long de ce parcours académique.

À mes parents, pour leur sagesse, leurs sacrifices et leur foi inébranlable en moi. Votre soutien et vos encouragements m'ont donné la force de surmonter tous les obstacles.

À ma chère soeur ASMA, dont l'amour et le soutien signifient beaucoup pour moi. Que cette dédicace soit un témoignage du lien spécial que nous partageons et des souvenirs que nous avons créés ensemble. Je suis reconnaissante de ta présence dans ma vie et de la joie que tu m'apportes chaque jour.

À mes professeurs[Dr NOUIOUA FARID , OUAREM OUALID], mentor inspirant et guide précieux, dont la sagesse et le dévouement ont façonné ma vision et enrichi mon parcours académique, ce mémoire est dédié avec une profonde gratitude.

Je tiens à consacrer une mention spéciale à mon ami LAMIS, qui a été un soutien inestimable tout au long de la réalisation de ce mémoire. Sa présence, son écoute attentive et ses encouragements ont été des piliers essentiels dans les moments de doute et de difficulté. Sa chaleur humaine et son amitié sincère ont rendu ce parcours académique plus enrichissant et plus significatif. Je suis reconnaissante d'avoir un ami aussi exceptionnel à mes côtés, et je tiens à lui exprimer toute ma gratitude pour sa confiance, son soutien et sa présence constante. Merci, LAMIS, pour ta précieuse amitié et ton soutien indéfectible..

- Ahlem -

Dédicace

Je tiens à exprimer ma plus profonde gratitude à ceux qui ont contribué à la réalisation de ce mémoire.

Tout d'abord, je souhaite dédier ce travail à ma famille, qui a toujours été une source inépuisable de soutien et d'encouragement. À mes parents, merci pour votre amour inconditionnel, votre sacrifices. Votre confiance en moi m'a donné la force de surmonter tous les obstacles et d'atteindre mes objectifs.

À mes sœurs [roumaissa, chaima, arwa, malak], merci pour votre soutien constant.

À mon fiancé, Halim, je te remercie pour ton soutien et ton encouragement tout au long de ce parcours et merci pour ton amour.

Un salut spécial à mon ami de toujours Ahlam. Merci pour votre amitié et votre compréhension. Je chéris les moments que nous avons partagés et vécus tout au long de notre parcours universitaire. Vous avez été cette amie et cette sœur encourageante dans les moments de désespoir et de déception. Merci. Et sans oublier votre aide précieuse, votre coopération et votre esprit d'équipe que vous appréciez. Votre soutien et votre amitié ont rendu ce voyage beaucoup plus facile.

Un autre salut à mes amis, Khawla, Samia, meriem et Nihad.

Je tiens également à exprimer ma gratitude à mon professeur superviseur, [Dr NOUIOUA FARID , OUAREM OUALID]. Merci pour votre encadrement, vos conseils avisés et votre disponibilité tout au long de ce projet. Votre expertise et votre bienveillance ont été des éléments clés dans l'aboutissement de ce mémoire. Votre soutien académique et moral a été indispensable, et je vous en suis profondément reconnaissante.

- Lamis -

Remerciement

Nous tenons à exprimer notre plus sincère gratitude à tous ceux qui ont contribué à la réalisation de ce travail de mémoire.

Tout d'abord, nous voudrions remercier notre directeur de mémoire, [Dr NOUIOUA FARID], pour son soutien, ses conseils et son encouragement tout au long de ce projet. Son expertise et ses retours constructifs nous ont été d'une grande aide pour mener à bien cette recherche.

Nous tenons à exprimer notre profonde gratitude au Professeur[OUAREM OUALID,] pour son aide inestimable et son soutien constant tout au long de ce projet. Son expertise, ses conseils avisés et son encouragement continu ont été essentiels à l'aboutissement de ce travail. Merci pour votre patience, votre disponibilité et votre volonté de partager vos idées avec nous. Nous sommes immensément reconnaissants pour tout ce que vous avez fait.

Nous remercions le Dr Attia Abdelouhab et Saha Adel pour le temps qu'ils ont consacré à discuter de notre humble travail.

Enfin, nous remercions notre famille et nos proches pour leur amour, leur soutien et leur compréhension.

Résumé

Dans un environnement numérique en constante évolution, l'analyse des parcours de navigation des utilisateurs sur les sites web représente un enjeu majeur pour les entreprises souhaitant optimiser leur stratégie digitale. Ce mémoire explore l'application de l'algorithme EMDO (Episode Mining under Distinct Occurrence) pour prédire le comportement des utilisateurs sur le Web en analysant leurs séquences de visites de pages web. L'algorithme propose une approche innovante pour extraire des règles d'épisodes basées sur les occurrences distinctes, améliorant ainsi la précision des prédictions.

Mots clés : EMDO, occurrences distinctes, visite de page web, fouille d'épisodes, règles d'épisode, fouille de données.

Abstract

In a constantly evolving digital environment, the analysis of user navigation on websites represents a major challenge for companies wishing to optimize their digital strategy. This dissertation explores the application of the EMDO (Episode Mining under Distinct Occurrence) algorithm to predict user behavior on the Web by analyzing their web page visit sequences. The algorithm offers an innovative approach to extracting episode-based rules based on distinct occurrences, thus improving the accuracy of predictions.

Keywords : EMDO, distinct occurrences, web page visit, episode mining, episode rules, data mining.

ملخص

في ظل البيئة الرقمية المتطورة باستمرار، يمثّل تحليل مسارات تصفّح المستخدمين لمواقع الويب قضية رئيسية بالنسبة للشركات التي ترغب في تحسين استراتيجيتها الرقمية. يستكشف هذا المشروع تطبيق خوارزمية ٱ ح و ج دْ (استخراج الحلقات تحت وجود متميز) لتوقع سلوك المستخدمين على الويب من خلال تحليل تسلسل زيارتهم لصفحات الويب. تقترح هذه الخوارزمية نهجاًا مبتكراً لاستخراج قواعد الحلقات استناداً إلى الوجود المتميز، مما يحسن دقة التنبؤات.

الكلمات المفتاحية: ٱ ح و ج دْ، الوجود المتميز، زيارة صفحة الويب، استخراج الحلقات، قواعد الحلقات، تنقيب البيانات.

Table des matières

Liste des abréviations

IA	Intelligence Artificiel
ECD	Extraction de Connaissance à partir de Données
FP-Growth	Mining Frequent patterns
Conf	Confiance
MINSUP	Support Minimum
MINCONF	Confiance Minimale
EMDO	Episode Mining under Distinct Occurrence
LMSE	Laboratoire des Matériaux et des Systèmes Électroniques
SPMF	Sequential Pattern Mining Framework
NUMPY	Numerical Python
PANDAS	Python Data Analysis library
PYCHARM	Python charme
MINEPI	MINimal occurrance-based EPIsode miner
WINEPI	WINdow-based EPIsope mining
NONEPI	Non-overlapping episode rule miner
TID	Transaction Identifier
CSV	Comma Separeted Values
BDD	Base de données

Table des figures

Liste des tableaux

List of Algorithms

Introduction générale

La fouille de données (en Anglais : Data Mining) est une discipline qui consiste à extraire des connaissances utiles à partir de gros volumes de données brutes. La fouille des épisodes à partir de séquences d'événements est une branche clé de la fouille de données. Notre mémoire se situe dans ce domaine et vise à appliquer les techniques récentes de la fouille d'épisodes pour l'analyse de l'historique de navigation d'utilisateurs sur le Web.

Contexte

La fouille d'épisodes constitue une branche éminente de la fouille de données axée sur l'analyse approfondie des séquences temporelles. Les techniques de la fouille d'épisodes ont été utilisées dans plusieurs applications réelles telles que l'analyse des alarmes dans un réseaux de télécommunication, la prédiction des actions d'achat et de vente à partir de l'historique des prix des stocks, la détection et la prévention des attaques malicieuse dans les réseaux et plein d'autre applications dans différents domaines. En particulier, l'une des applications potentielles de la fouille d'épisodes est l'analyse de l'historique des visites des pages Web par les utilisateurs. Elle permet d'offrir la possibilité de comprendre le comportement des utilisateurs, notamment sur les sites de la vente en ligne. Cette dernière tâche devient essentiel pour les entreprises et les professionnels du Web afin de réaliser des moteurs de recommandation qui contribuent à l'augmentation de leurs chiffres d'affaires. Dans un monde où les interactions en ligne sont omniprésentes, comprendre le comportement des utilisateurs à travers leurs visites sur le réseau internet devient crucial aussi bien pour les entreprises que pour les professionnels du Web.

Objectifs

Ce mémoire vise à utiliser appliquer un algorithme de la fouille des épisodes pour la prédiction des comportements des utilisateurs sur le Web à partir d'une séquence des visites des pages web. Pour cela nous utilisons l'algorithme EMDO (**E**pisode **M**ining under **D**istinct **O**ccurrence) qui propose une nouvelle approche pour l'extraction des règles d'épisodes en se basant sur les occurrences distincts pour le calcul des fréquences des épisodes.

Structure du rapport

En plus d'une introduction générale et d'une conclusion générale, ce mémoire comporte trois chapitres organisés comme suit :

Dans le premier chapitre, nous nous concentrons sur les concepts généraux de la fouille de données et plus généralement, sur l'extraction de connaissances à partir des données. Nous abordons les domaines d'application de cette approche. Nous présentons ensuite la fouille de motifs en examinant ses diverses formes (fouille de motifs fréquents, fouille de motifs périodiques, fouille d'épisodes, etc). Enfin, nous nous focalisons sur la fouille d'épisodes en examinant les différentes définitions de fréquences d'épisodes ainsi que certains algorithmes existants de fouille d'épisodes.

Dans le deuxième chapitre, nous exposons les définitions des notions de base en fouille d'épisodes : événement, séquence, épisode, fréquence, etc. Nous présentons ensuite un algorithme récent de fouille d'épisodes fréquents, baptisé EMDO, qui se base sur les occurrences distinctes des événements plutôt que sur leur fréquence totale [1]. Nous détaillons les différentes étapes de cet algorithme pour extraire efficacement les épisodes fréquents et les règles d'épisodes.

Dans le troisième chapitre, nous mettons l'accent sur la dimension pratique de notre mémoire. Nous présentons la configuration matérielle et logicielle nécessaires pour mettre en oeuvre l'algorithme EMDO. Nous décrivons l'implémentation de cet algorithme ainsi que les expérimentations réalisées sur des jeux de données réels représentant l'historique de navigation web. Nous analysons les performances de l'algorithme en termes de temps d'exécution et de pertinence des résultats obtenus. Cette analyse permettra de démontrer l'efficacité et la robus-

tesse de l'algorithme EMDO dans des scénarios d'application pratique, fournissant ainsi une validation expérimentale de notre approche.

Enfin, le mémoire se termine par une conclusion générale qui dresse un bilan sur les résultats obtenus et indique des perspectives pour des travaux futurs.

Chapitre 1.

La fouille de motifs

Chapitre 1

La fouille de motifs

1.1 Introduction

La fouille de données est une discipline puissante de l'analyse des données. Elle se distingue par sa capacité à extraire des connaissances utiles à partir de données brutes. Elle contribue à la révolution numérique en fournissant un moyen efficace de repérer les tendances, les modèles et les informations.

Ce chapitre offre une introduction générale à la fouille de données. Après une définition assez claire de cette discipline, nous explorons les divers domaines d'application, mettant en lumière son impact dans des secteurs tels que la finance et la santé. Une attention particulière est accordée à l'extraction de connaissances en environnement collaboratif, suivie d'une exploration de la relation entre la fouille de données et l'apprentissage automatique.

Nous abordons ensuite la fouille de motifs, en se concentrant sur la recherche d'itemsets fréquents dans les bases de données transactionnelles. Les concepts de base et les algorithmes d'extraction de règles d'association sont détaillés. Nous allons ensuite citer brièvement les autres types de motifs qu'on peut extraire à partir de diverses formes de données.

1.2 La fouille de donnée (Data Mining)

1.2.1 Definition

La fouille de données est un processus d'exploration et d'analyse de grandes quantités de données pour découvrir des modèles significatifs, des tendances ou des relations cachées. Il s'agit d'un domaine multidisciplinaire qui combine des techniques de statistiques, d'apprentissage automatique, d'intelligence artificielle, et de bases de données pour extraire des informations exploitables à partir de données brutes[2][3].

1.2.2 Extraction des Connaissances à partir des Données (ECD)

L'ECD est le processus complet utilisé pour explorer de grandes quantités de données afin de découvrir les relations implicites dans ces données. Ce processus comporte une succession d'étapes mettant en oeuvre diverses techniques statistiques, d'apprentissage automatique et d'exploration et de visualisation de données pour extraire des informations exploitables à partir de sources de données variées [4]. Ainsi, la fouille de données représente juste une étape dans le processus ECD.

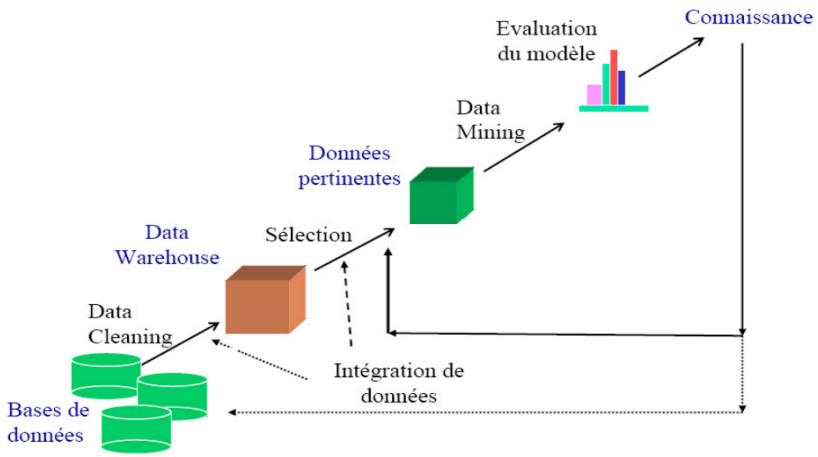

FIGURE 1.1 – Processus de découverte de connaissances à partir de données.

Le processus d'extraction des connaissances à partir de données est illustré dans la Figure 1.1. Il consiste en la succession des étapes suivantes :

1. **Compréhension du problème.**

   La compréhension de problème permet aux algorithmes de produire des résultats fiables,

et permet également de préparer les données pour l'exploration et l'interprétation correcte des résultats[5].

2. **Pré-traitement des données.**

   Les informations recueillies doivent être préparées en premier lieu. Les données doivent être nettoyées car elles peuvent contenir diverses anomalies. Par exemple, les données peuvent être omises en raison d'erreurs de frappe ou des erreurs dues au système, dans ce cas les données doivent être remplacées ou éliminées complètement.

   Le pré-traitement implique également la réduction des données, ce qui permet d'accélérer les calculs et de représenter les données sous un format optimal pour l'exploration.

   Le nettoyage des données, comprend un ensemble de techniques visant à identifier et à corriger les erreurs, les incohérences et les valeurs aberrantes dans les ensembles de données. Parmi les techniques les plus couramment utilisées, on trouve la suppression des valeurs manquantes, la détection et le traitement des valeurs aberrantes, la gestion des doublons, la normalisation des données, la correction des erreurs de saisie, la gestion des valeurs extrêmes et la vérification de la cohérence des données[5].

3. **Intégration des données :**

   Les données peuvent provenir de diverses bases de données, de fichiers et de textes. Ces données recueillies de différentes sources peuvent être regroupées et organisées dans une seule base de données grâce à cette étape.

   Les données comparables sont également nettoyées et codées de manière uniforme dans un stockage de données. Le but des opérations d'intégration et de nettoyage est de créer des stocks de données spécialisés contenant des données modifiées afin de rendre leur traitement plus faciles[5].

4. **Fouille de données (Data mining).**

   C'est l'étape qui constitue véritablement le cœur du processus d'Extraction de Connaissances à partir des Données. C'est ici que l'on va chercher à extraire automatiquement des modèles à partir des données. Cette étape est la plus critique du point de vue algorithmique.Le choix de la meilleure méthode pour résoudre un problème est le cœur de toute tâche de fouille de données. Pour essayer d'obtenir une solution optimale, il est possible de combiner plusieurs méthodes.

   Les méthodes de fouille de donnée qui sont les plus couramment utilisée dans les sys-

témes ECD sont :

- Méthodes de classification et structuration (algorithme de k-means,algorithme du plus proche voisin, etc).

- Méthode d'explication et de prédiction (arbre de decision,réseaux de neurons) [5].

5. **Evaluation et présentation :**

   Cette phase s'intéresse à l'évaluation des modèles obtenus en mesurant l'intérêt des motifs extraits. Elle s'occupe aussi de la présentation des résultats à l'utilisateur grâce à plusieurs techniques de visualisation.

   Cette étape est dépendante de la tâche de la fouille de données employée. En effet, les utilisateurs ne demandent pas des pages et des chiffres, mais des interprétation des modèles obtenus, bien que l'interaction avec l'expert soit importante pour que les motifs soient validés par ces experts du domaine[5].

1.2.3 Apprentissage automatique (Machine Learning)

La fouille de données est largement fondée sur les techniques et algorithmes de l'apprentissage automatique. Celui-ci est un domaine de l'intelligence artificielle (IA) qui consiste en l'utilisation d'algorithmes pour analyser et apprendre des données sans nécessiter une programmation explicite. Les données sont analysées, générées et exploitées à l'aide des techniques d'apprentissage automatique, notamment en traitement automatique du texte, la vision artificielle, etc. Il existe deux principaux types d'apprentissage automatique : Apprentissage supervisé et Apprentissage non supervisé [6]. Chacun de ces types permet de réaliser une multitude de tâches en pratique (voir Figure 1.2)

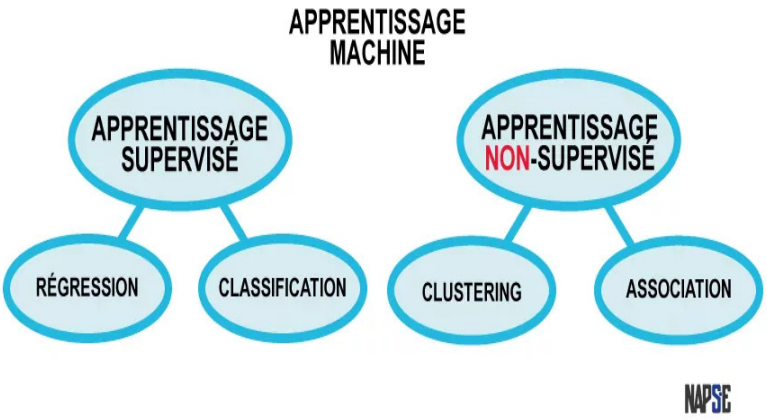

FIGURE 1.2 – les types d'apprentissage automatique.

1.2.3.1 Apprentissage supervisé :

L'apprentissage supervisé est un type d'apprentissage automatique basé sur le principe d'utilisation de données étiquetés (exemples) pour extraire des modèles qui tentent de capter la relation entre les entrée et les sorties (les classes) dans les exemples. Ces modèles, une fois validés, permettent ensuite de classer les nouvelles dont l'étiquete est inconnues. Ceci est utile pour la prédiction de résultats avec précision dans différentes situations pratiques [7][8].

L'apprentissage supervisé peut réaliser deux principaux types de tâche en pratique lors de l'extraction de données, à savoir la classification et la régression :

- **Classification :**

  La classification consiste à trouver un modèle ou une fonction qui distingue les catégories (appelées étiquettes) de classes présentes dans les données, à partir des attributs qui caractérisent ces données. Par exemple, la classification des e-mails comme spam ou non spam, la classification des images comme chats ou chiens, etc.

- **Régression :**

  Contrairement à la classification, qui vise à prédire la classe d'une observation, la régression vise à prédire une valeur numérique. Par exemple, prédire le prix d'une maison en fonction de ses caractéristiques ou prédire la quantité de ventes d'un produit.

Il y a plusieurs algorithmes qui ont été proposés dans la littérature pour l'apprentissage supervisé. Parmi les plus utilisés nous pouvons citer les exemples suivants :

- **Méthode des K plus proches voisins (KNN) :**

  L'algorithme des k plus proches voisins, également connu sous le nom de KNN ou kNN,

est un classificateur d'apprentissage supervisé non paramétrique, qui utilise la proximité pour effectuer des classifications ou des prédictions sur le regroupement d'un point de données individuel. Il est généralement utilisé comme algorithme de classification en partant de l'hypothèse que des points similaires peuvent être trouvés les uns à côtés des autres. Cependant, avant qu'une classification puisse être faite, la distance doit être définie[9].

- **Arbres de décsion :**

  Un arbre de décision est un modèle d'apprentissage automatique utilisé pour prendre des décisions. Il s'agit d'une structure arborescente où chaque noeud représente une décision basée sur une caractéristique et chaque feuille représente le résultat de la décision. Dans le domaine de l'apprentissage automatique, les arbres de décision sont largement utilisés. Ils sont particulièrement populaires en raison de leur simplicité et de leur facilité d'interprétation[10].

- **Classification Bayésienne :**

  La Classification Bayésienne est une approche statistique efficace pour le diagnostic de défauts dans les systèmes automatiques. Cette méthode se base sur le calcul de probabilités a posteriori des différentes classes de défauts, en utilisant le théorème de Bayes. Elle permet de prendre en compte l'incertitude dans les données d'entrée et de s'adapter à des distributions de probabilité complexes. En modélisant la distribution des observations pour chaque classe de défaut, la Classification Bayésienne peut affecter de nouvelles observations à la classe la plus probable, fournissant ainsi une mesure de confiance dans le diagnostic. Cette approche constitue une alternative intéressante aux méthodes de classification traditionnelles pour l'identification de pannes dans les systèmes automatisés[11].

- **Machines à vecteur de support (SVM) :**

  Les Machines à Vecteurs de Support (Support Vector Machines - SVM) sont une famille de techniques d'apprentissage automatique très performantes pour la classification et la régression. Le principe des SVM consiste à trouver l'hyperplan qui sépare au mieux les différentes classes dans l'espace des caractéristiques, en maximisant la marge entre cet hyperplan et les données les plus proches, appelés vecteurs de support. Cette approche permet de gérer efficacement les problèmes de classification linéairement séparables, mais aussi les cas non linéaires en utilisant des fonctions noyaux (kernel) qui projettent

les données dans un espace de plus grande dimension. Les SVM ont montré leur efficacité dans de nombreuses applications, notamment en reconnaissance de formes, en traitement du langage naturel et en bioinformatique. Leur robustesse face au bruit et leur capacité à généraliser à partir d'un nombre limité d'exemples en font des outils de choix pour de nombreuses tâches d'apprentissage supervisé[12].

1.2.3.2 Apprentissage non supervisé :

Contrairement à l'apprentissage supervisé, l'apprentissage non supervisé utilise des données non étiquetées. À partir de ces données, il découvre des modèles qui aident à résoudre les problèmes de regroupement ou d'association[13]. Voici les principales tâches de l'apprentissage non supervisé :

- **Clustering (Segmentation).**

  Le clustering est un processus qui divise automatiquement les données en groupes ou en classes. Un clustering ou un schéma de clustering est l'ensemble de clusters provenant d'un seul processus. Un modèle de clustering est appris à partir de données non étiquetées, contrairement à la technique de classification.

  Il existe différents algorithmes de clustering, chacun ayant ses propres avantages, inconvénients et applications appropriées. Voici quelques-uns des algorithmes de clustering les plus couramment utilisés :

  — **K-Means.** C'est l'un des algorithmes de clustering les plus populaires. Il partitionne les données en K clusters en essayant de minimiser la variance intra-cluster. Il nécessite de spécifier le nombre de clusters à l'avance.

  — **Algorithme EM.** c'est une méthode de clustering qui utilise une approche itérative pour maximiser la probabilité des données en fonction du modèle de clustering.

- **Recherche d'associations.**

  La recherche d'association permet de trouver des liens cachés entre différentes variables dans les bases de données. Il expose des modèles ambigus dans les données et choisit différentes mesures d'importance pour tirer les meilleures règles à partir d'un ensemble de règles pertinente. Les seuils de support et de confiance sont des mesures utilisées pour évaluer la qualité des ùotifs fréquents et des règles valides.

  Les algorithmes de recherche de règles d'association sont utilisés pour identifier les relations entre différents éléments dans une base de données transactionnelle (algorithme

A Priori, algorithme FP-Growth, etc.). D'autres formes adaptées de règles ont été proposées pour d'autres formes de données.

1.2.4 Les domaines d'applications du Data Mining

Avec la performance des systèmes informatiques actuels et le développement rapide des méthodes d'apprentissage automatique, la fouille de données est devenue largement utilisée dans de nombreux domaines d'applications de la vie courante [14, 15]. On peut citer certains exemples de ses domaines comme suit :

- **Médical/Pharmaceutique.**
  - — Expliquer ou prédire la réponse d'un patient à un traitement.
  - — Étude des corrélation entre le dosage dans un traitement et l'apparition d'effets secondaires.
  - — Etudier comment les changements dans la séquence d'ADN d'une personne affectent le risque de développer des maladies courantes telles que le cancer.
- **Marketing et Publicité.**
  - — Analyse des comportements des consommateurs.
  - — personnalisation des offres.
- **Éducation.**
  - — Suivie les performance des étudiants.
  - — Analyse des tendances éducatives

1.3 Fouille de motifs

1.3.1 Fouille d'itemset fréquentes à partir de BD transactionnelles

1.3.1.1 Concepts de base

- **Item.** un item est une unité de donnée individuelle qui peut être utilisée pour représenter, traiter, ou analyser l'information dans un contexte particulier. Par exemple, chaque produit à vendre dans un super-marché est un item. L'ensemble de tous les items est noté I.
- **Itemset.** Un itemset $X \subseteq I$ est un sous-ensemble non vide d'items (éléments ou articles dans une base de données). Par exemple $\{lait, sucre, sel\}$ est un exemple d'un itemset

dans un super-marché.

- *k*-**itemset.** Un ensemble d'items (itemset) X de cardinalité $k = |X|$ est appelé un k-itemset. Par exemple, $Caf, Moutarde, Saucisse, lait$ est un 4-itemset.
- **Transaction.** Une transaction (ligne de la base de données) est un ensemble d'items. Pour un itemset X de I, on dit qu'une transaction t contient X si et seulement si $X \subseteq t$, i.e., tous les items de X figurent dans t.
- **Base de données transactionnelle :** Une base de données transactionnelle est généralement un fichier avec chaque enregistrement représentant une transaction[16]. Une transaction contient essentiellement une liste d'éléments (e.g. les achats d'un client lors d'une visite) et un identifiant unique de transaction (TID). D'autres informations supplémetaires peuvent également figurer dans une transaction. Dans le cas des achats dans un super-marché par exemple, on peut trouver : la date de la transaction, l'identité du consommateur, l'identité de la personne qui a vendu, etc. Un exemple d'un gragment d'une BD transactionnelle est donné dans le Tableau 1.1

Transaction ID	lait	pain	sucre	salade
t_1	1	1	1	0
t_2	1	1	1	0
t_3	1	1	1	0
t_4	1	0	1	0
t_5	0	1	1	0
t_6	0	0	0	1

TABLE 1.1 – Exemple d'une base de données transactionnelle

- **Support d'un itemset.** le support d'un itemset X dans une BD transactionnelle D est le nombre de transactions qui contiennent X. Le support relatif de X dans D est le pourcentage de transactions de D qui contiennent X. C'est le rapport entre le nombre de transactions contenant X et le nombre total de transactions de D. Le support relatif d'un itemset est donné par la formule suivante :

$$\text{Sup}(X) = \frac{|\{t \in D \mid X \subseteq t\}|}{|D|}$$

- **Itemset fréquent.** On dit qu'un itemset X est fréquent si est seulement si son support

est supérieur ou égal à un seuil donné de support minimum noté *minsup*, i.e., $Sup(X) \geq minsup$.

- **Règles d'association.** Une règle d'association est une expression $X \Rightarrow Y$ entre deux itemsets fréquents X et Y, avec $X \neq \emptyset,\ Y \neq \emptyset,\ X \cap Y = \emptyset$. Une règle d'association permet de capter les dépendances cachées antre des ensembles d'items.
- **Support et confiance d'une règle d'association.**

  Le **support** d'une règle est le nombre de transactions dans la base de données transactionnelle contenant à la fois les itemsets X et Y :

$$Sup(X \Rightarrow Y) = Sup(X \cup Y)$$

  La **confiance** d'une règles $X \implies Y$ est le rapport entre le support de la règle $X \Rightarrow Y$ et le support de l'itemset X.

$$\text{Conf}(X \Rightarrow Y) = \frac{\text{Sup}(X \cup Y)}{\text{Sup}(X)}$$

  Une règle d'association $X \Rightarrow Y$ est dite valide si sont support (resp. confiance) est supérieur ou égal à un seuil minimum de support *minsup* (resp. de confiance $minconf$), i.e., si : $Sup(X \Rightarrow Y) \geq minsup$ et $Conf(X \Rightarrow Y) \geq minconf$.

1.3.1.2 Algorithmes d'extraction des règles d'association

Plusieurs algorithmes ont été proposés dans littérature pour l'extraction d'itemsets fréquents et de règles d'association valides à partir de bases de données transactionnelles. Nous présentons ici trois de ces algorithmes parmi les plus utilisés en pratique.

- **Algorithme Apriori**

  L'algorithme Apriori[17] est un algorithme très connu pour extraire les règles d'association des bases de données transactionnelles. Il est basé sur le principe de l'anti-monotonie. Il consiste à identifier les itemsets fréquents et à extraire des règles d'association valides. L'algorithme itère différentes étapes, éliminant progressivement les itemsets non fréquents (ayant un support inférieur au seuil de support minimum spécifié par l'utilisateur) en se basant sur le principe d'ant-imonotonie qui stipule que chaque

sur-ensemble d'un ensemble non fréquent est également non fréquent. L'exploitation de cette prpriété permet de réduire la complexité de l'exploration des itemsets fréquents et des règles d'association valide.

Exemple.

L'exemple ci-dessous montre le processus d'extraction des itemsets fréquents à partir d'une base de transactions. Considérant la base de donnée transactionnelle *D* illustrée dans la table 1.2 et supposons que le seuil minimum de support est $minsup = 0.33$ et le seuil minimum de confiance est $minconf = 60\%$

TID	Items
t_1	Hotdogs,Buns,Ketchup
t_2	Hotdogs,Buns
t_3	Hotdogs,Coke
t_4	Chips,Coke
t_5	Chips,Ketchup
t_6	Hotdogs,Coke,Chips

TABLE 1.2 – Exemple de base de données transactionnelle des achats d'un client

Dans un premier temps on calcule les itemsets fréquents de taille = 1

itemset	Support
Hotdog	0.66
Buns	0.33
Ketchup	0.33
Coke	0.5
Chips	0.66

TABLE 1.3 – les itemset fréquentes de taille=1 (L1)

Construction des itemsets fréquents de taille 2 (L_2)

On construit tous les ensembles possibles de taille 2 à partir des ensembles L_1 trouvés de l'étape précédente. Pour cela on applique des jointures sur les itemsets de L_1

et on calcule pour chaque 2-itemset candidat son support pour garder uniquement les 2-itemsets fréquents. Voici les 2-itemsets fréquents trouvés :

itemsets	Support
Hotdogs,Buns	0.33
Hotdogs,Coke	0.33
Hotdogs,Chips	0.33
Coke,Chips	0.5

TABLE 1.4 – Itemsets fréquents de taille 2 (L2)

Construction des itemsets fréquents de taille 3 (L_3)

On construit tous les 3-itemsets candidats à partir de l'ensemble L_2 des 2-itemsets fréquents en appliquants les opérations de jointure et élimination. ensuite, on calcule pour chaque 3-itemset candidat son support afin de ne garder que les 3-itemsets fréquents. Voici les 3-itemsets fréquents trouvés :

itemset	Support
Hotdog,Coke,Chips	0.33

TABLE 1.5 – Les itemset fréquentes de taille=3(L_3)

Donc, l'ensemble de tous les itemsets fréquents dans cet exemple est $F = L_1 \cup L_2 \cup L_3$.

Génération des règles d'associations

Pour chaque k-itemset f fréquent avec $k \geq 2$, on génère toutes les règles d'association de la forme $X \Rightarrow Y$ avec $X \cup Y = f$. Pour cela, on détermine d'abord les règles valides (ayant une confiance supérieure ou égale à $minconf$) de la forme $X \Rightarrow Y$ avec $|Y| = 1$. Ensuite itérativement, on applique des jointures sur les conséquences des règles valides de la forme $X \Rightarrow Y$ avec $|Y| = k$, $k \geq 1$, pour trouver les règles candidats ayant des conséquences de taille $k+1$. Enfin, le calcul des confiances de ces règles candidats permet de déterminer les règles valides ayant des conséquences de taille $k+1$. Ce processus est répété jusqu'à la génération de toutes les règles valides à partir de l'itemset fréquent f. Les règles d'association valides obtenus dans notre exemple avec

$minconf = 60\%$ sont illustrées dans le tableau suivant :

R1 :{B} → {Hd}	conf(R)=2/2=100%
R3 :{CO} → {HD}	conf(R)=2/3=66.66%
R7 :{CH} → {CO}	conf(R)=3/4=75%
R8 :{CO} → {CH}	conf(R)=3/3=100%
R9 :{CO,CH} → {HD}	conf(R)= 2/3=66.66%
R10 :{HD,CH} → {CO}	conf(R)=2/2=100%
R11 :{HD,CO} → {CH}	conf(R)=2/2=100%
R13 :{CO} → {HD,CH}	conf(R)=2/3=66.66%

TABLE 1.6 – les resultats des régles avec minconf=60%

- **Algorithme FP-growth**

  L'algorithme FP-Growth, abréviation de Frequent Pattern Growth[17], est un algorithme efficace en exploration de données pour extraire des motifs fréquents sans générer explicitement tous les itemsets fréquents. Voici les étapes clés de cet algorithme :

  — **Construction de l'arbre FP.** L'algorithme commence par construire un arbre FP (Frequent Pattern) à partir de la base de données en utilisant une structure d'arbre compacte pour représenter les itemsets fréquents.

  — **Extraction des motifs fréquents.** En parcourant l'arbre FP, les motifs fréquents sont extraits en utilisant une approche récursive qui évite la génération explicite des itemsets.

  — **Génération des règles d'association.** Une fois les motifs fréquents identifiés, l'algorithme peut être utilisé pour générer des règles d'association basées sur ces motifs.

  L'algorithme FP-Growth est particulièrement efficace pour traiter de grandes bases de données et est souvent préféré à l'algorithme Apriori en raison de sa capacité à réduire le temps de calcul et l'espace mémoire requis pour l'exploration des motifs fréquents

- **Algorithme Eclat**

  L'algorithme Eclat est un algorithme d'extraction de règles d'association similaire à l'algorithme Apriori, mais utilisant une méthode différente. L'algorithme Eclat se base sur une structure de données appelée tableau de fermeture, contrairement à Apriori, qui

utilise un processus itératif pour explorer des ensembles d'articles fréquents de différentes tailles.

L'algorithme Eclat explore le tableau de fermeture pour identifier les itemsets fréquents. Sans générer explicitement tous les candidats, il utilise une approche récursive pour parcourir les combinaisons d'items et créer les itemsets fréquents. En particulier, dans les cas où la base de données contient un grand nombre d'articles, l'algorithme Eclat est souvent plus efficace en termes de temps de calcul et d'utilisation de la mémoire que l'algorithme Apriori. Il est utilisé dans des applications d'exploration de données, de recommandation de produits et d'autres domaines où l'identification de motifs fréquents est cruciale.

1.3.2 Autres types de motifs

1.3.2.1 Fouille de motifs séquentiels

Les motifs séquentiels sont des structures ou des séquences récurrentes identifiables dans des ensembles de données séquentielles. Ces motifs peuvent révéler des tendances, des comportements ou des caractéristiques importantes dans les données ordonnées. La fouille de motifs séquentielle est utilisée dans divers domaines, tels que :

- **Bioinformatique** : Découverte de motifs dans les séquences d'ADN et d'ARN.
- **Analyse de texte** : Découverte de thèmes et de sentiments dans les textes.
- **Finance** : Détection de fraudes et d'anomalies dans les transactions financières.
- **Sécurité informatique** : Détection d'intrusions et d'attaques dans les réseaux informatiques.

1.3.2.2 Fouille de motifs périodiques

La fouille de motifs périodiques se concentre sur l'identification de motifs à la fois fréquents et périodiques, i.e., dont les séparations entre les occurrences consécutives ne sont pas très éloignées, dans des bases de où les transactions sont numérotées ou datées. Ces motifs périodiques peuvent représenter des tendances, des schémas ou des comportements répétés à intervalles réguliers dans des séquences temporelles.

La fouille de motifs périodiques a de nombreuses applications dans divers domaines, tels que :

- **Commerce électronique** : Dans le domaine du commerce électronique, la fouille de motifs périodiques peut être utilisée pour identifier des habitudes d'achat récurrentes chez les clients. Par exemple, un motif périodique pourrait révéler que certains produits sont achetés fréquemment à des intervalles réguliers, comme chaque mois.
- **Santé** : En médecine, la fouille de motifs périodiques peut être appliquée pour détecter des schémas dans les données de surveillance médicale. Par exemple, la fréquence des pics de température chez un patient peut être analysée pour identifier des motifs périodiques qui pourraient indiquer un état de santé particulier.
- **Le domaine des clics en ligne** : la fouille de motifs périodiques peut être appliquée pour analyser les habitudes de navigation des utilisateurs et identifier des modèles récurrents dans les clics sur des liens, des publicités ou des éléments interactifs.

1.3.2.3 Fouille de motifs à haute utilité

La fouille de motifs à haute utilité est une approche dans le domaine de la fouille de données qui se concentre sur l'identification de motifs ou de modèles particulièrement significatifs et utiles dans les ensembles de données, en accordant une attention particulière aux critères d'utilité définis par l'utilisateur ou l'analyste. Cette méthode diffère de la fouille de motifs fréquents en mettant l'accent sur l'importance et la pertinence des motifs extraits par rapport à des objectifs spécifiques, plutôt que sur leur fréquence d'apparition. La fouille de motifs à haute utilité a de nombreuses applications dans divers domaines, tels que :

- **Commerce électronique** : Optimisation des recommandations de produits pour les clients en identifiant des motifs d'achat à haute utilité, amélioration de la personnalisation des offres.
- **Santé** : Détection de motifs cliniques significatifs dans les données médicales, amélioration de la prédiction des maladies et des traitements personnalisés.
- **Finance** : Identification de schémas financiers pertinents, détection de fraudes, anticipation des tendances du marché.

1.4 Fouille d'épisodes

1.4.1 Principe et objectif

La découverte des épisodes fréquents est une approche largement utilisée pour extraire des informations significatives à partir de séquences d'événements. De nombreux algorithmes ont été développés dans ce domaine pour identifier et déduire des règles d'épisodes fréquents en fonction d'une définition donnée de la fréquence et en exploitant des propriétés particulières comme l'anti-monotonie pour accélérer le processus de recherche.

L'état de l'art du domaine de la fouille d'épisodes a reconnu un grande volume d'outils et d'algorithmes. Les algorithmes WINEPI, MINEPI sont les deux premiers algorithmes proposés par Mannila et al [18] dans ce domaine. Récemment, ce domaine rconnaît une évolution importante en termes de nouvelles techniques ainsi que des applications qui intègrent la découverte des épisodes comme tâche principale pour la reconnaissance des motifs intéressants dans les données.

L'extraction des règles d'épisodes est une tâche très intéressante qui étend la tâche traditionnelle de la découverte des épisodes fréquents. Cette tâche consiste à exprimer les relations implicites entre les épisode fréquents en se basant sur leurs occurrences dans les séquence d'évènements. Les règles extraites dans cette phase peuvent être également utilisées pour des tâches de prédiction.

Cependant, en plus des types existants des épisodes, la façon de définir et de calculer la fréquence de ces épisodes dans les séquences pose un défi majeur pour les techniques proposées. Pour cela, on peut distinguer deux grandes familles de définitions de fréquences. Dans cette section, nous allons aborder ces types de fréquences ainsi que quelques algorithmes de découverte des épisodes.

1.4.2 Les différents types de fréquence

- **Fréquence basée sur les fenêtres (window-based frequency)**

  Elle consiste à compter le nombre de fenêtres de largeur fixe dans lesquelles un épisode donné apparaît au moins une fois. Elle repose sur l'idée de diviser la séquence d'événements en fenêtres d'une certaine taille et de compter les occurrences d'épisodes au sein

de ces fenêtres [19].

- **Fréquence basée sur les occurrences (occurrence-based frequency)**
  Ce type de fréquence, se focalise sur les occurrences effectives des épisodes dans une séquence d'événements. Plusieurs définitions sont possibles. Par exemple, la fréquence sans chevauchements compte le nombre d'occurrences effectives de l'épisode qui ne se chevauchent jamais. Un autre exemple est la fréquence basée sur les occurrences distinctes qui se concentre sur les occurrences uniques d'événements au sein des épisodes, permettant des occurrences qui peuvent se chevaucher sans partager d'événements en commun. Ceci garantit que chaque événement n'est compté qu'une seule fois.

1.4.3 Quelques algorithmes existants de fouille d'épisodes

- **WINPI (window-based episode mining.)** L'algorithme WINEPI [20] est conçu pour extraire des motifs d'utilité élevée à partir de séquences d'événements. Il vise à découvrir des motifs fréquents qui sont non seulement fréquents, mais aussi utiles pour l'utilisateur. WINEPI utilise une approche basée sur des fenêtres pour identifier les motifs temporels significatifs dans les séquences d'événements, ce qui le rend utile pour l'analyse de données temporelles complexes.

  WINEPI peut être appliqué pour analyser les interactions des utilisateurs, les sujets d'actualité et les modèles de comportement des utilisateurs sur les plateformes de médias sociaux, permettant ainsi des recommandations de contenu personnalisées, la détection d'anomalies et l'analyse des tendances.
- **MINEPI.** L'algorithme MINEPI [20] est conçu pour extraire des épisodes fréquents à partir de séquences d'événements. Il vise à découvrir des motifs temporels significatifs dans les séquences d'événements en utilisant une approche basée sur des occurrences. MINEPI utilise une définition de fréquence basée sur des occurrences distinctes pour calculer le support des épisodes, ce qui lui permet de découvrir des épisodes fréquents dans les données d'événements complexes.

  MINEPI peut être utilisé pour découvrir des modèles ou des épisodes récurrents au sein de séquences d'événements, fournissant ainsi un aperçu des relations temporelles entre les événements.
- **NONEPI.** L'algorithme NONEPI [20] est un algorithme de fouille d'épisodes fréquent

sous la fréquence basée sur les occurrences non superposées. Il permet également de générer, à partir des épisodes fréquents, un ensemble de règles d'épisodes. Il peut être utilisé pour analyser le comportement d'achat d'un client (une séquence d'achats) ou pour trouver des relations entre certains mots dans un texte (une séquence de mots).

- **EMDO.** EMDO [1] se concentre sur l'extraction d'épisodes fréquents dans des séquences d'événements complexes en comptant des occurrences distinctes, garantissant que chaque événement n'est compté qu'une seule fois tout en permettant le chevauchement des occurrences.

1.5 Conclusion

Ce chapitre explore la fouille de données, de sa définition à ses applications variées. L'accent est mis sur l'extraction de connaissances à partir de données complexes, l'apprentissage automatique, et la fouille de motifs, notamment la recherche d'épisodes fréquents et de règles d'épisodes. Les concepts de base, les algorithmes, et la diversité des types de motifs sont présentés en détail.

Dans le prochain chapitre, nous mettons l'accent sur l'algorithme EMDO (**E**pisode **M**ining under **D**istinct **O**ccurences).

Chapitre 2 :

Algorithme de fouille d'épisodes fréquents basé sur les occurrences distinctes

Chapitre 2

Algorithme de fouille d'épisodes fréquents basé sur les occurrences distinctes

2.1 Introduction

La fouille d'épisodes est un domaine très important pour découvrir les motifs intéressants à partir de séquences d'évènements. Elle permet d'identifier des sous-séquence d'événements qui se produisent souvent dans un ordre spécifique. Ces motifs sont appelés *épisodes fréquents*. Ce chapitre présente les concepts fondamentaux de la fouille d'épisodes. Nous expliquerons également la forme de règles d'épisodes découverte par cet algorithme. Pour cela, nous commencerons par définir les notions de base de l'algorithme EMDO (**E**pisode **M**ining under **D**istinct **O**ccurrences), telles que les séquences, la forme d'épisodes à découvrir par EMDO [1] et comment principes de calcule du support d'un épisode. Pour plus d'information sur ce domaine, le lecteur peut suivre un survey sur la fouille d'épisode [21]

2.2 Définitions des notions de base

Avant de plonger dans les détails de l'algorithme de fouille d'épisodes fréquents basé sur les occurrences distinctes, il est essentiel de clarifier certaines notions de base qui sous-tendent cette approche :

Definition 1 (Évènement) *Un événement est une paire* (e,t) *où e est un élément d'un ensemble*

E (ensemble de tous les types d'événements) qui représente le type d'événement et t est un entier qui indique l'instant de temps où cet évènement se produit [1].

Definition 2 (Séquence) *Une séquence est une succession d'événements triés par leur temps d'occurrences. La séquence elle même est un triplet $S = (s, T_s, T_e)$ ou T_s et T_e sont respectivement temps de début et temps de fin de la séquence [1] :*

$$S =< (e_1,t_1)(e_2,t_2)(e_3,t_3)....(e_n,t_n) > \ avec\ T_s \leq t_1 < t_2 < \cdots < t_n \leq T_e$$

.

Un exemple d'une séquence simple d'évènements est illustrée dans la figure 2.1. Cette séquence S contient 10 évènements de 3 types d'événements différents $E = \{a, b, c\}$.

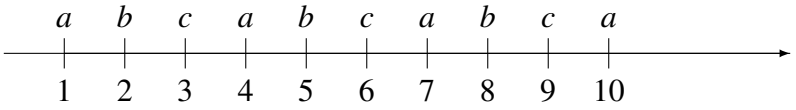

FIGURE 2.1 – Exemple d'une séquence d'événements

Definition 3 (Épisode) *Un épisode α est un triplet $(V, \prec_\alpha, g_\alpha)$ où V est un ensemble de noeuds $v_1, v_2, ..., v_n$, $\prec_\alpha$ est un ordre sur V et $g_\alpha : V \to E$ est une application qui associe un type d'événement à chaque noeud. L'entier n indique le nombre de noeuds de l'épisode α ($n = |\alpha|$).*

Selon l'ordre $\prec_\alpha$, on peut distinguer entre différents types d'épisodes : lorsque l'ordre α est **total**, l'épisode α est un épisode en **série**. Dans ce cas, α est simplement noté $\alpha = A_1 \to A_2 \to \cdots \to A_n$ où chaque A_i est un type d'événement. Lorsque l'ordre $\prec_\alpha$ est **vide**, l'épisode α est dit **parallèle** et il est noté $\alpha = A_1 A_2 ... A_n$.

Un épisode est dit *injectif* s'il ne contient aucun type d'événement répété, c'est-à-dire pour tout $1 \leq i, j \leq n$, si $i \neq j$ alors $g(v_i) \neq g(v_j)$. La **taille** d'un épisode est le nombre de ses évènements (noeuds) [1].

Il est à noter que l'algorithme EMDO, que nous allons présenter dans les sections qui suivent, découvre les épisodes parallèles. Pour cela, et pour simplifier les détails de l'algorithme, dans ce qui suit, le mot *épisode* désigne simplement un épisode parallèle [1].

Definition 4 (Sous-épisode) *Soit $\alpha = A_1 A_2 \ldots A_n$ et $\beta = B_1 B_1 \ldots B_m$ deux épisodes. L'épisode*

β est un sous-épisode de α (noté $\beta \sqsubseteq \alpha$) si et seulement s'il existe m entiers $i_1, i_2, \ldots, i_m$ avec : $1 \leq i_1 < i_2 < \cdots < i_m \leq n$ tel que $B_1 = A_{i_1}$, $B_2 = A_{i_2}$, ... $B_m = A_{i_m}$. L'ordre $\prec_\beta$ des noeuds de l'épisode β (sous-épisode) doit être le même que celui dans l'épisode α (super-épisode). Dans ce cas l'épisode α est appelle ***super-épisode*** *[1].*

Definition 5 (Occurrences d'un épisode) *Une occurrence de l'épisode α dans la séquence S est un vecteur d'entiers $h = [t_1, t_2, ...t_n]$ tel que chaque t_i est le temps d'occurrence du i^{eme} noeud (événement) de l'épisode α, c'est-à-dire A_i se produit au temps t_i dans S [1].*

Definition 6 (Occurrences distinctes d'un épisode) *Soient $h_1 = [t_1\ t_2\ \ldots\ t_n]$ et $h_2 = [t'_1\ t'_2\ \ldots\ t'_n]$ deux occurrences d'un épisode α. Les deux occurrences h_1 et h_2 sont dites* ***occurrences distinctes*** *si et seulement si $t_i \neq t'_j$ pour $1 \leq i, j \leq n$ avec $t_i \in h_1$ et $t'_j \in h_2$ [1].*

Definition 7 (Ensemble maximum d'occurrences distinctes) *Soit H un ensemble d'occurrences distinctes de l'épisode α. L'ensemble H est dit maximum si et seulement si pour tout ensemble H' d'occurrences distinctes, $|H| \geq |H'|$. L'ensemble maximum d'occurrences distinctes de α est notée par do_α [1].*

Definition 8 (Support d'un épisode) *Le support d'un épisode α, noté $sup(\alpha)$, représente le nombre maximum des occurrences distinctes :$sup(\alpha) = |do_\alpha|$. L'épisode α est considéré comme un épisode fréquent dans la séquence si $sup(\alpha) \geq minsup$, où minsup est un seuil de support prédéfini [1].*

L'algorithme EMDO cherche en deuxième phase les règles d'épisodes. Les règles d'épisodes expliquent les relations implicites entre un pair d'épisodes tel que ces deux épisodes soient fréquents sous la définition de fréquence basée sur les occurrences distincte. La forme de règles à chercher est donnée dans la définition qui suit.

Definition 9 (Règle d'épisode) *Une règles d'épisodes est une expression de la forme $\beta \Rightarrow \alpha$ où α et β deux épisodes fréquents au sens de la fréquence basée sur les occurrences distinctes [1].*

Definition 10 (Support d'une règle d'épisode) *le support d'un règle d'épisode $\alpha \implies \beta$, notée $supER(\alpha \implies \beta)$, est définie comme suit : $supER(\alpha \implies \beta) = |occER(\alpha \implies \beta)|$ où $occER(\alpha \implies \beta) = do_{\alpha \cup \beta}$, i.e., les occurrences distinctes de l'episode $\alpha \cup \beta$ [1].*

Definition 11 (Confiance d'un règle d'épisode) *La confiance d'une règle d'épisode* $\alpha \Rightarrow \beta$ *est notée conf* $(\alpha \Rightarrow \beta)$*, et elle est définie comme suit :* $\text{conf}(\alpha \Rightarrow \beta) = \frac{\|\text{occER}(\alpha \Rightarrow \beta)\|}{\text{support}(\alpha)}$ *[1].*

2.3 Découvrir les épisodes fréquents en se basant sur les occurrences distinctes

EMDO signifie «Episode Mining under Distinct Occurrences (Fouille d'Episodes sous les occurrences distinctes)». Il s'agit d'un algorithme conçu pour trouver des épisodes fréquents dans des séquences d'événements en comptant les occurrences distinctes. L'algorithme surmonte les limites des algorithmes de fouille d'épisodes existants en considérant des séquences d'événements complexes, i.e., dans lesquelles plusieurs événements peuvent se produire au même instant. EMDO introduit le concept d'occurrences distinctes, qui permet de compter les occurrences qui peuvent se chevaucher tout en garantissant que le même évènement n'est pas compté plusieurs fois. Cette approche vise à fournir une vision plus précise de la fréquence des épisodes dans des séquences d'événements complexes.

Ainsi, l'algorithme EMDO utilise une nouvelle définition de fréquence moins stricte que la fréquence basée sur les occurrences sans chevauchement. Il utilise une stratégie de recherhce en profondeur d'abord pour découvrir des épisodes parallèles fréquents et des règles d'épisode valides, en tenant compte des occurrences distinctes d'événements dans les séquences. De plus, EMDO intègre une stratégie d'élagage pour générer efficacement des règles d'épisode valides.[1].

2.3.1 Extraction des épisodes fréquents

L'algorithme Mine Frequent episodes [1] est utilisé pour extraire les épisodes fréquents à partir d'une séquence d'événements complexes. L'algorithme initialise l'ensemble des épisodes fréquents et l'ensemble des règles, puis parcourt la séquence d'événements pour extraire les épisodes de taille 1 et les stocker dans un ensemble P. Pour éviter d'explorer tout l'espace de recherche, l'algorithme exploite la propriété d'anti-monotonie qui permet d'éliminer les épisodes non fréquents en se basant sur la fréquence de leurs sous-ensembles. Cela permet de réduire l'espace de recherche et d'identifier les épisodes fréquents de manière efficace.
À la fin de l'algorithme, l'ensemble F contient tous les épisodes fréquents extraits de la sé-

quence d'événements complexes.

Algorithme 1: Mine_Frequent_Episodes

Input: *minsup* - a minimum support threshold.
Output: F - the set of all frequent parallel episode

```
P ⇐ {} ; // Initialise l'ensemble des épisodes candidats fréquents
F ⇐ {} ; //Initialise l'ensemble des épisodes fréquents
α ⇐ ∅;   //Initialise α comme vide pour construire de nouveaux épisodes
// Initialiser les occurrences pour chaque événement individuel
for (each individual event e ∈ E) do
    do_e ⇐ {} ;
end
  // scan the sequence S
for (k ⇐ 1 to n) do
    {% scan the event set found at time t_k %}
    for (j ⇐ 1 to m) do
        e_j ⇐ the event found at time t_k ;
        do_ej ⇐ do_ej ∪ {t_k};
    end
end
// Filtrer les événements fréquents basés sur le seuil de support
for (each individual event e ∈ E) do
    if (‖do_e‖ ≥ minsup) then
        P ⇐ P ∪ {(e)};
    end
end
F ⇐ P ; // Initialise F avec les épisodes fréquents trouvés dans P
 // Étendre chaque épisode fréquent trouvé
for (each individual episode α ∈ F) do
    for (each individual episode β ∈ P) do
        do_α⊔β ⇐ Distinct_Occurrence_Recognition(α,β);
        if (‖do_α⊔β‖ ≥ minsup) then
            γ ⇐ α ⊔ β ;
            do_γ ⇐ do_α⊔β ;
            F ⇐ F ∪ {γ} ;
            α ⇐ γ ;
        end
    end
end
return F ;  // Retourne l'ensemble complet des épisodes fréquents
```

L'algorithme Roconnaissances des occurences [1] présente en détail la fonction de reconnaissance d'occurrences. Il prend en entrée un épisode α à étendre et un unique épisode ayant un seul évènement β ($|\beta| = 1$) par lequel α est prolongé. En sortie, il identifie les occurrences

distinctes du nouvel épisode $\alpha \sqcup \beta$. Pour chaque occurrence de α, notée $O_i \in do_\alpha$, l'algorithme explore les occurrences do_β pour trouver un événement $O_j \in do_\beta$ et concevoir une nouvelle occurrence de $\alpha \sqcup \beta$ contenant l'horodatage de O_i joint avec celles de O_j.

Ensuite, l'algorithme compare la nouvelle occurrence avec celui dans $do_{\alpha \sqcup \beta}$. Si la dernière occurrence se chevauche avec une ou plusieurs occurrences courantes de $\alpha \sqcup \beta$, l'algorithme vérifie simplement quelle occurrence à supprimer (ignorer) de do_α où bien de do_β. Pour cela, l'algorithme vérifie si l'occurrence O_j de do_β courante existe dans $newocc.timestamps$ (c-à-d : $O_j.timestamps[1] \in newocc.timestamps$) alors l'algorithme supprime l'occurrence O_j de do_β. Sinon, les horodatages de l'occurrence $O_i.timestamps$ sûrement existent dans $newocc.timestamps$ (ligne 10-22). Il est à noter que la fonction $exists(t_k, O_i)$ vérifie si l'entier t_k existe dans le vecteur O_i. A la fin de chaque itération, l'algorithm ajoute une occurrence valide à l'ensemble $do_{\alpha \sqcup \beta}$ (line 23). Finalement, il retourne l'ensemble $do_{\alpha \sqcup \beta}$.

Algorithm 2: Reconnaissance des occurrences

```
Input: episode α - un épisode à étendre
       episode β - un épisode d'un seul événement à utiliser pour étendre α
Output: doα⊔β - l'ensemble des occurrences distinctes du nouvel épisode α ⊔ β
//Initialiser les indices
j ← 0 ;
i ← 0 ;
// Parcourir chaque occurrence de α
for (each Oi ∈ doα ) do
    found ← false ;
    // Initialiser la variable trouvée pour l'occurrence actuelle de α
    // Parcourir chaque occurrence de β jusqu'à ce qu'une correspondance soit
       trouvée
    for (each O j ∈ doβ and f ound = f alse) do
        // Combiner les timestamps des occurrences de α et β
        newocc.timestamps ← Oi.timestamps ∪ O j.timestamps ;
        // Initialiser les variables de contrôle pour la boucle interne
        stop ← false ;
        k ← 1 ;
        // Vérifier si la nouvelle occurrence chevauche des occurrences existantes
        while (not stop and k ≤ |doα⊔β |) do
            if (newocc.timestamps ∩ Ok.timestamps/= /0) then
                if (exists(O j.timestamps[1], Ok.timestamps) = true) then
                    // Supprimer l'occurrence actuelle de β si elle chevauche
                    remove O j from doβ ;
                else
                    // Supprimer l'occurrence actuelle de α si elle chevauche
                    remove Oi from doα ;
                end
                stop ← true ; //Arrêter la boucle interne si un chevauchement est trouvé
            end
            Ok ← Ok+1 ; // Passer à l'occurrence suivante
        end
        // Mettre à jour la variable trouvée si un chevauchement a été détecté
        if (stop) then
            found ← true ;
        end
        doα⊔β ← doα⊔β ∪newocc ;
    end
end
return doα⊔β ;
```

2.3.2 Extraction des règles d'épisodes

L'algorithme 3 [1] est utilisé pour calculer le support d'une règle d'épisode, c'est-à-dire le nombre d'occurrences valides dans la séquence. L'algorithme initialise le support de la règle à 0 et définit deux indices i et j à 1. Ensuite, l'algorithme parcourt les occurrences des épisodes α et β pour calculer le support de la règle. Il itère sur les occurrences de l'épisode a et pour chaque occurrence, il recherche une occurrence valide de l'épisode β en vérifiant si $start(O_i) < start(O_j)$ et $end(O_i) < end(O_j)$. À chaque fois qu'une occurrence valide de la règle $\alpha \Rightarrow \beta$ est trouvée, le support est augmenté de 1. Une fois toutes les occurrences vérifiées, l'algorithme retourne le support de la règle $\alpha \Rightarrow \beta$.

Algorithm 3: Calcul de support d'une règle d'épisode

Input: Deux épisodes α et β
Output: *ruleSupport* : Le support de la règle $\alpha \Rightarrow \beta$

```
% Initialisation
{% Initialisation %}
ruleSupport ← 0;
i ← 1;
j ← 1;
% Parcourir toutes les occurrences de α
while (i ≤ |do_α|) do
    stop ← false;
    % Parcourir les occurrences de β tant que l'on n'a pas trouvé une correspondance
    while (j ≤ |do_β| and not stop) do
        % Parcourir les occurrences de β à partir de l'indice actuel
        for (each O_k in do_β s.t : j < k ≤ |do_β|) do
            % Vérifier si les occurrences satisfont les conditions de la règle
            if (start(O_i) < start(O_j) and end(O_i) < end(O_j)) then
                i ← i + 1;
                % Passer à l'occurrence suivante de α
                ruleSupport ← ruleSupport + 1;
                % Incrémenter le support de la règle
                stop ← true;
                % Arrêter la recherche pour l'occurrence actuelle de α
            end
        end
        j ← k;
        % Mettre à jour l'indice pour les occurrences de β
    end
end
% Retourner le support de la règle
return ruleSupport;
```

L'algorithme **Extract_Episode_Rules_With_Pruning** (Extraction de Règles d'Épisodes avec Élagage) est conçu pour identifier efficacement les règles d'épisodes dans des séquences d'événements complexes en utilisant une stratégie d'élagage pour réduire l'espace de recherche. Il commence par initialiser l'ensemble des épisodes fréquents F en utilisant l'algorithme **Mine_Frequent_Episodes** avec un seuil de support minimal minsup. Ensuite, pour chaque épisode fréquent α de l'ensemble F, l'algorithme génère des candidats pour les conséquents β des règles en combinant les épisodes simples de l'ensemble P. Pour chaque candidat β, il calcule le support de la règle $\alpha \Rightarrow \beta$ en utilisant la fonction **Episode_Rule_Support** et la confiance de la règle comme conf $= \frac{\text{ruleSupport}}{|\alpha.\text{occ}|}$. Si la confiance dépasse le seuil minimal minconf, la règle est ajoutée à l'ensemble R des règles valides. Si une règle candidate est invalide, l'algorithme utilise la propriété d'anti-monotonie pour élaguer les super-épisodes de cette règle, évitant ainsi des calculs inutiles. Cette approche permet de découvrir toutes les règles d'épisodes valides tout en optimisant l'efficacité grâce à l'élagage. En fin de compte, l'algorithme retourne l'ensemble complet R des règles d'épisodes valides, facilitant ainsi l'analyse des séquences d'événements complexes.

Algorithm 4: Extraction des règles d'épisodes avec élagage

```
1 center[H]
   Input: S : séquence d'événements complexes sur E (ensemble de types d'événements)
   minsup : seuil de support
   mincon f : seuil de confiance .
   Output: R : ensemble complet des règles d'épisodes
2  F ← FrequentEpisodes(minsup) ;      // Trouver tous les épisodes fréquents
3  R ← /0 ;                            // Initialiser l'ensemble des règles d'épisodes
4  P ← {γ|γ ∈ F ∧ ‖γ‖ = 1} ;   // Ensemble des épisodes fréquents de longueur 1
5  for (each α in F) do          // Parcourir chaque épisode fréquent
6     j ← 0 ;
7     // Parcourir chaque épisode fréquent de longueur 1
8     while ( j < |P|) do
9           β ← P[ j] ;
10          root ← β ;
11          k ← j ;
12          stop ← false ;
13          // Construire des règles d'épisodes
14          while (not stop) do
15              if (β ∈ F) then
16                  // Calculer le support de la règle d'épisode
17                  ruleSupport ← EpisodeRuleSupport(α, β ) ;
18                  // Calculer la confiance de la règle d'épisode
19                  con f ← ruleSupport/|al pha.occ| ;
20                  // Vérifier si la confiance est supérieure au seuil
21                  if (con f >= mincon f ) then
22                      // Ajouter la règle à l'ensemble des règles
23                      R ← R ∪ {α → β } ;
24                      root ← β ;
25                  else
26                      β ← root ;
27                  end
28              else
29                  β ← root ;
30              end
31              // Vérifier si tous les épisodes fréquents ont été parcourus
32              if (k > |P|) then
33                  stop ← true ;
34              else
35                  // Combiner β avec le prochain épisode fréquent
36                  β = β ⊔ P[k] ;
37                  k ← k + 1 ;
38              end
39          end
40          j ← j + 1 ;
41    end
42 end // Retourner l'ensemble des règles d'épisodes
```

2.4 Conclusion

Pour conclure ce chapitre sur l'algorithme de fouille d'épisodes fréquents basé sur les occurrences distinctes, nous pouvons souligner plusieurs points clés. Tout d'abord, nous avons défini les notions de base nécessaires à la compréhension de l'algorithme, notamment ce que c'est un épisode, une fréquence, les notions de support et de confiance, etc. Ensuite, nous avons présenté l'algorithme EMDO utilisé dans ce travail en détaillant les tâches respectives des différents sous-algorithmes qui le composent. A présent, nous allons montrer dans le chapitre suivant comment l'algorithme EMDO est utilisé dans un contexte applicatif pour analyser les séquences qui représentent l'historique de navigation web d'un utilisateur.

Chapitre 3 :

Implémentation et Etude Expérimentale

Chapitre 3

Implémentation et Etude Expérimentale

3.1 Introduction

Ce chapitre se concentre sur la mise en oeuvre pratique et l'évaluation expérimentale des techniques et algorithmes discutés précédemment. Nous détaillons ici les étapes d'implémentation de l'algorithme EMDO (Episode Mining under Distinct Occurrences) et présentons le matériel et les logiciels utilisés pour cette tâche. L'objectif est de démontrer l'efficacité de l'algorithme à travers des expériences concrètes. Nous discutons également des résultats obtenus, en fournissant une analyse approfondie et une interprétation des données expérimentales. Cette approche permet de valider la pertinence et la performance de notre méthode dans un contexte d'application réel.

3.2 Implémentation

3.2.1 Matériel

Afin de montrer l'efficacité de l'algorithme que nous avons choisi, nous avons utilisé une station de calcul installé au niveau du laboratoire LMSE (Laboratoire de Matériaux et Systèmes Electroniques) dont les caractéristiques sont citées ci-dessous :

- **Processeur :** 2 processeur r Intel (R) Xeon 2.27 GHZ et 2.26 GHZ.
- **RAM :** 48 GO.
- **Type de systéme :** Windows 7 Professional 64 bits.

3.2.2 Logiciels

- **Eclips IDE**

  "Eclipse IDE" est un environnement de développement intégré (IDE) populaire utilisé principalement pour le développement en Java, bien qu'il prenne en charge plusieurs autres langages de programmation grâce à des plugins. Eclipse offre un ensemble complet d'outils pour écrire, déboguer et déployer des applications logicielles. Il est largement utilisé dans l'industrie du développement de logiciels et dispose d'une large communauté d'utilisateurs et de contributeurs qui développent des plugins et des extensions pour améliorer ses fonctionnalités[22].

- **JAVA**

  Java est un langage de programmation polyvalent, orienté objet, conçu pour être portable et sécurisé. Il offre une syntaxe simple et élégante, une gestion automatique de la mémoire via le ramasse-miettes (garbage collector) et une vaste bibliothèque standard. [23, 24].

- **SPMF**

  SPMF (Sequential Pattern Mining Framework)[25] est une bibliothèque open-source en Java dédiée à l'exploration et à l'analyse de données séquentielles, en particulier l'extraction de motifs séquentiels à partir de séquences d'éléments ordonnés. Cette bibliothèque offre une gamme de techniques de fouille de données séquentielles, notamment l'extraction de motifs fréquents, l'extraction de motifs séquentiels fermés, la prédiction de séquences, etc.

  Concernant la phase de préparation de données, elle consiste à convertir des données stockées dans un fichier CSV (Comma Separeted Values) vers une séquence d'événements. Pour cela, nous avons utilisé l'environnement de Python (3.10) JUPYTER Note book configurés sous PYCHARM (édition professionnelle), ainsi que les bibliothèques de python NUMPY et PANDAS pour la manipulation des tables et des fichiers d'analyse de données respectivement.

3.3 Étude Expérimentale

3.3.1 Base de données

Pour la mise en oeuvre du côté applicatif de note mémoire, nous avons utilisé une base de données réelle publiquement partagée sur la plate-forme Kaggle [26]. Cette base de données a été collectée par une société de réalisation de moteurs de recherche appelé Outbrain [27]. La table 3.1 montre quelques détailles sur la base de données qu'on a utilisé dans le contexte de notre mémoire.

Le taille de la base de données	10 000 000 lignes
Nombre de documents (pages)	59849

TABLE 3.1 – Détails sur la base de données utilisées

L'ensemble de données contient un échantillon de pages vues et de clics d'utilisateurs observés sur plusieurs sites d'éditeurs aux États-Unis entre le 14 juin 2016 et le 28 juin 2016. Cette table contient plus de **2 milliards de lignes** et occupe **100 Go d'espace non compressé**. Cependant, on a utilisé une échantillons de **10 millions** de lignes afin de bien évaluer les performances de l'algorithme. La base de données est composée de 6 colonnes :

- **uuid :** cette colonne représente un identifiant unique pour chaque utilisateur qui visite un document.
- **id_document :** contient l'identifiant du document que l'utilisateur a visité.
- **horodatage (timestamp en ms) :** enregistre l'heur de chaque visite d'un utilisateur sur un document.
- **plateforme (ordinateur de bureau =** 1**, mobile =** 2**, tablette =** 3**) :** indique la plateforme de laquelle l'utilisateur à accéder au document.
- **géo_localisation :**capture des informations sur l'emplacement géographique de l'utilisateur à déférentes niveaux.
- **traffic_source (interne =** 1**, recherche =** 2**, social =** 3**) :** catégorise la manière de visite des documents par les utilisateurs.

Cependant, la structure actuelle de la table n'est pas adaptée à la fouille d'épisodes. Pour cette raison, on a fait d'abord une opération de pré-traitement. Nous avons sélectionnés unique-

ment les colonnes qui nous intéressent et supprimé le reste des colonnes dont nous n'avons pas besoin, comme : uuid, géo_localisation et traffic_source comme indiqué dans la Figure 3.1.

L'étape suivante consiste à ne garder que les lignes dont l'attribut plateforme = 3. L'objectif derrière ce choix est que la majorité des utilisateurs utilisent beaucoup les mobiles et/ou les tablettes. À la fin, on obtient une table moins chargée que la précédente. Cette dernière est ensuite convertit en une forme adaptée à la fouille d'épisodes par l'algorithme EMDO.

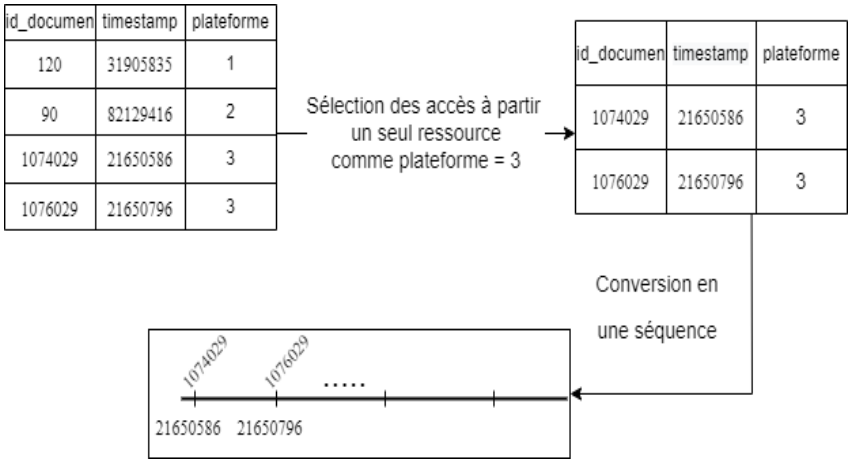

FIGURE 3.1 – Préparation des données au format adaptée à EMDO

3.3.2 Interprétation des résultats

Pour montrer l'efficacité de notre travail, nous avons mené plusieurs tests concernant : (1) l'extraction des épisodes fréquents et (2) la génération des règles d'épisode valides.

3.3.2.1 Extraction des épisodes fréquents

Dans ces expérimentations, nous avons utilisé plusieurs valeurs pour le seuil du support minimum comme indiqué dans la table suivante. Cette table nous donne pour chaque valeur de *Minsup* utilisée, le nombre d'épisodes fréquents trouvés, la taille maximale des épisodes trouvés et le nombre d'épisodes candidats :

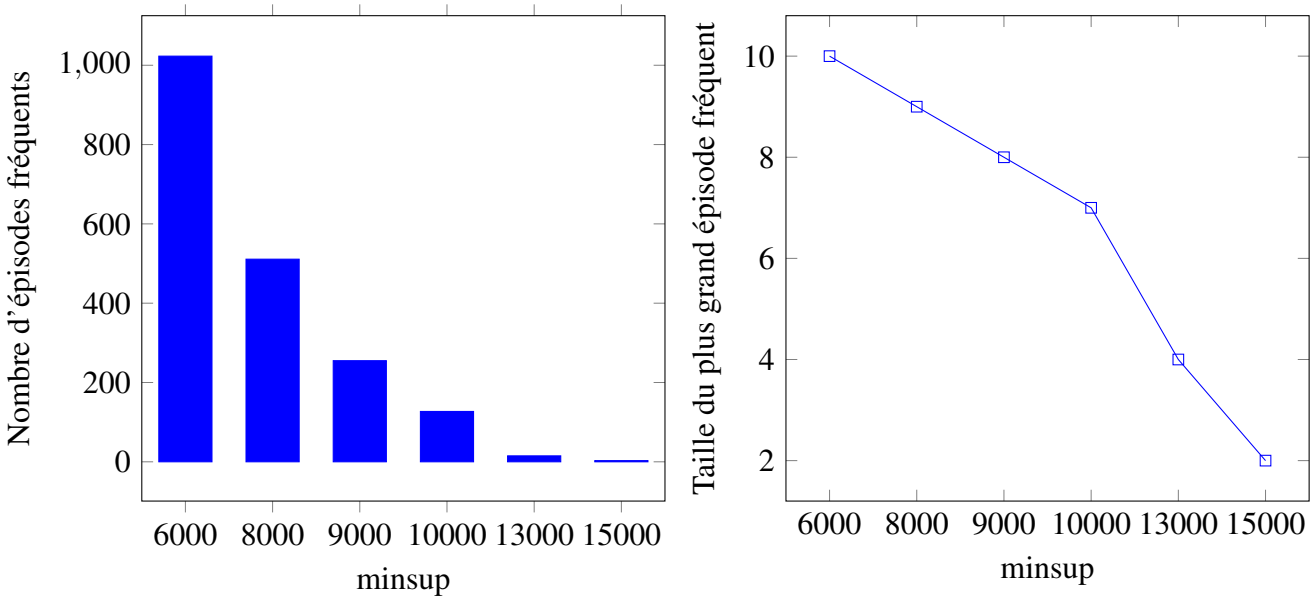

FIGURE 3.2 – Influence de *minsup* sur le nombre d'épisodes fréquents et la taille du plus grand épisode sur des jeux de données réels

Minsup	Nbr d'episode frequent	Taille max	Episode condidats
6000	1023	10	40560
8000	511	9	40049
9000	255	8	39794
10000	127	7	39667
15000	3	2	39548

TABLE 3.2 – Extraction des épisodes fréquents

Lorsque la valeur du *Minsup* est modifiée à chaque fois, on observe une relation inverse entre le Minsup, le nombre d'épisodes fréquents identifiés et le temps de calcul nécessaire. Plus la valeur du *Minsup* est élevée, moins d'épisodes fréquents sont identifiés car le support minimum requis est plus strict. Cela conduit à une diminution du nombre d'épisodes identifiés.

En revanche, à mesure que le Minsup diminue (6000), le nombre d'épisodes fréquents identifiés augmente (1023) car le support minimum requis est moins contraignant, permettant d'identifier plus d'occurrences.

Cependant, cette augmentation du nombre d'épisodes associée à un *Minsup* plus bas s'accompagne généralement d'un temps de calcul plus long, car le processus d'analyse doit exami-

ner un plus grand nombre de candidats potentiels.

3.3.2.2 Génération des règles d'épisode

Pour tester notre travail nous avons effectués plusieurs expériences a partir de la bibliothèque SPMF et à l'aide d'environnement de développement populaire Eclipse.

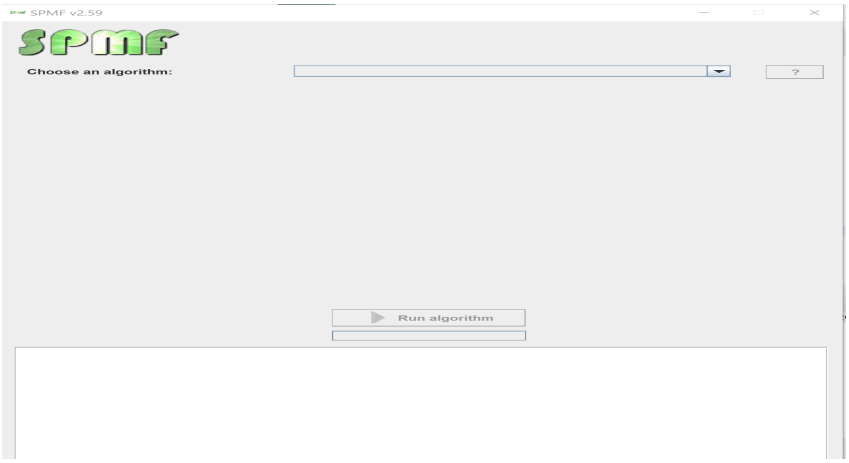

FIGURE 3.3 – Interface de SPMF (Sequential Pattern Mining Framework)

Nous avons utilisé l'interface de SPMF en suivant les étapes suivantes :

- Nous optons pour l'algorithme EMDO et sélectionnons le fichier d'entrée (resultat.txt), puis spécifions le fichier de sortie (out_resultat.txt).
- Nous choisissons ensuite la valeur de $Minsup$ par exemple 9000 et de $Minconf$, par exemple 0.1.
- Nous choisissons 'pattern viewer' dans le fichier de sortie ouvert en utilisant (open output file using).
- L'algorithme est lancé en appuyant le bouton 'Run'.

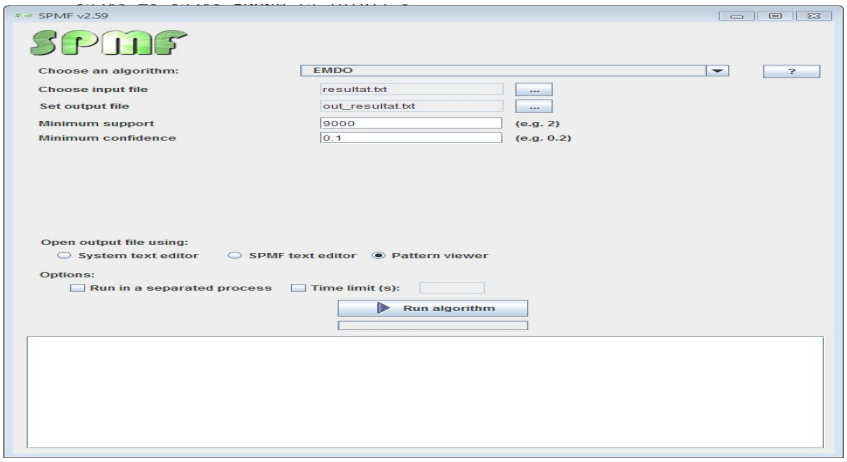

FIGURE 3.4 – Spécification des paramètres sur SPMF

Le tableau suivant montre les résultats d'une étude avec la définition de la valeur *minsup* à 10000 et en faisant varier la valeur *minconf* à chaque fois. Voici les résultats obtenus :

Minconf	Nbr de règle
0.1	11785
0.3	6889
0.5	3983
0.7	2003
1	569

TABLE 3.3 – Variation de nombre de règles par rapport aux valeurs de *Minconf*

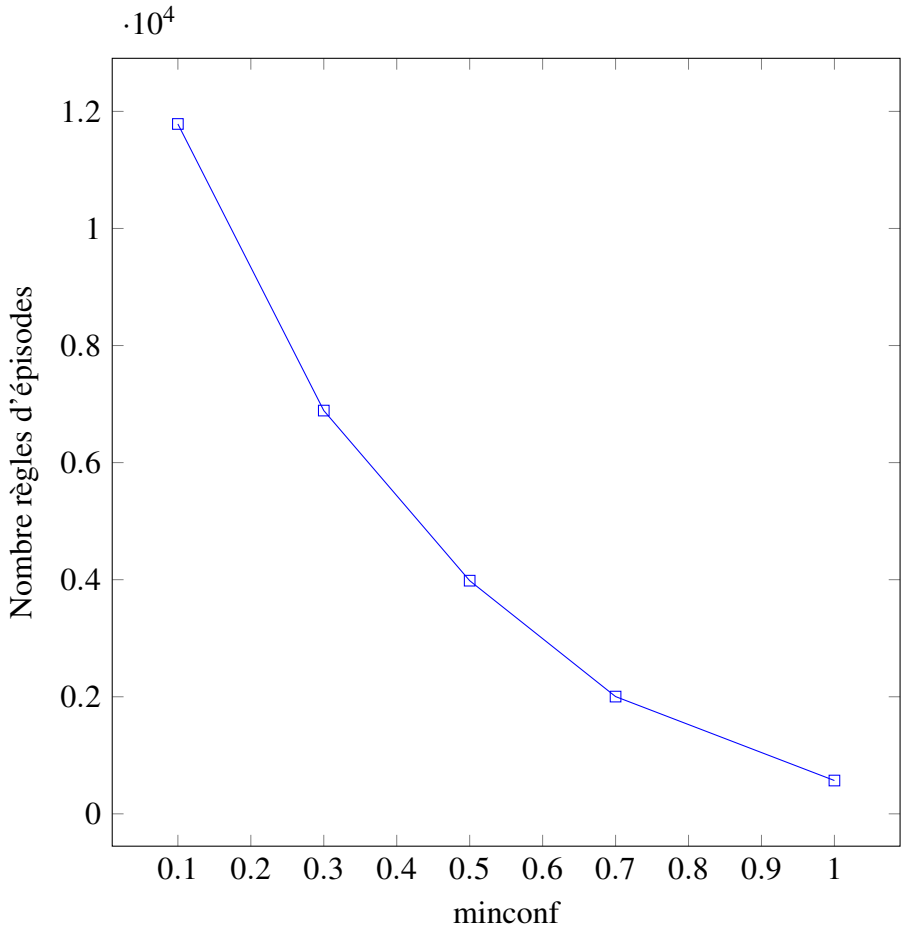

FIGURE 3.5 – Influence de $minconf$ sur le nombre de règles d'épisodes pour $minsup = 10000$

Selon les valeurs trouvées, on remarque une relation inverse entre la valeur du seuil de confiance et le nombre de règles trouvées par l'algorithme EMDO. Cela confirme fortement l'efficacité de la propriété d'anti-monotonie entre les règles d'épisodes. Par conséquent, l'algorithme fonctionne sur toute taille de base de données.

3.3.2.3 Exploitation des régles :

Pour montrer plus clairement les résultats de notre travail, nous avons lancé un test avec une valeur de $minsup = 13000$ et $minconf = 1$. La Table 3.4 illustre toutes les règles d'épisode obtenus :

N°	Règles d'épisodes	Confiance
1	<42744> ⇒ <234>	100.0 %
2	<42744> ⇒ <1811567>	100.0 %
3	<234,42744> ⇒ <234>	100.0 %
4	<234,42744> ⇒ <1811567>	100.0 %
5	<234,42744> ⇒ <234,1811567>	100.0 %
6	<42744,1811567> ⇒ <234>	100.0 %
7	<42744,1811567> ⇒ <1811567>	100.0 %
8	<42744,1811567> ⇒ <234,1811567>	100.0 %
9	<234,42744,1811567> ⇒ <234>	100.0 %
10	<234,42744,1811567> ⇒ <1811567>	100.0 %
11	<234,42744,1811567> ⇒ <234,1811567>	100.0 %

TABLE 3.4 – Exemple de règles d'épisodes trouvées pour $minsup = 13000$ et $minconf = 100\%$

D'après la table 3.4, on peut déduire que l'algorithme EMDO applique efficacement la propriété d'anti-monotonie [28] entre les règles d'épisodes. Cette propriété indique que la confiance d'une règle $\alpha \Rightarrow \beta$ ne dépasse pas celle d'une règle$\alpha \Rightarrow \beta'$telle que $\beta' \sqsubseteq \beta$.
Il est à noter que l'algorithme EMDO a pû trouver des règles avec un taux de confiance de 100 % comme l'avant dernière règle de la Table 3.4. On peut également suggérer des publicités à afficher dans les pages vue que les publicités aussi jouent un rôle très important dans le côté marketing.

Les règles d'épisodes découvertes dans notre analyse présentent de nombreux avantages pratiques et peuvent être utilisées de diverses manières pour optimiser nos stratégies et améliorer l'expérience utilisateur. Voici quelques-unes de leurs utilisations :

- **Optimisation de contenu :**
  consiste à améliorer la qualité, la pertinence et l'attrait du contenu d'une page web pour mieux répondre aux attentes des utilisateurs et des moteurs de recherche.
- **Navigation Guidée :**
  consiste à structurer le parcours des utilisateurs sur un site web de manière logique et intuitive, facilitant ainsi la découverte et l'interaction avec le contenu.

- **Amélioration de l'engagement :**
  peut augmenter le temps passé sur le site en orientant les utilisateurs vers des contenus liés.
- **Identifier et faciliter les flux de navigation :**
  consiste à comprendre et améliorer les chemins que les utilisateurs empruntent sur un site web pour atteindre leurs objectifs.
- **Optimisation des parcours utilisateurs :**
  consiste à améliorer les chemins que les utilisateurs suivent sur un site web pour atteindre leurs objectifs de manière efficace et satisfaisante.
- **Comprehension des séquences complexes :**
  consiste à analyser et interpréter les chemins de navigation non linéaires et variés que les utilisateurs suivent sur un site web. par exemple sur un site de voyage, certains utilisateurs peuvent d'abord consulter des guides de destinations, puis comparer des offres de vol, revenir aux guides, vérifier des avis sur des hôtels, et enfin réserver un package complet. Comprendre cette séquence complexe permet au site de proposer des recommandations pertinentes à chaque étape, facilitant ainsi la prise de décision et améliorant l'expérience utilisateur.
- **Amélioration de contenu :**
  consiste à mettre à jour et à optimiser le contenu d'un site web pour qu'il soit plus pertinent, attractif et utile pour les utilisateurs.
- **Structuration des visites :**
  consiste à organiser et à guider les utilisateurs à travers les pages d'un site web de manière à optimiser leur expérience utilisateur.
- **Réduction du taux de Rebond :**
  implique de minimiser le pourcentage de visiteurs qui quittent un site web après avoir consulté une seule page, sans interagir davantage.
- **Analyse des parcours :**
  consiste à examiner et à comprendre les chemins que les utilisateurs empruntent lors de leur navigation sur un site web.
- **Personnalisation du contenu :**
  implique d'adapter le contenu d'un site Web en fonction des préférences, des comportements et des caractéristiques individuelles des utilisateurs.

Les utilisateurs peuvent consulter une grande variété de sujets dans le cadre du concours **Outrbrain Click Prediction** [26], car la plate-forme collabore avec une variété d'éditeurs et d'annonceurs qui proposent des contenu dans divers domaines. Dans le contexte de ventes et achat, la Table 3.5 montre quelques explication sur l'utilité des règles dans le cas où ces cliques viennent de sites de ventes en ligne (e-commerce) avec conf=100% :

N° de la règle	Explication
1	Si la page 42744 est la page d'accueil et la page 234 est une page de produit phare. Cela pourrait signifier que les utilisateurs trouvent rapidement le produit phare attrayant depuis la page d'accueil. Cela indique que le produit en question est bien mis en avant sur la page d'accueil, ce qui justifie son positionnement stratégique pour maximiser les visites et potentiellement les ventes.
2	Si la page 42744 est une page de promotion et la page 1811567 est une page de formulaire de demande de devis Cela montre l'efficacité de la page de promotion pour diriger les utilisateurs vers une action concrète, comme demander un devis. Optimiser cette transition pourrait augmenter le taux de conversion.
3	Si la page 234 est une page de produit et 42744 est une page de témoignages clients. Les utilisateurs qui consultent les témoignages reviennent ensuite au produit, ce qui peut indiquer que les témoignages sont convaincants et renforcent la décision d'achat. Mettre des liens de retour bien visibles vers le produit depuis les témoignages pourrait renforcer cette boucle positive.
4	Si la page 1811567 est une page de checkout ou de panier. Les utilisateurs passent de la page produit à la page témoignages puis au checkout, ce qui montre une progression naturelle vers l'achat. Assurer que ces pages sont bien connectées et sans obstacles pourrait améliorer l'expérience d'achat.
7	Les utilisateurs revisitent la page de formulaire ou de checkout après avoir vu la page de promotion. Cela montre un intérêt continu pour compléter l'action. Assurer que cette page est optimisée pour les conversions (chargement rapide, formulaires clairs) est crucial.
9	Les utilisateurs reviennent souvent à la page produit après avoir visité des pages d'information et de commande. Indique un potentiel manque d'informations ou d'éléments rassurants sur la page produit, nécessitant une optimisation pour minimiser les retours et accélérer la décision d'achat.
11	Les utilisateurs naviguent entre la page produit et la page de commande après avoir visité une page d'information. Souligne l'importance de présenter des informations cohérentes et complémentaires entre ces pages pour soutenir la décision d'achat et réduire les hésitations.

TABLE 3.5 – d'utilisation des règles sur la vente et l'achat de produit

3.4 Conclusion

Ce chapitre a exposé la mise en oeuvre réussie de l'algorithme EMDO et son efficacité à extraire des motifs temporels à partir de données séquentielles. Les résultats expérimentaux ont confirmé la robustesse et la rapidité de l'algorithme, ainsi que sa capacité à améliorer les prédictions. Cette étude expérimentale valide l'approche proposée et ouvre la voie à des améliorations et applications futures de l'algorithme EMDO..

Conclusion générale

Ce mémoire a présenté des travaux de recherche sur la fouille d'épisodes à partir de séquences d'événements, avec une application concrète à l'analyse de l'historique des visites de pages web. La conclusion générale de ce mémoire récapitule les contributions essentielles et les résultats obtenus dans le domaine de la fouille de données séquentielles. En se focalisant sur l'algorithme EMDO (Episode Mining under Distinct Occurrence), le mémoire démontre comment cette méthode innovante permet d'extraire des motifs temporels efficaces à partir de données de navigation web. Cela ouvre la voie à des applications pratiques telles que l'amélioration des moteurs de recommandation en ligne.

L'étude met en évidence l'importance de la fouille d'épisodes pour mieux comprendre et prédire le comportement des utilisateurs, ce qui est crucial pour le développement de solutions personnalisées dans le commerce électronique. En outre, elle propose des directions pour des recherches futures, notamment l'optimisation de l'algorithme et son application à d'autres types de données séquentielles. En conclusion, ce mémoire offre des perspectives prometteuses pour l'application pratique des techniques de fouille d'épisodes.

Bibliographie

[1] O. Ouarem, F. Nouioua, and P. Fournier-Viger, "Discovering frequent parallel episodes in complex event sequences by counting distinct occurrences," *Applied Intelligence*, vol. 54, no. 1, pp. 701–721, 2024.

[2] J. Han, M. Kamber, and J. Pei, *Data Mining : Concepts and Techniques*, 4th ed. Morgan Kaufmann, 2023, a comprehensive introduction to data mining.

[3] Kobia1. (2024) Qu'est-ce que le data mining ? [Online]. Available : https://kobia.fr/quest-ce-que-le-data-mining/

[4] S. Boudrandi and M. Meknassi, "Extraction de la connaissance à partir des données : un trait d'union entre le management de la connaissance et l'intelligence organisationnelle," in *Association pour la Gestion des Connaissances dans la Société et les Organisations : AGeCSO*, Montreal, Canada, 2017.

[5] M. A. Nedoui, "La fouille de données," Mémoire de master, Université Elchahid Hama Lakhder El Oued, El Oued, Algérie, 2018.

[6] Eduscol, "Introduction à l'apprentissage automatique," 2023. [Online]. Available : https://eduscol.education.fr/sti/sites/eduscol.education.fr.sti/files/ressources/pedagogiques/14512/14512-introduction-lapprentissage-automatique-ensps.pdf

[7] Blent(Equipe). (2024) Apprentissage supervisé : définition. [Online]. Available : https://blent.ai/blog/a/apprentissage-supervise-definition

[8] Salesforce. (2024) Qu'est-ce que l'apprentissage supervisé ? [Online]. Available : https://www.salesforce.com/fr/resources/definition/apprentissage-supervise/

[9] M. Bensedddik, "L'apprentissage statistique pour le diagnostic de défauts dans un

système automatique," 2022-2023. [Online]. Available : http://dspace.univ-ouargla.dz/jspui/bitstream/123456789/34677/1/MOUANE-BENSEDDIK.pdf

[10] IBM. (2024) Qu'est-ce qu'un arbre de décisions | ibm. [Online]. Available : https://www.ibm.com/fr-fr/topics/decision-trees

[11] K. P. Murphy, *Machine learning : a probabilistic perspective*. MIT press, 2012.

[12] B. Brossault. (2024) Machines à vecteurs de support : définition et utilisation. [Online]. Available : https://blog.hubspot.fr/marketing/support-vector-machine

[13] Linedata. (2024) Qu'est-ce que l'apprentissage non supervisé ? [Online]. Available : https://fr.linedata.com/quest-ce-que-lapprentissage-non-supervise#:~:text=L'apprentissage%20non%20supervis%C3%A9%20est,tr%C3%A8s%20peu%20d'intervention%20humaine.

[14] B. Senai, "La fouille des images medicales," Ph.D. dissertation, Faculté des Mathématiques et Informatique, Université des Sciences et de la Technologie d'Oran Mohamed Boudiaf, 2016.

[15] Kobia2. (2024) Data mining : définition, fonctionnement, domaine d'application. [Online]. Available : https://kobia.fr/quest-ce-que-le-data-mining/

[16] D. Chami, "Une plate forme orientée agent pour le data mining," Ph.D. dissertation, Université HADJ LAKHDAR – BATNA, BATNA, Algérie, 2010.

[17] G. Calas, "Études des principaux algorithmes de data mining," http://guillaume.calas.free.fr/data/Publications/DM-Algos.pdf, 2009.

[18] L. Fahed, "Prédire et influencer l'apparition des évènements dans une séquence complex," Ph.D. dissertation, Université de Lorraine, France, 2016.

[19] A. Achar, S. Laxman, and P. S. Sastry, "A unified view of automatabased algorithms for frequent episode discovery," *arXiv preprint arXiv :1007.0690*, 2024.

[20] H. LAIB and Y. BOUKHARI, "Fouille d'épisodes à partir de séquences d'événements," Mémoire de Master, Université de Mohamed El Bachir El Ibrahimi de Bordj Bou Arreridj, Faculté des Mathématiques et d'Informatique, Département d'informatique, 2023.

[21] O. Ouarem, F. Nouioua, and P. Fournier-Viger, “A survey of episode mining,” *WIREs Data. Mining. Knowl. Discov.*, vol. 14, no. 2, 2024. [Online]. Available : https://doi.org/10.1002/widm.1524

[22] “«Éclipse - définition et explications»,” https://www.techno-science.net/definition/517.html, [En ligne, consulté le 1er juin 2024].

[23] Oracle Corporation. (2023) «java documentation». [Online]. Available : https://docs.oracle.com/en/java/

[24] TutorialsPoint. «java tutorial». [Online]. Available : https://www.tutorialspoint.com/java/index.htm

[25] “Spmf :bibliothèque de data mining,” En ligne, disponible à : https://www.philippe-fournierviger.com/spmf/index.php?link=algorithms.php.

[26] “Kaggle web site,” https://www.kaggle.com/ Dernière accès 16 Mai 2024.

[27] https://www.outbrain.com/fr/, Dernière accès le 10 Janvier 2024.

[28] O. Ouarem, F. Nouioua, and P. Fournier-Viger, “Mining episode rules from event sequences under non-overlapping frequency,” in *Proceedings of IEA/AIE-2021, Advances and Trends in Artificial Intelligence. Artificial Intelligence Practices*, ser. Lecture Notes in Computer Science, vol. 12798. Cham : Springer, 2021.