Projet EDA

Tanguy Ducrocq - Charles Stephann - Clément MARTIN

Professeur : Lamine DIOP
EPITA - Ing3

30 Novembre 2025

Table des matières

Résumé

Ce rapport présente la conception et l'implémentation d'un outil interactif d'exploration de motifs fréquents. Face au problème de l'explosion combinatoire des motifs dans les algorithmes classiques, nous proposons une approche hybride intégrant deux pipelines distincts : (1) une extraction exhaustive suivie d'un échantillonnage interactif guidé par l'utilisateur, et (2) un échantillonnage direct en sortie (*Output Sampling*). La solution, développée en **Python** avec une interface **Streamlit**, permet l'analyse de données transactionnelles et séquentielles (via une approche Bag-of-Items), tout en intégrant des métriques de diversité, de qualité et un mécanisme de feedback utilisateur.

Introduction et Contexte

Les jeux de données transactionnels (paniers d'achats) et séquentiels (logs, parcours) recèlent des informations précieuses pour la prise de décision. Cependant, les algorithmes classiques tels que Apriori ou FP-Growth génèrent souvent une quantité massive de motifs, dont beaucoup sont redondants ou peu informatifs pour l'analyste métier. Ce phénomène, connu sous le nom de **pattern explosion**, rend l'analyse manuelle impossible. L'objectif de ce projet est de développer une solution logicielle permettant de :

— Extraire efficacement un pool de motifs candidats.
— Sélectionner un sous-ensemble pertinent via des stratégies de scoring composites.
— Intégrer l'expert métier dans la boucle via un mécanisme de feedback interactif.
— Proposer une alternative probabiliste pour les grands jeux de données (*Output Sampling*).

Méthodologie et Choix Techniques

3.1 Préparation des Données et Architecture

Le système repose sur une **architecture modulaire** séparant la logique métier (`tools.py`) de l'interface utilisateur (`app.py`).

Gestion des formats L'outil est capable d'ingérer des fichiers `CSV`, `JSON` et `Parquet`. Une détection automatique du format est implémentée pour distinguer :

— **Format Wide** : Une ligne par transaction (items séparés par des espaces).

— **Format Long** : Une ligne par item (`TransactionID`, `Item`).

Traitement des données séquentielles Pour les données séquentielles (comportant une dimension temporelle), nous avons adopté une approche pragmatique dite "**Bag-of-Items**". Notre pipeline utilise l'encodage **One-Hot** (via `TransactionEncoder` de `mlxtend`) pour transformer les séquences en ensembles d'items. Cela permet de réutiliser les algorithmes de fouille d'ensembles fréquents (Apriori) tout en capturant les co-occurrences globales au sein d'une même session utilisateur.

3.2 Pipeline 1 : Extraction et Échantillonnage Interactif

3.2.1 Extraction Exhaustive

La première étape consiste à générer un pool candidat P à l'aide de l'algorithme **Apriori** avec un seuil de support minimal configurable (par défaut $sup = 0.01$). Pour chaque motif, nous calculons les métriques standard : **Support**, **Confidence**, **Lift**, et **Longueur**.

3.2.2 Stratégies de Scoring

Pour filtrer ce pool, nous avons implémenté plusieurs **stratégies de scoring composite**, normalisant les métriques entre $[0, 1]$. La formule générale pour le score d'un motif m est :

$$\text{Score}(m) = \alpha \cdot \text{Norm}(\text{Lift}) + \beta \cdot \text{Norm}(\text{Support}) + \gamma \cdot \text{Norm}(\text{Surprise})$$

Redondance La redondance est calculée par la similarité moyenne de **Jaccard** avec les autres motifs du pool. Trois profils sont disponibles :

— **Balanced** : Compromis entre fréquence, corrélation et surprise.

— **Quality** : Favorise les règles fortes (haute confiance) et le lift.

— **Diversity** : Pénalise fortement la redondance pour maximiser la couverture.

3.2.3 Feedback Utilisateur

L'interactivité est assurée par un mécanisme de **ré-pondération dynamique**. Lorsqu'un utilisateur "Like" ou "Dislike" un motif, le système ajuste les poids de tous les motifs similaires (basé sur l'indice de **Jaccard**) dans le pool candidat, influençant ainsi les futurs tirages aléatoires (*Importance Sampling*).

3.3 Pipeline 2 : Output Pattern Sampling

Pour répondre aux limitations de mémoire sur les très grands datasets, nous avons implémenté un **échantillonneur direct** (*Output Sampling*). Au lieu de miner tous les motifs, l'algorithme :

1. Génère aléatoirement des itemsets candidats de longueur k.
2. Vérifie leur existence et calcule leur mesure d'intérêt (**Support** ou **Lift**) à la volée.
3. Sélectionne les motifs finaux selon une distribution de probabilité proportionnelle à leur intérêt (**Méthode de Monte Carlo**).

Implémentation et Interface

L'application est développée avec **Streamlit**, offrant une interface web réactive sans besoin de compétences front-end avancées.

FIGURE 4.1 – Interface de l'application : Sélection des paramètres et visualisation des motifs.

Les fonctionnalités clés incluent :
— **Chargement dynamique** : Upload de fichiers et configuration des colonnes (*Wide/Long*).
— **Onglets Pipeline** Séparation claire entre l'approche exhaustive et l'échantillonnage direct.
— **Visualisation** Affichage tabulaire des motifs avec indicateurs colorés et boutons de feedback.
— **Export** : Possibilité de télécharger les échantillons générés en `CSV`.

Évaluation et Résultats

Pour garantir la pertinence des motifs extraits, nous avons intégré un module d'évaluation calculant plusieurs métriques.

5.1 Métriques de Qualité

— **Diversité** $(1 - \text{Jaccard moyen})$: Mesure à quel point les motifs de l'échantillon sont différents les uns des autres. Nos tests montrent que la stratégie "Diversity" augmente ce score de 15% par rapport à un échantillonnage aléatoire simple.
— **Couverture** Proportion de transactions contenant au moins un des motifs de l'échantillon.
— **Stabilité** : Robustesse de l'échantillonnage face aux changements de graine aléatoire.

5.2 Performance

Les tests ont été réalisés sur des jeux de données synthétiques et réels.

TABLE 5.1 – Comparaison des performances des deux approches.

Métrique	**Pipeline Interactif**	**Output Sampling**
Latence (1k transactions)	~ 0.5 sec	~ 0.1 sec
Latence (50k transactions)	~ 12 sec (Apriori)	~ 2.5 sec
Diversité moyenne	0.72	0.85
Mémoire	Élevée (Stockage Pool P)	Faible

On constate que l'*Output Sampling* est nettement plus performant en temps et mémoire pour les grands volumes de données, bien qu'il puisse manquer certains motifs très spécifiques que l'approche exhaustive (Apriori) capture systématiquement.

Discussion et Conclusion

Ce projet a permis de mettre en œuvre une chaîne complète de fouille de données, allant du pré-traitement à l'interaction utilisateur.

Limites : Le choix principal de traiter les séquences comme des "**sacs d'items**" (*Bag-of-Items*) simplifie grandement l'implémentation et permet l'usage d'Apriori, mais entraîne la **perte de l'information d'ordre** (ex : $A \rightarrow B$ est confondu avec $B \rightarrow A$). Pour une analyse séquentielle pure, l'intégration d'algorithmes comme **SPADE** ou **PrefixSpan** serait nécessaire.

Perspectives : Les améliorations futures pourraient inclure :

— L'implémentation réelle de la fouille séquentielle (respectant l'ordre temporel).

— Une visualisation graphique des relations (graphes de réseaux) dans Streamlit.

— La persistance du modèle de feedback utilisateur entre les sessions.

En conclusion, l'outil développé offre un compromis pragmatique entre exhaustivité et interactivité, permettant à un analyste d'extraire rapidement de la connaissance exploitable à partir de données brutes.