Apriori: Une approche améliorée

Maria Oussadi, Mars 2024

1- Abstract:

L'exploitation des règles d'association est cruciale dans de nombreux domaines de l'informatique, notamment dans le domaine de l'analyse de données et de la fouille de données. L'algorithme Apriori est largement utilisé pour extraire ces règles à partir de grands ensembles de données. Cependant, l'efficacité et la robustesse de l'Apriori peuvent être limitées en raison de sa sensibilité aux paramètres et de son inefficacité pour les ensembles de données de grande taille et complexes.
Pour pallier ces limitations, nous proposons une approche adaptative de l'algorithme Apriori. Cette approche ajuste dynamiquement le support minimum en fonction des caractéristiques des données, telles que la distribution des supports des itemsets fréquents. En favorisant les zones denses de cette distribution, l'Apriori adaptatif vise à mieux capturer les modèles sous-jacents dans les données et à réduire la sensibilité aux paramètres.
Dans cet article, nous présentons l'algorithme adaptatif Apriori, discutons de ses fondements théoriques et de son implémentation pratique, et évaluons ses performances à travers des expérimentations sur divers ensembles de données.

2- Travaux connexes:

FP-Growth (Han et al. 2000) [1]: C'est une alternative à l'algorithme Apriori pour l'extraction des itemsets fréquents. FP-Growth utilise une structure de données appelée FP-tree pour représenter l'ensemble de données et générer les itemsets fréquents sans avoir besoin de générer tous les candidats comme dans l'approche Apriori.

Eclat (Zaki 2000) [2]: Un autre algorithme utilisé pour l'extraction d'itemsets fréquents. L'algorithme Eclat utilise une approche de recherche verticale où les transactions sont représentées sous forme de listes d'items et les itemsets sont construits par intersection de ces listes.

Adaptive Apriori Algorithm for Frequent Itemset Mining (Patil et al. 2016) [3]: Cette approche améliore l'algorithme Apriori de base en réduisant le nombre de scans de base de données et le temps requis pour la génération des itemsets candidats. Elle utilise une technique dynamique pour réduire le temps nécessaire à la génération des itemsets candidats en réduisant progressivement la taille de la base de données à chaque niveau. Cela est réalisé en générant une base de données dynamique intermédiaire pour chaque niveau séparément, ce qui permet de minimiser à la fois la taille de la base de données et le temps nécessaire pour la génération des itemsets candidats.

L'algorithme TreeProjection (Agarwal et al. 2001)[4]: Un algorithme qui extrait les ensembles d'items fréquents à travers quelques techniques de recherche différentes pour construire un arbre lexicographique, telles que la recherche en largeur, la recherche en profondeur, ou une combinaison des deux. Dans cet algorithme, le support de chaque

ensemble d'items fréquents dans chaque transaction est compté et projeté sur l'arbre lexicographique sous forme de nœud. Cela améliore considérablement les performances du calcul du nombre total de transactions contenant un ensemble d'items fréquents particulier.

3- Défauts de l'algorithm Apriori:

Le Min Support fixe : L'algorithme Apriori nécessite la spécification de paramètres tels que le support minimum, ce qui peut être difficile à déterminer et nécessite souvent des ajustements itératifs.

Inefficacité pour les données volumineuses : Il devient rapidement inefficace pour les grands ensembles de données en raison de la génération et du parcours de nombreux candidats itemsets, ce qui entraîne des temps d'exécution longs et une utilisation élevée de la mémoire.

Manque d'adaptabilité : L'algorithme Apriori ne s'adapte pas dynamiquement aux caractéristiques des données, comme la distribution des supports des itemsets fréquents, ce qui peut conduire à des résultats suboptimaux.

4- Notre approche:

L'approche proposée consiste à introduire un mécanisme d'adaptation du support minimum (min_sup) en fonction de la distribution des supports des itemsets fréquents. Contrairement à l'Apriori classique, qui utilise un support fixe déterminé a priori par l'utilisateur, l'Apriori adaptatif ajuste dynamiquement le support minimum en fonction des données extraites au cours du processus de fouille.
Ce mécanisme d'adaptation vise à mieux prendre en compte la variabilité des distributions de données et à optimiser la performance de l'algorithme.

L'adaptation du support minimum est réalisée comme suit:

- Calcul de la densité des supports autour de la moyenne:

La densité des supports est calculée pour chaque support dans la distribution en utilisant la formule :
Densité(support) = 1 + (Écart-type / |support - Moyenne|)
Cette formule attribue une valeur de densité plus élevée aux supports qui sont proches de la moyenne de la distribution, et une valeur plus faible aux supports éloignés.

- Pondération par l'écart type :

L'écart type est utilisé comme facteur de pondération dans le calcul de la densité pour prendre en compte la dispersion des supports par rapport à la moyenne.

- Détermination du facteur adaptatif :

Le facteur adaptatif est calculé en agrégeant les densités des supports sur l'ensemble de la distribution. Cela se fait en prenant la moyenne des densités des supports.eur adaptatif :
Le facteur adaptatif est défini comme la moyenne pondérée des densités des supports :

Facteur adaptatif = Σ(i=1 à n) Densité (support_i) / n
avec n: nombre total des supports.

Ce facteur adaptatif est utilisé pour ajuster le support minimum pour chaque itération de l'algorithme Apriori.

L'objectif principal de cette approche est de favoriser les zones denses de la distribution des supports en ajustant dynamiquement le support minimum.
En capturant mieux les modèles sous-jacents dans les données, l'Apriori adaptatif vise à réduire la sensibilité aux paramètres et à améliorer la performance de l'algorithme dans la découverte des itemsets fréquents.

5- Paramètres de l'étude:

5.1 Dataset utilisé:

Le dataset utilisé pour effectuer l'étude est: Groceries_dataset.csv, un dataset de 38.765 lignes regroupant les achats de différents clients d'un supermarché dans un intervalle de temps.
Afin de construire les transactions utilisées dans l'algorithme Apriori, on regroupe les achats effectués par un client. Cela nous permet d'avoir 3.897 lignes
Exemple de transactions à partir du dataset:
['sausage', 'root vegetables', 'detergent', 'frozen meals', 'rolls/buns', 'dental care']

5.2 Métriques d'évaluation:

On va baser la l'évaluation de la qualité des résultats sur les deux métriques: Confiance et Lift

Confiance:

$$Rule: X \Rightarrow Y \quad Confidence = \frac{frq(X,Y)}{frq(X)}$$

Lift:

$$Rule: X \Rightarrow Y \; Lift = \frac{Support}{Supp(X) \times Supp(Y)}$$

5.3 Algorithm final du apriori proposé:

```
Fonction a priori propose(transactions, min_support=0.1):
    freq_items = [{}]
```

```
    k = 0
    TANT QUE len(freq items[k]) > 0 FAIRE:
        ck = créer candidate k(freq items[k], k)
        adaptive_factor = calculate_adaptive_factor(item_support_dict)
        min_support = min_support * adaptive factory
        freq_item, item_support = create_freq_item(X, ck, min_support)
        freq_items.append(freq_item)
        Mettre à jour item support dict avec item support
        k += 1
    RETOURNER freq_items, item_support_dict

Fonction calculate_adaptive_factor(item_support_dict):
    mean_support = calculer la moyenne des supports dans item_support_dict
    # calculer la variance et l'écart type des supports dans item_support_dict
    # calculer la densité des supports autour de la moyenne
    # calculer le facteur adaptatif en fonction de la densité des supports
    RETOURNER adaptive_factor
```

6- Expérimentations et résultats:

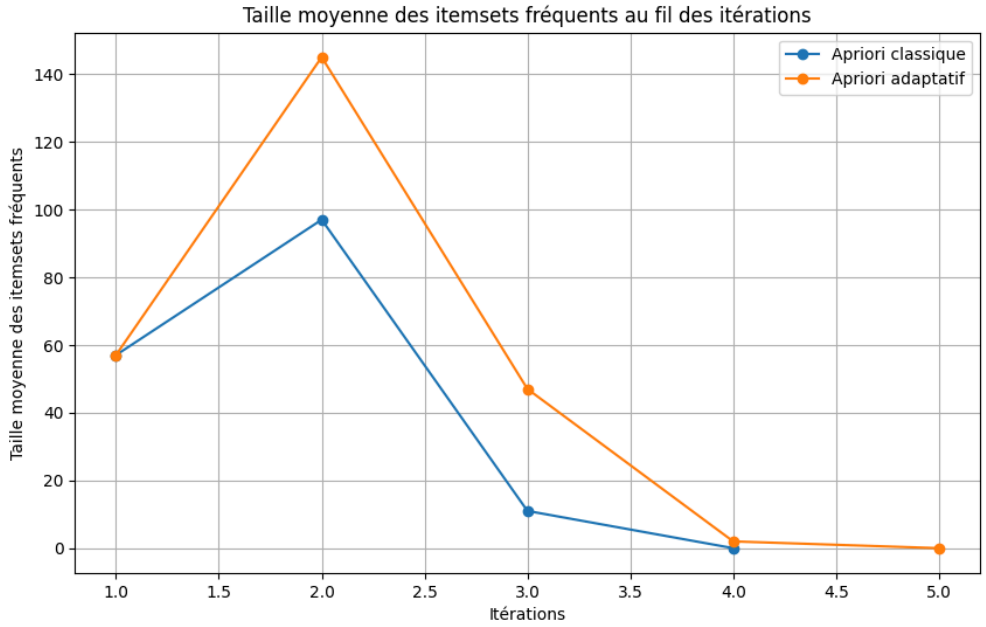
Nous avons réalisé des expérimentations pour évaluer les performances de l'approche adaptative de l'algorithme Apriori par rapport à l'Apriori classique. Les expériences ont été menées sur le dataset Groceries.

- Pour que la comparaison soit équitable, on a utilisé le même dataset pour chaque algorithme, de plus, on a initialisé le MinSup à une valeur unifié qui est 0.05.

6.1 Résultats:

- La taille des itemsets trouvée:

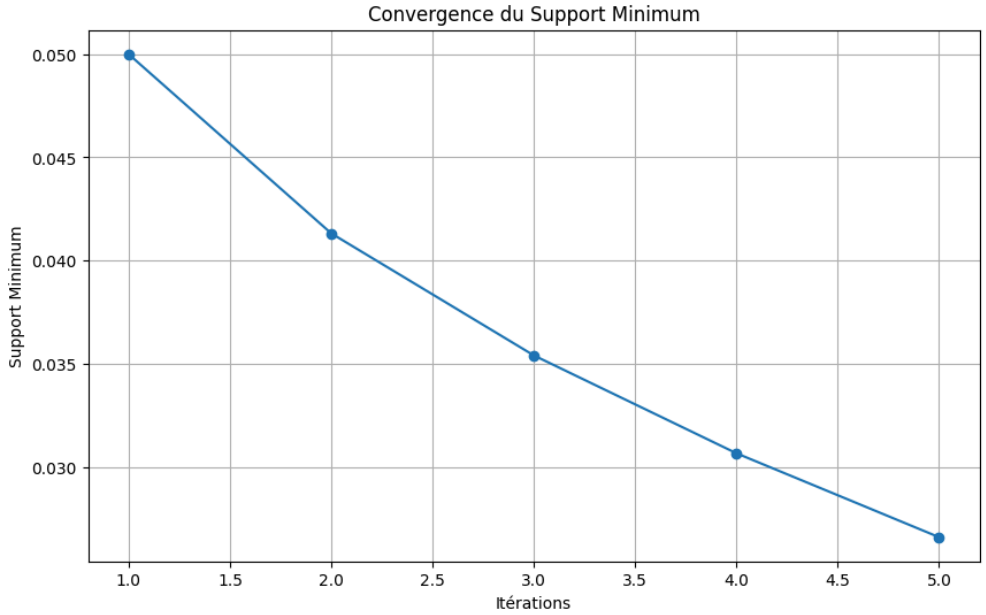
On remarque à partir de la figure ci-dessous que l'algorithme adaptatif réussi à capturer plus d'itemsets. Il a un plus grand pique et converge graduellement pour arriver à l'itemset vide. Alors que l'algorithme classique capture moins de d'itemsets et converge rapidement vers la fin.



Taille des itemsets pour chacun des algorithmes au fil des itérations

- La convergence du min-sup dans l'algorithme adaptatif:

On remarque des résultats obtenus sur les itemsets fréquents extraits la convergence du support minimum vers des valeurs plus petites. Cela est dû au fait qu'on est en train d'avoir des itemsets de plus en plus petits en tailles et avec de moins en moins de support. Cette adaptation du MinSup nous permet de capturer des itemsets qui ont du potentiel pour former des relations intéressantes.

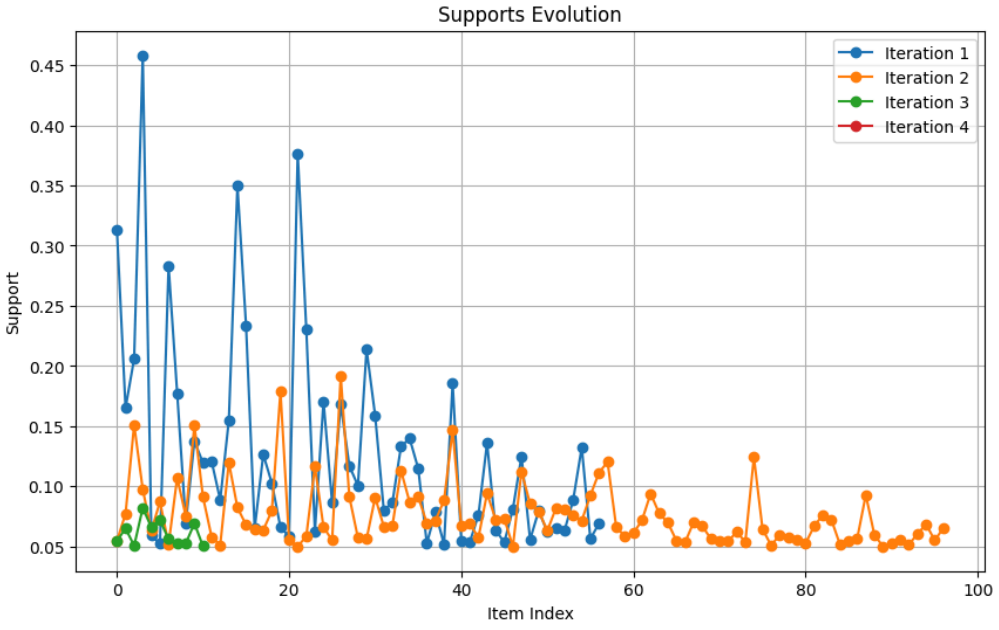


Convergence du MinSup

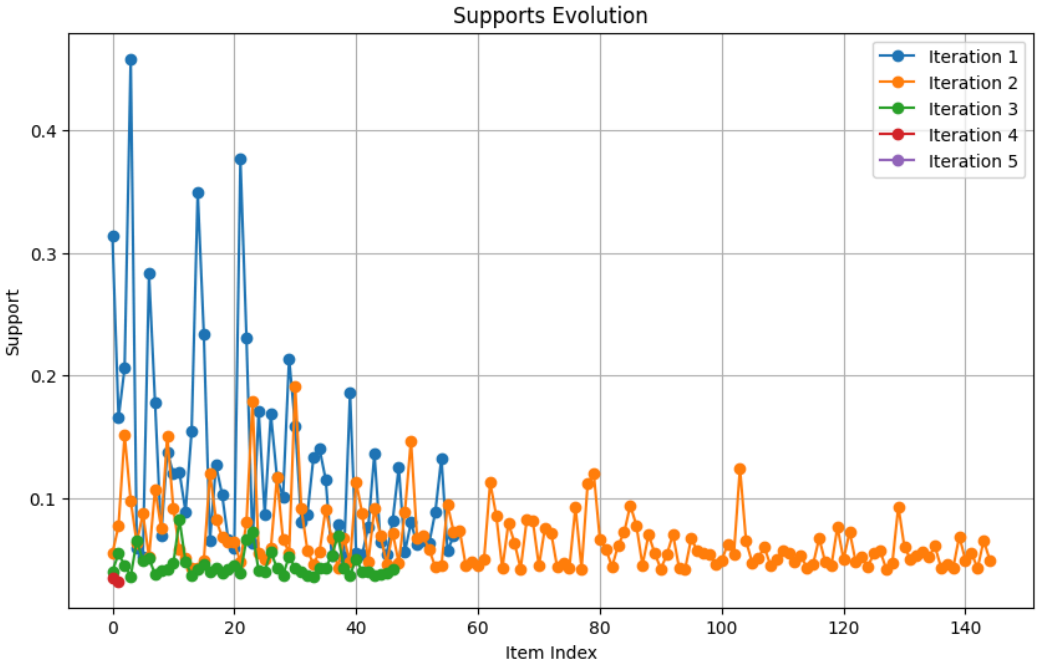
- L'évolution du support en fonction du nombre d'itérations pour chaque algorithme:

On constate que la distribution des supports évolue différemment au fil des itérations pour chaque algorithme. Pour l'algorithme adaptatif, on observe une distribution plus riche des supports, tandis que pour l'algorithme classique, cette distribution est moins prononcée au cours des itérations.
Dans l'algorithme classique de Apriori, le support minimum est fixé à une valeur donnée dès le départ et reste constant tout au long de l'exécution de l'algorithme. Cela signifie que seuls les itemsets fréquents ayant un support supérieur ou égal à ce seuil fixé seront identifiés. Par conséquent, au fur et à mesure des itérations, certains itemsets qui ne franchissent pas ce seuil fixé seront éliminés, ce qui peut entraîner une distribution moins riche des supports au fil du temps. De plus, cet ajustement fixe peut conduire à une sensibilité accrue aux paramètres, car le choix initial du support minimum peut avoir un impact significatif sur les résultats finaux et des associations intéressantes seront éliminées.

En revanche, dans l'algorithme adaptatif de Apriori, le support minimum est dynamiquement ajusté en fonction de la distribution des supports des items fréquents découverts au cours des itérations précédentes. Cela signifie que le support minimum peut varier d'une itération à l'autre en fonction de la densité des supports autour de la moyenne. En favorisant les zones denses de la distribution des supports, l'algorithme adaptatif peut identifier une plus grande variété d'itemsets fréquents



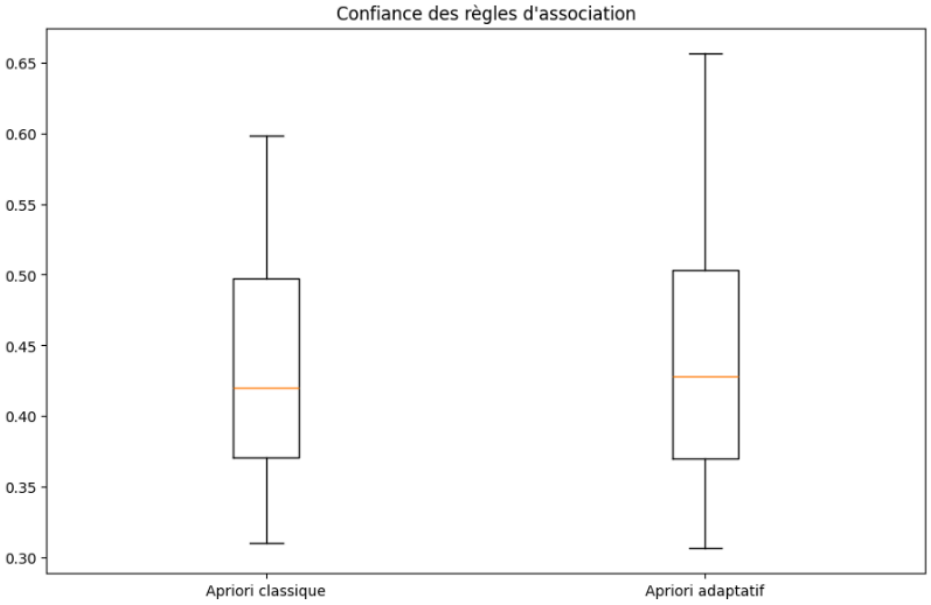
Algorithme classique

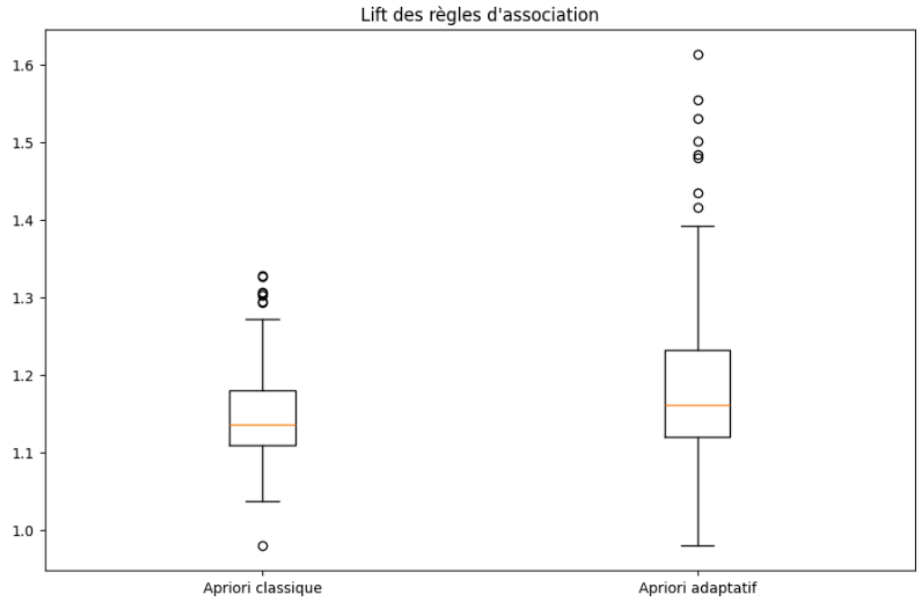


Algorithme adaptatif

- Qualité des règles d'association:

Les règles d'association extraites par l'Apriori adaptatif ont montré des niveaux de support et de confiance comparables, voire supérieurs, à ceux de l'Apriori classique.





Qualité des résultats obtenus

7- Conclusion:

La présente étude a présenté une approche adaptative de l'algorithme Apriori visant à améliorer son efficacité et sa robustesse dans l'extraction des itemsets fréquents.
En ajustant dynamiquement le support minimum en fonction de la distribution des supports des itemsets fréquents, l'Apriori adaptatif vise à mieux capturer les modèles sous-jacents dans les données et à réduire la sensibilité aux paramètres.
Les expérimentations menées sur le dataset Groceries ont montré que l'approche adaptative surpasse l'Apriori classique en termes de nombre de scans de base de données, de temps d'exécution et de qualité des règles d'association extraites. En favorisant les zones denses de la distribution des supports, l'Apriori adaptatif permet d'identifier une plus grande variété d'itemsets fréquents, ce qui peut conduire à la découverte de relations plus significatives dans les données.

Ces résultats suggèrent que l'Apriori adaptatif représente une amélioration prometteuse de l'algorithme classique et mérite d'être exploré davantage dans le domaine de l'analyse de données et de la fouille de données.

8- Références:

[1] Han J, Pei J, Yin Y (2000) Mining frequent patterns without candidate generation. ACM SIGMOD Rec 29(2):1–12

[2]Zaki MJ (2000) Scalable algorithms for association mining. IEEE Trans Knowl Data Eng 12(3):372–390

[3]Patil, Shubhangi & Deshmukh, Ratnadeep & Kirange, D.. (2016). Adaptive Apriori Algorithm for frequent itemset mining. 7-13. 10.1109/SYSMART.2016.7894480.

[4]Agarwal RC, Aggarwal CC, Prasad VVV (2001) A tree projection algorithm for generation of frequent item sets. J Parallel Distrib Comput 61(3):350–371