Article

High Dimensional Data Differential Privacy Protection Publishing Method Based on Association Analysis

Wei Shi 1,2, Xiaolei Zhang 1,*, Hao Chen 1 and Xing Zhang 1

1 School of Electronics & Information Engineering, Liaoning University of Technology, Jinzhou 121001, China; shiwei@lnut.edu.cn (W.S.); eurasip@gmail.com (H.C.)
2 Key Laboratory of Security for Network and Data in Industrial Internet of Liaoning Province, Jinzhou 121001, China
* Correspondence: 219807029@stu.lnut.edu.cn

Abstract: In order to solve the problem of privacy disclosure when publishing high-dimensional data and to protect the privacy of frequent itemsets in association rules, a high-dimensional data publishing method based on frequent itemsets of association rules (PDP Growth) is proposed. This method, in a distributed framework, utilizes rough set theory to improve the mining of association rules. It optimizes association analysis while reducing the dimensionality of high-dimensional data, eliminating more redundant attributes, and obtaining more concise frequent itemsets, and uses the exponential mechanism to protect the differential privacy of the simplest frequent itemset obtained, and effectively protects the privacy of the frequent itemset by adding Laplace noise to its support. The theory validates that the method satisfies the requirement of differential privacy protection. Experiments on multiple datasets show that this method can improve the efficiency of high-dimensional data mining and meet the privacy protection. Finally, the association analysis results that meet the requirements are published.

Keywords: high-dimensional data; differential privacy; distributed framework; association rule; frequent itemsets; rough set

1. Introduction

With the wide application of big data technology, more and more high-dimensional data need to be processed. In the process of mining them based on association rules, analysts often face the original high-dimensional data which contain a lot of sensitive information. If the high-dimensional data are directly used for publishing after mining based on association rules, and the mining results are not privacy protected, it is very easy to cause the privacy disclosure of target users. For example, when an association rule is applied, the products purchased by users change, and the resulting frequent items also change. Then, the attacker can infer the user’s purchase information from the frequent items, thus leading to the disclosure of the user’s personal information. In addition, there are also user’s natural person identity and identification marks, service content information, consumption bills, private social content, and location data, among others. A large number of researchers have conducted privacy protection on association rule mining results based on homomorphic encryption technology [1]. Homomorphic encryption [2] allows data analysts to perform specific algebraic operations on the encrypted ciphertext, and the results of the ciphertext operations, after decryption, are the same as those obtained by directly operating on the plaintext. K-anonymity [3] and its extended anonymity protection models have been extensively studied and have had a significant impact. This privacy protection model defines data attributes that are highly correlated with the background knowledge of attackers as quasi-identifiers. By classifying and compressing the quasi-identifiers in the data, all data are divided into multiple equivalence classes, where
the quasi-identifiers of the data are completely identical, thereby hiding a record within a group of data records. This type of privacy protection model that achieves rule hiding through grouping is referred to as a group-based privacy protection model. However, studies have found two major flaws in such privacy protection models [4,5]. Firstly, these models have insufficient security, as they always have vulnerabilities that can be exploited by new attacks [6–8]. The essence of this issue lies in the fact that the security of such models is related to the background knowledge of attackers, but group-based privacy protection models cannot fully define all possible background knowledge.

The traditional privacy protection methods for association rules are mainly implemented through data encryption and limited publishing. When the amount of data are large, the operation efficiency of the former is low, while the latter is less robust and is vulnerable to background maximization knowledge attacks. Differential privacy can resist any background knowledge attack, so differential privacy can be used to provide protection, but high-dimensional data often need to add too much noise, making the published results less available. Therefore, in the distributed framework, this paper improves the mining method based on association rules through rough set theory to obtain the simplest frequent itemsets, and then adds differential privacy protection to frequent itemsets and support, finally realizing the privacy publishing of high-dimensional data association rule itemsets.

The contributions of this paper are as follows:

1. Propose a method for high-dimensional data based on frequent itemsets under a distributed framework.
2. Address the issue of memory overflow when dealing with large-scale data by utilizing rough set theory.
3. Introduce an optimization method for attribute information entropy in a distributed environment. This method performs a secondary screening on the data after dimensionality reduction using rough set theory, eliminating redundant items and improving the efficiency of the algorithm.
4. Apply differential privacy to protect the data and demonstrate that the proposed algorithm satisfies the differential privacy mechanism.

2. Related Work

The research on association analysis was first proposed by Rakesh Agrawal in 1993 in the analysis of the database itemset relationship. The classic case of association rules is the “shopping basket analysis” event, that is, it is found that most people who buy diapers like to buy beer. In the process of mining based on association rules, there is plenty of user-sensitive information that needs to be protected. The traditional association analysis method [9–11] is not applicable in the big data environment. Because the processing capacity of a single machine is limited, when facing massive data, it often causes memory overflow, low execution efficiency, and other problems. In 2015, Zhang Xu [12] and others put forward a parallel association rule mining algorithm based on GPU, which can effectively use the support of GPU parallel computing candidate sets to speed up computing efficiency. Chon et al. [13] proposed the BIGMiner algorithm based on MapReduce to generate candidate k + 1 itemsets and local support, broadcast its network overhead, and improve the scalability of the algorithm. The literature [14] proposed a frequent pattern mining method based on multi project support, which recursively mined frequent patterns by constructing the suffix conditional pattern base and conditional pattern tree of MISFP-Tree. In 2018, Zhu Haodong et al. [15] applied the FP Growth algorithm to the distributed framework Hadoop, realizing parallel computing of the FP Growth algorithm. However, frequent interactions during execution caused excessive memory overhead. In 2019, Jiang Dongjie [16] and others proposed a tree structure called UFP Tree to reduce frequent data interaction through compression. In 2021, Zhao Xincan [17] designed a PJPFP Tree structure to optimize the stored procedures, reducing the computing time. However, this
method is also implemented based on MapReduce framework, which cannot be compared with Spark in terms of performance. To ensure the privacy of association rule mining, Ning Bo et al. [18] quantified the privacy leakage caused by association attributes by using information entropy and Markov chain. At present, many association rule methods satisfying differential privacy [19–25] have been proposed, but there are still shortcomings in the balance between privacy and availability. As mentioned in the literature [26], most methods have a poor processing effect for high-dimensional data. In 2018, Yu Yihan et al. [27] implemented the association rule differential privacy mining algorithm based on the combination of parallel computing and differential privacy methods under the distributed Hadoop framework. However, Hadoop does not adopt the memory level processing method; each read requires storing the data in HDFS, so a large number of I/O operations will be generated, which takes a long time. Xu [28] proposed a differentially private algorithm called PFP-growth for frequent itemset mining. This algorithm employs a split strategy for longer records, which reduces the amount of added noise and enhances the usability of the mining results. However, this method causes damage to the original data, thereby reducing the usability of the final mining results. Wang [29] introduced a differentially private algorithm called PrivSuper for frequent itemset mining. This algorithm utilizes the exponential mechanism to select the most frequent k itemsets and adds Laplace noise to the selected itemsets. PrivSuper satisfies differential privacy and offers a high level of security. However, as the value of k increases, the amount of added noise also increases significantly, resulting in lower usability of the mining results. From this, it can be seen that current methods do not effectively balance the usability and privacy of mining results for massive association rules. Firstly, high-dimensional data suffer from the curse of dimensionality. As the dimensionality of the data increases, the volume of the data space grows exponentially, leading to sparsity in the data. The number of variable combinations also explodes, making optimization problems and parameter tuning challenging. As a result, general methods struggle to be effective, leading to low processing efficiency.

Secondly, with the increase in data dimensions, the degree of correlation between data also increases correspondingly. However, the protection level of differential privacy for massive correlated data is often very low. The reason is that the increase in dimensions leads to an increase of noise. If the traditional noise adding method is used, the correlation between attributes may be damaged. Therefore, this paper proposes PDP Growth, a high-dimensional data publishing method based on association rules and frequent itemsets, which satisfies differential privacy protection under the Spark distributed framework. The contributions of this paper mainly include the following:

(1) Under the distributed framework, this paper proposes using rough set theory to improve the mining method of association rules, reduce the dimensions of datasets while optimizing the analysis of association rules, eliminate more redundant attributes, and obtain the simplest frequent itemset, which effectively solves the problem of low efficiency of traditional high-dimensional data processing.

(2) Then, a scoring function based on real support is designed. First, the index mechanism of differential privacy is used to randomly score the simplest frequent itemsets obtained to meet differential privacy. Then, Laplace noise is added to the support of frequent itemsets to achieve differential privacy protection of the associated data in a distributed environment.

3. Differential Privacy Protection

Differential privacy is to disturb the original dataset by adding appropriate noise to the output results, so as to achieve data privacy protection. The strict mathematical definition of differential privacy is as follows:
Theorem 1 (Differential Privacy). For the sum of sibling datasets (with a difference of one record) and all their subsets $S$, if there is a random algorithm $A$, it satisfies the following formula:

$$Pr[A(D_1) \in S] \leq e^\varepsilon Pr[A(D_2) \in S] \quad (1)$$

Then, algorithm $A$ satisfies $\varepsilon$-differential privacy, which $\varepsilon$ is the privacy budget (measuring the degree of privacy protection). Generally, differential privacy can be achieved through Laplace mechanism (for numerical data) and exponential mechanism (for discrete data). Because both mechanisms rely on global sensitivity, define global sensitivity first.

Definition 1 (Global Sensitivity). Given the query function $f$, $D_1$ is the input dataset and $D_2$ is the output dataset. On any pair of $D_1$ and $D_2$, the global sensitivity of function $f$ is as follows:

$$\Delta f = \max_{D_1,D_2} \|f(D_1) - f(D_2)\|_1 \quad (2)$$

where $\|f(D_1) - f(D_2)\|_1$ is the first order distance between $f(D_1)$ and $f(D_2)$. When the input dataset changes, the global sensitivity can measure the impact of the function on the corresponding output results.

Theorem 2 (Laplace Mechanism). Given dataset $D$, privacy budget $\varepsilon$, the global sensitivity $\Delta f$ of function $f$, when the output of $f$ satisfies the following formula:

$$A(D) = f(D) + \text{Lap}(\Delta f / \varepsilon) \quad (3)$$

Then, algorithm $A$ is said to satisfy $\varepsilon$ difference privacy, where $\text{Lap}(\Delta f / \varepsilon)$ is random noise satisfying Laplace distribution, where lambda ($\Delta f / \varepsilon$) is a scale parameter that determines the width of the distribution. A larger lambda value corresponds to a wider distribution, while a smaller lambda value corresponds to a narrower distribution. The $x$-axis represents the range of values for the random variable, and the $y$-axis represents the probability density, which is the ratio of the probability of an event occurring in a certain interval to the length of that interval, as shown in Figure 1.

![Laplace distribution](image)

Figure 1. Laplace distribution.

Theorem 3 (Exponential Mechanism). Given dataset $D$, the output is an entity object $\gamma \in R, \mu(D, R)$, which is the availability function, and $\Delta \mu$ is the sensitivity of function $\mu(x, R)$. If $r$ is selected from the input and output with a probability proportional to $\exp[\frac{-\Delta \mu}{\varepsilon}]$, then algorithm
A satisfies $\varepsilon$-differential privacy. The exponential mechanism is mainly used to deal with non-numerical attributes.

Differential privacy protection has sequential combination and parallel combination, which play an important role in proving whether the algorithm satisfies differential privacy and in the process of privacy budget allocation. Property 1 (sequence combination): if $n$ random algorithms $A_1, A_2, ..., A_n, A_i (1 \leq i \leq n)$ provide $\varepsilon_i$-differential privacy protection, then there is a sequence combination property on the same dataset $M$, that is, \{A_1, A_2, ..., A_n\} provides $\varepsilon$-differential privacy protection on $M$, where $\varepsilon = \sum_{i=1}^{n} \varepsilon_i$.

Property 2 (parallel combination): There are random algorithms $A$ and dataset $M$. For disjoint subsets $M_1, M_2, ..., M_n$, if algorithm $A$ satisfies $\varepsilon$-differential privacy protection, then the combination operation of algorithm $A$ on $\{M_1, M_2, ..., M_n\}$ also satisfies $\varepsilon$-differential privacy protection.

These differential privacy mechanisms provide means to protect individual privacy by introducing noise or adjusting the selection probability of the output results, thus achieving privacy protection. The degree of privacy protection is controlled by global sensitivity and privacy parameters. In practical applications of differential privacy, it is also necessary to consider the impact of the composition properties of differential privacy.

4. Association Analysis

Association analysis is a research hotspot in the field of data mining. The main purpose of association analysis is to discover the association between data. The association rules obtained through analysis can help businesses make reasonable decisions, improve sales, and increase turnover.

Association analysis generally involves two processes: finding frequent itemsets (support) from datasets, and generating association rules (confidence) from frequent itemsets.

Definition 1 (Support). If there is an itemset $X$, the probability of $X$ appearing in dataset $D$ is called the support of itemset $X$, and it is recorded as support ($X$).

$$\text{support}(X) = \frac{\text{count}(X)}{\text{count}(D)} = P(X)$$  \hspace{1cm} (4)$$

where $\text{count}(X)$ represents the number of occurrences of itemset $X$, and $\text{count}(D)$ represents the total number of data. When the support is greater than the minimum support (support_min), the itemset generated is called a frequent itemset.

Definition 2 (Confidence). The probability of occurrence of itemset $B$ when itemset $A$ occurs is called confidence and is recorded as confidence($A \Rightarrow B$).

$$\text{confidence}(A \Rightarrow B) = \frac{\text{count}(A \cup B)}{\text{count}(A)} = P(B|A)$$  \hspace{1cm} (5)$$

Confidence reflects the reliability of association rules. If the confidence level of association rule $\{A \Rightarrow B\}$ is high, it means that when $A$ occurs, $B$ will occur with a high probability. Applying it to reality can bring great research value, for example, bundling goods can increase turnover.

Association rules can be divided into single-layer and multi-layer according to hierarchy; It can also be divided into single dimension and multi dimension by dimension. The Apriori algorithm is a classical association analysis algorithm, which is generally only used for single dimension and single-layer association rules.

Qiu et al. [30] first proposed a YAFIM algorithm by combining an Apriori algorithm with a Spark computing framework. The algorithm consists of two stages: in the first
stage, the read datasets are written to the HDFS distributed file system, and then all frequent 1-itemsets are found after being converted by Spark’s RDD conversion operator. In the second stage, optimize the stored procedure, use the hash tree to store data and iterate, and generate frequent (k + 1)-itemsets from frequent k-itemsets. However, with the increase in data size, a large number of temporary itemsets are generated, and the efficiency of the algorithm is low. To solve this problem, this paper combines the attribute reduction of rough sets to optimize the generation of redundant itemsets, which can play a better role.

For multi-layer and more complex cases, the second FP Growth classical algorithm is generally used to achieve multi-level and multi-dimensional association analysis. FP Growth is a frequent pattern growth algorithm proposed by Han et al. in 2000. It solves the problem of the high memory share and long computing time of Apriori algorithm when computing large datasets. The design idea of the FP Growth algorithm is to adopt the divide and conquer strategy. It only needs to scan the database twice to obtain all frequent itemsets. Compress all possible frequent itemsets into a frequent pattern tree (PF Tree), mine the corresponding frequent itemsets by traversing PF Tree, and then generate all association relationships through PF Tree. Finally, the results will be used for data analysis and data publishing.

An FP Growth algorithm is generally divided into two steps: building the FP Tree and mining frequent itemsets. The specific process is as follows:

Step 1: Build FP Tree.

1. First, scan dataset D for the first time and build an item header table H, which is composed of items, support, and node chains. Each item in the dataset is marked as a temporary l-itemset (candidate itemset), marked as \( L_{\text{temp}} \), and then compared with the preset minimum support (support_min); if it is less than support_min, the min itemset is deleted, while the remaining itemsets are retained, and frequent 1-itemsets are obtained, which are recorded as \( L_1 \). Then, dataset D is sorted in descending order according to the support of the frequent 1-itemset, and the sorted dataset D is recorded as \( D' \).

2. Scan dataset \( D' \) for the second time, create the root node (root), mark it as null, and build the branch of the FP Tree. First, add the first transaction directly to the branch, and set its corresponding item count value to 1. Start from the second transaction and add it to FP Tree circularly. If the corresponding node already exists, directly add one to the count of each item in the corresponding existing node. If it does not exist, initialize a new branch and set its corresponding item count value to 1 until the last item is scanned. The cycle ends and a PF Tree is generated.

Step 2: Mining frequent itemsets

1. Rebuild a new itemset table, which consists of four parts: items (with the last element of the header table as the header and the second element of the header table as the tail), conditional pattern base, conditional FP Tree, and frequent itemsets.

2. According to the arrangement order in the item header table, start from the tail of the item header table to calculate the conditional mode base. Use the conditional pattern base to rebuild a new FP Tree, which is called the conditional FP Tree. Construction method: For the conditional pattern base corresponding to each frequent item, construct the condition FP Tree according to the comparison between the support count size and the threshold value.

3. Using the idea of recursion, all frequent itemsets \( L_k \) are mined according to the constructed condition FP Tree.

The first step of the FP Growth algorithm is to build the FP Tree item header table \( H \), which can quickly access all of the items that make up the FP Tree, and also save the count information of the support of each item of the FP Tree. The second step of the FP Growth algorithm is to find all the prefix paths from the item to the root node in the table with the
item root node as the endpoint. The set of these prefix paths constitutes the conditional pattern base.

The advantage of the FP Growth algorithm is that it only needs to scan the database twice to obtain all frequent itemsets. Unlike the Apriori algorithm, it needs to generate a large number of candidate itemsets, and it also takes up too much space with a large amount of computation. However, the shortcomings of the FP Growth algorithm are also obvious. When processing massive data, the memory of a single machine is limited. With the increase in dimensions, the memory consumed by storage will also increase, which can easily cause memory overflow and program crash. Therefore, a parallel PF Growth algorithm based on a distributed platform has emerged. In 2016, Fang et al. [31] proposed an IPFP Growth algorithm by combining the FP Growth algorithm with Spark distributed framework for the first time. The algorithm consists of three parts: first, write the read dataset to the HDFS distributed file system, and find all frequent 1-itemsets and arrange them in descending order after being converted by Spark’s RDD conversion operator. The second step is to divide the dataset into several groups in an equal way, and mine frequent itemsets on each group.

Step 3: Consolidate all of the results obtained from grouping into a single file, sorting them based on the length and lexicographic order of frequent itemsets. In the end, a global frequent itemset list is obtained, which includes all of the groups’ frequent itemsets along with their support. However, this algorithm does not eliminate the redundancy of data. When the data size are too large, the conditions for data generation PF Tree are too many, which will still lead to memory overflow. Therefore, this paper considers improving the rough set first, and then adjusts the grouping structure to realize the application of large data scale.

As far as big data are concerned, it is appropriate to use the FP Tree algorithm to obtain multidimensional rules, but there are still problems such as long mining time and a large number of invalid or redundant rules. Therefore, in order to solve the above problems, this paper proposes an attribute information entropy optimization method in a distributed environment, which can filter the reduced dimension data of a rough set twice, eliminate redundant items, and improve the execution efficiency of the algorithm.

This section introduces the importance of association analysis in data mining and its related knowledge content, including two processes of association analysis: finding frequent itemsets and discovering association rules from frequent itemsets. Then, we learn about the FP-Growth algorithm from the research of several scholars, and describe the workflow of the FP-Growth algorithm, as well as its advantages and disadvantages. Some improvements are also proposed to address its limitations.

5. Implementation of a Privacy Protection Method for Distributed Association Analysis

5.1. Association Analysis Method Based on Rough Set Optimization

Section 3 introduces two commonly used association analysis methods and gives the general implementation flow of the distributed PF Growth algorithm. However, both the Apriori algorithm and PF Tree algorithm have certain problems, because many rules in association analysis are uncertain, and rough set theory can only find uncertain rules without relying on any prior knowledge. Therefore, this paper considers applying rough set theory to association analysis, optimizing the association analysis while reducing the dimensions of high-dimensional data, eliminating more redundant attributes, and obtaining a more concise frequent itemset.

There are three main types of feature selection methods in rough set, which are the exhaustive method, heuristic method, and random method.

The exhaustion method is to directly calculate all feature subsets, and then select the feature subset with the least cardinality from the feature subset. It is often solved by constructing the differentiation function between different objects. However, with the expansion of the data scale, the feature space will also increase. The number of logical
expressions obtained is too great and there are a lot of duplicates, so the calculation amount is too large. The heuristic method is an approximate solution method, and the implementation process is relatively simple. It redefines the feature importance with different average standards, and regards it as heuristic information. Starting from an empty set, it uses backward deletion or forward selection to obtain the optimal feature subset. The random method is a relatively new method, which uses the GA algorithm and simulates the annealing algorithm to construct the minimum reduction.

This paper adopts a heuristic method. The heuristic rough set feature selection algorithm mainly has three improved ideas: feature selection method based on positive field, feature selection algorithm based on conditional information entropy, and feature selection method based on mutual information. Among them, the feature selection algorithm based on positive domain [32] is the simplest.

Algorithm 1 redefines the importance of features as the change degree of the positive domain after adding features. The algorithm first sets the optimal feature subset T to an empty set, and then adds the feature importance to subset T in the order of large to small, until the attribute dependency of the added feature subset is equal to that of the original feature subset. The algorithm ends, and finally outputs the optimal feature subset that meets the conditions. The following is a parallel frequent itemset mining algorithm based on rough set theory optimization (Algorithm 2).

**Algorithm 1** Feature selection algorithm based on positive domain

**Input:** DT = (U, C U D)

**Output:** Optimal characteristic subset T

**Step 1:** T = Ø

**Step 2:** Order S = T

**Step 3:** ∀x ∈ C − T, when γ_{TU\{x\}}(D) > γ_{T}(D), there is S = T ⋃ \{x\}

**Step 4:** T = S

**Step 5:** When γ_{TU\{x\}}(D) > γ_{T}(D),

- then jump to step 6, otherwise return to step 2

**Step 6:** Enter T to end the algorithm

**Algorithm 2** Parallel frequent itemset mining algorithm based on rough set theory optimization

**Input:** Dataset D, minimum support\_min;

**Output:** minimalist frequent itemsets T_{k};

**Step 1:** Clean, transform and discretize dataset D to obtain original itemset T;

**Step 2:** Count the number of occurrences of each item in item T;

- If the number of times is less than the minimum support number, delete it,
- If it is greater than, keep it, sort it, once, and skip to Step 3;

**Step 3:** Get the decision table with attributes;

- If it is a core attribute, parallel FP growth analysis based on Spark will be performed directly;
- If it is a non core attribute, perform a heuristic attribute reduction, and then perform parallel FP growth analysis based on Spark for the reduction results;

**Step 4:** After parallel FP Growth analysis, attribute value reduction is performed again for the obtained frequent itemset;

**Step 5:** Final Output Minimal frequent Itemset T_{k};

Through association analysis mining, we will find a large number of hidden associations or correlations in the dataset, and then we can infer the association attribute set of some sensitive attributes. In this paper, we add reasonable noise to the attribute set to achieve differential privacy protection.
5.2. Privacy Preserving Method of Associated Itemsets Based on Rough Sets

The above section optimizes the method of generating associated itemsets. However, if the results are published without any processing, malicious attackers can obtain specific user data through the analysis of the results, which may still cause the disclosure of sensitive information of private data.

Therefore, this section continues to improve on the above association analysis methods to achieve differential privacy protection in a distributed environment. Specifically, it includes using the index mechanism of differential privacy to define the scoring function, using the scoring function to achieve $\epsilon_1$ differential privacy for the simplest frequent itemset, and adding Laplacian distribution to support frequent itemsets to achieve $\epsilon_2$ compliant differential privacy protection. Finally, the noisy correlation analysis results meeting the requirements of differential privacy protection are published.

Figure 2 shows the flow chart of the association analysis protection method based on differential privacy. First, input dataset $D$ and the corresponding minimum support (min_sup) and privacy budget $\epsilon$, then use the association analysis method based on rough set optimization proposed in subsection 4 to obtain all of the simplest frequent itemsets. At the same time, use the built distributed data processing platform Spark to process the parallel differential privacy method to generate the frequent itemsets that meet privacy protection based on the exponential mechanism and the noisy support based on the Laplace mechanism. Finally, the top-k itemset satisfying the differential privacy requirement is output, and the algorithm ends.

![Flow chart of association analysis protection method based on differential privacy.](image)

The parallel differential privacy algorithm mainly includes two parts. The first part uses the exponential mechanism to output k frequent itemsets satisfying differential
privacy, and the second part uses the Laplace mechanism to achieve support privacy protection.

The first part of the index mechanism depends on the selection of the scoring function. Here, according to the support degree of the real simplest frequent items, the set is arranged from large to small, and the scoring process is performed in turn according to the order. Then, according to the final actual score, k itemsets are extracted from the real frequent itemset to form a noisy k-itemset output.

Definition scoring function (Scoring-Function) \( u(D, r_j) = \text{sort}(\text{Sup}(k, r_j)) \) indicates the position of itemset \( s \) sorted in ascending order of support. Therefore, the PDP Growth algorithm is given.

The Algorithm 3 is divided into two stages. The first stage: because \( k \) itemsets are selected to satisfy differential privacy, \( k \) times of selection are required, so \( \Delta \mu = k \). The weight value of \( r_j \) obtained by using the index mechanism is \( \exp\left(\frac{\epsilon \mu(D, r_j)}{2K \Delta \mu}\right) \). To prove that the algorithm satisfies \( \epsilon - \)differential privacy, it is necessary to prove that one cycle in step 4 satisfies differential privacy and then according to the serial combination property of two differential privacy sections, the algorithm satisfies \( \epsilon_1 - \)differential privacy. It is proven that in step 7, one cycle satisfies differential privacy, and then according to the serial combination of two differential privacy sections, the algorithm satisfies \( \epsilon_2 - \)differential privacy. Finally, according to the combination property of differential privacy, it is indirectly proven that the algorithm satisfies differential privacy as a whole. The specific certification process is as follows:

Stage 1: Let \( R_i \) be the \( i \)th cycle of dataset \( D \) and \( R'_i \) represents the \( i \)th cycle of adjacent dataset \( D' \). According to the differential privacy index mechanism, the following formula can be deduced:

\[
\frac{p_i[R_i = r]}{p'_i[R'_i = r]} = \frac{\exp\left(\frac{\epsilon_1 \mu(D, r)}{2K \Delta \mu}\right)}{\exp\left(\frac{\epsilon_1 \mu(D', r')}{2K \Delta \mu}\right)} = \frac{\sum_{r'' \in R} \exp\left(\frac{\epsilon_1 \mu(D', r'')}{2K \Delta \mu}\right)}{\exp\left(\frac{\epsilon_1 \mu(D', r')}{{2K \Delta \mu}}\right)} \leq \exp\left(\frac{\epsilon_1}{k}\right)
\]

It can be found that each cycle process in the above formula satisfies \( \epsilon_1 / k \)-differential privacy protection. It can be seen from the serial nature of property 1 in Section 2 that after \( k \) cycles are completed, \( \epsilon_1 \)-differential privacy protection is satisfied.

The second stage: after the protection of differential privacy index mechanism, the frequent itemsets generated meet the differential privacy requirements. As the support of all itemsets is based on the real support as the weight, if directly released, users’ privacy data may still be disclosed. Therefore, for the support of itemsets, it is necessary to add a privacy budget of \( \epsilon_2 \) conforming to the Laplace distribution to achieve differential privacy protection.
Algorithm 3 Parallel differential privacy preserving algorithm PDP

- Growth

Input: Minimal frequent itemset $T_p$, Privacy budget $\epsilon_1$, Privacy budget $\epsilon_2$

Output: $K$-itemset $T^*_k$ satisfying differential privacy, Corresponding noise support $N^*_k$

Step 1: Count the number of the simplest frequent itemsets (count),

and initialize two empty collections, One for storing $k$ itemsets $T^*_k$, The other is used to store the corresponding support $N^*_k$;

Step 2: Traverse all itemsets, Get the area score of each $r_i$ and the corresponding weight value based on $\mu(D, r_i) = \text{sort} \left( \text{Sup}(k, r_i) \right)$;

Step 3: The obtain weight values are sorted in descending order, and the total weight is defined as the sum of each weight $\sum_{i=1}^{\text{count}} r_i$;

Step 4: Cyclic extraction accounts for the first $k$ items in each total and adds them to the new set;

Step 5: End the loop and output the $K$-item set satisfying the differential privacy $T^*_k$;

Step 6: Calculate the number $k$ of all frequent itemsets contained in $T^*_k$;

Step 7: Cyclically add Laplacian noise $\epsilon_2/k$ to the support corresponding to each frequent itemset, and add it to set $N^*_k$;

Step 8: End the cycle and output the set $N^*_k$ corresponding to the noisy frequent support;

The certification process is as follows:

Prove: The second stage is satisfaction $\epsilon_2$-differential privacy protection. Similar to the exponential mechanism, $R_i$ is the $i$th cycle of dataset $D$, and $R'_i$ b is the result of adjacent dataset $D'$. According to the Laplace mechanism of differential privacy, the following formula can be deduced:

$$\frac{P_i[R_i = r]}{P'_i[R'_i = r]} = \prod_{i=1}^{k} \frac{\exp \left( - \frac{\epsilon_2 |R_i - Z_i|}{k} \right)}{\exp \left( - \frac{\epsilon_2 |Z_i|}{k} \right)}$$

$$= \prod_{i=1}^{k} \exp \left( \frac{\epsilon_2 |R_i - Z_i|}{k} \right)$$

$$\leq \prod_{i=1}^{k} \exp \left( \frac{\epsilon_2 |R' - K'|}{k} \right)$$

$$= \exp \left( \frac{\epsilon_2 |K|}{k} \right)$$

$$\leq \exp \left( \frac{\epsilon_2}{k} \right)$$

(7)

It can be found that each of the above cycle processes satisfies $\epsilon_2/k$-differential privacy protection. It can be seen from the serial nature that $\epsilon_2$-differential privacy protection is satisfied after $k$ cycles are completed.

To sum up, as the first stage and the second stage both meet the requirements of differential privacy protection, the algorithm proposed in this chapter also meets the requirements of $\epsilon$-differential privacy protection.

6. Experimental Results and Analysis

6.1. Experimental Environment and Dataset

In order to verify the effectiveness of the algorithm in this paper, an experimental platform was built to connect 1000 Mbit/s Ethernet. The cluster environment of Spark on Yarn was built using Hadoop 2.7.2 and Spark 3.0. Python was used as the development environment. The datasets used in the experiment were Pumsb Star dataset, Ta Feng dataset, and retail dataset. The dataset description is shown in Table 1, where $D$ is the
dataset, \( |D| \) is the quantity, \( n \) is the dimension of the data, and \( \text{Avg} \ (N) \) is the average length of the data. Pumsb Star dataset is widely used in frequent pattern mining, and transaction items with more than 80% support are finally filtered out. The Ta Feng dataset is a real user consumption record dataset published by Recsys Conference. The product categories include food, fruit, office supplies, etc. Retail datasets are the sales records of a retail store on the UCI website. They are commonly used datasets for association rule mining.

Table 1. Dataset.

| D     | \( |D| \)  | \( n \)  | \( \text{Avg} \ (N) \) |
|-------|----------|--------|-----------------|
| Pumsb-Star | 49,046   | 2088   | 50              |
| Ta-Feng   | 32,266   | 2012   | 17.7            |
| retail    | 88,162   | 16,470 | 10.3            |

6.2. Algorithm Evaluation Criteria

The algorithm is evaluated by three indicators: the parallel efficiency of the algorithm is tested by using the acceleration ratio. That is, the ratio of time spent by a single processor and multiple processors under the same workload. The higher the value is, the less time is consumed in parallel computing, and the higher the execution efficiency is. The formula is as follows:

\[
\text{speedup} = \frac{T_s}{T_c}
\]  

where \( T_s \) and \( T_c \) represents the running time of a single processor environment and a multiprocessor environment, respectively. \( F\text{score} \) and average relative error \( \text{ARE} \) are used to measure the accuracy of the algorithm. The \( F\text{score} \) is defined as follows:

\[
F\text{score} = \frac{1}{k} |T_k \cap T'_k|
\]

where \( T_k \) and \( T'_k \) represent the original frequent itemsets and noisy frequent itemsets, respectively. Generally, the larger the \( F\text{score} \), the higher the accuracy of the top-\( k \) itemset returned. The average relative error is generally expressed in \( \text{ARE} \), which is mainly used to measure the availability of mining results, and is defined as follows:

\[
\text{ARE} = \frac{1}{k} \sum_{i \in N_k} \frac{|NS(k_i, N_k) - N'\cdot S(K_i, N'_k)|}{NS(k_i, N_k)}
\]

where \( NS(k_i, N_k) \) and \( N'\cdot S(K_i, N'_k) \) represent the true support and noisy support of a \( k \)-degree itemset, respectively. In general, the smaller the \( \text{ARE} \), the higher the availability of mining results.

6.3. Experimental Results and Analysis

In order to test the execution efficiency of the PDP Growth algorithm, the speedup ratio of the algorithm is compared with that of IPFP Growth algorithm on three datasets. The minimum support threshold is 1000, and different nodes are set to complete the comparison between the PDP Growth algorithm and IPFP Growth algorithm. The two algorithms are executed 10 times independently, and the average value is taken. The results are shown in Figure 3.
Figure 3. Effect of node number on acceleration ratio: (a) Pumsb-Star dataset, (b) Tai-Feng dataset, and (c) retail dataset.
It can be seen from Figure 3 that the efficiency of PDP Growth is better than that of the IPFP Growth algorithm on the three datasets. PDP Growth performs better in retail datasets with large data dimensions and Pumsb Star datasets with long average data length than in Ta Feng datasets with relatively low dimensions, with an average increase of 1.48 per node. The reason for the improvement in the speedup ratio is that the attributes filtered by rough sets are more accurate when the dimensions are large. The reason for the decline in the speedup ratio is that when the data dimension is low and the length is low, the processing capacity of the cluster cannot be effectively used, and more time is wasted by allocating data to multiple nodes. Therefore, in the case of big data and high dimensions, PDP Growth can provide full play to the efficiency of the algorithm, and more efficient dimensionality reduction and more accurate mining results also greatly improve the overall mining performance of the algorithm. Therefore, the experiment shows that PDP Growth algorithm is more suitable for mining frequent itemsets of high-dimensional data in the big data environment.

In order to verify the size of the k-value of the algorithm and its impact on different datasets, first set the k-value to 100, and take 0.2, 0.4, 0.6, 0.8, and 1.0 as the corresponding privacy budget. Conduct experiments, compare the PFP Growth algorithm with the classical PrivBasis [33] algorithm, and obtain the changes in the Fscore value and ARE value on different datasets; Fscore is an evaluation metric that combines precision and recall. It measures the ability of a classification model to simultaneously consider both accuracy and recall in its predictions. A higher Fscore indicates a better classification performance of the model. ARE (absolute relative error) is a metric that measures the absolute relative difference between predicted values and true values. The ARE value calculates the ratio of the absolute difference between predicted and true values to the true value itself, and it is used to assess the regression performance of the model. A smaller ARE value indicates better regression capability of the model, as shown in Figures 4–6.

![Graph](image.png)
Figure 4. Fscore and ARE in retail dataset: (a) Fscore and (b) ARE.

Figure 5. Fscore and ARE in Ta-Feng dataset: (a) Fscore and (b) ARE.
Figure 6. Fscore and ARE in Pumsb-Star dataset when $\varepsilon$ changes: (a) Fscore and (b) ARE.

It can be seen from experimental Figures 4a, 5a and 6a that when the fixed k value is unchanged, the Fscore value is also increased with the increase in $\varepsilon$. According to experimental Figures 4b, 5b and 6b, the ARE value is decreased with the increase in $\varepsilon$, which is consistent with previous expectations. Because with the continuous increase in $\varepsilon$, the smaller the amount of noise to be added, the higher the data availability, and the lower the average relative error. In addition, the experimental graph shows that under the same privacy budget, the algorithm PDP Growth proposed in this paper is better than the PFP Growth algorithm and PrivBasis algorithm in the three datasets, with a smaller average error and higher data availability.

In order to further verify the impact of noisy support on the algorithm, experiments will continue on the three datasets. For the selection of the privacy protection intensity k value, set different k values, fix the size of privacy budget, and measure the availability of the published data. Because the larger the $\varepsilon$ value is, the less noise is added, and the lower the privacy protection level is, this experiment sets $\varepsilon = 0.4$ to change the k value from 50 to 200. The experimental results are shown in Figures 7–9.
Figure 7. Fscore and ARE in retail dataset when k changes. (a) Fscore; (b) ARE.
Figure 8. Fscore and ARE in Ta-Feng dataset when k changes. (a) Fscore; (b) ARE.

Figure 9. Fscore and ARE in Pumsb-Star dataset when k changes: (a) Fscore and (b) ARE.
According to experimental Figures 7a, 8a and 9a, when $\varepsilon = 0.4$ is fixed, with the increase in $k$, the Fscore values of the three algorithms under each dataset show a decreasing trend. According to Figures 7b, 8b and 9b, the ARE value increases with the increase in $k$. This shows that the increase in $k$ value increases the loss error of information and improves the data privacy to a certain extent. Under the same $k$ value, the accuracy and loss error of PDP Growth algorithm on the three datasets are better than the PFP Growth algorithm and PrivBasis algorithm, which shows that the PDP Growth algorithm increases the availability of published data compared with the other two algorithms on the premise of satisfying privacy protection.

7. Conclusions

This paper studies the privacy protection in high-dimensional data association analysis, and proposes a new association rule-based high-dimensional data publishing method of PDP Growth under the Spark framework. This paper analyzes the existing problems in differential privacy algorithms in the field and proposes a parallel frequent itemset mining strategy based on rough set theory optimization to improve the processing efficiency of high-dimensional data. In this step, rough set theory is used to preliminarily reduce the dimensionality of the dataset. Then, an optimization method based on attribute information entropy is proposed to further eliminate redundancy in the data. Subsequently, a frequent itemset mining algorithm is applied to obtain the most concise frequent itemsets. In terms of privacy protection, the frequent itemsets are processed using the exponential mechanism. To avoid privacy leakage caused by directly releasing the protected dataset, Laplace noise is added to the obtained support values, achieving differential privacy protection.

This paper analyzes the flow of the algorithm, and theoretically proves that PDP Growth algorithm satisfies $\varepsilon$-differential privacy. Finally, through experimental verification, compared with similar algorithms, the PDP Growth algorithm can improve the operation efficiency of the algorithm and improve the availability of data while ensuring that the risk of data leakage is small. In the future, we will integrate other theories to better handle high-dimensional data by reducing data dimensionality while preserving the information contained in the data. This addresses some of the current shortcomings of our algorithm, as it sacrifices data information in order to protect the data. Additionally, we will continue researching how to better utilize the computational resources of nodes in a distributed framework to improve the efficiency of the algorithm.

**Author Contributions:** conceptualization, W.S.; method, W.S.; software, X.Z. (Xiaolei Zhang); verify, X.Z. (Xiaolei Zhang) and H.C.; formal analysis, H.C.; survey, X.Z. (Xing Zhang); resources, X.Z. (Xing Zhang); data management, X.Z. (Xing Zhang); writing—manuscript preparation, W.S.; writing—commenting and editing, X.Z. (Xiaolei Zhang); visualization, H.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** This paper was supported in part by the Applied Basic Research Project of Liaoning Province under Grant 2022JH2/101300280, Scientific Research Fund Project of Education Department of Liaoning Province under Grant LJKZ0625.

**Data Availability Statement:** The dataset used in this paper can be found on the website provided: https://archive.ics.uci.edu/ml/datasets.php (access date: 17 December 2021).

**Acknowledgments:** All of the authors of this article have been reviewed and confirmed.

**Conflicts of Interest:** The authors declare no conflict of interest.
References

17. Zhao, X.C. Research on Association Rule Mining Based on High-Dimensional Data and Incremental Data in Big Data Environment; Jiangxi University of Technology: Ganzhou, China, 2021.
28. Xu, S. Key Technologies Research on Frequent Pattern Mining with Differential Privacy Protection; Beijing University of Posts and Telecommunications: Beijing, China, 2016.

33. Li, N.; Qardaji, W.; Su, D.; Cao, J. *PrivBasis: Frequent Itemset Mining with Differential Privacy; Very Large Data Bases; VLDB Endowment: Greater Los Angeles, CA, USA, 2012.*

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.