EFFICIENT VERTICAL MINING OF HIGH AVERAGE-UTILITY ITEMSETS BASED ON NOVEL UPPER-BOUNDS

A

Major project report submitted in partial fulfilment of the requirement

for the award of the Degree of

BACHELOR OF TECHNOLOGY
In
COMPUTER SCIENCE AND ENGINEERING

By

A.Radhika patali	**(17841A0552)**
Manish Vemula	**(17841A0534)**
Vinay Reddy	**(17841A0548)**
Vinay Ragipani Manideep	**(15841A05F9)**

Under the esteemed guidance of
Mrs. G. ANITHA
Assistant Professor
Computer Science Engineering

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

AURORA'S TECHNOLOGICAL AND RESEARCH INSTITUTE

(Affiliated to JNTU, Hyderabad and approved by AICTE,New Delhi)
Parvathapur, Uppal, Hyderabad-500 039

(2020-21)

DECLARATION

We hereby declare that the work described in this major project, entitled "**EFFICIENT VERTICAL MINING OF HIGH AVERAGE-UTILITY ITEMSETS BASED ON NOVEL UPPER-BOUNDS**" which is been submitted by us **A.RadhikaPatali (17841A0552),ManishVemula(17841A0534), VinayReddy(17841A0548), Ragipani Manideep(15841A05F9)** in partial fulfillment for the award of bachelor of technology in Computer Science and Engineering to the Jawaharlal Nehru Technological University is a record of Bonafide work carried out by us under the guidance of **Ms. G Anitha, Asst.professor**.The results embodied in this major report have not been submitted to any other university or institute for the award of any Degree.

Place: Hyderabad

Date:

A.RadhikaPatali(17841A0552)

ManishVemula(17841A0534)

Vinay Reddy(17841A0548)

Ragipani Manideep(15841A05F9)

CERTIFICATE

This is to certify that the major Project Report entitled "**EFFICIENT VERTICAL MINING OF HIGH AVERAGE-UTILITY ITEMSETS BASED ON NOVEL UPPER-BOUNDS"** being submitted by **(A.RADHIKA PATALI, MANISH VEMULA, VINAY REDDY, RAGIPANI MANIDEEP),** in partial fulfillment for the award of Degree of Bachelor of Technology in **COMPUTER SCIENCE AND ENGINEERING** to the **JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, Hyderabad** is a record of Bonafide work carried out by them under my guidance and supervision. The results embodied in this major project report have not been submitted to any other university or institute for the award of any degree or diploma.

InternalGuide	**Head of the Department**
Mrs. G.Anitha	Mrs. A.Durga Pavani
Assistant professor, IT	CSE
Major Project Coordinator	**Director**
Mrs. M. Sowmya	Mr. Srikanth Jatla
Associate Professor	
CSE	

EXTERNAL EXAMINER

BRAIN O
VISION

10-06-2021

To
The Head Of Department,
Computer Science and Engineering
Aurora's Technological and Research Institute.

COMPLETION CERTIFICATE

This is to certify that the following students, who are pursuing **B.TECH**(Computer Science and Engineering) at **Aurora's Technological and Research Institute**, has successfully completed their major project in
Efficient Vertical Mining of High Average-Utility Itemsets based on Novel Upper-Bounds
Under the guidance of Mr. Bala Maheswar for 2 Months.

A.Radhika Patali	17841A0552
Manish Vemula	17841A0534
Nalla Vinay Reddy	17841A0548
Ragipani Manideep	15841A05F9

Yours faithfully
Name:Bala Maheshwar
Technical Manager
BRAIN O VISION SOLUTIONS (INDIA) PVT LTD

ACKNOWLEDGEMENT

This project would not have been successful without the help of several people. We would like to thank the personalities who were part of our project in numerous ways, those who gave us outstanding support from the birth of this work.

We would also like to express our gratitude to **Mr. Srikanth Jatla**, Director, **Aurora's Technological And Research Institute** for providing us congenial atmosphere and encouragement.

We express our sincere thanks to Head of the Department **Mrs. A.Durga Pavani** for giving us the support and her kind attention and valuable guidance to us throughout this project.

We express our sincere thanks to Project Coordinator **Mrs.M.Sowmya**,Ass. Professor, for helping us to complete our project by giving valuable suggestions.

We hereby wish to express our deep sense of gratitude to **Mrs. G.Anitha** Assistant Professor,Department of CSE, for the esteemed guidance, moral support and invaluable Advice provided by her for the success of the project.

We are also thankful to all the staff members of CSE department who have co-operated in making our project a success. We would like to thank all our parents and friends who extended their help, encouragement and moral support either directly or indirectly in our project work.

Sincerely,

A.Radhika Patali(17841A0552)

Manish Vemula(17841A0534)

Vinay Reddy (17841A0548)

Ragipani manideep(15841A05F9)

ABSTRACT

- Mining High Average-Utility Itemsets (HAUIs) in a quantitative database is an extension of the traditional problem of frequent itemset mining, having several practical applications. Discovering HAUIs is more challenging than mining frequent itemsets using the traditional support model since the average-utilities of itemsets do not satisfy the downward-closure property. To design algorithms for mining HAUIs that reduce the search space of itemsets, prior studies have proposed various upper-bounds on the average-utilities of itemsets. However, these algorithms can generate a huge amount of unpromising HAUI candidates, which result in high memory consumption and long runtimes.
- To address this problem, this paper proposes four tight average-utility upper-bounds, based on a vertical database representation, and three efficient pruning strategies. Furthermore, a novel generic framework for comparing average-utility upper-bounds is presented. Based on these theoretical results, an efficient algorithm named dHAUIM is introduced for mining the complete set of HAUIs. dHAUIM represents the search space and quickly compute upper-bounds using a novel IDUL structure. Extensive experiments show that dHAUIM outperforms three state-of-the-art algorithms for mining HAUIs in terms of runtime on both real-life and synthetic databases. Moreover, results show that the proposed pruning strategies dramatically reduce the number of candidate HAUIs.

INDEX

LIST OF FIGURES

1. INTRODUCTION

1.1 PROJECT STATEMENT

In this section, we present new theoretical results for the design of an efficient algorithm for mining high average-utility itemsets. In particular, this section explains how the utility measure *u* and novel upper-bounds on the average-utility measure can be defined using a vertical form (columns of the matrix), contrarily to previous studies that have used the horizontal form. These results are the basis of the proposed algorithm. Let x be the column utility of an itemset in the h *column* of the matrix. Furthermore, let X>Y be the set of all column utility values associated to X starting from the X *column,* whereX=Y is the minimum index (or first index) of items in. As a convention,. Based on these definitions, the utility of is defined in vertical form as follows. During the expansion of the search space of HAU candidate itemsets, an itemset is extended with an itemset such thatX<Y The result is an extension X (also called a forward extension) of X (with itemset), denoted as X= Y ⊕ X *i.e.* Y= X ∪ Yunder the condition that X<Y In this situation, the itemset *P* is said to be a *prefix*of X. For example, if X=X<Y U X, an item-extension of X (with item) isX ⊕ {Y}, concisely denoted as . Note that the measure is *neither monotonic n or anti-monotonic*. *Indeed*, for example, X,Y,Z, but (X) = 90 < (Y) = 290/3 < Z(X) = 304. To design pruning strategies for reducing the search space in HAUIM, an UB named has been proposed [14]. The measure is an *upper bound*on *au,* which satisfies the *DCP* (or *anti-monotonicity* property), i.e. it follows that(X) ⩽ Y(X) andX(Z)⩽Y(X). Therefore, if is loW, i.e.(X) < Y, then all its super itemsets are also low-X and LAU. To prune more unpromising candidates from the search space, several upper-bounds on the X measure have been proposed [16],[18],[19]. How-ever, the set of candidate itemsets considered when using these upper-bounds is often very large. To overcome this problem, this article proposes four new UBs that are tighter than, named1, which can be quickly calculated using the proposed *vertical form* (X).

Association Mining searches for frequent items in the data-set. In frequent mining usually the interesting associations and correlations between item sets in transactional and relational databases are found. In short, Frequent Mining shows which items appear together in a transaction or relation.

Need of Association Mining:

Frequent mining is generation of association rules from a Transactional Dataset. If there are 2 items X and Y purchased frequently then its good to put them together in stores or provide some discount offer on one item on purchase of other item. This can really increase the sales. For example it is likely to find that if a customer buys Milk and bread he/she also buys Butter.

So the association rule is [‘milk]^[‘bread’]=>[‘butter’]. So seller can suggest the customer to buy butter if he/she buys Milk and Bread.

Important Definations :

Support : It is one of the measure of interestingness. This tells about usefulness and certainty of rules. 5% Support means total 5% of transactions in database follow the rule.

Support(A -> B) = Support_count(A ∪ B)

Confidence: A confidence of 60% means that 60% of the customers who purchased a milk and bread also bought butter.

Confidence(A -> B) = Support_count(A ∪ B) / Support_count(A)

If a rule satisfies both minimum support and minimum confidence, it is a strong rule.

Support_count(X) : Number of transactions in which X appears. If X is A union B then it is the number of transactions in which A and B both are present.

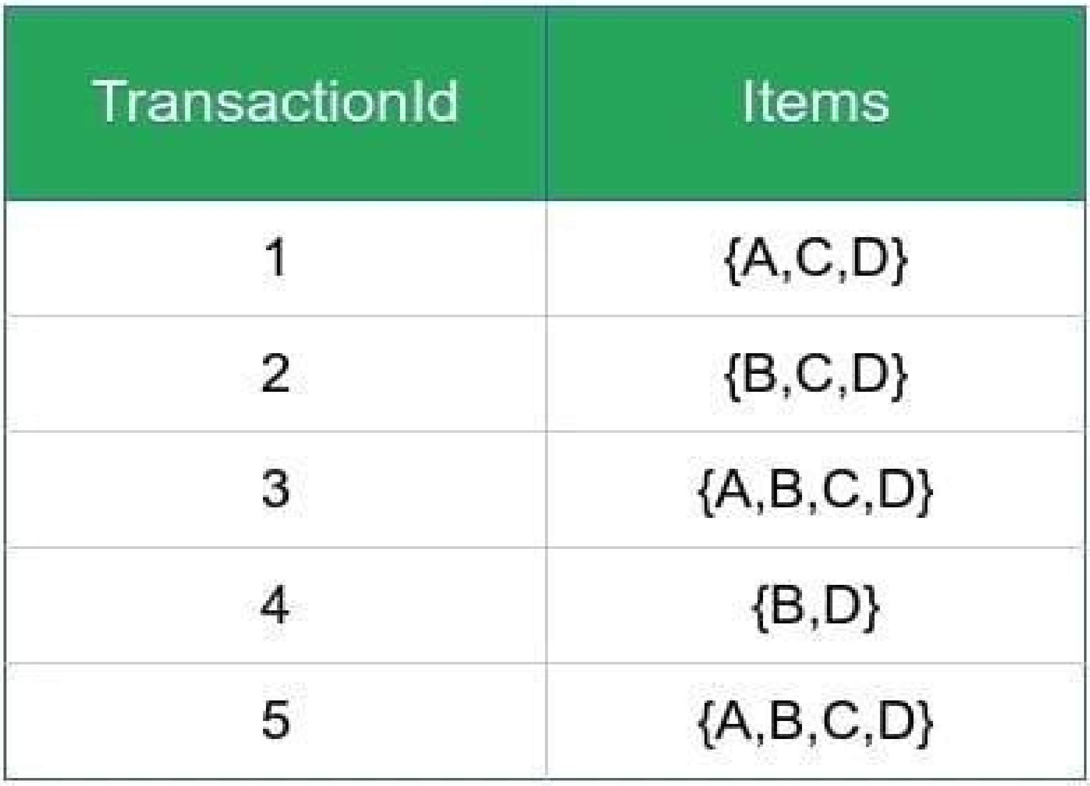
Maximal Itemset: An itemset is maximal frequent if none of its supersets are frequent.

Closed Itemset:An itemset is closed if none of its immediate supersets have same support count same as Itemset.

K- Itemset:Itemset which contains K items is a K-itemset. So it can be said that an itemset is frequent if the corresponding support count is greater than minimum support count.

Example On finding Frequent Itemsets –

Consider the given dataset with given transactions.



TransactionId	Items
1	{A,C,D}
2	{B,C,D}
3	{A,B,C,D}
4	{B,D}
5	{A,B,C,D}

FIG 1.1 ITEMSETS

1.2 EXISTING SYSTEM

Facing these limitations of tradition and weighted frequent itemset mining, the problem of high utility item set mining (HUIM) has emerged as a important research problem. The goal is to extract item sets that have a high utility (yield a high profit), where the utility of an item set is defined as the sum of the utilities of its items in transactions where the item set appears. An item-set is of high utility (HU) if its utility is larger than or equal to a minimum utility threshold mu, defined by the user. The key challenge for designing efficient HUIM algorithms is that the DCP does not hold for the utility measure. To address this issue, an upper bound (UB) on the utility of item sets that respect the DCP was proposed, named transaction-weighted utilization (TWU). The DCP property of the TWU upper bound states that suspects of an item set having a TWU value that is less than the mu threshold also have low TWU values, and are low utility item sets. The TWUpper-bound has been used in numerous HUIM algorithms to prune unpromising item sets and thus reduce the search space.

➢ DISADVANTAGES

- Fog architecture is decentralized for processing.
- Requires high amount of energy.
- Privacy and security issues.
- Complexity is more, hence making it difficult to understand.
- It cannot tolerate high latency.
- In distributed computing, the enormous measures of data must be transmitted to data focuses on cloud, yielding critical performance overhead.

1.3 PROPOSED SYSTEM

Although high utility item set mining is useful, the standard utility measure does not take the length of item sets into account. But in real life, an item set containing many items is often uninteresting as it is difficult to co-promote a large set of products. Besides, item sets containing several items often have a higher utility than those containing fewer items, which may be seen as unfair. For instance, many high utility items such as laptop produce high utility item-sets when they are combined with items having a lower utility such as USB drives or mouse pads.

➢ **DISADVANTAGES**

- Mainly focuses on manufacturing industries.
- Reduces the cost along with higher productivity.
- Data can be uploaded by admin without any issues.
- Use of generalised network.

2. SYSTEM STUDY

2.1 SOFTWARE REQUIREMENTS

•	**Operating system**	: Windows 7 Ultimate.
•	**Coding Language**	: Python.
•	**Front-End**	: Python.
•	**Designing**	: Html,css,javascript.
•	**Data Base**	: MySQL.

2.1.2 Functional Requirements

- Graphical User interface with the User.

2.1.3 Software Requirements

For developing the application the following are the Software Requirements:

1.Python

2.Django

3.Mysql

4.Wampserver

2.1.4 Operating Systems supported

1.Windows 7

2.Windows XP

3.Windows 8

2.1.5 Technologies and Languages used to Develop

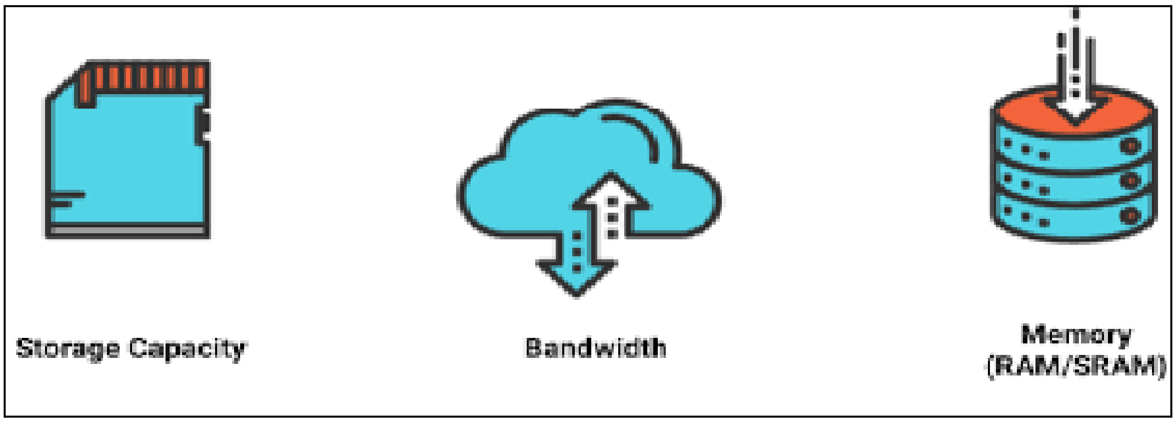
1.Python

2.1.6 Debugger and Emulator

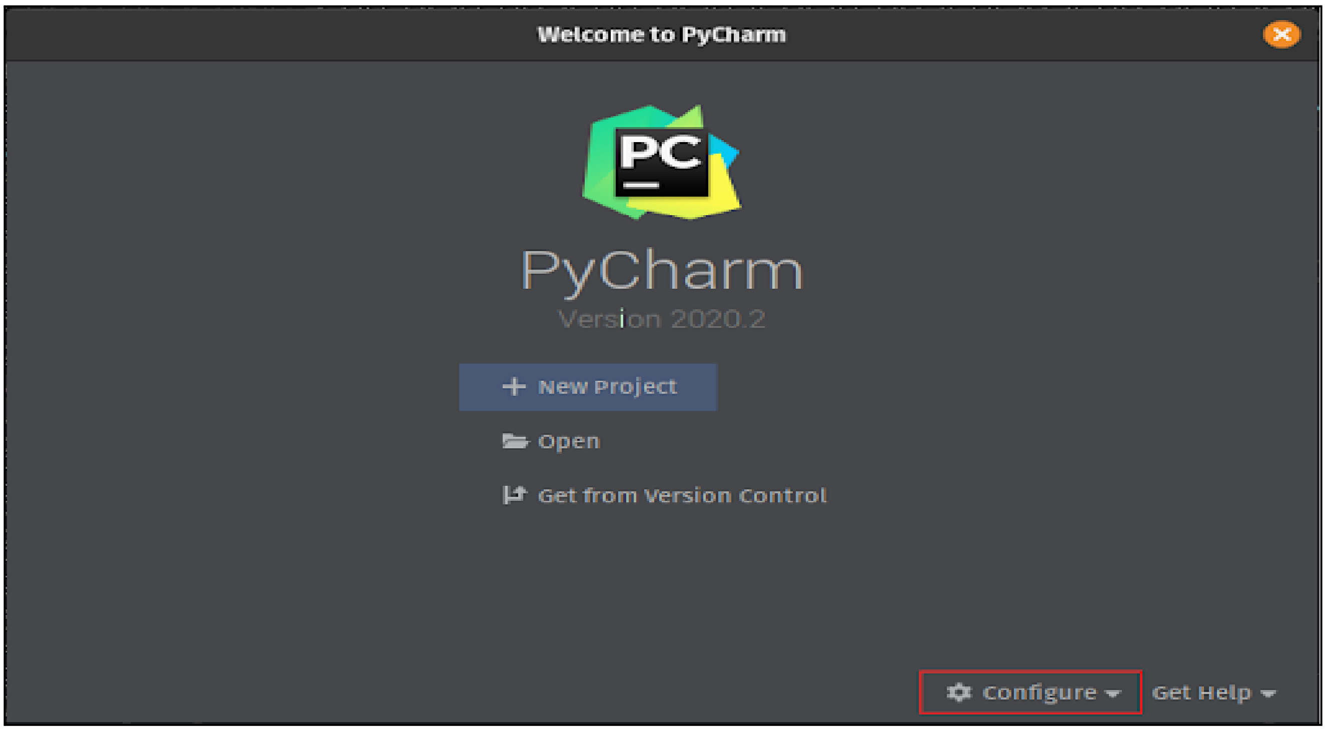
Any Browser (Particularly Chrome)

2.2 HARDWARE REQUIREMENTS

- Processor: Pentium IV or higher
- RAM: 256 MB



- Space on Hard Disk: minimum 512MB



2.2 PYCHARM

2.3 SYSTEM ANALYSIS

In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviours. They are contrasted with functional requirements that define specific behaviour or functions. Non-functional requirements add tremendous value to business analysis. It is commonly misunderstood by a lot of people. It is important for business stakeholders, and Clients to clearly explain the requirements and their expectations in measurable terms. If the non-functional requirements are not measurable then they should be revised or rewritten to gain better clarity. For example, User stories help in mitigating the gap between developers and the user community in Agile Methodology.

- **Usability** Prioritize the important functions of the system based on usage patterns. Frequently used functions should be tested for usability, as should complex and critical functions. Be sure to create a requirement for this.
- **Reliability** Reliability defines the trust in the system that is developed after using it for a period of time. It defines the likeability of the software to work without failure for a given time period. The number of bugs in the code, hardware failures, and problems can reduce the reliability of the software. Your goal should be a long MTBF (mean time between failures). It is defined as the average period of time the system runs before failing. Create a

requirement that data created in the system will be retained for a number of years without the data being changed by the system. It’s a good idea to also include requirements that make it easier to monitor system performance.

- **Performance** What should system response times be, as measured from any point, under what circumstances? Are there specific peak times when the load on the system will be unusually high? Think of stress periods, for example, at the end of the month or in conjunction with payroll disbursement.
- **Supportability** The system needs to be cost-effective to maintain. Maintainability requirements may cover diverse levels of documentation, such as system documentation, as well as test documentation, e.g., which test cases and test plans will accompany the system.

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are,

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

2.4.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

3. LITERATURE SURVEY

3.1 PAPERS REVIEWED

3.1.1 Secure Mining of Association Rules in Vertically Distributed Databases

3.1.2

by Tamir Tassa

A protocol for secure mining of association rules in vertically distributed databases.

The current leading protocol is that of Kantarcioglu and Clifton. Our protocol, like theirs, is based on the Fast Distributed Mining (FDM) algorithm of Cheung et al., which is an unsecured distributed version of the Apriori algorithm. The main ingredients in our protocol are two novel secure multi party algorithms one that computes the union of private subsets that each of the interacting players hold, and another that tests the inclusion of an element held by one player in a subset held by another. Our protocol offers enhanced privacy with respect to the protocol in. In addition, it is simpler and is significantly more efficient in terms of communication rounds, communication cost and computational cost.

3.1.3 Security in Outsourcing of Association Rule Mining

3.1.4

by W. K. Wong & David W. Cheung

Outsourcing association rule mining to an outside service provider brings several important benefits to the data owner. These include (i) relief from the high mining cost, (ii) minimization of demands in resources, and (iii) effective centralized mining for multiple distributed owners. On the other hand, security is an issue; the service provider should be prevented from accessing the actual data since (i) the data may be associated with private information, (ii) the frequency analysis is meant to be used solely by the owner. This paper proposes substitution cipher techniques in the encryption of transactional data for outsourcing association rule mining. After identifying the non-trivial threats to a straightforward one-to-one item mapping substitution cipher, we propose a more secure encryption scheme based on a one-to-n item mapping that transforms transactions non-deterministically, yet guarantees correct decryption. We develop an effective and efficient encryption algorithm based on this method. Our algorithm performs a single pass over the database and thus is suitable for applications in which data owners send streams of transactions to the service provider. A comprehensive cryptanalysis study is carried out. The results show that our technique is highly secure with a low data transformation cost.

3.1.4 An Overview of Secure Mining of Association Rules in Vertically Distributed Databases

3.1.5

by Ms. Sonal PatilMr. Harshad S. Patil

In this paper, propose a protocol for secure mining of association rules in vertically distributed databases. Now a day the current leading protocol is Kantarcioglu and Clifton. This protocol is based on the Fast Distributed Mining (FDM) algorithm which is an unsecured distributed version of the Apriori algorithm. The main ingredients in this protocol are two novel secure multi_x005F_x0002_party algorithms 1. That computes the union of private subsets that each of the interacting players hold, and 2. Tests the inclusion of an element held by one player in a subset held by another. In this protocol offers enhanced privacy with respect to the other one. Differences in this protocol, it is simpler and is significantly more efficient in terms of communication rounds, communication cost and computational cost.

3.1.5 problem of secure multi-user computation

3.1.6

by W. K. Wong & David W. Cheung

The goal defines a problem of secure multi-user computation. In such problems, there are N users that hold private inputs of data, x1, . . . , xN, and they wish to securely compute y = f(x1, . . . , xN) for some public function f [1]. If there existed a trusted third party, the users could submit to him their inputs and he would perform the function evaluation and send to them the resulting output. In the absence of such a trusted third party, it is needed to devise techniques that the users can run on their own in order to arrive at the required output y. The next goal is to secure the inputs of each user. If the both are combined together (data mining and Secure) the third party involvement is avoided.

3.1.6 Mining Algorithm Strategies

A drawback of FIM and ARM is that they only consider the occurrence frequencies of itemsets. The other factors such as weights, interestingness, or unit profits of items are notconsidered to select more useful and meaningful patterns.High-utility itemset mining (HUIM) [32, 33] was developed to consider both the quantities and unit profits of item/sets to derive the set of high-utility itemsets (HUIs). To maintain the downward closure (DC) property and reduce the search space for mining HUIs, the transaction-weighted utilization(TWU) model [16] was designed. It provides the transaction-weighted downward closure (TWDC) property of the high transaction-weighted utilization itemsets (HTWUIs) to speedup the mining process and has become a standard mechanism for HUIM. Li etal. [18] then designed the isolated items discarding strategy (IIDS) to further reduce the number of can-didates generated when mining HUIs using the TWU model.The incremental high-utility pattern (IHUP) algorithm [3] was developed to incrementally and interactively mine HUIs basedon a tree structure similar to the FP-tree approach. The HUI-Miner algorithm [21] was proposed, which relies on a utility-list structure for mining HUIs without generating candidates and using a depth-first search. HUI-Miner constructs a vertical database representation to avoid performing multiple databases cans while still deriving the set of high-utility k-itemsets,thanks to a simple join operation. To further improve the performance of HUIM, the FHM algorithm [7] was designed.It stores the relationships between all pairs of items (2-itemsets) to reduce the search space and prune a large amount of unpromising candidates early. Liu et al [26] designed thed2HUP algorithm with several pruning strategies to efficiently discover HUIs. Several algorithms [24, 28, 29] were also proposed for different applications and to address other issues related to HUIM. HUIM is a very active research area.

4 . SOFTWARE REQUIREMENT ANALYSIS

4.1 PROBLEM DEFINITION

Before the beginning of every farm season, most farmers prefer to plan potential yields. On the other hand, some farmers chose to skip planning. Whether a farmer plans the potential yield or not, certain expectations are still present. While hoping for the best, farmers are often presented with various challenges and obstacles that require them to constantly question their productivity and resulting final success.

The greatest importance is usually given to crop protection from diseases, insect pests, and weeds, as well as to protection from unfavorable weather events such as frost or hail ,along with other crop maintenance practices. The aforementioned challenges are well-known and often discussed. However, farmers also face another interesting challenge, often forgotten about or not realized; wild animal crop protection.

Wild animals are a special challenge for farmers throughout the world. Animals such as deer, wild boars, rabbits, moles, elephants, monkeys, and many others may cause serious damage to crops. They can damage the plants by feeding on plant parts or simply by running over the field and trampling over the crops. Therefore, wild animals may easily cause significant yield losses and provoke additional financial problems. Another aspect to consider is that wild animal crop protection requires a particularly cautious approach. In other words, while utilizing his crop production, every farmer should be aware and take into consideration the fact that animals are living beings and need to be protected from any potential suffering.

4.2. THE MODULES AND THEIR FUNCTIONALITY

4.2.1 PYTHON MODULES

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++or Java. It provides constructs that enable clear programming on both small and large scales. Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open-source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library

4.2.2 DJANGO

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source.

Django's primary goal is to ease the creation of complex, database-driven websites. Django emphasizes reusability and "pluggability" of components, rapid development, and the principle of don't repeat yourself. Python is used throughout, even for settings files and data models.

4.2.3 ITEMSET UPLOADS

To initiate the system, we need to upload the data (i.e., Products). The products can be uploaded with the attributes that are promoted to improve the utility of itemset. The uploads of itemset is done with the help of admin side. Admin is the only authorized person to upload the data to database and have rights to modify or delete the product or its details.

4.2.4 ANALYSIS OF ITEMSET

The rate of buying product by the users is noticed. The Itemset can be analyzed according to the purchasing chart. The lower numbers of purchased products are so called average-utility itemset. To up the utility of itemset we need to do analysis of what need to do, in order to improve the sale.

4.2.5 PRODUCT RE-ARRANGE

The average-utility itemset can be improved to High Average-utility Itemset. To improve this we going to handle some functional techniques. The Itemset can be promoted and add or remove some features to effectively sale the product. The re-arrangement of product can be done with the high utilization of itemset property. Based on this module Itemset again publish to users and analysis the Itemset again.

4.2.6 GRAPH ANALYSIS OF SYSTEM

The graphs are utilized to analysis the proposed system based on the selling details. The more important part is to analysis the data which are user bought products and number that lies on the average utility. The graphs are plot to improve the way of analysis and ease of making understand the pictured data.

5. SOFTWARE DESIGN

5.1 SYSTEM ARCHITECTURE

- ❖ The following diagram is a strategic diagram that allows the steps in a process, workflow or algorithm to be ordered in a sequence and displayed graphically.
- ❖ The steps and choices are modelled, and there are inputs/outputs and data stores available for modelling the things that are consumed, produced and stored by the process.

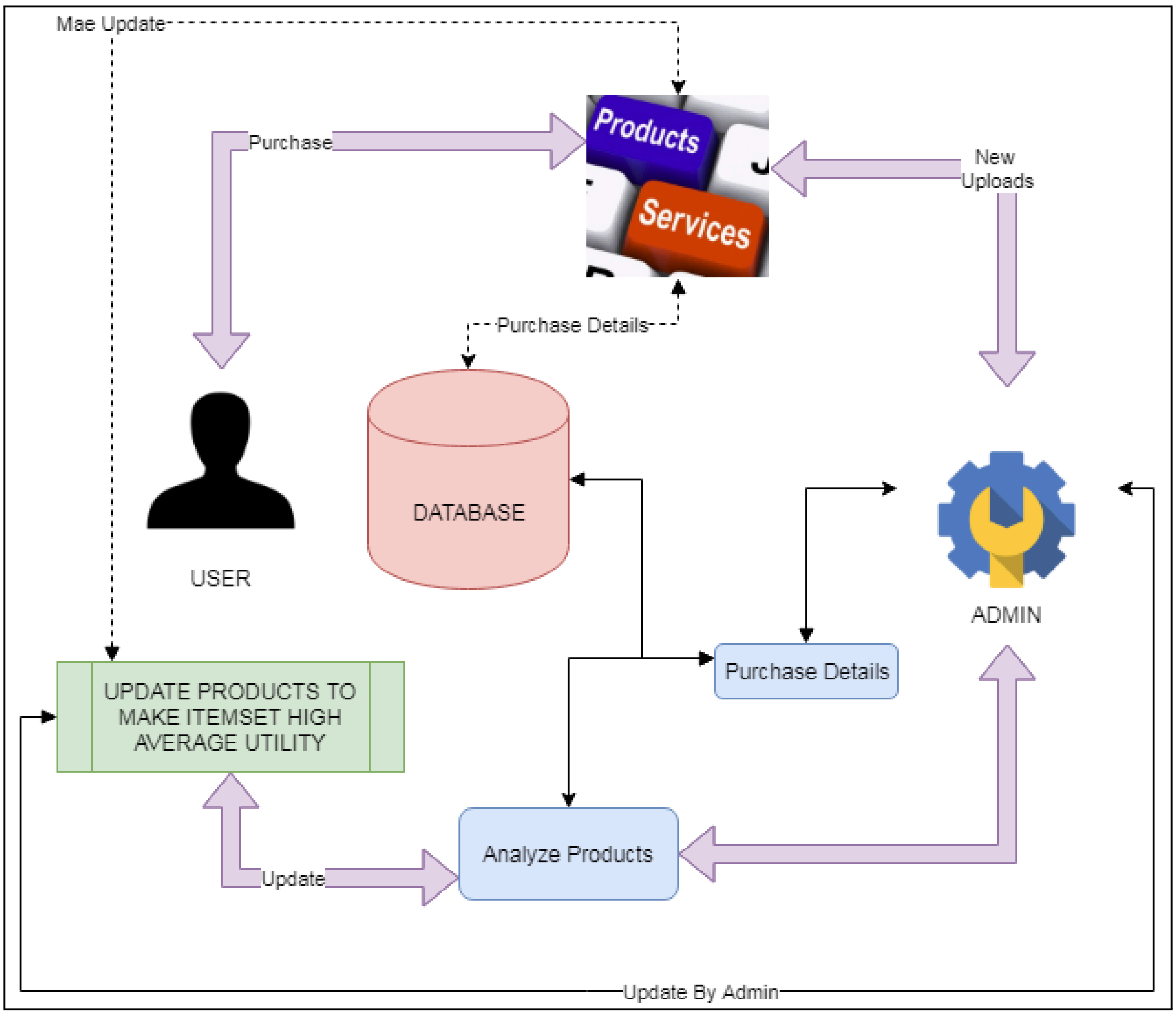


FIG 5.1 ARCHITECTURE

5.2 ALGORITHM:
THE DHAUIM ALGORITHM

Based on the novel and UBs and the recursive formulas of Proposition 3, this section presents an efficient algorithm, named dHAUIM (diffset-based High Average Utility Itemset Miner), for mining the set of all high average-utility itemsets. The proposal of this algorithm answers the third research question. The pseudocode of the algorithm is shown in Fig. 2. During the mining process, HAU candidate itemsets are stored in a prefix-tree using a novel structure named IDUL (Itemset-Diffset-Util-ityList). This structure contains the information of a node of the form. Recall that the notation [ ] denotes the set of item-extensions of a parent node. The dHAUIM algorithm (Fig. 2) takes as input a QDB and the threshold. It first computes the integrated matrix and the vectors using formulas. Then, the algorithm calculates the vectors using formula. The set of all high– 1 items is then obtained by applying the strong width pruning strategy SWP. Afterwards, the recursive procedure HAU-Search is called. In terms of implementation, to reduce memory us-age, the integrated QDB can be represented using well-known techniques for storing sparse matrices. Consider the running. Because, the procedure receives. The IDUL tree representing the search space is shown in Fig. 4. In this figure, for each node, for the sake of brevity, the field is omitted, while the values are represent (i). For example, although the improved UB is tighter than, both have different pruning effects (DP and WP). Thus, using the width pruning condition at line 11 of the HAU-Search procedure is necessary. Indeed, for 🕴 = = 250, if the condition is removed, two redundant itemsets and will be kept. As a result, two additional redundant candidates will need to be considered.

(ii). Similarly, if we replace the , UBs with the stronger in the depth pruning condition at line 3 of HAU-Search, the algorithm may consider much more un-promising candidate itemsets. In fact, for the IDUL tree in Fig. 4 and = 200, the algorithm would have to consider ten additional redundant candidates having average-utility values that are less than. On the contrary, if or are used at line 3, then and these unpromising candidates are not considered.

(iii). since the UBs do not satisfy the property, we cannot replace the strong UB with weaker or UBs in the width pruning condition at line 11 of HAU-Search. Indeed, assume that we re-place the condition. For the two nodes, we have and the modified condition (*) does not hold. Hence, will not be and thus the itemset which is the union of and will be missing in the result.

5.2 DATA FLOW DIAGRAM

a. User

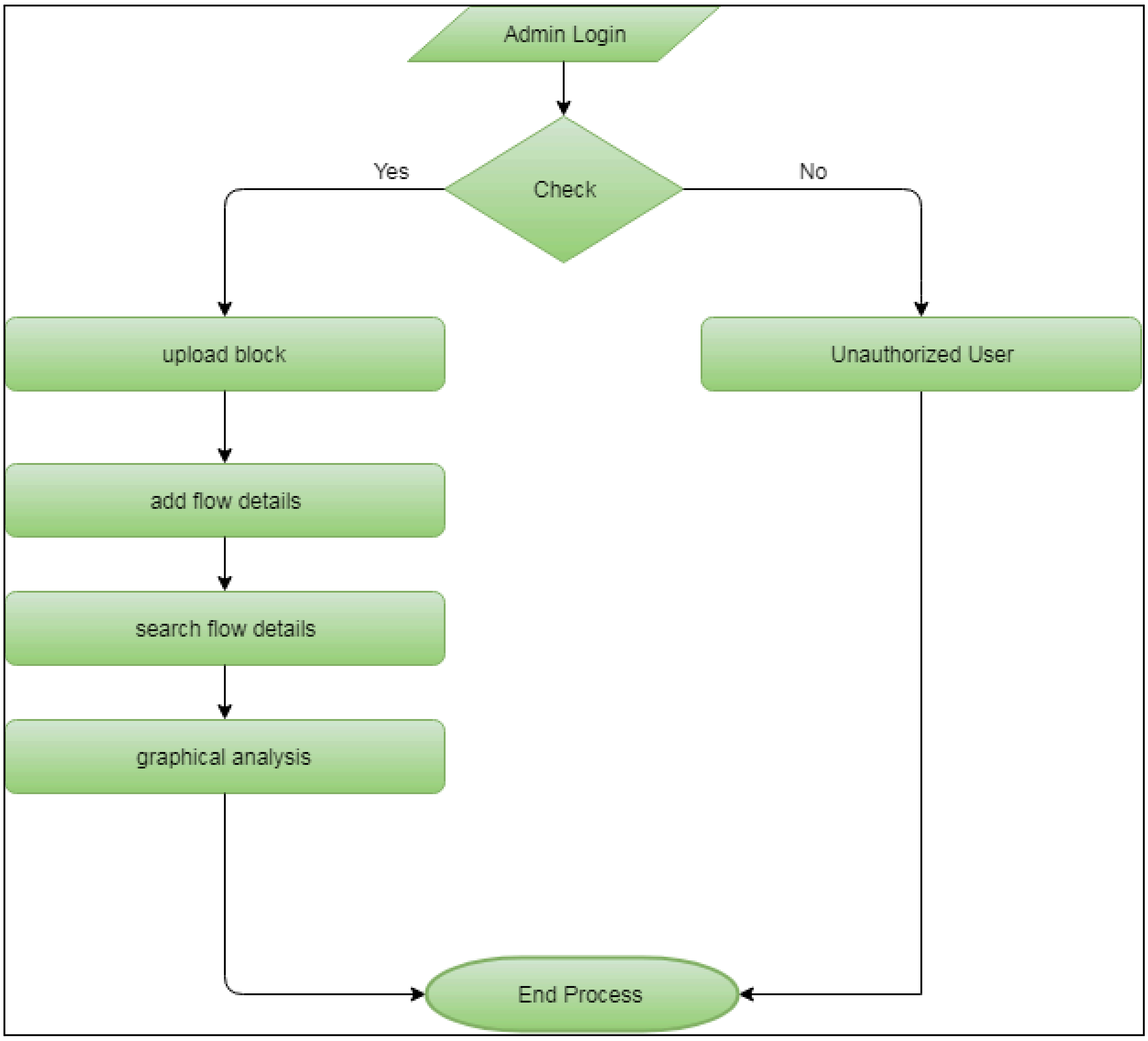


FIG 5.2 data flow diagram

6. UML DIAGRAMS

UNIFIED MODELING LANGUAGE DIAGRAMS

The unified modeling language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic semantic and pragmatic rules. A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagrams, which is as follows.

USER MODEL VIEW

This view represents the system from the users perspective.The analysis representation describes a usage scenario from the end-users perspective.

STRUCTURAL MODEL VIEW

In this model the data and functionality are arrived from inside the system.
This model view models the static structures.

BEHAVIORAL MODEL VIEW

It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

IMPLEMENTATION MODEL VIEW

In this the structural and behavioral as parts of the system are represented as they are to be built.

ENVIRONMENTAL MODEL VIEW

In these the structural and behavioral aspects of the environment in which the system is to be implemented are represented.
UML is specifically constructed through two different domains they are
UML Analysis modeling, which focuses on the user model and structural model views of the system.
UML design modeling, which focuses on the behavioral modeling, implementation modeling and environmental model views.
The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

USES OF UML DIAGRAMS

The Unified Modelling Language (UML) is a standard language for writing software blue prints. The UML is a language for

● Visualizing

● Specifying

● Constructing

● Documenting the artifacts of a software intensive system.

The UML is a language which provides vocabulary and the rules for combining words in that vocabulary for the purpose of communication. A modelling language is a language whose vocabulary and the rules focus on the conceptual and physical representation of a system. Modelling yields an understanding of a system.

THINGS IN UML

There are four kinds of things in the UML:

• Structural things

• Behavioural things

• Grouping things

• Annotational things

Structural Things

Structural things are the nouns of UML models. These are the mostly static parts of a model, representing elements that are either conceptual or physical. In all, there are seven kinds of structural things. A class is a description of a set of objects that share the same attributes, operations, relationships, and semantics. A class implements one or more interfaces. Graphically, a class is rendered as a rectangle, usually including its name, attributes, and operations. An interface is a collection of operations that specify a service of a class or component. An interface therefore describes the externally visible behaviour of that element. Graphically, an interface is rendered as a circle together with its name. An interface rarely stands alone. Collaboration defines an interaction and is a society of roles and other elements that work together to provide some cooperative behaviour that's bigger than the sum of all the elements. Therefore, Collaborations have structural, as well as behavioural, dimensions. A given class might participate in several collaborations. These collaborations therefore represent the implementation of patterns that make up a system. Graphically, collaboration is rendered as an ellipse with dashed lines, usually including only its name.

Behavioural Things

Behavioural things are the dynamic parts of UML models. These are the verbs of a model, representing behaviour over time and space. In all, there are two primary kinds of behavioural things.

An interaction is a behaviour that comprises a set of messages exchanged among a set of objects within a particular context to accomplish a specific purpose. The behaviour of a society of objects or of an individual operation may be specified with an interaction. An interaction involves a number of other elements, including messages, action sequences and links graphically; a message is rendered as a directed line, almost always including the name of its operation.

A state machine is a behaviour that specifies the sequences of states that an object or an interaction goes through during its lifetime in response to events, together with its responses to those events. The behaviour of an individual class or a collaboration of classes may be specified with a state machine. A state machine involves a number of other elements, including states, transitions, events, and activities. Graphically, a state is rendered as a rounded rectangle, usually including its name and its sub states, if any.

Grouping Things

Grouping things are the organizational parts of UML models. These are the boxes into which a model can be decomposed. In all, there is one primary kind of grouping thing, namely, packages.

A package is a general-purpose mechanism for organizing elements into groups. Structural things, behavioural things, and even other grouping things may be placed in a package.

Unlike components (which exist at run time), a package is purely conceptual (meaning that it exists only at development time). Graphically, a package is rendered as a tabbed folder, usually including only its name and, sometimes, its contents.

Annotational Things

Annotational things are the explanatory parts of UML models. These are the comments you may apply to describe, illuminate, and remark about any element in a model. There is one primary kind of Annotational thing, called a note. A note is simply a symbol for rendering constraints and comments attached to an element or a collection of elements. Graphically, a note is rendered as a rectangle with a dog-eared corner, together with a textual or graphical comment.

6.1 CLASS DIAGRAM

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of objectoriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

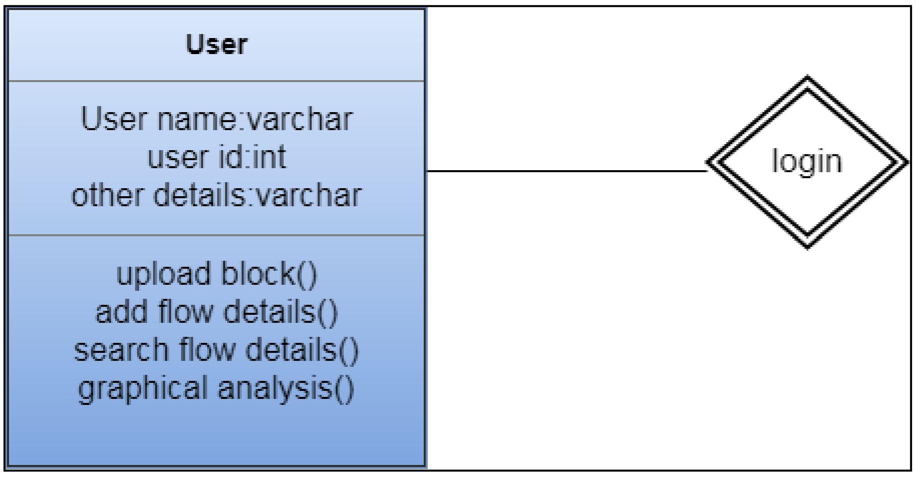


Fig 6.1 class diagram

6.2 SEQUENCE DIAGRAM

a. User

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario

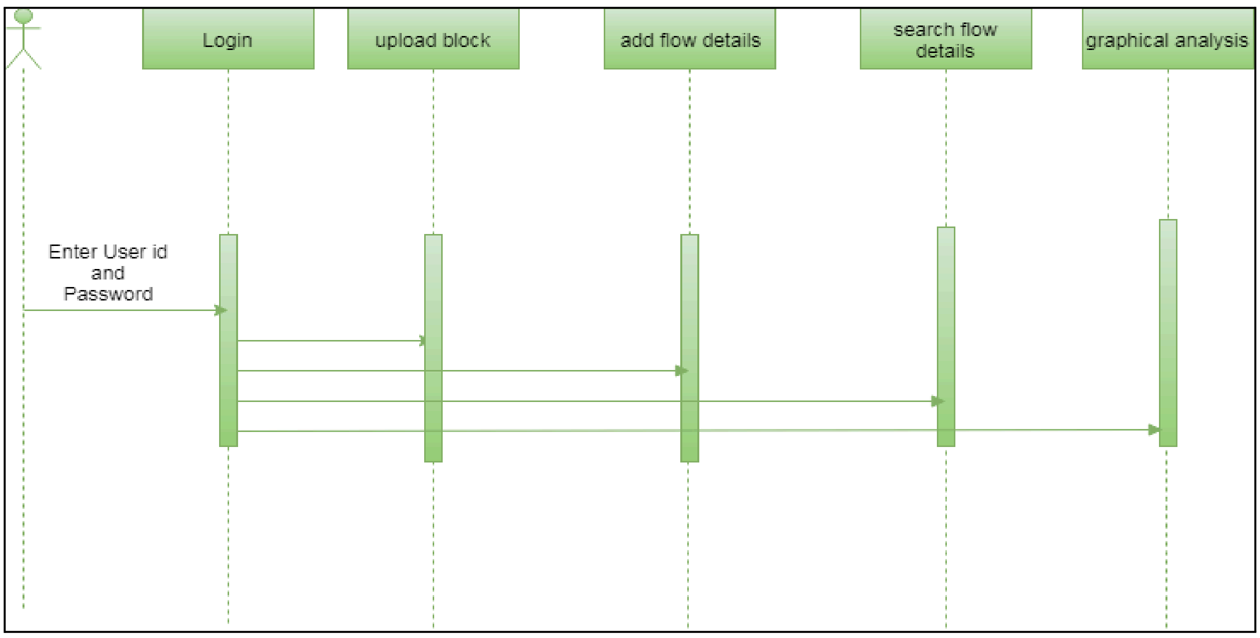


FIG6.2 sequence diagram

6.3 USE CASE DIAGRAM

a. User

The purpose of use case diagram is to capture the dynamic aspect of a system. Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified.

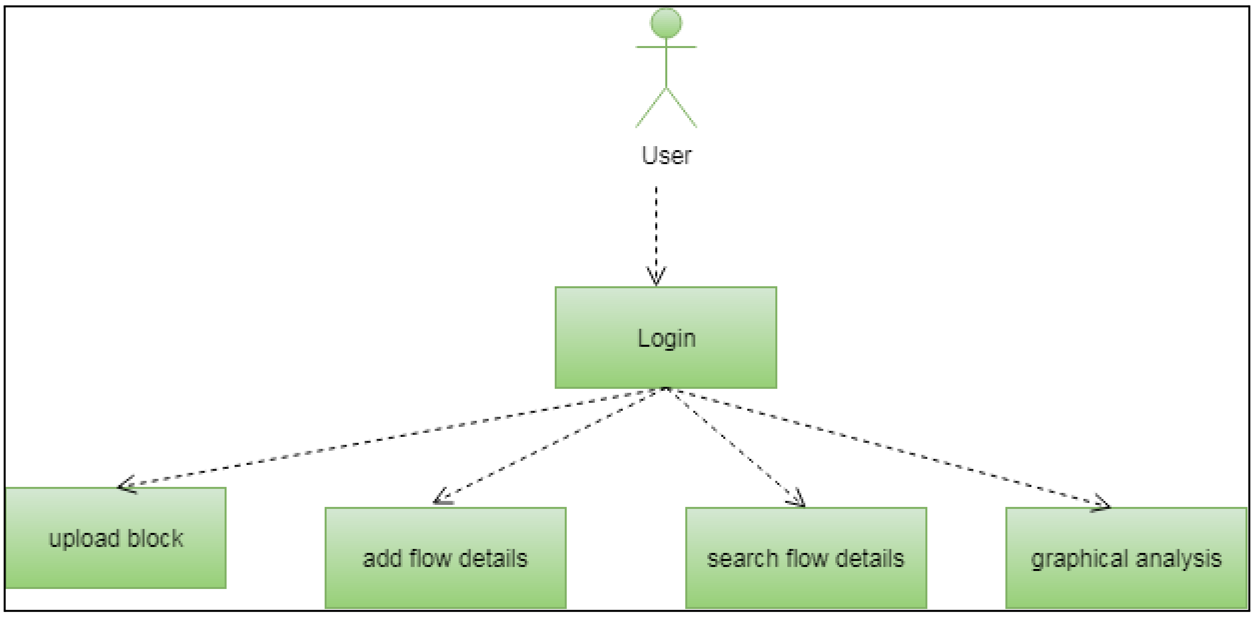


FIG6.3 use case diagram

6.4 ACTIVITY DIAGRAM

a. User

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. It captures the dynamic behavior of the system.

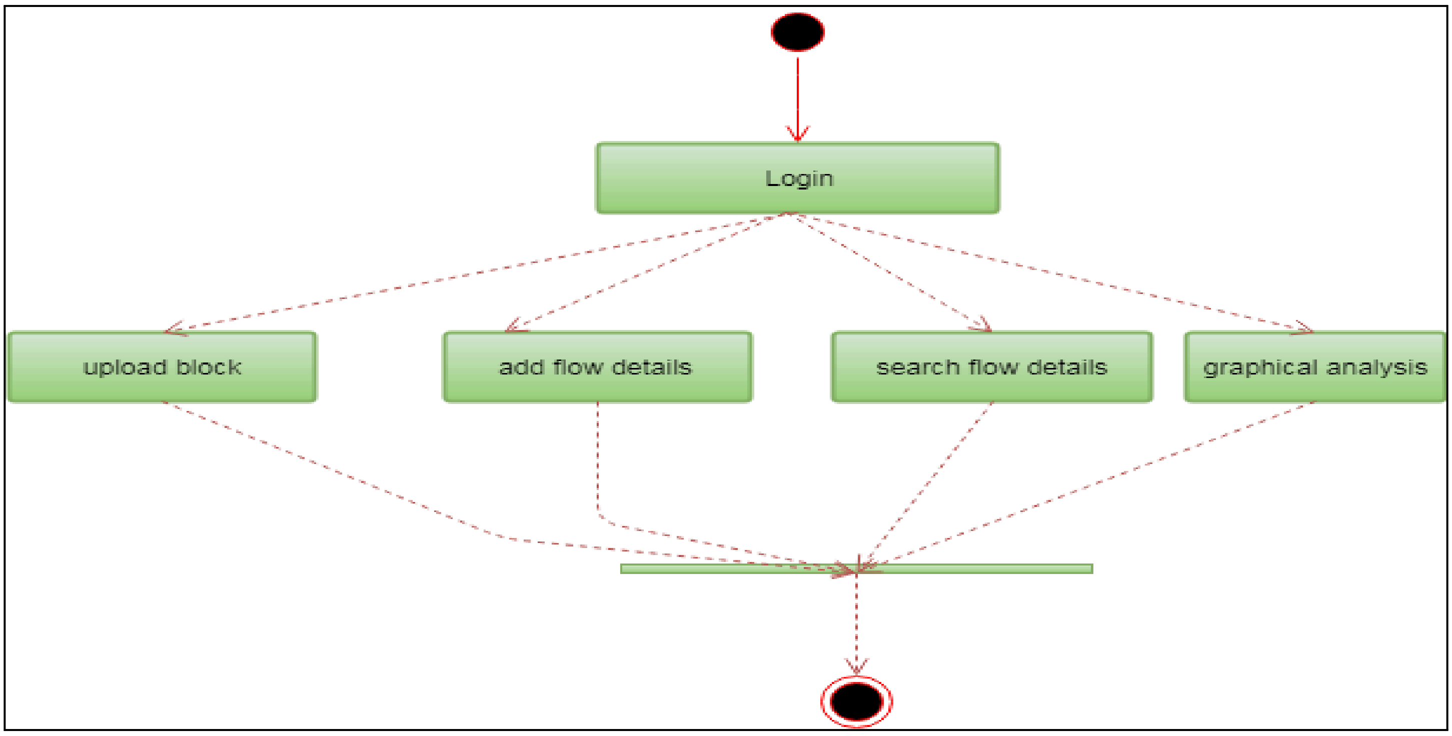


FIG6.4 activity diagram

6.5 COMPONENT DIAGRAM

a. User

a component diagram depicts how components are wired together to form larger components or software systems. They are used to illustrate the structure of arbitrarily complex systems

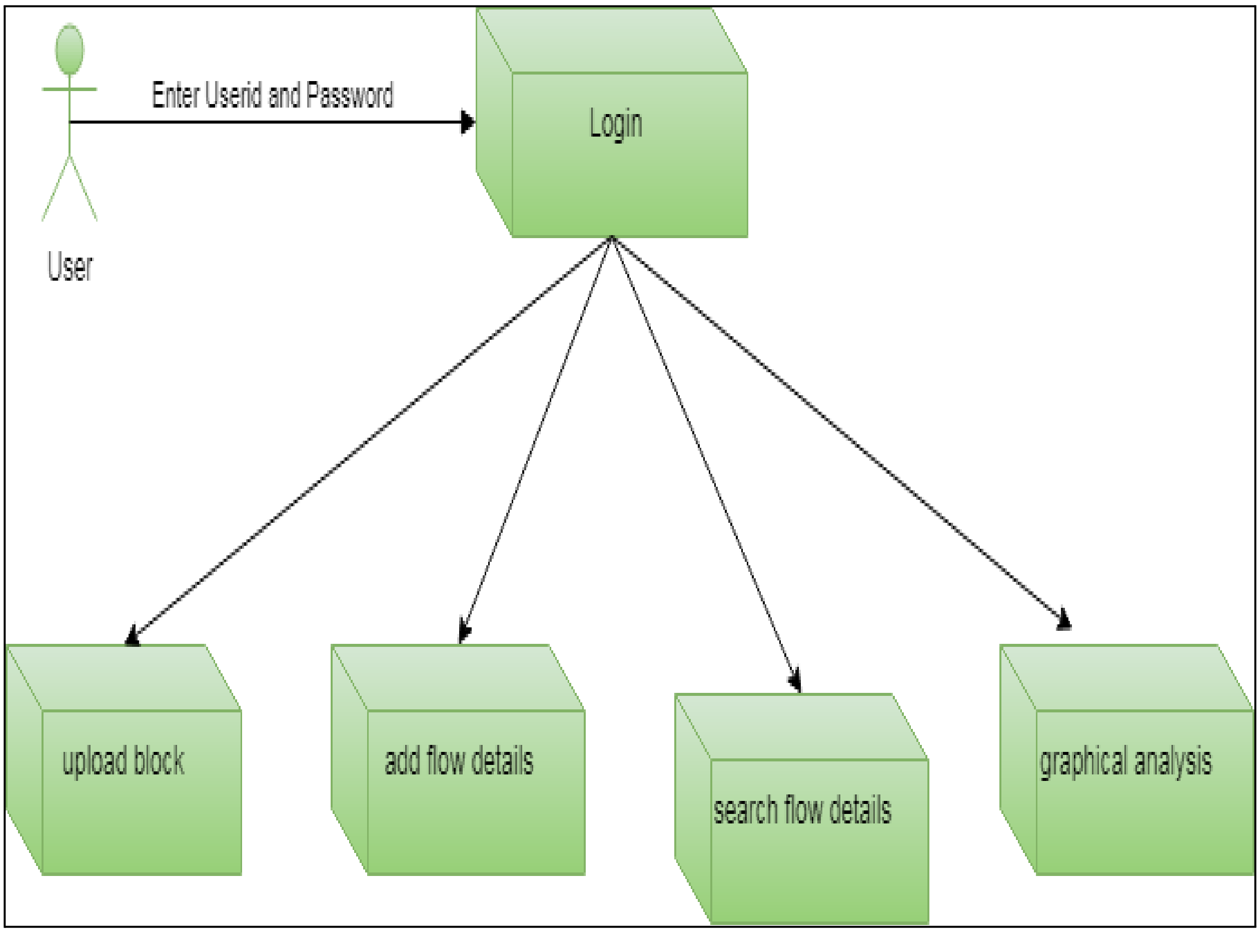


FIG6.5 component diagram

6.1.1 DATABASE DESIGN

6.1.2 *E-R* DIAGRAM

a. USER

An entity–relationship model describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types and specifies relationships that can exist between entities

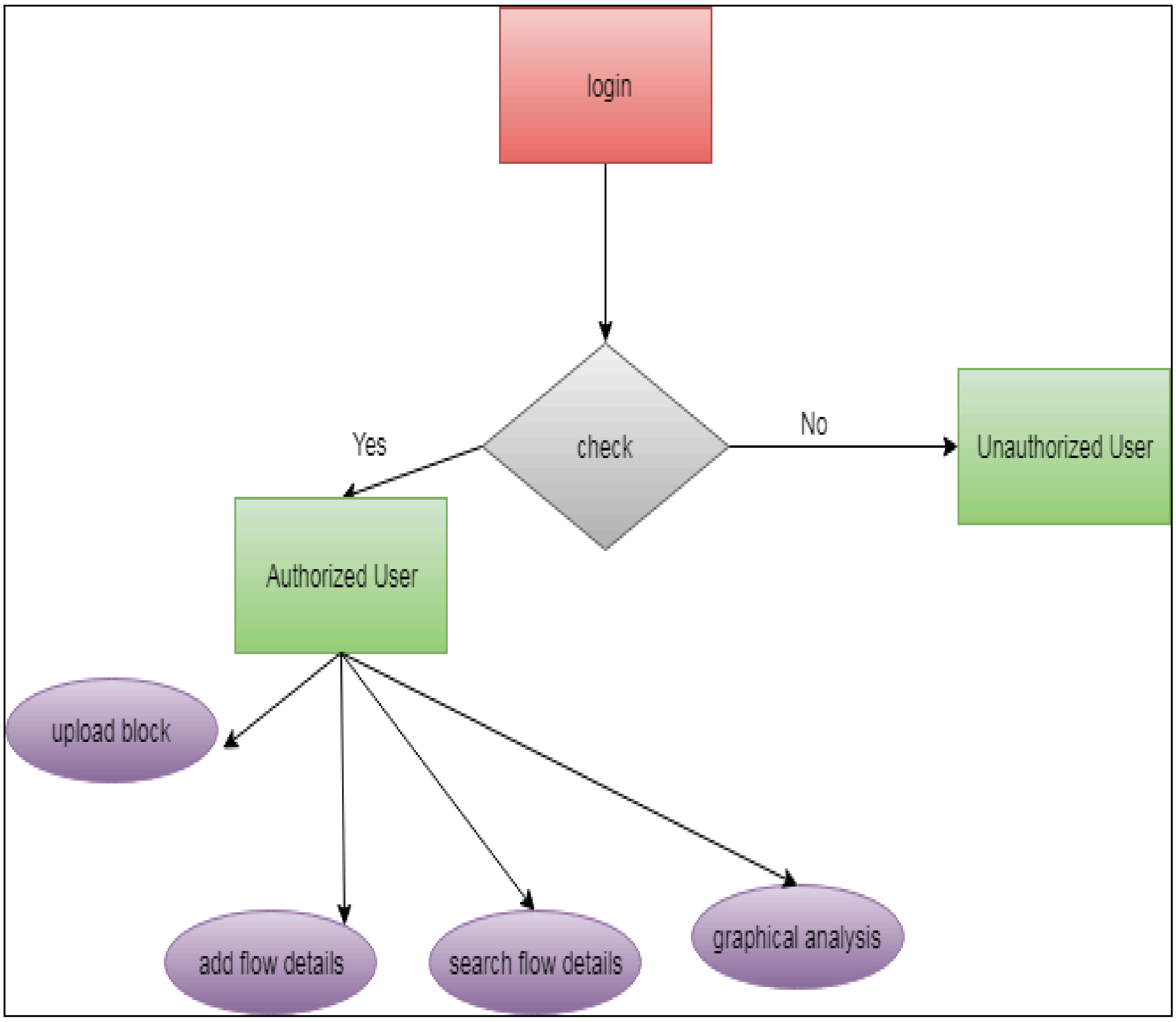


FIG 6.6 E-R DIAGRAM

7. CODING

7.1 DJANGO CONNECTION

```
import os

# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))


# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/1.11/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = '2rl!&twi#=wui@*+rf*gmu!_+gkkmi$j+imzjlbu++m9ah*$gr'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []


# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'user',
    'admins',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]
```

```
ROOT_URLCONF = 'Efficient_Vertical_Mining.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [(os.path.join(BASE_DIR,'assets/templates'))],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]

WSGI_APPLICATION = 'Efficient_Vertical_Mining.wsgi.application'


# Database
# https://docs.djangoproject.com/en/1.11/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'vertical_mining',
        'USER': 'root',
        'PASSWORD': '',
        'HOST': '127.0.0.1',
        'PORT': '3306',
    }
}


# Password validation
# https://docs.djangoproject.com/en/1.11/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME':
'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]


# Internationalization
# https://docs.djangoproject.com/en/1.11/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True

USE_TZ = True


# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/1.11/howto/static-files/

STATIC_URL = '/static/'
STATICFILES_DIRS=[os.path.join(BASE_DIR,'assets/static'),]
MEDIA_URL='/media/'
MEDIA_DIR=os.path.join(BASE_DIR,'assets/media')
```

7.2 USER DESIGN

```
<!DOCTYPE html>
{% load staticfiles %}
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
    body{
    background: url("{% static 'images15.jpg' %}");
 background-size: cover;
```

```
    }
        .menu table{
          width:100%;
          text-align:center;

          }
          .menu table td:hover{
          background:rgba(0,0,255,0.3);

          }
          .menu table td{
          background:rgba(0,0,255);

         }
          .menu table,.menu table th,.menu table td {
    border: ;
    border-collapse: collapse;

}
.menu table th,.menu table  td {
    padding: 15px;

    }


          .topic h1{
          color:black;
          padding:2px;
          text-align:center;
          border-style:none;
          height:100px;
          width:1330px;
          float:left;


          }
          .mainholder{

          position:relative;
          top:50px;
          left:50px;
          z-index:999;
          float:left;
          }


    </style>
</head>
<body>
<div class="background-image">
    <div class="topic"><h1
```

```
style="color:yellow;margin-top:10px;margin-left:30px;border-style:none;width:1300px;height:40px;bo
rder-color:black;background:;">DATA SECURITY APPROACH ON CYBER CRIME WITH WEB
VULNERABILITY</h1></div>
<div class="menu">
      <table>
         <tr>
            <td><a  style="color:yellow;text-decoration: none;" href="{% url 'user_page' %}">MY
DETAILS</a></td>
            <td><a  style="color:yellow;text-decoration: none;" href="{% url 'category_page' %}">
HOME PAGE</a></td>
            <td><a  style="color:yellow;text-decoration: none;" href="{% url
'select_purchase_item_number' %}"> SELECT COUNT OF ITEMS</a></td>
            <td><a  style="color:yellow;text-decoration: none;" href="{% url 'user_page' %}">
FEEDBACK</a></td>

            <td><a  style="color:yellow;text-decoration: none;" href="{% url 'index'
%}">LOGOUT</a></td>


         </tr>
      </table>
   </div>
</div>
</div>
<div class="marqee">

</div>
<div class="mainholder">
{% block userblock %}
{% endblock %}
</div>

</body>
</html>
```

7.3 ADMIN DESIGN

```
<!DOCTYPE html>
{% load staticfiles %}
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
    body{
    background: url("{% static 'images15.jpg' %}");
 background-size: cover;
    }
        .menu table{
          width:100%;
          text-align:center;
          }
          .menu table td:hover{
          background:rgba(0,0,255,0.3);

          }
          .menu table td{
          background:rgba(0,0,255);

         }
          .menu table,.menu table th,.menu table td {
    border: ;
    border-collapse: collapse;

}
.menu table th,.menu table  td {
    padding: 15px;

    }


          .topic h1{
          color:black;
          padding:2px;
          text-align:center;
          border-style:none;
          height:100px;
          width:1330px;
          float:left;


          }
          .mainholder{

          position:relative;
          top:50px;
          left:50px;
          z-index:999;
          float:left;
          }


    </style>
</head>
<body>
<div class="background-image">
    <div class="topic"><h1
style="color:yellow;margin-top:10px;margin-left:30px;border-style:none;widt
h:1300px;height:40px;border-color:black;background:;">DATA SECURITY
APPROACH ON CYBER CRIME WITH WEB VULNERABILITY</h1></div>
<div class="menu">
```

```
        <table>
            <tr>
                <td><a  style="color:yellow;text-decoration: none;"
href="{% url 'admin_page' %}">USER DETAILS</a></td>
                <td><a  style="color:yellow;text-decoration: none;"
href="{% url 'upload_page' %}"> UPLOAD PAGE</a></td>
                <td><a  style="color:yellow;text-decoration: none;"
href="{% url 'admin_page' %}"> FEEDBACK</a></td>
                <td><a  style="color:yellow;text-decoration: none;"
href="{% url 'admin_page'  %}"> CHART PAGE</a></td>
                <td><a  style="color:yellow;text-decoration: none;"
href="{% url 'login' %}">LOGOUT</a></td>


            </tr>
        </table>
    </div>
</div>
</div>
<div class="marqee">

</div>
<div class="mainholder">
{% block adminblock %}
{% endblock %}
</div>

</body>
</html>
```

7.4 TECHNOLOGIES USED

PYTHON

One of the best (and only) full-featured, dedicated IDEs for Python is PyCharm. Available in both paid (Professional) and free open-source (Community) editions, PyCharm installs quickly and easily on Windows, Mac OS X, and Linux platforms. Out of the box, PyCharm supports Python development directly.

An IDE (or Integrated Development Environment) is a program dedicated to software development. As the name implies, IDEs integrate several tools specifically designed for software development. These tools usually include:

• An editor designed to handle code (with, for example, syntax highlighting and auto-completion)

• Build, execution, and debugging tools

• Some form of source control

Most IDEs support many different programming languages and contain many more features. They can, therefore, be large and take time to download and install. You may also need advanced knowledge to use them properly.
In contrast, a dedicated code editor can be as simple as a text editor with syntax highlighting and code formatting capabilities. Most good code editors can execute code and control a debugger. The very best ones interact with source control systems as well. Compared to an IDE, a good dedicated code editor is usually smaller and quicker, but often less feature rich. Of course, there are lots of other features you might want, like source code control, an extension model, build and test tools, language help, and so on. But the above list is what I'd see as "core features" that a good editing environment should support.

Installation Procedure

Installing Python is generally easy. Nowadays many Linux and UNIX distributions include a recent Python. Even some Windows computers (notably those from HP) now come with Python already installed. If you do need to install Python, you can download from Python official website i.e., http://www.python.org.The engine that translates and runs Python is called the python Interpreter.
Steps for Installing:

• Open a Web browser and go to https://www.python.org/downloads/.

• Follow the link for the Windows installer python-XYZ.msi file where XYZ is the version you need to install.

• To use this installer python-XYZ.msi, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.

• Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.

There are two ways to use it:

1. Immediate mode
2. Script mode

Immediate mode

In this mode, you type Python expressions into the Python Interpreter window, and the interpreter immediately shows the results. The >>> is called the Python prompt. The interpreter users the prompt to indicate that it is ready for instructions. We type 2+2 , and the interpreter evaluated our expression, and replied 4, and on next line it gave a new prompt ,indicating that it is ready for more input.

Script mode

Alternatively, you can write a program in a file and use the interpreter to execute the contents of the file. Such a file is called a script.With these features in mind, let’s take a look at some general purpose tools we can use for Python development.

DJANGO

There are many different web frameworks under Python. Django is the most representative of the heavyweight players. Many successful websites and apps are based on Django.Django is an open source web application framework, written in Python. Django abides by the BSD copyright and was first released in July 2005, and the first official version 1.0 was released in September 2008 Django adopts the MVT software design pattern, namely model (Model), view (View) and template (Template). Django is an open source web application framework written in Python.Using Django, with very little code, Python program developers can easily complete most of the content required by a formal website, and further develop a full-featured Web service. Django itself is based on the MVC model, that is, Model + View (view) + Controller (controller) design mode, MVC mode simplifies the subsequent modification and expansion of the program, and makes it possible to reuse a certain part of the program.

• Low coupling
• Fast development
• Easy to deploy
• High reusability
• Low maintenance cost
• Powerful database function
• Comes with powerful background functions
• Elegant URL
• Django's MTV model is essentially the same as MVC, and it is also for maintaining loose coupling between components, but with a slightly different definition. Django's MTV refers to:M represents the model (Model): Write the functions that the program should have, and is responsible for the mapping (ORM) between business objects and the database.T stands for Template: responsible for how to display the page (html) to the user.V stands for View:

Responsible for business logic, and call Model and Template when appropriate.
• In addition to the above three layers, a URL distributor is also needed. Its function is to distribute URL page requests to different Views for processing. The View then calls the corresponding Model and Template.
• Build environment: The programming environment used by this machine is Python3.6, the coding tool is PyCharm, and the virtual environment is virtual env 。I like the latest, please make changes according to personal taste.
• Virtual environment: solve the problem of different versions of python and various toolkits. cmd enter the command line, and then start the following operations:
• Install the virtual environment: pip install virtualenv.
• Create a virtual environment: virtualenv virtual environment name. For convenience, I created a folder on the desktop, first enter: cd desktop, change the current directory to the desktop, and then create a virtual environment.
• Enter the virtual environment: enter: cd virtual environment name, enter the virtual environment folder, then enter: cd scripts, enter the secondary scripts folder, then enter: activate.bat, enter the virtual environment. In the virtual environment, "(virtual environment name)" will be prefixed before the code, and other codes in the virtual environment will also have this prefix until exiting the virtual environment.
• Enter python in the virtual environment: Enter: python, press enter, enter python and see its version information. ctrl+z: Quit python.

WAMPSERVER

Stands for "Windows, Apache, MySQL, and PHP." WAMP is a variation of lamp for Windows systems and is often installed as a software bundle (Apache, MySQL, and PHP). It is often used for web development and internal testing, but may also be used to serve live websites.
The most important part of the WAMP package is Apache (or "Apache HTTP Server") which is used run the web server within Windows. By running a local Apache web server on a Windows machine, a web developer can test webpages in a web browser without publishing them live on the Internet.
WAMP also includes MYSQL and PHP, which are two of the most common technologies used for creating dynamic websites. MySQL is a high-speed database, while PHP is a scripting language that can be used to access data from the database. By

installing these two components locally, a developer can build and test a dynamic website before publishing it to a public web server.
While Apache, MySQL, and PHP are opensource components that can be installed individually, they are usually installed together. One popular package is called "WampServer," which provides a user-friendly way to install and configure the "AMP" components on Windows.
The "P" in WAMP can also stand for either perl or python, which are other scripting languages. The Mac version of LAMP is known as MAMP.
Windows which is an ultimate platform for beginners and advanced users to operate, process and manage the different day to day computing tasks, however, if you are a developer and want to experience some of the most powerful software without paying a single penny then you should think about Linux. There are so many software packages that are only designed to run efficiently on Linux platforms such as Apache web server, PHP interpreter and MySQL database (LAMP).
In simple WAMPServer is software that bundled all these packages – Apache, PHP and MySQL for Windows environment to run them efficiently. It eliminates the developer's time spent in the cumbersome configuration environment process and freeing up more energy for development.
All this in Wamp can be operated from a simple graphical menu and configuration environment. PHP extension, Apache module, open / close the mouse to get it, no longer have to modify the configuration file in person, WAMP will do it. No longer need to ask PHP installation problems everywhere, WampServer will do everything for you. This software is completely free and can be downloaded to the latest version.

8. TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

Software Testing is the process used to help identify the correctness, completeness, security, and quality of developed computer software. Testing is a process of technical investigation, performed on behalf of stakeholders, that is intended to reveal quality-related information about the product with respect to the context in which it is intended to operate. This includes, but is not limited to, the process of executing a program or application with the intent of finding errors. Quality is not an absolute; it is value to some person. With that in mind, testing can never completely establish the correctness of arbitrary computer software; testing furnishes a criticism or comparison that compares the state and behavior of the product against a specification. An important point is that software testing should be distinguished from the separate discipline of Software Quality Assurance (SQA), which encompasses all business process areas, not just testing.

There are many approaches to software testing, but effective testing of complex products is essentially a process of investigation, not merely a matter of creating and following routine procedure. One definition of testing is "the process of questioning a product in order to evaluate it", where the "questions" are operations, the tester attempts to execute with the product, and the product answers with its behavior in reaction to the probing of the tester [citation needed]. Although most of the intellectual processes of testing are nearly identical to that of review or inspection, the word testing is connoted to mean the dynamic analysis of the product—putting the product through its paces. Some of the common quality attributes include capability, reliability, efficiency, portability, maintainability, compatibility and usability. A good test is sometimes described as one which reveals an error; however, more recent thinking suggests that a good test is one which reveals information of interest to someone who matters within the project community.

8.1 INTRODUCTION TO TESTING

In general, software engineers distinguish software faults from software failures. In case of a failure, the software does not do what the user expects. A fault is a programming error that may or may not actually manifest as a failure. A fault can also be described as an error in the correctness of the semantic of a computer program. A fault will become a failure if the exact computation conditions are met, one of them being that the faulty portion of computer software executes on the CPU. A fault can also turn into a failure when the software is ported to a different hardware platform or a different compiler, or when the software gets extended.Software testing is the technical investigation of the product under test to provide stakeholders with quality related information.

Software testing may be viewed as a sub-field of Software Quality Assurance but typically exists independently (and there may be no SQA areas in some companies). In SQA, software process specialists and auditors take a broader view on software and its development. They examine and change the software engineering process itself to reduce the amount of faults that end up in the code or deliver faster.

Regardless of the methods used or level of formality involved the desired result of testing is a level of confidence in the software so that the organization is confident that the software has an acceptable defect rate. What constitutes an acceptable defect rate depends on the nature of the software. An arcade video game designed to simulate flying an airplane would presumably have a much higher tolerance for defects than software used to control an actual airliner.

A problem with software testing is that the number of defects in a software product can be very large, and the number of configurations of the product larger still. Bugs that occur infrequently are difficult to find in testing. A rule of thumb is that a system that is expected to function without faults for a certain length of time must have already been tested for at least that length of time. This has severe consequences for projects to write long-lived reliable software.

A common practice of software testing is that it is performed by an independent group of testers after the functionality is developed but before it is shipped to the customer.

This practice often results in the testing phase being used as project buffer to compensate for project delays. Another practice is to start software testing at the same moment the project starts and it is a continuous process until the project finishes.

Another common practice is for test suites to be developed during technical support escalation procedures. Such tests are then maintained in regression testing suites to ensure that future updates to the software don't repeat any of the known mistakes.

It is commonly believed that the earlier a defect is found the cheaper it is to fix it. Unit tests are maintained along with the rest of the software source code and generally integrated into the build process (with inherently interactive tests being relegated to a partially manual build acceptance process). The software, tools, samples of data input and output, and configurations are all referred to collectively as a test harness.

8.2 TYPES OF TESTS

8.2.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

8.2.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at   exposing the problems that arise from the combination of components.

8.2.3 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.
Systems/Procedures	: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

8.2.4 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

8.2.5 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

8.2.6 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

8.2.7 Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

8.3 Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

8.3.1 Test objectives

All field entries must work properly.

Pages must be activated from the identified link.

The entry screen, messages and responses must not be delayed.

8.3.2 Features to be tested

Verify that the entries are of the correct format

No duplicate entries should be allowed

All links should take the user to the correct page.

8.4 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

8.5 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

8.6 Test Results

All the test cases mentioned above passed successfully. No defects encountered.

9. OUTPUT SCREENS

9.1 USER INTERFACE

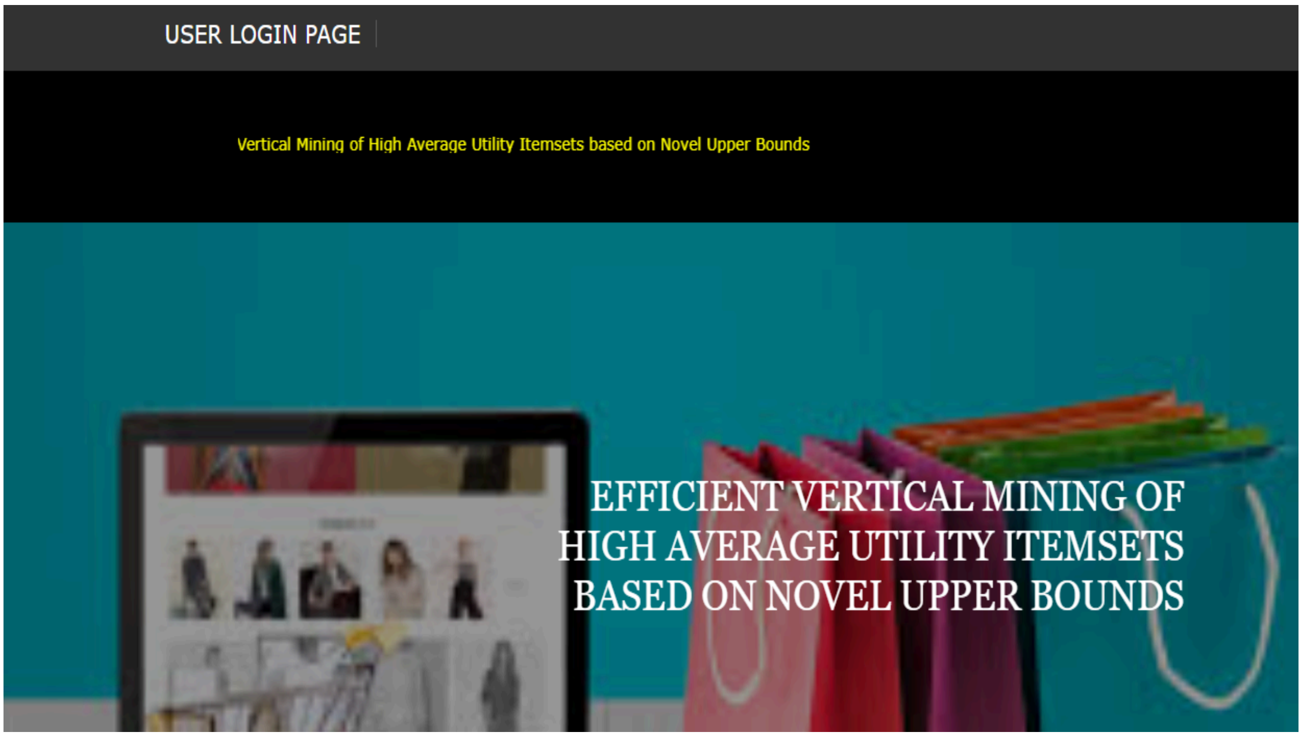

USER LOGIN PAGE
Vertical Mining of High Average Utility Itemsets based on Novel Upper Bounds
EFFICIENT VERTICAL MINING OF HIGH AVERAGE UTILITY ITEMSETS BASED ON NOVEL UPPER BOUNDS

Fig 9.1:USER INTERFACE

9.2 USER LOGIN

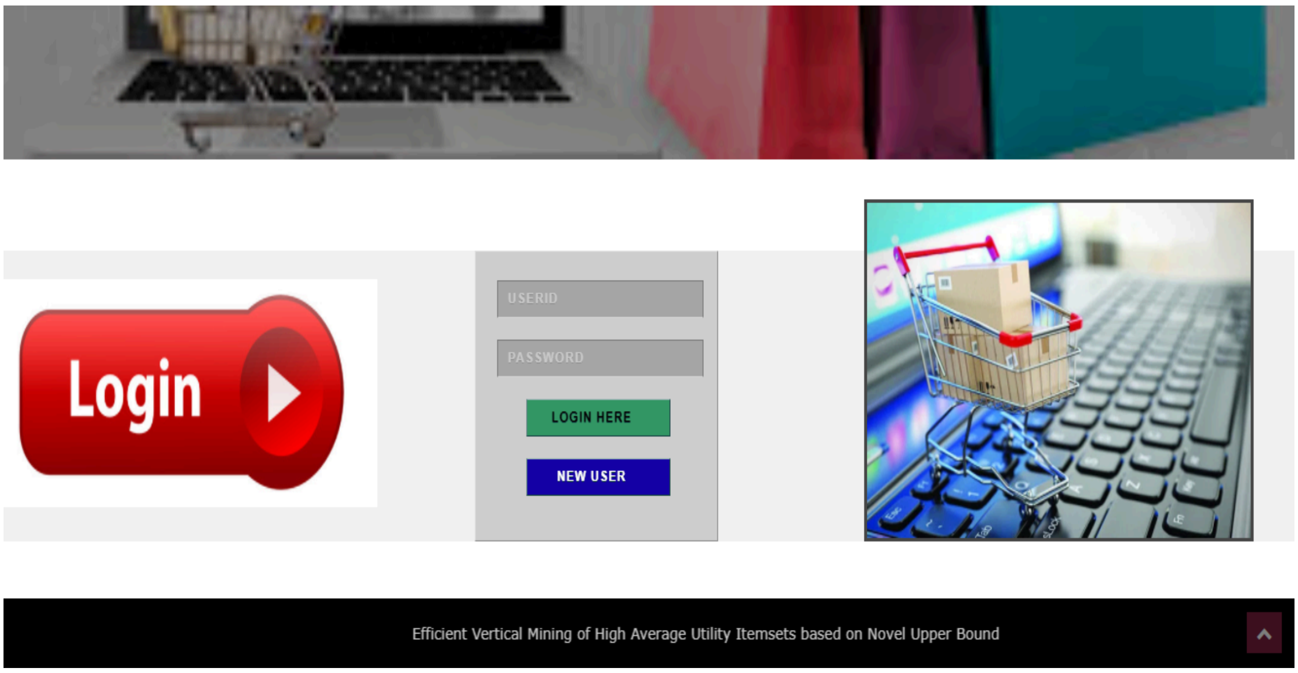

Login
USERID
PASSWORD
LOGIN HERE
NEW USER
Efficient Vertical Mining of High Average Utility Itemsets based on Novel Upper Bound

Fig9.2:USER LOGIN

9.3 WELCOME PAGE

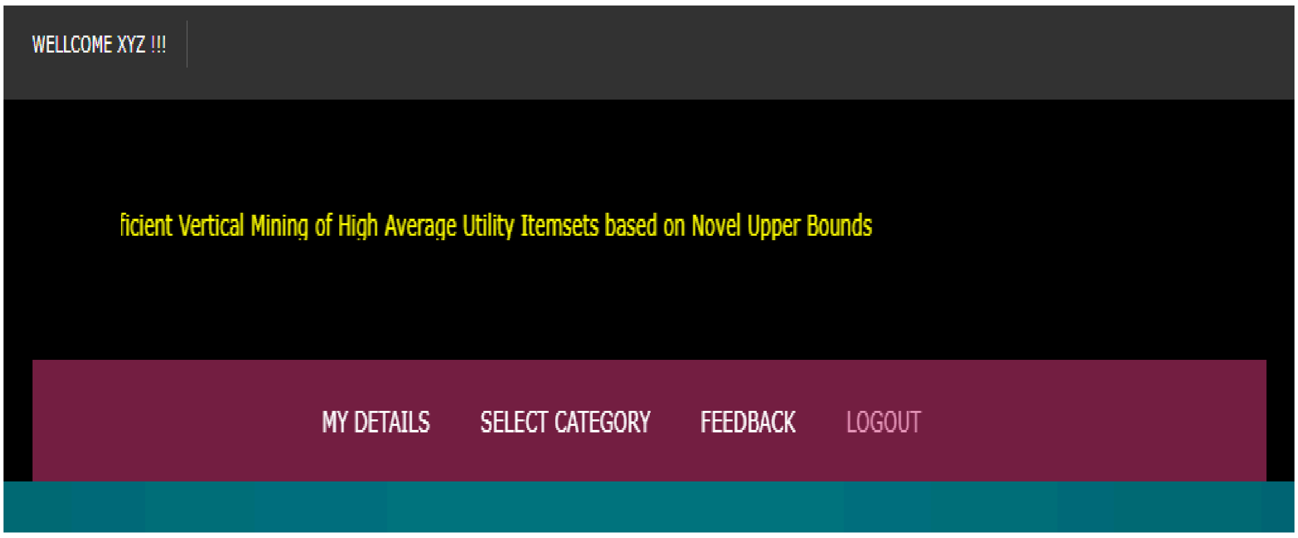
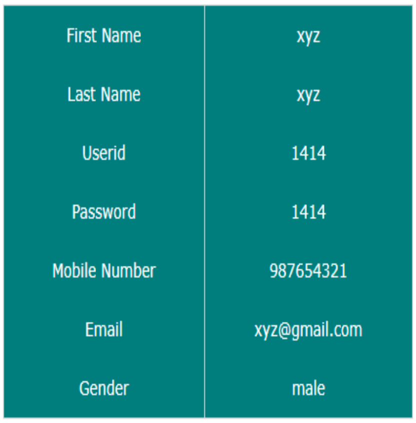

WELLCOME XYZ !!!
ficient Vertical Mining of High Average Utility Itemsets based on Novel Upper Bounds
MY DETAILS
SELECT CATEGORY
FEEDBACK
LOGOUT

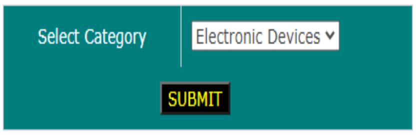
Fig9.3:WELCOME PAGE



First Name	xyz
Last Name	xyz
Userid	1414
Password	1414
Mobile Number	987654321
Email	xyz@gmail.com
Gender	male

Fig9.4:WELCOME PAGE2

9.3 CATAGORY SELECTION PAGE



Select Category | Electronic Devices

SUBMIT

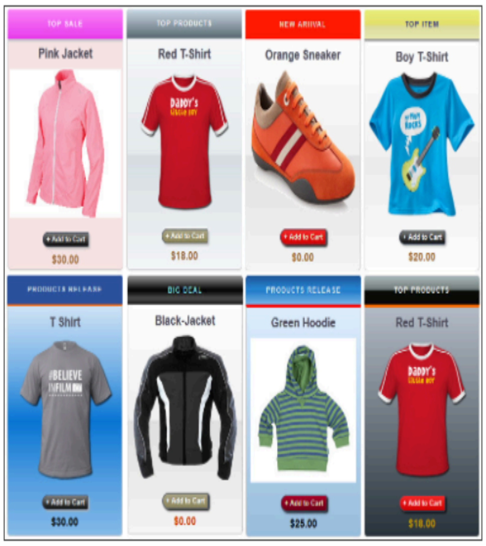
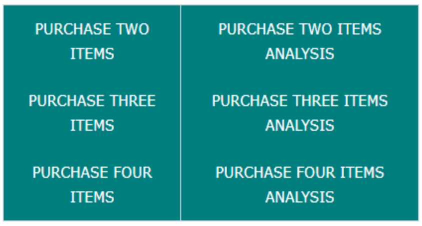
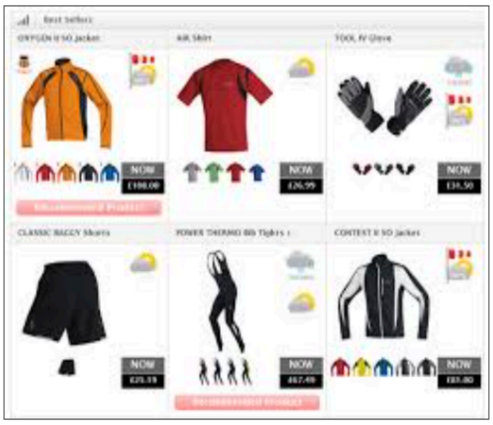


Fig9.5:CATAGORY SELECTION



PURCHASE TWO ITEMS	PURCHASE TWO ITEMS ANALYSIS
PURCHASE THREE ITEMS	PURCHASE THREE ITEMS ANALYSIS
PURCHASE FOUR ITEMS	PURCHASE FOUR ITEMS ANALYSIS



Efficient Vertical Mining of High Average Utility Ite

Fig9.6:ITEM SELECTION

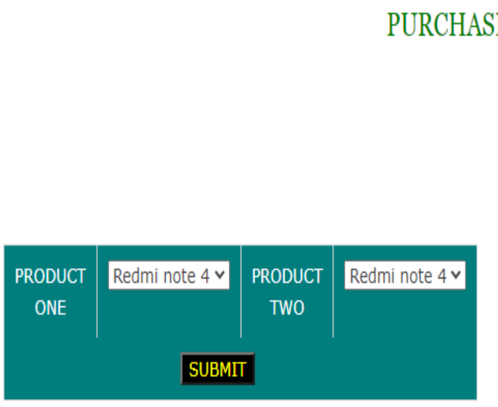


Fig9.7:DATA ENTRY

9.4 OUTPUT AND ANALYSIS

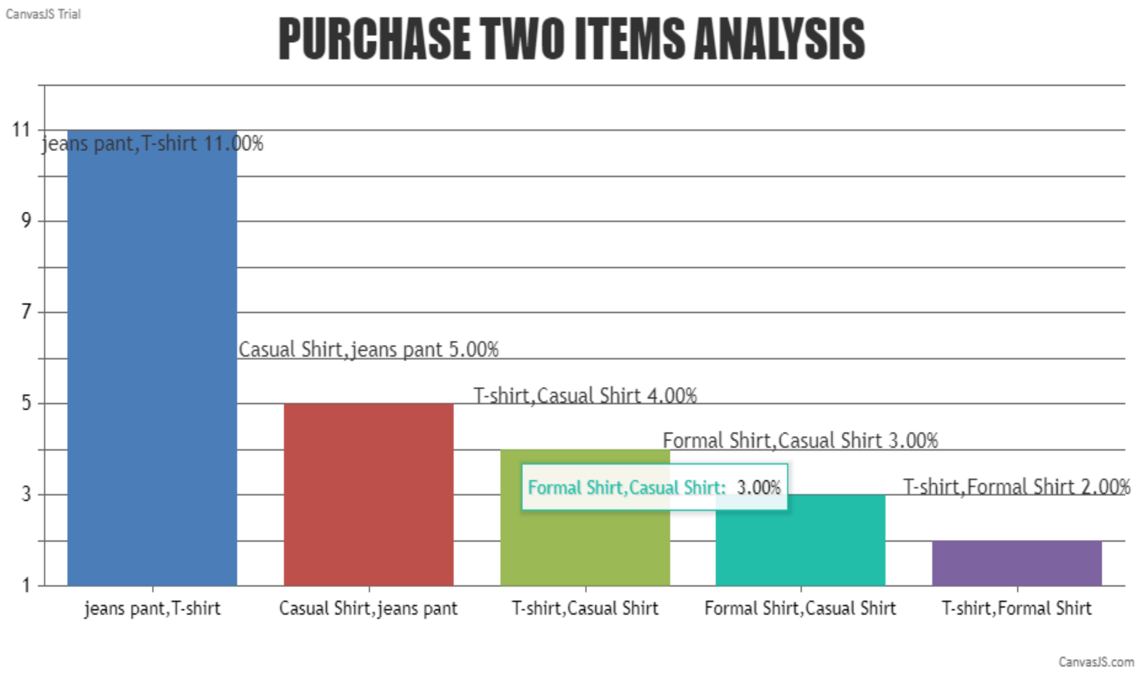


Fig9.8:OUTPUT

10. CONCLUSION

Based on the idea of computing utility values using a vertical form in quantitative databases, this paper has pro-posed four tight UBs, called aub1, iaub and laub, a generic framework to evaluate UBs in terms of their pruning effects, and three pruning strategies to eliminate unpromising candidates early. A new IDUL tree structure was also developed to quickly calculate the average utility and UBs of itemsets using a recursive process. A novel algorithm named dHAUIM has been further presented to efficiently mine high average-utility itemsets. An extensive experimental evaluation was carried out. Results have shown that dHAUIM outperforms four state-of-the-art HAUI mining algorithms in terms of execution time and number of join operations on both real-life and synthetic databases.

11 .FUTURE ENHANCEMENTS

In the past, the auub model was adopted in HAUIMto provide an upper-bound on the utilities of itemsets for maintaining the downward closure property. Because the auub model is based on the maximal utility in each transaction, it is greatly influenced by the utility values of items in transactions.In this paper, we present two efficient upper-bounds to further reduce the upper-bound values compared to the traditional auub model. The MAU-list structure is also developed to keep the necessary information for the mining process, and avoid performing multiple database scans. Three pruning strategies are respectively developed to prune unpromising itemsets early. Thus, the computational time and search space can be greatly reduced. Experiments on both real-life and synthetic datasets showed that the proposed algorithm significantly outperforms the state-of-the-art HAUI-Miner algorithm and the developed pruning strategies are efficient to speed up the mining performance, and can also be used with the traditional auub model.

12. BIBLIOGRAPHY

[1] Tamir Tassa."Secure Mining of Association Rules in Horizontally Distributed Databases," In *IEEE Transactions On Knowledge And Data Engineering,*

[2] R. Fagin, M. Naor, and P. Winkler."Comparing Information Without Leaking It," *Communications of the ACM*, 39:77–85, 1996

[3] Murat Kantarcıoˇglu and Chris Clifton," Privacy-preserving Distributed Mining of Association Rules on Horizontally Partitioned Data," in *Ieee Transactions On Knowledge And Data Engineering, To Appear, 29 Jan. 2003; revised 11 Jun. 2003, accepted 1 Jul. 2003.*

[4] D. W.-L. Cheung, V. Ng, A. W.-C. Fu, and Y. Fu, "Efficient mining of association rules in distributed databases," *IEEE Trans. Knowledge Data Eng.*, vol. 8, no. 6, pp. 911–922, Dec. 1996.

[5] D. Agrawal and C. C. Aggarwal, "On the design and quantification of privacy preserving data mining algorithms," in *Proceedings of the Twentieth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems* Santa Barbara, California, USA: ACM, May 21-23 2001, pp. 247– 255. [Online]. Available: http: //doi.acm.org/10.1145/375551.375602

[6] Y. Lindell and B. Pinkas, "Privacy preserving data mining," in *Advances in Cryptology –CRYPTO 2000*. Springer-Verlag, Aug. 20-24 2000, pp. 36–54. [Online]. Available: http://link.springer.de/link/ service/series/0558/bibs/1880/18800036.htm

[7] A. C. Yao, "How to generate and exchange secrets," in *Proceedings of the 27th IEEE Symposium on Foundations of Computer Science*. IEEE, 1986, pp. 162–167.