DATA MINING AND AN APPLICATION IN THE OPEN EDUCATION SYSTEM OF ANADOLU UNIVERSITY

Master Thesis

BILAL AL-RUBAIEE

Eskişehir, 2018

DATA MINING AND AN APPLICATION IN THE OPEN EDUCATION SYSTEM OF ANADOLU UNIVERSITY

BILAL AL-RUBAIEE

MASTER THESIS

Department of Statistics

Supervisor: Assoc. Prof. Dr. Atilla ASLANARGUN

Eskişehir

Anadolu University

Graduate School of Sciences

May 2018

FINAL APPROVAL FOR THESIS

This thesis titled "**Data Mining and an Application in The Open Education System of Anadolu University**" has been prepared and submitted by **Bilal Al-rubaiee** in partial fulfilment of the requirements in "Anadolu University Directive on Graduate Education and Examination" for the Degree of Master of Science in Statistics Department has been examined and approved on 31/05/2018.

Committee Members		**Signature**
Member (Supervisor)	: Assoc. Prof. Dr. Atilla ASLANARGUN	....
Member	: Prof. Dr. H. Kıvanç AKSOY	....
Member	: Assist. Prof. Dr. Sinan AYDIN	....

Prof.Dr. Ersin YÜCEL
Director of Graduate School of Sciences

ABSTRACT

DATA MINING AND AN APPLICATION IN THE OPEN EDUCATION SYSTEM OF ANADOLU UNIVERSITY

BILAL AL-RUBAIEE

Department of Statistics

Anadolu University, Graduate School of Sciences, May 2018

Supervisor: Assoc. Prof. Dr. Atilla ASLANARGUN

In Data mining, the discovery of frequent sets of items that occur together in a dataset is a fundamental task, particularly in transaction datasets. This thesis focus on closed frequent itemsets, a compressed representation of frequent patterns. Since mining all frequent itemsets generates duplicates and subsets, and with dense dataset, it gives a huge number of generated frequent itemsets which may be not understandable, then to avoid generate duplicates and subsets, and to get useful and compressed information about data under study, closed frequent itemset technique was adopted in this study.

In this thesis, the CHARM algorithm to discover closed frequent itemsets is proposed to identify frequent itemsets for Anadolu University Open Education System data to find out the collection of books taken together. The reason of choosing CHARM algorithm in this study is that it employs a vertical layout which facilitates the process of computing support count, moreover, it employs a novel way to traverse the search space.

The solution is implemented in R software environment for statistical computing and graphics. The result obtained shows that the encoded CHARM algorithm can get an efficient performance. Moreover, the result obtained could be useful to help decision makers to package the books in packets to achieve the aim of the study.

Keywords: data mining, frequent patterns, closed frequent itemsets, CHARM algorithm, Open Education System.

ÖZET

VERİ MADENCİLİĞİ VE ANADOLU ÜNİVERSİTESİ AÇIKÖĞRETİM SİSTEMİNDE BİR UYGULAMA

BILAL AL-RUBAIEE

İstatistik Anabilim Dalı

Anadolu Üniversitesi, Fen bilimleri Enstitüsü, Mayıs 2018

Danışman: Doç. Dr. Atilla ASLANARGUN

Veri madenciliğinde, veri kümesinde yaygın rastlanan öğelerin bulunması, özellikle işlem veri kümelerinde temel bir görevdir. Bu tez kapalı yaygın öğe setlerine odaklanmakta, yaygın görülen desenlerin sıkıştırılmış bir temsilidir. Madencilik, yoğun veri kümesi ile, tüm yaygın öğe çiftleri ve altkümeleri ürettiğinden, çok sayıda anlaşılabilir olmayan yaygın öğeler oluşturur, daha sonra kopyalamaları ve altkümeler oluşturmaktan kaçınmak için ve çalışma verileri hakkında yararlı ve sıkıştırılmış bilgi almak için, bu çalışmada kapalı yaygın öğe setleri tekniği benimsenmiştir.

Bu tez çalışmasında, Anadolu Üniversitesi AÇIKÖĞRETİM Sistemi verileri ile birlikte alınan kitapların koleksiyonunu öğrenmek için, yaygın öğeleri belirlemek için önerilmiş kapalı yaygın kümelerini keşfetmeye yönelik CHARM algoritması önerilmiştir. Bu çalışmada CHARM algoritmasının seçilmesinin nedeni, destek sayım işlem sürecini kolaylaştıran dikey bir düzen kullanmasıdır, ayrıca arama alanını geçmeye yönelik yeni bir yol kullanır.

Çözümler, istatistiksel bilgi işlem ve grafikler için geliştirilmiş R yazılımı ortamında uygulanmaktadır. Elde edilen sonuç kodlanmış CHARM algoritmasının verimli bir performans gösterebileceğini göstermektedir. Dahası, elde edilen sonuç, karar vericilerin kitapların paketlenmesini, çalışmanın amacına ulaşması için paketlemelerine yardımcı olmak açısından yararlı olabilir.

Anahtar Kelimeler: veri madenciliği, yaygın desenler, kapalı yaygın öğeler, CHARM algoritması, açıköğretim sistemi.

ACKNOWLEDGMENTS

I would like to express my sincerest gratitude to my supervisor, Associate Professor Dr. Atilla ASLANARGUN, I thank him for his guidance and encouragement, where good advice, support has been invaluable on both academic and personal levels.

I would like to specially thank Dr. Sinan AYDIN who has supported me throughout my thesis with his patience and unsurpassed knowledge.

Also, I would like to thank Prof. Dr. H. Kıvanç AKSOY for being on my thesis committee and for his valuable contributions.

Finally, I would like to thank my parents for their unconditional support over the years.

Bilal AL-RUBAIEE

May, 2018

31/05/2016

STATEMENT OF COMPLIANCE WITH ETHICAL PRINCIPLES AND RULES

I hereby truthfully declare that this thesis is an original work prepared by me; that I have behaved in accordance with the scientific ethical principles and rules throughout the stages of preparation, data collection, analysis and presentation of my work; that I have cited the sources of all the data and information that could be obtained within the scope of this study, and included these sources in the references section; and that this study has been scanned for plagiarism with “scientific plagiarism detection program” used by Anadolu University, and that “it does not have any plagiarism” whatsoever. I also declare that, if a case contrary to my declaration is detected in my work at any time, I hereby express my consent to all the ethical and legal consequences that are involved.

. . . .

Bilal AL-RUBAIEE

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

Data Mining (DM) has received much attention in recent decades and has been recognized as a promising filed due to the great improvement and high potential of computer techniques which helped to generate and store a huge amount of data in various fields. The basic issue addressed by (DM) process is to represent unclear and incomprehensible data into other forms that might be more understandable and more useful [6], therefore Data Mining is defined as the process of discovering insightful, interesting, previously unknown, potentially useful, and novel patterns, as well as understandable knowledge from large scale data [27]. (DM) is an interdisciplinary field, it involves an integration of a set of disciplines including, database system, statistics, machine learning, visualization, and information science [11].

Generally, there are two fundamental tasks of data mining; descriptive mining tasks which summarize and characterize the general properties of data in the database. While second task is predictive mining tasks which perform inference on the current data in order to make predictions. Below we described briefly the main descriptive and predictive functionalities of data mining.

Classification and Prediction, are two forms that can be used to extract models describing important data classes or to predict future data trends.

Classification, is the process of finding a model which describe data classes or concepts, for the purposes of being able to use the model to predict the classes of objects whose class label is unknown. To represent classification models there are various forms such as decision tree, Bayesian classification, and neural network techniques. Whereas classification technique predicts categorical (discrete, unordered) labels, prediction models as regression analysis predicts continues value.

Clustering, is a descriptive approach. Unlike classification that analyse class-labelled data objects. Clustering analyse data objects without a known class-labelled. The principle of clustering is to maximize the intra-class similarity and minimize the interclass similarity. That is objects within same cluster have high similar attribute values between each other, but are very dissimilar to objects in other clusters. Then, to gain some useful knowledge from the clusters, statistics can be used to describe the clusters.

Association Rules, one of the fundamental methods of data mining is association rules that describe relationships between items in datasets. The basic motivation for searching association rules came from the need to analyse so-called supermarket transaction data, or basket analysing, that is, to explore customer behaviour in term of purchased products. Association rules describe how often items are purchased together.

Outlier Detection, a database may contain data objects that do not follow the general behaviour or model of the data. These data objects are outliers. In some applications the rare events can be more interesting than others. To analyse outlier, we can use statistics tests or using distance measures or deviation methods to identify outliers by examining differences in the main characteristics of objects in a group.

Evolution Analysis, describe trends for subjects whose behaviour change over time it may include classification, prediction, or clustering of time related data, distinct feature of this analysis that it includes time-series data analysis.

A key component of many data mining tasks is frequent pattern mining (FPM) which play an essential role in many data mining tasks such as mining association rules, correlations, clustering, classification, outlier detection [1]. In general, (FPs) are referred to itemsets, subsequences, or substructure which appear in the target dataset no less than a user-specified threshold value.

Frequent itemsets consider one of the important kinds of (FPs) and it aims to extract useful information from the binary datasets or transactional datasets [14]. Frequent itemsets mining presented first time by introducing the “market-basket” model, where each basket (transaction) consists of a set of items called itemset. Usually the number of items in a basket smaller than the total number of items.

Since the introduction of frequent pattern mining in [2]. It has attracted great attention and many algorithms have been designed for finding the frequent sets.

Appriori algorithm [3] consider as the most popular algorithm to generate frequent itemsets, that It applies a generate-and-test approach to mine frequent patterns in bottom-up strategy. The algorithm first generates candidate itemsets of length k and then tests if they are frequent or not. The algorithm then generates candidates of length $k+1$ and so on.

However, aforementioned algorithm finds the set of all frequent itemsets. One of the major challenges is that, most of discovered itemsets are subset of another bigger itemsets. Therefore, it gives a lot of duplicated and redundant itemsets. However, with dense database where there are many frequent itemsets aforementioned algorithm will give redundant itemsets which can be unuseful. On the other hand, in many real world problems finding all frequent itemsets is not useful. One solution to avoid create unuseful itemsets is to discover compact representation of the frequent itemsets. This representation paves an easier way to understand the knowledge. From this point of view, two types of compact representation: maximal and closed frequent itemsets are alternative approaches attempt to achieve better efficiency and with less redundant itemsets.

Maximal frequent itemsets (MFI), itemsets that are frequent and cannot be extended because it has no frequent superset. Many algorithms have been designed for mining efficiently the MFI, e.g. the Max-Miner [4], MAFIA [5], and GenMax [9].

Closed frequent itemsets (CFI), is defined as an itemset that is frequent and its extension does not have the same support, i.e. does not have any frequent superset with the same support [1]. There are several CFI algorithms, e.g. close [18], CHARM [25, 26] , CLOSET [18] and CLOSET+ [22].

Many remarkable algorithms have been developed, in addition a lot of researches have contributed to frequent itemsets mining. One common point a lot of these researches share is that they focus on the performance of algorithms in terms of speed and time and attempting to get better performance.

In this thesis, we attempt to apply the concept of closed frequent itemsets (FCI) in a real world problem. In brief, the present study aims to identify books that are taken together by students in Anadolu University Open Education System to help decision maker in storage and distribution processes.

After an intensive search and for reasons that will be explained in chapter 4, we decided to implement the CHARM algorithm presented in [25, 26]. In addition, we also utilize the statistical language R to write our algorithm's code which is known for its ability in statistical analysis and handling of large data.

The rest of thesis is organised as follows:

- **Chapter 2:** a brief summary on data mining and the main tasks of data mining
- **Chapter 3:** addresses the concept of frequent pattern mining. It also gives a comprehensive survey of the most influential algorithms of frequent pattern mining, the compact representation of frequent patterns.
- **Chapter 4:** discusses the implementation of CHARM algorithm and also some descriptive statistics on the data presented in this chapter.
- **Chapter 5:** preparation the dataset and the result obtained presented in this chapter.
- **Chapter 6:** the final chapter presents the conclusion based on the finding and include recommendations of the thesis.

2. DATA MINING

In this preliminary chapter data mining and the relationship between data mining and various disciplines that contribute to this field has discussed. Then a brief explanation of the main tasks of data mining has been given.

2.1 Data Mining

Simply, data mining (DM) defined as the process of automatically discovering useful knowledge and information in big amounts of datasets [11,21]. Data mining considered as an interdisciplinary field that is combining concepts from associated areas like statistics, machine learning, database systems, and other disciplines.

Not all information discovery processes are considered to be as a data mining task. For example, finding particular web page using a keyword to an Internet search engine is not data mining technique [21].

2.2 Data Mining and Knowledge Discovery

Data mining is a fundamental step in the process of knowledge discovery in database (KDD). The overall (KDD) process as shown in figure 2.1 consists of three main steps as follows:

- Data pre-processing
- Data mining
- Data post-processing

Data pre-processing consists of four stages aims to collect and prepare raw data and transform it into an appropriate format for mining process. Data preprocessing stages are:

1. Data cleaning
2. Data integration
3. Data selection
4. Data transformation

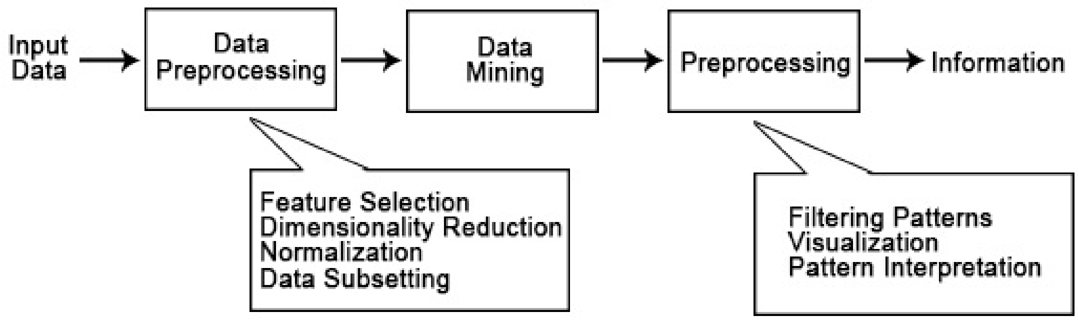

Figure 2.1: *The process of knowledge discovery in database (KDD)*

Next, after data prepared, intelligent methods are applied to extract high-level information and useful knowledge, this process is known as the data mining process.

The post-processing phase which aims to identify the most interesting pattern and then present the obtained information in different ways as visualization and other knowledge representation techniques [21].

2.3 The Origins of Data Mining

The steady and significant advances in computer techniques led to generate and collect datasets with sizes of gigabytes, terabytes, and even petabytes. Further the high dimensionality of collected data, that becomes common to encounter data with thousands of attributes. All this made researchers from different disciplines to focus on establishing more efficient and measurable tools in which handles various types of data. In practical, the field of data mining derived from disciplines such as Statistics, machine learning, artificial intelligence, visualization, database technology, and some other disciplines [21].

2.4 Data Mining and Statistics

In this section, a brief explanation of the differences between statistical techniques and data mining is discussed.

Statistics is the way to create methods to extract information from data. However, with data mining problems, especially those emerging from large amounts of data, statistics may not be enough to find information from data. Even so, statistics play an essential role in data mining field. That is considered as a vital component in any data mining processes [13].

It is not possible to understand even simple facts about data properties in large amounts of data which explains the complex examination methods may be demanded to understand the simple feature of data which would be easily apparent in small datasets. Furthermore, in generally the object of data mining is to generate some inferences beyond the available database [13].

Since statistical methods are so fundamental to data mining, But large datasets are considered as the most substantial difference between standard statistical applications and data mining.

In addition to the size of data, the other aspect that distinguishes data mining from statistics is that classical statistics methods perform well with the classical format of data, where rows represent objects and columns represent variables. Many types of data have a different structure such as, datasets come from the internet which creates a need for appropriate tools and methods from outside the field of statistics [7].

In summary, while data mining does interfere with the standard exploratory data analysis techniques of statistics, it confronts new problems, many of which are outcomes of size and the non-traditional nature of concerned datasets [13].

2.5 Data Mining Tasks

In general, Data mining tasks are separated into two main categories:

Predictive tasks the purpose of these tasks is to predict the value of a particular attribute based on the values of other attributes. The attribute to be predicted is known as the target or dependent variable, while the attributes utilized for the prediction are recognized as the explanatory or independent variables [21]. Predictive data mining tasks contain classification and regression analysis.

Descriptive tasks the purpose of this task is to determine patterns (correlations, clusters, paths, and outliers) that summarize the relationships between variables in data. Descriptive data mining tasks often do not contain target variables [21]. Descriptive tasks contain clustering, association rules analyzing, and outlier detection.

Classification and Regression

Classification and regression both refer to the task of finding a model for predicting the target variable or dependent variable as a function of the explanatory or independent variables. While classification models used for discrete variables, regression models used for continuous variables [11].

Clustering

Clustering is a descriptive data mining task, it analysis unlabelled data and it aims to clustered or grouped data based on similarity. That is items that belong to the same class or cluster are more similar to each other, further it is dissimilar with the items that belong to another cluster of classes [11,21]. For instance, clustering can be applied in marketing to identify homogeneous groups of customers.

Outlier Detection

A dataset may contain data objects that whose characteristics are significantly different from the general model of the data. These data objects are outliers. Most methods treat outliers as exceptions or noise. However, in some implementations, such as fraud detection, the events occur rarely can be more interesting than the more often occurring ones [11].

Association Rules

Association rules are used to determine patterns that represent highly associated features in the dataset. It consists of two main phases: First, finding a pattern that appears together, in another word, discover frequent patterns which is the main phase of association rules, then find the relationship between this patterns. Association rules aim to discover the most interesting patterns in an efficient approach. It first used in the market basket analyses, to discover the items have purchased together and to learn about customer behavior [11,21].

3. FREQUENT PATTERN MINING

In this chapter, first the problem of frequent pattern mining explained, then the fundamental techniques used to solve frequent pattern mining described, after that, a comprehensive overview of the most influential algorithms proposed to mine frequent patterns. At last, a compressed representation to reduce the discovered patterns discussed.

3.1 Problem Definition

The formal model and problem statement of the frequent itemsets mining problem are introduced in [2, 27]. In this section, several vital concepts of frequent pattern mining (FPM) will be defined.

Let $I = \{x_1, x_2, \ldots, x_m\}$ be a set of elements called items in a given dataset. Let X be a set of some items in I, i.e. $X \subseteq I$, it called an itemset or pattern. The size of an itemset is defined by the number of items it contains i.e., if $I = \{x_1, x_2, \ldots, x_k\}$ is an itemset contain k items then it can be defined as an itemset of size k.

Let $T = \{t_1, t_2, \ldots, t_n\}$ be another set of elements called transaction identifiers or $tids$. A set of some $tids$, $t \subseteq T$ is called a $tidset$.

Each transaction is a couple of form $\langle t, X \rangle$, where $t \in T$ is a unique transaction identifier, and X is an itemset. A transaction t is said to contain itemset Y if and only if $Y \subseteq X$.

Let the database D be a set of transactions, which can be defined as a binary relation on the set of the transaction identifier and the set of items, i.e., $D = T \times I$. In the binary representation of the database, an item can be treated as a binary variable takes either *one* if the item presented in the transaction or *zero* if the item did not appear in the transaction.

Example: consider the binary dataset consists of five items and six transactions as shown in table 3.1a. If the item x present in the transaction tid then the corresponding cell takes one. Otherwise, it takes $zero$.

Table 3.1 – An example database

tid	A	B	C	D	E
1	1	1	0	1	1
2	0	1	1	0	1
3	1	1	0	1	1
4	1	1	1	0	1
5	1	1	1	1	1
6	0	1	1	1	0

***(a)** Binary Dataset*

tids	A
1	ABDE
2	BCE
3	ABDE
4	ABCE
5	ABCDE
6	BCD

***(b)** Transaction Database*

x	A	B	C	D	E
t(x)	1	1	2	1	1
	3	2	4	3	2
	4	3	5	5	3
	5	4	6	6	4
		5			5
		6			

***(c)** Vertical Database*

Definition 1. *The support of an itemset X in transaction database D defined as the number of transactions in D containing X and denoted as $sup(X,D)$ or $sup(X)$ [27].*

$$sup(X) = |\{tid|\langle tid, I\rangle \in D \quad and \quad X \subseteq I\}| \tag{3.1}$$

Definition 2. *The relative support of X defined as an estimate of the joint probability of the items containing X and denoted as $rsup(X)$ [27].*

$$rsup(X) = \frac{sup(X)}{|D|} \tag{3.2}$$

Definition 3. *An itemset X is called frequent itemset or frequent pattern if its support is equal or greater than a user-specified minimum support threshold σ, while $0 \leq \sigma \leq |T|$ [8, 19, 27].*

When minimun support is specified as fraction i.e., $0 \leq \sigma \leq 1$, then we assume that relative support is implied. Moreover, to calculate the relative minimum support threshold mathematically;

$$\sigma_r = \frac{\sigma}{|T|} \tag{3.3}$$

Definition 4. *The collection of all frequent itemsets in transaction database D with respect to a user-specified minimum support threshold σ is denoted by;*

$$F(D,\sigma) = \{X \subseteq I | sup(X) \geq \sigma\} \tag{3.4}$$

In this section, the concept of frequent itemsets and minimum support threshold discussed and mathematical forms have given. Next section, a brief survey on the computational complexity of finding frequent itemsets will be described.

3.2 Itemset Mining

The task of discovering all frequent itemsets is quite challenging. First of all, the search space grows exponentially with the number of items occurring in the database. That is every itemset $X \subseteq I$ is potentially frequent. Therefore, the initial search space of frequent itemsets consists of the power set of I without the empty set i.e., $2^{|I|} - 1$. If I is large enough, then search space will be extremely large therefore it is not practicable to determine support count of all of candidate itemsets. That what makes the support count procedure a tough problem, especially that frequent pattern mining deals with massive databases containing millions of transactions [8].

In this section, some search space traversal and data representation methods are discussed in order to facilitate these two issues.

3.2.1 Search strategy

A search for frequent itemsets conceptually can be viewed as the traversal on the itemset search space [21]. The search strategy employed by an algorithm determines the way how the search space is traversed during the frequent itemset generation process [21]. Depending on the order of frequent itemsets in the search space some search strategies can be better than others. An overview of search strategies is given below:

Bottom-Up search versus Top-Down search

The bottom-up approach is when pairs of frequent $(k-1)-$itemsets are merged to obtain candidate $k-$itemset. This strategy is useful when the maximum length of frequent itemsets is not too long. Apriori algorithm [20] adopt a bottom-up search strategy. Alternatively, Top-Down approach starts with the top element of the search space. The advantage of this approach that it is useful to discover maximal frequent itemsets (will be explained later) where the border of frequent itemsets is located near the top of the search space then it can quickly determine the set of maximal itemsets [21, 28].

Equivalence classes

In this approach, first, partition the search space into disjoint groups, after that an algorithm start generate frequent itemsets within particular class, then moving to the next class. Also, it can be divided into two parts or groups, prefix-based equivalence classes, and suffix-based equivalence classes. The prefix-based approach, the algorithm starts with the first prefix and after moving to the next one and so on. In contrast, suffix-based approach starts from the maximal frequent itemset and moving to its subsets and so on. Both approaches can be explained using tree structure [21].

Breadth-First versus Depth-First

One of the commonly employed approaches to traverse the search space is breadth-first manner (BFS). In this approach, the algorithm first starts with discovering all the frequent 1-itemsets, then all the frequent 2-itemsets and so on, until all frequent itemsets are discovered. In contrast, depth-first approach (DFS), recursively visit the descending of an itemset until it infrequent level reached then it passes to the next item and so on. This approach allows to detect the border of frequent itemsets more quickly than breadth-first, and for this reason often employed in algorithms that designed to discover maximal frequent itemsets [21].

3.2.2 Data representation

A vital consideration in most algorithms is the dataset representation. Conceptually, transaction databases represented by a two-dimension matrix in which every row represents an individual transaction $t_i \in T$, and every column represent an individual item $x_i \in I$. Each transaction represented by one for the items occurring in the transaction and zero for that item that does not occur in the transaction as shown in Table 3.1a

Up to date, there are two main database representation layouts. *The horizontal data layout*; that is for each transaction there is a transaction identifier and list of items occurring in that transaction. This manner adopted by *Apriori-like* algorithms. Another representation, *the vertical data layout*; in which the database consists of a set of items, each item followed by its transaction identifier set $tidset$. Both layouts are shown in figure 3.1b, 3.1c sequentially.

Data representation plays an essential role in computing support count. For instance, in a horizontal layout to compute the support of an itemset $X \subseteq I$ one needs to scan the dataset completely and check for every transaction t whether $X \subseteq t$ or not [8]. In contrast, the vertical layout has more advantage in computing support count in which the support of an itemset $X \subseteq I$ can be computed merely by intersecting the *tidsets* of its subsets [8, 28].

3.3 The Common Algorithms of Frequent Pattern Mining

Frequent pattern mining (FPM) is considered as one of the most intensively researched problems in data mining. In the last decades, many algorithms in different frameworks to solve the problem of discovering frequent pattern mining have been designed. The support-based framework is the most common framework which took the most significant attention [1]. In this framework, the frequent itemsets are that one with a frequency equal to or greater than a user-specified threshold. A fundamental issue of generating candidate itemsets in (FPM) appear in eliminating non-relative and duplicate candidates [1].

In this section, common algorithms for discovering frequent patterns will be discussed. These algorithms can be labeled according to the way or strategy of discovering the search space such as join-based exploration, prefix-based depth-first exploration, or suffix-based depth-first exploration [1]. The other aspect is the database representation layout, such as horizontal or vertical layout.

3.3.1 Apriori algorithm

The most common algorithm for generating all frequent itemsets is Apriori algorithm. Its improved and designed in [3], after the introduction of frequent pattern mining in a short period. In the same time, the same technique independently proposed by [16].

The Apriori algorithm [3] employs a breadth-first approach to generate all frequent itemsets in which all frequent itemsets of length $k-1$ are generated before that those of length k. The essential technique which is employed in Apriori algorithm is the downward closure property.

Definition 5. *Downward Closed Property, Given a transaction database D over a set of items I, let* $X, Y \subseteq I$*be two itemsets. Then*

$$X \subseteq Y \Longrightarrow sup(Y) \leq sup(X) \tag{3.5}$$

This property means, all subsets of frequent itemset are frequent, and all supersets of infrequent itemsets are not frequent. From this concept, the Apriori method uses the join-based approach that is generated candidates itemsets of length $k+1$ from known frequent itemsets of length k as long as they have $k-1$ items in common [1].

Apriori algorithm consists of two phases. The first phase, the algorithm passes over the data to count items occurrence to determine the set of all frequent 1-itemsets. The second phase, a subsequent pass called pass k that is repeated over and over again, consists of two steps. The first, join step, the algorithm iteratively generate new candidate itemsets C_k by in which joins the frequent $(k-1)-$itemsets F_{k-1} found in the previous iteration to generate frequent $k-$itemsets F_k. Next the prune step, in this step eliminates all infrequent candidates that is not belong to the frequent $(k-1)-$itemsets F_{k-1}. The algorithm terminates when the iteration cannot generate any new frequent itemsets. The pseudo-code for generating frequent itemsets in Apriori algorithm shown in algorithm 1. Let C_k denote to the set of candidate $k-$itemsets and F_k denote the set of frequent $k-$itemsets.

Algorithm 1: APRIORI Algorithm

```
APRIORI ($D, I, minsup$)
$F = \phi$
$C^{(1)} = \{\phi\}$ // Initial prefix tree with single items
foreach $x \in I$ do
    Add $x$ as child of $\phi$ in $C^{(1)}$ with $sup(x) = 0$
$k = 1$ // $k$ denotes the level
while $C^{(k)} \neq \phi$ do
    COMPUTE SUPPORT ($C^{(k)}, D$)
    foreach $leaf\, X \in C^{(k)}$ do
        if $sup(X) \geqslant minsup$ then
            $F \longleftarrow F \cup \{\langle X, sup(X) \rangle\}$
        else
            Remove $X$ from $C^{(k)}$
    $C^{(k+1)} \longleftarrow$ EXTEND PREFIXTREE $C^{(k)}$
    $k \longleftarrow k+1$
return $F^{(k)}$
COMPUTE SUPPORT ($C^{(k)}, D$)
foreach $\langle t, i(t) \rangle \in D$ do
    foreach $k-subset\, X \subseteq i(t)$ do
        if $X \in C^{(k)}$ then
            $sup(X) \longleftarrow sup(X) + 1$
EXTEND PREFIXTREE $C^{(k)}$
foreach $leaf\, X_a \in C^{(k)}$ do
    foreach $leaf\, X_b \in SIBLING(X_a)$, such that $b > a$ do
        $X_{ab} \longleftarrow X_a \cup X_b$
        // prune candidate if there are any infrequent subsets
        if $X_j \in C^{(k)}$, for all $X_j \subset X_{ab}$ such that $|X_j| = |X_{ab}| - 1$ then
            Add $X_{ab}$ as a child of $X_a$ with $sup(X_{ab}) = 0$
    if no extensions from $X_a$ then
        remove $X_a$ and all ancestors of $X_a$ with no extensions from $C^{(k)}$
return $C^{(k)}$
```

3.3.2 Eclat algorithm

The earlier version of the Equivalence CLAss Transformation (Eclat) algorithm was presented in [28]. The key feature of this algorithm is that it used the vertical database layout to represent the dataset, where *tid-list* for each item. The vertical representation facilitates the computing process of the support count by simple tid-list intersections.

Another feature of Eclat in comparison with Apriori is candidate generation process. Whereas, It generates candidate as similar as to Apriori in which, each candidate $k-$itemset within a lattice partition generated from frequent $(k-1)-$itemsets. However, Eclat uses prefix-based equivalence to decompose the lattice into small sub-lattices and it traverse on first prefix and its sub-lattice then backward to the next prefix and son on.

Definition 6. *Equivalence relation, on P is a binary relation $\equiv$ such that for $X,Y,Z \in P$ can be define as follows:*

1. *Reflexive $X \equiv X$*
2. *Symmetric $X \equiv Y$ implies $Y \equiv X$*
3. *Transitive $X \equiv Y$ and $Y \equiv Z$ implies $X \equiv Z$*

Equivalence relation partitions the set into disjoint sets called Equivalence classes. The Equivalence classes of item $X \in P$ is given as $[X] = \{Y \in P | X \equiv Y\}$.

Difine an Equivalence relation θ_k on the lattice $P(I)$ as follows:

$$\forall X,Y \in P(I), X \equiv_{\theta_k} Y \Leftrightarrow p(X,k) = p(Y,k) \tag{3.6}$$

For every two itemsets in the same class if they have a $k-$length prefix in common, and θ_k a prefix-based equivalence relation. The pseudo-code of Eclat algorithm is given in algorithm 2:

3.3.3 FP-Growth algorithm

This algorithm takes a radically different approach to discover frequent itemsets. This algorithm avoids the candidate generation method which is used by Apriori-like methods. Instead, it uses a novel data structure denoted by frequent pattern tree (FP-tree) [12].

Algorithm 2: ECLAT Algorithm

```
initial call: $F \longleftarrow \phi, P \longleftarrow \{\langle x, t(x) \rangle | x \in I, sup(x) \geqslant minsup\}$
ECLAT $(P, minsup, F)$
initialization;
foreach $\langle x_a, t(x_a) \rangle \in P$ do
    $F \longleftarrow F \cup \{\langle x_a, sup(x_a) \rangle\}$
    $P_i \longleftarrow \phi$
    foreach $\langle x_b, t(x_b) \rangle \in P$ with $X_b > X_a$ do
        $X_{ab} = X_a \cup X_b$
        $t(X_{ab}) = t(X_a) \cap t(X_b)$
        if $sup(X_{ab}) \geqslant minsup$ then
            $P_a \longleftarrow P_a \cup \{\langle X_{ab}, t(X_{ab}) \rangle\}$
    if $P_i \neq \phi$ then
        ECLAT $(P_a, minsup, C)$
```

In this algorithm, the process of frequent pattern mining consists of two steps [19].

1. Construct FP-tree data structure to store the information in a compact space.
2. Develop an FP-tree based pattern growth (FP-Growth) to discover all frequent itemsets.

FP-Tree representation

To design a compact data structure, it requires two scans on transaction database. The first scan computes the support count of each item and infrequent items discarded and then stores frequent items in frequency decreasing order. In fact, frequent 1-itemsets are generated in this procedure. The second scan constructs FP-tree [12].

FP-tree is a prefix-tree structure; it consists of one root named *"null"* as the children of the root there is a set of *item prefix subtree*. Each node of the *item prefix subtree* consists of three parts; *item name*, *frequency*, and *node-link*. For any frequent item *x* by following *x node-link* all possible frequent itemsets contain *x* can be obtained.

Mining frequent itemsets using FP-tree

FP-Growth approach that generates frequent itemsets based on FP-tree structure. The algorithm traverses the data from the least frequent item in the set of frequent itemsets, then to discover frequent itemsets ending with a particular item using the suffix-based approach. The key strategy employed by the FP-Growth algorithm is a *divide-and-conquer* strategy that divides the mining tasks into small sub-databases tasks. To illustrate how the *divide-and-conquer* approach performs as follows:

1. Generate sub-databases by gathering all paths that contain a particular node, as mentioned before; starting from the least frequent item. These sub databases called prefix paths.

2. Convert the prefix path into conditional FP-tree, which have the same structure of FP-tree except that it is used to find frequent itemsets which are ending with a particular suffix.

3. Discover frequent itemsets ending with a particular suffix from the conditional FP-tree. Next, move on to the next sub-dataset to find frequent itemsets ending with another suffix and so on.

The pseudo-code of FP-Growth algorithm shown in algorithm 3. [12].

3.4 Summarizing Itemsets

Frequent itemsets mining is usually deal with vast amounts of data. One of the main challenges of frequent itemsets mining from a large dataset is the fact that it often generates a vast number of frequent itemsets. In another word, the number of generated frequent itemsets undesirably large and contains redundant itemsets, especially in dense databases when the minimum support threshold is set low. It is because if an itemset is frequent, all of its subsets are frequent as well. The compact representation of frequent itemsets

Algorithm 3: FP-GROWTH Algorithm

```
initial call: $R \longleftarrow FP-tree(D), P \longleftarrow \phi, F \longleftarrow \phi$
FP-GROWTH $(R, P, F, minsup)$
Remove infrequent items from R;
if Is Path R then
    Insert subset of $R$ into $F$ foreach $Y \subseteq R$ do
        $X \longleftarrow P \cup Y$
        $sup(X) = min_{x \in Y}\{cnt(x)\}$
        $F \longleftarrow F \cup \{\langle X, sup(X) \rangle\}$
else
    // process projected FP-trees for each frequent item $i$
    foreach $i \in R$ in increasing order of $sup(i)$ do
        $X = P \cup \{i\}$
        $sup(X) = sup(i)$ sum of $cnt(i)$ for all nodes labeled $i$
        $F \longleftarrow F \cup \{\langle X, sup(X) \rangle\}$
        $R_X \longleftarrow \phi$ // projected FP-tree for $X$
        foreach $path \in PATH\ FROM\ ROOT(i)$ do
            $cnt(i) \longleftarrow$ count of $i$ in path
            Insert path, excluding $i$ into FP-tree $R_X$ with count $cnt(i)$
        if $R_X \neq \phi$ then
            FP-GROWTH $(R_X, P, F, minsup)$
```

considered as a meaningful solution to overcome this problem, in which it gives a summary of the set of itemsets. The compact representations can reduce computational and memory storage demands. Also, it can make an easier way to analyze the discovered itemsets [1,27].

In this section two types of compact representation of frequent itemsets maximal and closed frequent itemsets are discussed, and a brief review of its algorithms are discussed as well.

3.4.1 Maximal frequent itemsets

Suppose D is the transaction database, I is the set of all items in D and F is the set of all frequent itemsets that given as follows:

$$F = \{X | X \in I \quad and \quad sup(X) \geq \sigma\} \tag{3.7}$$

***Table 3.2** – Frequent Itemsets with minsup = 3*

sup	Itemsets
6	B
5	E, BE
4	A, C, D, AB, AE, BC, BD, ABE
3	AD, CE, DE, ABD, ADE, BCE, BDE, ABDE

Definition 7. *a frequent itemset $X \in F$ is called maximal if it has no frequent superset. Let M be the set of all maximal frequent itemsets, which is given as follows:*

$$M = \{X | X \in F \quad and \quad \nexists Y \supset X \quad such \quad that \quad Y \in F\} \tag{3.8}$$

The set M can determine any set X whether it is frequent or not, since there is no frequent superset of it and all of its subsets are frequent. If X is a maximal frequent itemset, and $Y \subseteq X$ then Y must be frequent and $sup(Y) \geqslant sup(X)$. Otherwise, Y cannot be frequent. As a conclusion the set of all maximal frequent itemsets, M considered as a compact representation of the set of all frequent itemsets F [27].

To explain this concept, consider the dataset given in Table 3.1 and minimum support equal to 3, the frequent itemsets in Table 3.2 have obtained.

Form table 3.2 we can notice that there are 19 generated frequent itemsets in this example that have minimum support equal or greater than 3, out of this 19 frequent itemsets there are only two maximal frequent itemsets $ABDE, BCE$ as shown in figure 3.1. Therefore, all other frequent itemsets are subsets of this two maximal itemsets. There is a border divided the search space into two groups frequent and infrequent groups. The itemsets stay near the border in above group $\{B, D, E\}$, $\{B, C, E\}$, and $\{A, B, D, E\}$ as in figure 3.1 are considered to be maximal frequent itemsets since their supersets are infrequent. The itemset $\{B, D, E\}$ is non-maximal since one of its supersets, $\{A, B, D, E\}$, is frequent. All the remain frequent itemsets must be a subset of one of the maximal itemsets. Notice that, maximal frequent itemsets do not contain the support information of their subsets.

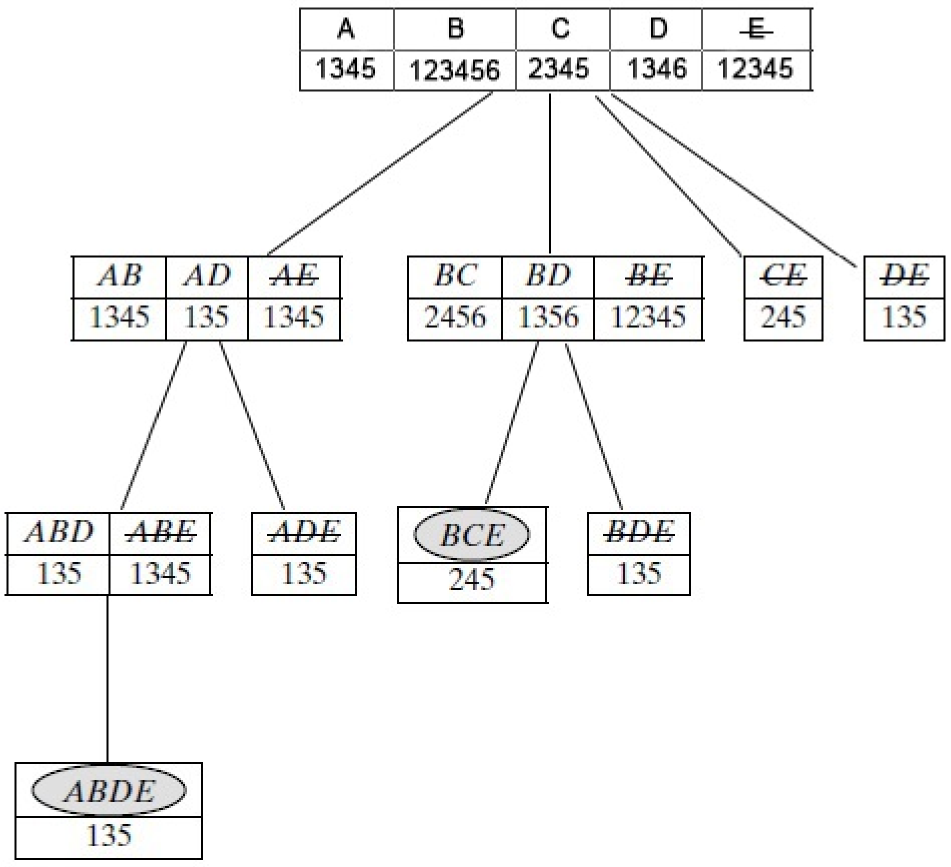

Figure 3.1: *Maximal Frequent Itemsets*

3.4.2 Closed frequent itemsets

Suppose D is the transaction database, I is the set of all items in D and F is the set of all frequent itemsets that given as:

$$F = \{X|X \in I \quad and \quad sup(X) \geq \sigma\} \tag{3.9}$$

Definition 8. *Closed Itemset, An itemset $X \in F$ is closed if it does not have any frequent supersets has exactly the same support count as X. That is, $sup(X) > sup(Y)$, for all $Y \supset X$.*

In another word, we can say; X is not closed if at least one of its frequent supersets has the same support count as X.

Definition 9. *Closed Frequent Itemset, An itemset $X \in F$ is Closed Frequent Itemset if it* *is closed and its support is greater or equal to minimum support threshold,* $sup(X) \geqslant \sigma$.

Let C be the set of all closed frequent itemsets, which is given as follows:

$$C = \{X | X \in F \quad and \quad \nexists Y \supset X \quad such \quad that \quad sup(Y) = sup(X)\} \tag{3.10}$$

By definition, none of the maximal frequent itemsets have frequent supersets. Thus all maximal frequent itemsets are closed. However, the main difference between maximal and closed frequent itemsets that, maximal itemsets loses information about the support count of the underlying itemsets. On the other hand, mining closed frequent itemsets does not lose any information about the support count [1].

To explain the concept of closed itemsets, for the same data from Table 3.1 and minimum support equal to 3. Figure 3.2 shows the closed itemsets.

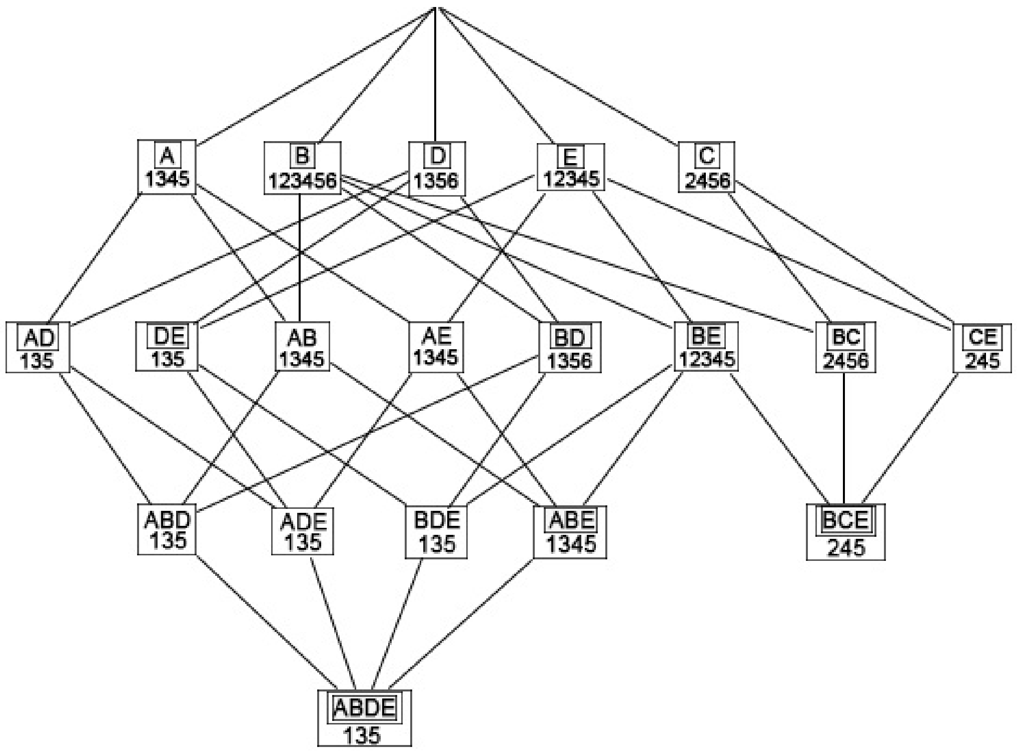

Figure 3.2: *Closed Frequent Itemsets*

The fellowing aquation and figure 3.3 illistrate the relationship between all the sets frequent, closed, and maximal frequent itemsets.

$$M \subseteq C \subseteq F \tag{3.11}$$

The advantage of using closed frequent itemsets can be illustrated by the dataset shown

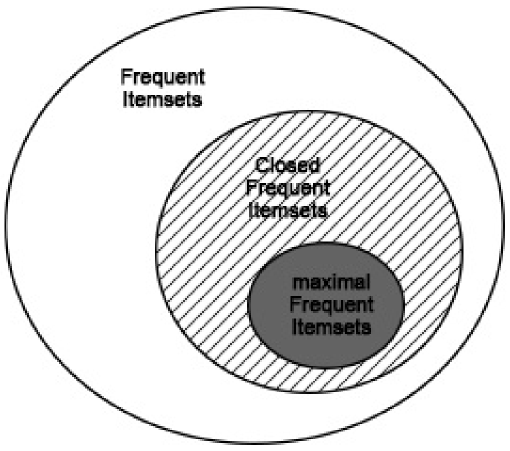

Figure 3.3: *The Relationship among Frequent, Maximal Frequent, Closed Frequent Itemsets*

in table 3.3 which contain ten transactions and fiften items. As shown in the table, the dataset can be divided into three groups. Group A which contain items a_1 through a_5; group B which contain items b_1 through b_5; group C which contain items c_1 through c_5. Notice that, transactions contain items which in the same group and there is no transaction contains items from different groups. To generate frequent itemsets as we montiond in section 2 search space grow exponentially, then in this example there are $3 \times (2^5 - 1)$ potential frequent itemset. However, there are only three closed frequent itemsets in the data $\{a_1, a_2, a_3, a_4, a_5\}$, $\{b_1, b_2, b_3, b_4, b_5\}$, $\{c_1, c_2, c_3, c_4, c_5\}$. It is sufficent to represent the entire set of frequent itemsets by closed frequent itemsets with this kinds of datasets [21].

Table 3.3 *– A Transaction Dataset for Mining Closed Itemsets*

TID	a_1	a_2	a_3	a_4	a_5	b_1	b_2	b_3	b_4	b_5	c_1	c_2	c_3	c_4	c_5
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
5	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
6	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
8	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
9	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
10	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

3.4.3 Closed frequent itemsets algorithms

The literature offers several algorithms for mining closed frequent itemsets. In general, these algorithms can be classified into three classes according to data representation and traversal of search space. A-Close algorithm which is Apriori-like algorithm [17], the pattern growth algorithms which represented by CLOSET algorithm [18, 22], and the vertical format based algorithms which represented by CHARM algorithm [25, 26].

In this section a survey of the CHARM algorithm which will be implemented in the thesis.

CHARM algorithm

CHARM [25, 26] is the characteristic algorithm of the vertical format based class; it implements a bottom-up depth-first search strategy on the data structure. Further, CHARM employed new ideas to improve the performance [25, 26], those ideas are:

- CHARM explores both the item space and the transaction space together over a novel data structure called $itemset - tidset$ tree in short IT-tree search space to improve the efficiency of counting frequent closed patterns. In contrast, most of FPM algorithms exploit only the itemset lattice. That allows the algorithm to use a novel search method that skips many levels of the IT-tree to determine frequent closed itemsets more quickly.

- CHARM uses two pruning strategies, in addition to use the monotonic approach to prune candidate based on subset infrequency, it also prunes branches by using a fast hash-based approach to eliminate non-closed itemsets discovered during the closure checking process.

- Also, CHARM employs a novel vertical data representation called $diffsets$ [24] for fast frequency computation. *diffsets* keep track of differences in the *tids* of a candidate itemset from its prefix pattern. Thus, *diffsets* can quickly compute the support of itemsets.

Finally; the main technique used in CHARM is a union of two itemsets and an intersection of their $tidsets$.

CHARM properties for IT-pairs

Let $f : P(I) \longrightarrow N$ as a one-to-one mapping from itemsets to integer. For any two itemsets X_i and X_j, we say $X_i \leqslant X_j$ if and only if $f(X_i) \leqslant f(X_j)$. f define as a total order over the set of all items. For example, if f sorts itemsets in increasing order of their support, then $X_i < X_j$ if support X_i less than support X_j.

Let $X_i \times t(X_i)$ and $X_j \times t(X_j)$ be any two members in the IT-tree, and $X_i \leqslant_f X_j$, where f is a total order (i.e. lexicographic or support based) the main computations in CHARM relies on the following four properties:

1. If $t(X_i) = t(X_j)$, then $t(X_i \cup X_j) = t(X_i) \cup t(X_j) = t(X_i) = t(X_j)$
   This property implies that we can replace every occurrence of X_i with $X_i \cup X_j$, and we can remove the element X_j from further consideration, since its closure is identical to the closure of $X_i \cup X_j$.

2. If $t(X_i) \subset t(X_j)$, then $t(X_i \cup X_j) = t(X_i) \cap t(X_j) = t(X_i) \neq t(X_j)$
   In this case, we can replace every occurrence of X_i with $X_i \cup X_j$, since they have

identical closure. But since $t(X_i) \neq t(X_j)$ we cannot remove X_j from class $[P]$. It generates a different closure.

3. If $t(X_i) \supset t(X_j)$, then $t(X_i \cup X_j) = t(X_i) \cap t(X_j) = t(X_i) \neq t(X_j)$

   Here, it is similar to property 2 above.

4. If $t(X_i) \neq t(X_j)$, then $t(X_i \cup X_j) = t(X_i) \cap t(X_j) \neq t(X_i) \neq t(X_j)$

   In this case, both X_i and X_j lead to a different closure, so no element can be removed.

Algorithm 4: The CHARM Algorithm

```
initial call: $C \longleftarrow \phi, P \longleftarrow \{\langle x, t(x)\rangle | x \in I, sup(x) \geqslant minsup\}$
CHARM $(P, minsup, C)$
Sort $P$ in increasing order of support;
initialization;
foreach $\langle x_i, t(x_i)\rangle \in P$ do
    $P_i \longleftarrow \phi$
    foreach $\langle x_j, t(x_j)\rangle \in P \quad with \quad j > i$ do
        $X_{ij} = X_i \cup X_j$
        $t(X_{ij}) = t(X_i) \cap t(X_j)$
        if $sup(X_{ij}) \geqslant minsup$ then
            if $t(X_i) = t(X_j)$ then
                Replace $X_i$ with $X_{ij}$ in $P$ and $P_i$
                Remove $\langle X_j, t(X_j)\rangle$ from $P$
            else
                if $t(X_i) \subset t(X_j)$ then
                    Replace $X_i$ with $X_{ij}$ in $P$ and $P_i$
                else
                    $P_i \longleftarrow P_i \cup \{\langle X_{ij}, t(X_{ij})\rangle\}$
    if $P_i \neq \phi$ then
        CHARM $(P_i, minsup, C)$
    if $\nexists Z \in C \; such \; that \; X_i \in Z \quad and \quad t(X_i) = t(Z)$ then
        $C = C \cup X_i$
```

Algorithm design

The pseudo-code of CHARM algorithm is shown in algorithm 4. It takes as input the set of all frequent single items along with their *tidsets*, $P = \{\langle X_i, t(X_i)\rangle, X_i \in I \quad and \quad sup(X_i) \geqslant \sigma\}$. Also, first the set of all closed itemsets, C is empty. The algorithm in the beginning arranges the frequent 1-itemsets in increasing order based on support count. Then the main computation to generate frequent itemsets, for each IT-pair set $\{\langle X_i, t(X_i)\rangle\}$ the algorithm tries to extend it with all other IT-pairs $\{\langle X_j, t(X_j)\rangle\}$ in the sorted order, and apply CHARM properties to eliminate infrequent branches. First the algorithm makes sure that $X_{ij} = X_i \cup X_j$ is frequent, by checking the support count of X_{ij}, which is given as the intersection of the transaction for both items $t(X_{ij}) = t(X_i) \cap t(X_j)$. If X_{ij} is frequent, then eliminating process starts by checking properties 1 and 2 (lines 8 and 12). Note that whenever we replace X_i with $X_{ij} = X_i \cup X_j$, we make sure to do in the current set P, as well as the new set P_i . Only when property 3 and 4 holds, the algorithm adds the new extension X_{ij} to the set P_i (line 14). If the set P_i is not empty, then a recursive call to CHARM will be made. Finally, if X_i is not a subset of any closed set in the set of closed itemsets, C, then the algorithm safely adds it to the set of closed itemsets, C (line 18).

Example 3.4.1. *To show how CHARM works, assume we have the database consist of six transactions and five items as shown in table 3.1 and assume that minimum support set equal to 3. First of all, the items should be sorted in increasing order of support. We will use the pseudo-code to explain the computation.*

At line 1, we start the root class as $[\phi] = \{A \times 1345, C \times 2456, D \times 1356, E \times 12345, B \times 123456\}$. At line 4, we first start with node $A \times 1345$, set $X = A$. At line 6 it will be combined with the remain items, AC is infrequent then it eliminated, since $t(A) \neq t(D)$ then property 4 applies, and then we insert AD in class $[A]$, then we found that $t(A) \subset t(E)$ then property 2 applies, and then replace all occurrence of A with AE and also replace class $[A]$ with $[AE]$. Then we make a recursive call to CHARM-EXTEND with class $[AE]$. Also, $t(A) \subset t(B)$ then, prepert 2 applied, replace $[AE]$ with $[AEB]$ and the obtain item is $ADEB$. The CHARM algorithm works recursively with all items and the obtained result shown in figure 3.2.

4. ALGORITHM IMPLEMENTATION

In the previous chapter, the concept of frequent pattern mining and various algorithms for discovering all frequent itemsets it has been presented and discussed, and also CHARM algorithm used for discovering closed frequent itemsets it has been presented in more details. From the perspective relating to the process of mathematical and computer calculation, it is known that mining frequent itemset is a difficult task and it costs memory and time, especially in dense datasets.

The objective of this thesis is to discover a significant way to determine which books are taken together in Anadolu University Open Education System. The data under study was obtained from 2015-2016 academic year, and it contains dense data about books which were taken by students in that year. The Anadolu university Open Education system is structured in four levels of education, and each level has two semesters. On the beginning of each education level the books for the current academic year are given.

In this chapter, the software programs developed for the implementation of the CHARM algorithm will be discussed and the reason behind choosing the closed frequent itemsets. The algorithm was encoded by using R language, Rstudio software.

4.1 The Reason of Using CFI and CHARM Algorithm

The use of algorithms to discover all frequent itemsets gives a massive amount of frequent itemsets especially with big datasets when minimum support threshold set low. Furthermore, it generates many of redundant and duplicated itemsets in which most of them are subsets of another itemset. On the other hand, maximal frequent itemsets which overcome the issue of generating redundant itemsets but its disadvantage in our case study that it gives very compressed information about the dataset.

The fact that makes mining closed frequent itemsets useful in our study is that CFI performs well in categorical datasets, i.e., when the dataset classified into more than one

category and items in every category dissimilar with the items from other categories, as shown in chapter 3 table 3.3. Since the data under study contains categories and the items inside every category are not similar to others from other categories as shown in the illustration from chapter 3, mining closed itemsets performs better.
The second stage is to choose an appropriate algorithm to utilize in our study. Since all algorithms give almost the same result, the difference is just in the implementation time, and after research and study CHARM algorithm has been chosen. The reason for choosing CHARM is that it has achieved a significant improvement for mining closed itemsets [25, 26]. Furthermore, the novel approach to traverse search space. Another reason it was that CHARM adopts a vertical format which facilitates support counting process.

4.2 Implementation Details

CHARM algorithm presented in [25, 26] and discussed in details in chapter 3 will be encoded using R statistical programming language. The implementation divided into three phases as follows:

- Eliminate infrequent items
- Run CHARM algorithm
- Eliminate duplicates and subsets from the obtained result

The first step, eliminate infrequent items to remove the items with a frequency smaller than the specified minimum support. Since CHARM algorithm works with the vertical data layout, then a function was encoded to eliminate the vertical form of the dataset in which it calculates row sums for each item and compare it with the specified minimum support and remove items with row sum smaller than minimum support. The obtained dataset from this step is the input dataset for the next step.

```
Charm <- function (p,c,MIN_SUP){
    SortBySupportCount(p)
    len4p <- GetLength(p)
    for(idx in 1:len4p){
          q <- GenerateFrequentTidSet()
          for(jdx in (idx+1):len4p){
              xij <- MergeTidSets(p[[idx]],p[[jdx]])
              if(GetSupport(xij)>=MIN_SUP){
                 if( IsSameTidSets(p,idx,jdx) ){
                        ReplaceTidSetBy(p,idx,xij)
                        ReplaceTidSetBy(q,idx,xij)
                        RemoveTidSet(p,jdx)
                   }else{
                   if( IsSuperSet(p[[idx]],p[[jdx]]) ){
                                ReplaceTidSetBy(p,idx,xij)
                                ReplaceTidSetBy(q,idx,xij)
                        }else{
                                Add2CFI(q,xij)
                 }
              }
          }
       }
          if( !IsEmpty(q) ){
             Charm(q,c,MIN_SUP)
          }
          if( !IsSuperSetExists(c,p[[idx]]) ){
             Add2CFI(m,p[[idx]])
     }
   }
}
```

Figure 4.1: *R source code of CHARM algorithm*

The most important phase of the implementation is to run CHARM algorithm in order to discover closed frequent itemsets. In this study is to discover the books that were taken together by students in order to make packets to facilitate the storage and distribution processes.

The R source code of CHARM algorithm as given in [15] shown in figure 4.1 which was used as a guide to encode the algorithm.

As shown in figure 4.1 the CHARM algorithm has three variables as input. The first one is the input dataset which resulted from the previous step, the second the closed itemsets variable which is a matrix with zero row and the same number of columns in the input dataset, and the third variable is the specified minimum support value.

In order to discover frequent itemsets, the first step in CHARM algorithm is to sort the input dataset in a total order. The best performance of CHARM when the dataset is sorted according to the support count in ascending order. Support count for each item in the input dataset is calculated as the row sum from the vertical layout.

After sorting the input dataset in ascending order, two loops traverse the input dataset. The main procedure of examining itemsets and discover closed itemsets happens inside these loops. Each item X_i generates a prefix class q that is the prefix class which stores all supersets of X_i. Furthermore, each X_i will be combined with the remain items in the input dataset.

The next stage is to check if the combination Xij has support count greater or equal to the specified minimum support. If it confirmed the condition, then the basic properties of CHARM will be examined to find which property is fitting the itemset Xij. The generated itemset will be stored in the prefix class q. After this step if the prefix class q is not empty then the function will be returned with q as input dataset, this time to discover if there are itemsets inside the prefix class for each item.

Finally, each prefix class will be added separately to the closed itemsets matrix. For each itemset before is inserted in the closed itemset matrix it should be checked if the itemset is subsumed by another itemset from the closed itemset matrix.

The process of checking to find if an itemset is subsumed by another itemset or not. First, it examines the support count of the itemset and if there is an itemset inside the matrix matching it. If the itemset confirms the condition, then the algorithm examines the items inside the itemsets. After, if the itemset from the matrix is a superset of an inserted itemset, then it will be eliminated because its superset already exists in the closed itemsets matrix. Otherwise, it will be added to the closed itemsets matrix.

The process continues on this way until all closed itemsets are discovered. The obtained result will be given as a matrix form.

The last stage, eliminate duplicates and subsets from the obtained results it consists in putting the result in an appropriate form and remove duplicates and subsets from the obtained result. The itemsets and its frequency are important in the study. Moreover, the obtained result in a sparse matrix form (which will be explained in the next chapter), then a way to facilitate the understanding of result is to put it in a data frame form which consists of three variables, itemsets, frequency, and the length of itemsets. The itemsets represented by row names in the sparse matrix while frequency as mentioned earlier by row sum and the length of itemsets by calculating how many books contains each itemset.

Itemsets with length 1 and 2 are not necessary for the study, because there are no packets with one book and with two books is not matching the purpose of the study to facilitate the storage and distribution processes of books then it will be removed from the results.

The closed frequent itemsets are the itemsets which its superset does not have the same support count number. There are subsets of itemsets which have different support count, but logically, the study aims to find big packets, and it is more important than small packets, and then, that subset will be eliminated as a duplicate.

A function was encoded to remove itemsets which have supersets regardless of its support count.

5. CASE STUDY

In the previous chapter, the implementation details of CHARM algorithm and the code description have been described. In this chapter, first the data under study is explained, then the dataset transformed into a suitable form to fit the encoded algorithm which was explained in the previous chapter. Finally, the result obtained from the algorithm is discussed.

5.1 Description of Dataset

The data under study obtained from Anadolu University Open Education System contains data of the books which were taken by students in 2015-2016 academic year, that consists of four variables as follows: registration number, books code, Semester, period code.

Preliminary data contains 6276073 row and four columns, table 5.1 shows the description of each variable in the data.

The purpose of the study (frequent itemsets mining), is to give information about the books that were taken together by students in that academic year 2015-2016 to make it in packets that it could facilitate storage and distribution processes of books.

***Table 5.1** – Description of Columns of Dataets*

registration number	Each number represents one student
books code	contains the code name of each book in Open Education System
Semester	Consists of four levels 1,3,5,7 and represent the semester in which the book was taken
period code	Represents the period of first or second semester from the academic year.

5.1.1 Input dataset preparation

As mentioned earlier the dataset consists of four columns and 6276073 row. In frequent itemset mining, the dataset should be in a transactional format as shown in chapter 3 table 3.1a. Since the dataset is not in the required form, then some pre-processes should be made to convert the data into a suitable format that fits the algorithm which will be applied in this study.

The following steps are done to prepare the data:

- Convert the dataset into basket format data
- Read the basket format data as transactional dataset
- Convert the transactional dataset into the vertical layout format

Before converting process starts, two things are important to be mentioned. Fist, in this study the target data is only the data from three, five, and seven semesters, the data from semester one is eliminated because the books in packages are unchangeable. Second, since the period code for whole data is equal to one, then it will be eliminated because it is not necessary to keep it.

The first step is to convert the data from the original format into a basket format $\langle TID, \{item_1, item_2, \ldots, item_n\}\rangle$ as shown in table 5.2 in which the first column represent registration number (student), second column semester, and the third column contains books code. The number of transactions obtained is equal to 823218, and each transaction represents a registration number in the dataset.

After converting the original data into basket format which was done by using “plyr” package [23] in R program, the result was saved with “csv” extension to be fit on R program.

The second step is to convert the basket format dataset into transactional dataset by using the “arules” package [10] from R. In this step the obtained data will be given as a

Table 5.2 *– Sample of Dataset in Basket Format*

KAYITNO	YARIYIL	BOOKSCODE
10000177	3	İLT205U, FEL203U, KYT201U, TAR201U, TÜR201U, ÖMB201U, PSİ201U, HUK215U
10000255	3	PSİ201U
10000543	3	HUK215U, İLT205U, FEL203U, KYT201U, ÖMB201U, TAR201U, PSİ201U
10000861	3	İLT205U, KYT201U
10000878	3	HUK215U, ÖMB201U, TAR201U, FEL203U
10001325	3	İLT205U, HUK215U, FEL203U
1000138	3	HUK211U, TAR201U

sparse matrix.

The sparse matrix is an efficient way to store the big data which consists of 0 and 1 when the majority of items in the matrix equal to zero and with just a few ones scattered throughout.

The reason of using the sparse matrix in R and statistical software that it does not record the zeros in the matrix where there is no item while it keeps the places empty and therefore it will save a lot of memory and gives structure to transactional datasets.

The third and last step is to convert the obtained sparse matrix dataset from the horizontal format into the vertical format which is the final form for the acceptable dataset from the CHARM algorithm.

The final dataset contains data of three semesters in which consists of 434 items or row, and 823218 transactions or columns.

Finally, since each semester is independent of the others, then the whole dataset can be divided into separated sub-dataset and for each sub-dataset to be treated separately. From this point, the result for each semester will be presented separately as well.

This step (split the main dataset into sub-datasets for each semester) has been taken to facilitate data analysis and algorithm running processes.

```
transactions as itemMatrix in sparse format with
823217 rows (elements/itemsets/transactions) and
434 columns (items) and a density of 0.01166929

most frequent items:
İŞL293U TAR201U TÜR201U MLY201U MLY301U (Other)
159668  127963  120211  114640  104971 3541706

element (itemset/transaction) length distribution:
sizes
1      2      3      4      5      6      7      8      9
102736  75659  67349  64317  45272 265151  68318  49437  84978

Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
1.000   3.000   6.000   5.064   6.000   9.000
```

Figure 5.1: *Summary of The Whole Dataset*

5.2 Summary of the Dataset

In this section, descriptive properties of data under study are shown before running the CHARM algorithm. In this step “arules” package [10] is used over the transactional form. This gives a brief image that it helps us to understand data under study.

Figure 5.1 presents a summary of transactional dataset: The first block information shows that the whole dataset consists of 823217 transactions and 434 columns which represent different books (items). Another important scale is the density number which refers to the total number of non-empty cells in the sparse matrix, in another word, the number of books that were taken by students. The total number of cells in the matrix can be computed by multiplying row with columns, in this case, $823217 \times 434 = 357276178$, and by multiplying the result number with density gives the total number of non-empty cells.

The second block gives information about the most frequent items and their frequencies (the support count). For instance, $\imath SL293U$ appeared 159668 times in the transactions $159668/823217 = 0.1939562$ percentage of the transactions which refer to the relative support of the item.

```
transactions as itemMatrix in sparse format with
307307 rows (elements/itemsets/transactions) and
237 columns (items) and a density of 0.01934966

most frequent items:
TAR201U TÜR201U ARA2001 İLH2002 İLH2001 (Other)
127963   92716   43099   42454   41496 1061542

element (itemset/transaction) length distribution:
sizes
1     2     3     4     5     6     7     8     9
46117 38341 30956 29078 21378 67336 26425 47674     2

Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
1.000   2.000   5.000   4.586   6.000   9.000
```

Figure 5.2: *Summary of Dataset of Semester 3*

The last block in the figure shows the transaction length distribution; here we can see that transactions contain minimum one item and maximum nine items; the median of transactions contains six items.

As a second step, a general presentation of each category separately.

As mentioned earlier the whole dataset was divided into three subsets according to the semester. The dataset of semester three as shown in figure 5.2 contains 307307 transaction or row and 237 items or books with a density of dataset equal to 0.01934966.

The second block of figure 5.2 gives information about the most frequent items and their support count. For instance, $TAR201U$ is the most appeared item in the dataset that is shown up 127963 times and as second place item $TUR201U$ is shown up 92716 times.

The transaction length distribution as shown in the last block of the figure shows that transactions ranging from one up to eight and only two transactions in the dataset contains nine items which are a trivial number. The density matrix shown in figure 5.3 visualizes a sample of 100 transactions represents the binary sparse matrix of dataset of semester three. The dark dots in the plot represent the ones in the matrix where the books were taken for each transaction. The plot shows that there are some items in the matrix which appear so much in the transactions represented by the dense vertical lines. From the other side, the

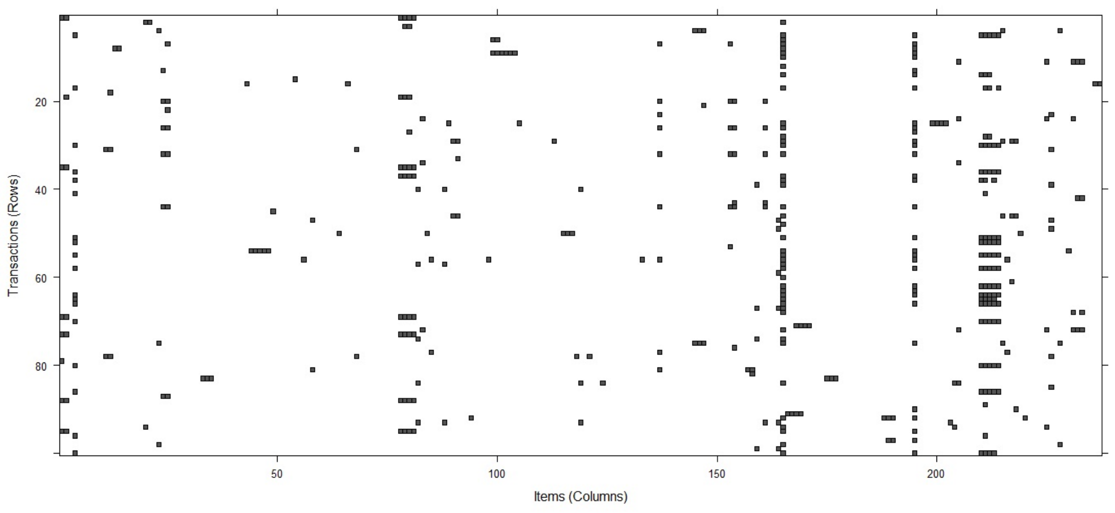

Figure 5.3: *The Density Matrix of Semester 3*

area with white represents few appearing of items which mean items could be eliminated from the dataset. Figure 5.4 shows the most important items that are most frequently present in the dataset. In this figure, only items with a frequency higher than specified minimum support which set as $\sigma = 5000$ have been presented. The X-axis represents the items, and Y-axis represents the absolute support count. Compatible with the summary in figure 5.2 $TAR201U$ is the most frequent item with support count exceeded 120000 and $TUR201U$ with support count around 100000. The figure also displays that the majority of items have support count ranging between 10.000-50.000

Next, it is presented a short description of the dataset of semester five. As shown in figure 5.5 the dataset consists of 274139 transactions and 101 items or books with density consider high compared with that one from semester three that is equal to 0.06441522.

The most frequent items in the second block of figure 5.5 are near to each other, and there is no significant difference between it as in the dataset of semester three. Moreover, transaction length shows that there are 84975 transactions which contain nine items or books and 146358 transactions with six and seven items in each transaction which explain there could be frequent itemsets of length nine, seven and six. The density matrix of semester 5 dataset is more dense as shown in figure 5.6. A sample of 100 transactions

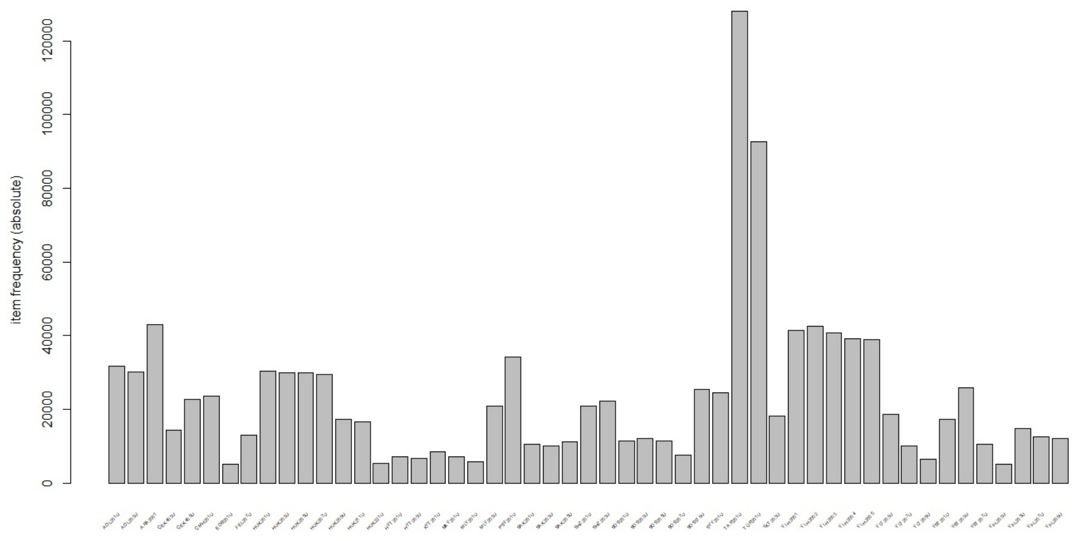

Figure 5.4: *Frequent Items of Semester 3*

```
transactions as itemMatrix in sparse format with
274139 rows (elements/itemsets/transactions) and
101 columns (items) and a density of 0.06441522

most frequent items:
MLY301U ARY101U MUH301U İKT103U HUK303U (Other)
104971   96426   91148   87617   82921 1320448

element (itemset/transaction) length distribution:
sizes
1      2      3      4      5      6      7      8      9
9819   7579  10098   9506   5036 130369  15989    768  84975

Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
1.000   6.000   6.000   6.506   9.000   9.000
```

Figure 5.5: *Summary of Dataset of Semester 5*

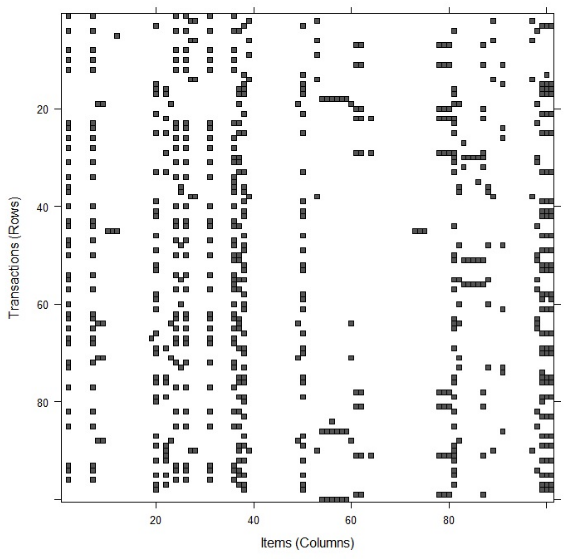

Figure 5.6: *The Density Matrix of Semester 5*

which shows that the X-axis is smaller than dataset of semester 3, but there are more dense vertical lines with dark dots which means frequent items or books with high frequency. Figure 5.7 shows the frequent items which have support count greater or equal to the specified minimum support. Compatible with the density matrix in figure 5.6 there are many items with very high support count, that is almost half of the frequent items with support count exceeded 40000 and the remain frequent items with support count around 10000. Since the average of frequency in semester five is high, then it is expected to have a high number of frequent itemsets in the result.

The last sub-dataset of the study for semester seven which contains 241772 transactions and 110 row as shown in figure 5.8,the density of dataset is equal to 0.03671227 which consider relatively less than that one of semester five but more than that one of semester three.

The transaction length distribution ranging from one to seven, and there are only a few transactions with length more than seven which will be ignored because of it is less than minimum support. The density matrix of semester seven dataset shows some very dense vertical lines as displayed in figure 5.9 which means frequent items with a very high support count. Compatible with the second block in figure 5.8 whereas first and second

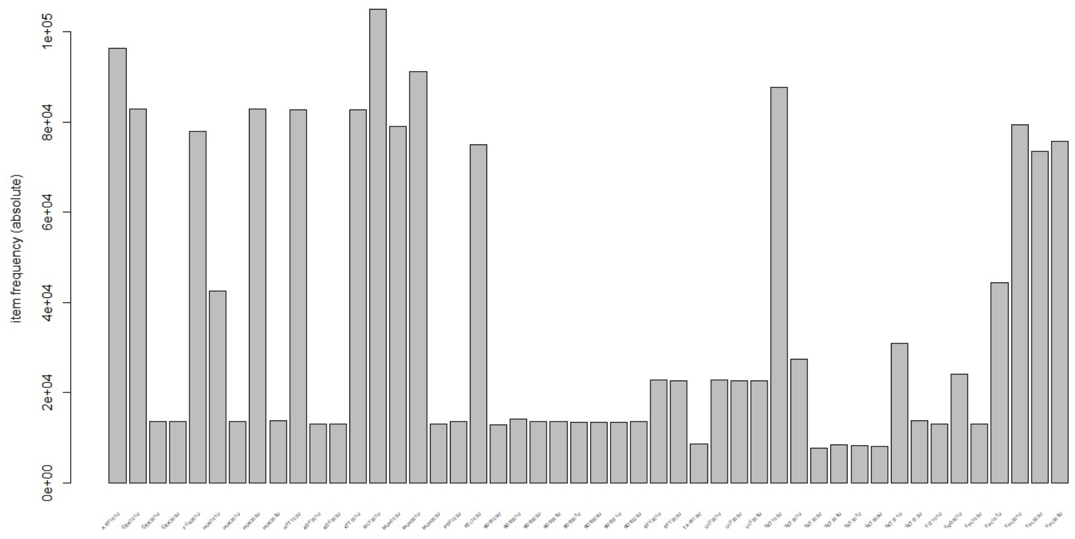

Figure 5.7: *Frequent Items of Semester 5*

```
transactions as itemMatrix in sparse format with
241772 rows (elements/itemsets/transactions) and
110 columns (items) and a density of 0.03671227

most frequent items:
İŞL293U MLY201U İŞL201U İŞL401U İŞL403U (Other)
159668  108796   66614   65064   63516  512702

element (itemset/transaction) length distribution:
sizes
1     2     3     4     5     6     7     8     9
46800 29740 26295 25733 18858 67446 25904   995     1

Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
1.000   2.000   4.000   4.038   6.000   9.000
```

Figure 5.8: *Summary of Dataset of Semester 7*

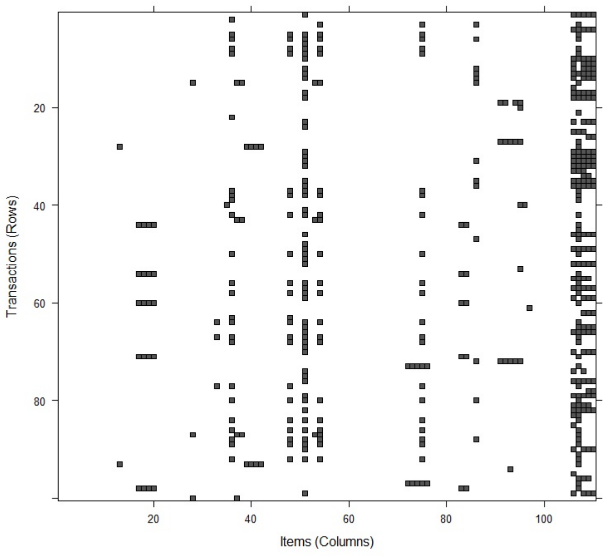

Figure 5.9: *The Density Matrix of Semester 7*

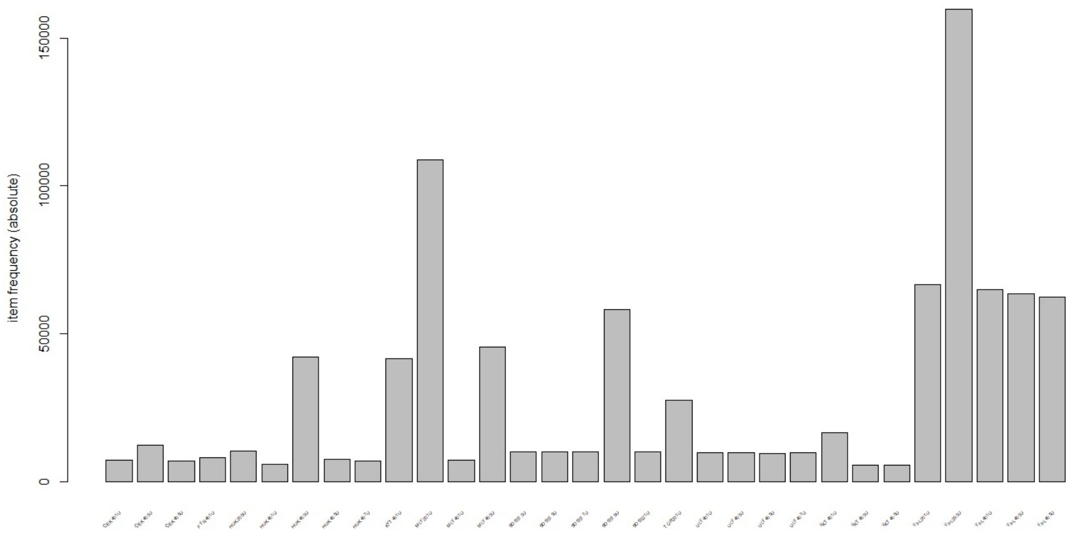

Figure 5.10: *Frequent Items of Semester 7*

items are the most frequent with a very high frequency and then it begins to decline significantly.

The bar chart in figure 5.10 shows the frequent items in semester seven dataset. The figure proves the density matrix shown in figure 5.9 that is only the items with long bars illustrated items with a very high frequency. Therefore, it could appear in the majority of the transactions in the datasets.

Up to now, a summary of the data under study which provided a clear vision on each sub-dataset separately. Next section, the result obtained from the encoded CHARM algorithm will be discussed for each sub-dataset separately as well.

5.3 Result and Discussion

This section presents the result of the dataset obtained from Anadolu University Open Education System after applying the CHARM algorithm. As presented earlier the input data were divided by semester into three groups, further the results will be presented for each group separately.

The purpose of this study was to discover the books that have been taken together and make packets from them. Execution time and storage memory were not taken into account.

It is important to mention that the absolute minimum support threshold was used in the implementation and it set as $minsup = 5000$ for all three groups in the study.

The Result Obtained From Semester Three

The result obtained from semester three with minimum support set equal to 5000 was 571 closed frequent itemsets. After that, duplicates and subsets were eliminated from the result obtained. Then the final result contains 178 itemsets without duplicates. Table 5.3 displays a sample of the result obtained for semester three.

***Table 5.3** – Sample of itemsets generated of semester 3*

	itemsets	frequency	books
287	SHZ201U,TAR201U,SHZ203U,SOS319U,TÜR201U	7449	5
633	İLH2005,TÜR201U,İLH2003,İLH2004	22800	4
422	ÇMH201U,SOS319U,PSİ201U	15420	3
124	SOS201U,FEL207U,SOS203U	8306	3
696	İLH2004,ARA2001,İLH2001,İLH2003,İLH2002	30993	5
446	HUK207U,TÜR201U,ADL201U,ADL203U	5010	4
553	HUK201U,TAR201U,TÜR201U	5469	3
693	İLH2004,TÜR201U,İLH2002,İLH2003	23308	4
20	SAK203U,TÜR201U,SAK201U	5047	3
398	ÇEK405U,TAR201U,ÇMH201U,TÜR201U	7685	4

The result after eliminating insignificant itemsets can be divided into six groups as follows:

- There are 43 itemsets which contain three books.
- 69 itemsets with four books
- 39 itemsets with five books
- 18 itemsets with six books
- 7 itemsets with seven books
- 2 itemsets with eight books

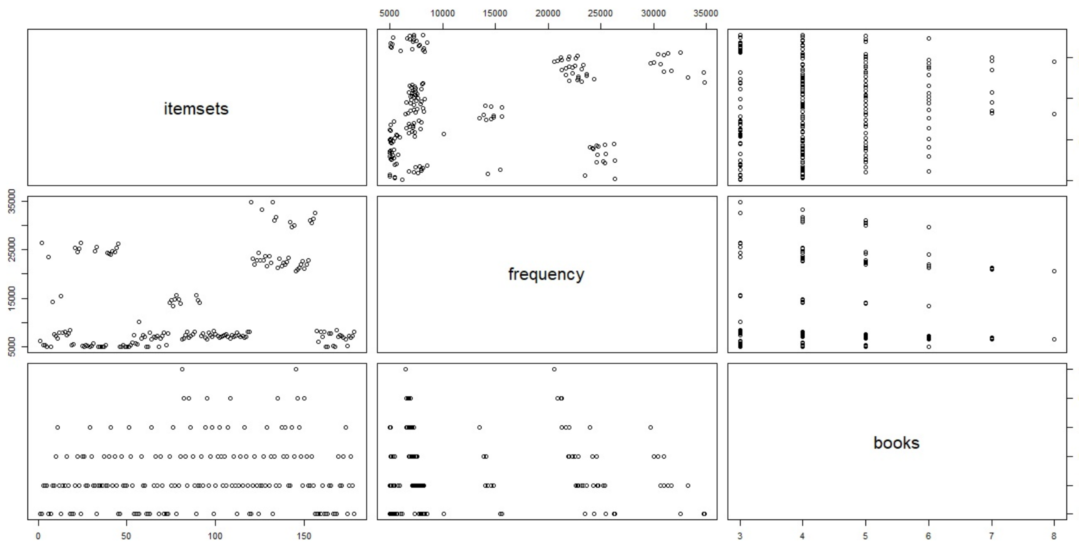

Figure 5.11: *The Matrix Plot of Semester 3*

The symmetric matrix given in figure 5.11 displays a comprehensive overview of the obtained result from semester three. The second rectangle shows the itemsets frequency, that is ranging between 5000 up to 35000, and it shows that the majority of itemsets have support count between 5000 and 10000 and in the second level there are a large number of itemsets that have support count ranging between 20000 and 25000. The rectangle beside it shows that itemsets in this semester consist of 3-8 books and the frequency of packets is shown in the rectangle under it.

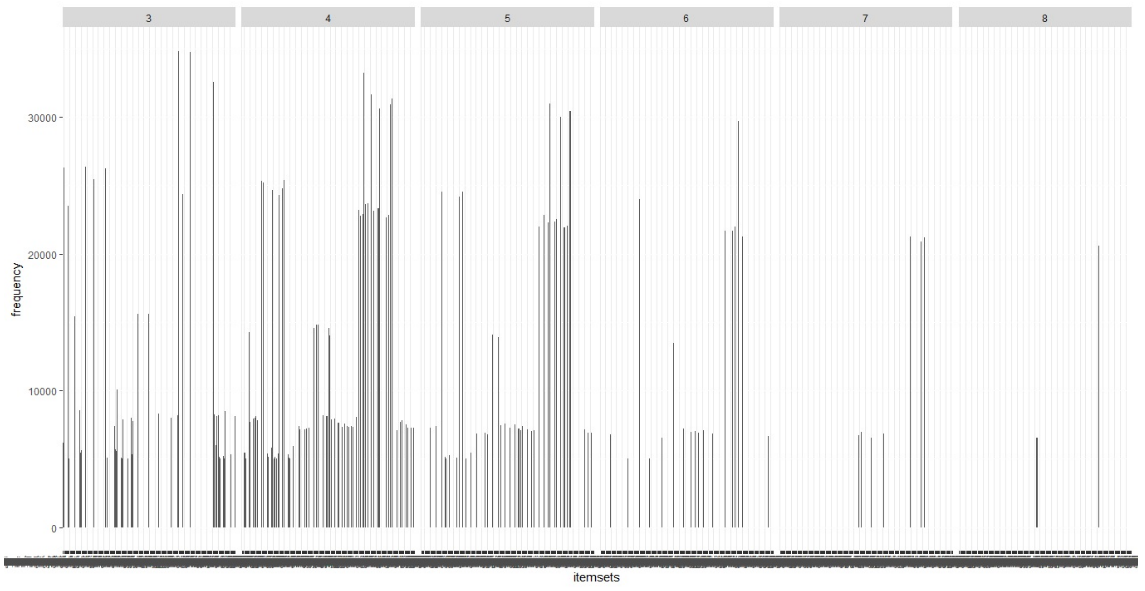

Figure 5.12: *Bar Chart of Frequent Itemsets of Semester 3*

In the figure 5.12, it is shown the itemsets frequency for each packet of semester three. The figure shows that packets of four and five books are very dense. Almost half of the packets contains 4 or 5 books. Furthermore, in all categories, there are packets with high frequency.

The Result Obtained From Semester Five

The result obtained from semester 5 was not accurate due to an error that appeared when dataset of semester five was run. Therefore, we attempt to give an approximate result. The dataset of semester 5 contains 101 items, and after eliminating infrequent items, 48 frequent items remain. These items have frequency greater than or equal to the specified minimum support. First, the dataset was divided into two groups: the first group contains from 1 up to 30 and second group from 31 up to 48, and each group was run separately, and the result obtained from each group merged together. The result obtained from this process contains 74 observations or itemsets as shown in Table 5.4

***Table 5.4** – Sample of itemsets generated of semester 5*

	itemsets	frequency	books
41	SOS311U,SOS303U,SOS307U,SOS305U	10718	4
221	ULİ305U,ULİ301U,ULİ303U	21827	3
160	HİT102U,KYT301U,ARY101U	81090	3
157	HİT102U,ÇEK101U,KYT301U,ARY101U	80574	4
163	KYT301U,ÇEK101U,MLY301U,ARY101U,HUK303U	80224	5
115	PZL103U,MUH103U,MUH301U,FİN201U	29857	4
225	ULİ303U,İKT311U,ULİ301U	22070	3
243	İŞL303U,İŞL305U,İŞL301U	69045	3
63	HUK301U,PSİ103U,SOS323U,ÇEK301U,ÇEK303U	13104	5
195	İKT303U,İKT309U,İKT305U,İKT307U	6700	4

The result after eliminating insignificant itemsets can be divided into four groups as follows:

- 33 itemsets contain three books
- 25 itemsets contain four books
- 12 itemsets contain five books
- Three itemsets contain six books
- Moreover, one itemset contains seven books.

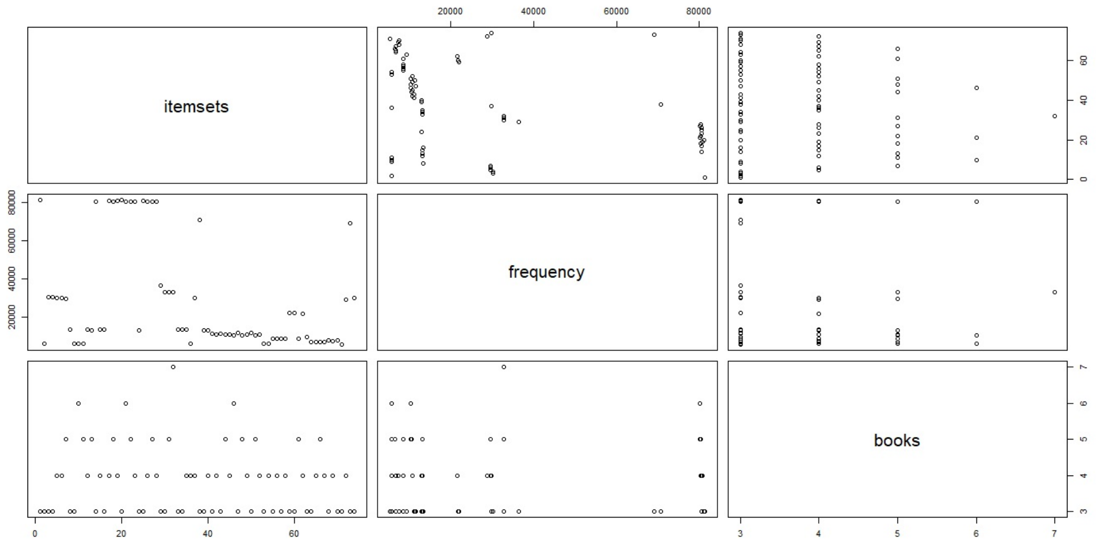

***Figure 5.13:** The Matrix Plot of Semester 5*

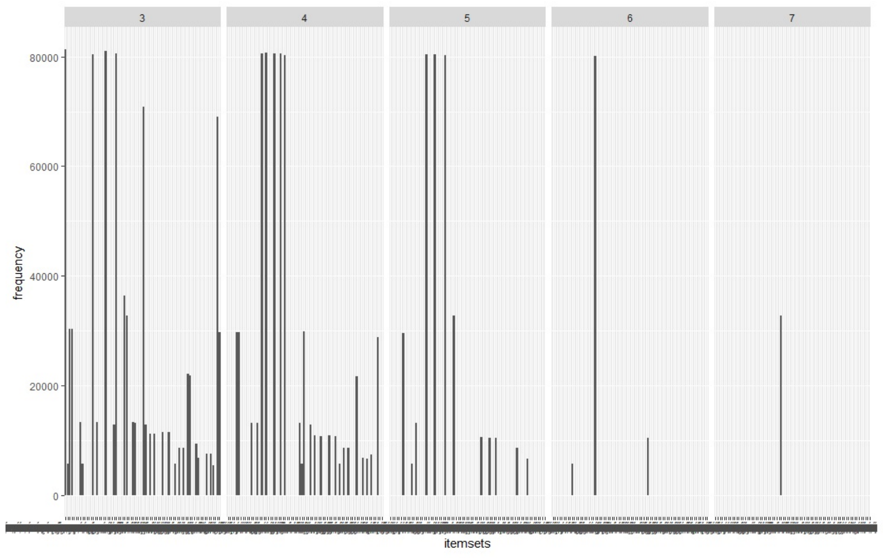

Figure 5.14: Bar Chart of Frequent Itemsets of Semester 5

The symmetric matrix in figure 5.13 shows a comprehensive overview of the obtained result from semester five. The second rectangle shows the itemsets frequency, and it is ranging from 5000 up to 80000, and it shows that the majority of itemsets have support count between 5000 and 20000. The rectangle beside it shows that itemsets in this semester consist of 3-7 books and the frequency of packets is shown in the rectangle under it.

To see the generated itemsets in semester five and their frequencies, figure 5.14 shows the books for each itemset and its support count. It is clear that the majority of generated itemsets are with three and four books.

It is interesting to see that itemset with seven books has a frequency around 30000 as shown in figure 5.14 and by comparing it with the summary of the dataset from figure 5.5 there is only 15989 transactions with seven items. As a conclusion, in this case, the itemset may contain nine items, and because of the error obtained in running this dataset, the full itemset could not be obtained.

The Result Obtained From Semester Seven

The result obtained from semester seven with minimum support equal to 5000 was 204 observations. After that, duplicates and subsets were eliminated from the result obtained.

The final result contains 80 itemsets without duplicates. Table 5.5 shows a sample of the result obtained.

***Table 5.5** – Sample of itemsets generated of semester 7*

	itemsets	frequency	books
60	ULİ405U,ULİ401U,ULİ403U	7313	3
105	SOS317U,SOS321U,SOS315U	8805	3
155	TÜR201U,KYT401U,MLY201U,SOS319U,MLY403U	5135	5
255	MLY403U,MLY201U,İŞL293U	28312	3
34	HUK405U,İŞL293U,FİN401U	5522	3
54	ULİ405U,ULİ407U,ULİ401U,ULİ403U	6581	4
123	SOS313U,SOS319U,SOS315U	8475	3
30	ÇEK401U,İŞL293U,ÇEK403U	5168	3
96	SOS317U,SOS319U,SOS313U,SOS321U,SOS315U	8070	5
270	İŞL405U,İŞL293U,İŞL403U,İŞL401U	37214	4

The result after eliminating insignificant itemsets can be divided into five groups as following:

- 29 itemsets contain three books
- 30 itemsets contain four books
- 15 itemsets contain five books
- Five itemsets contain six books
- One itemset contains seven books

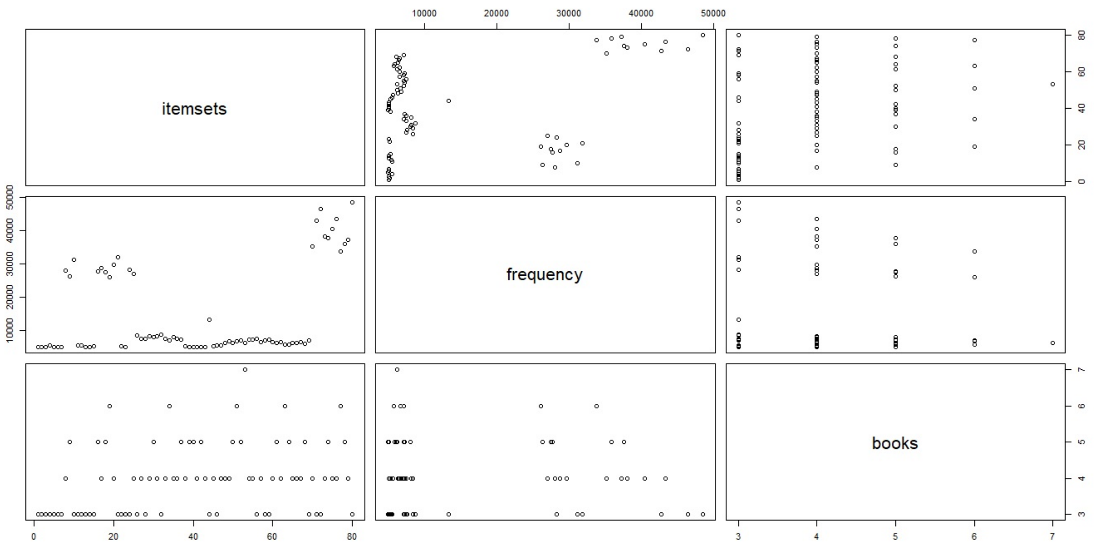

Figure 5.15: *The Matrix Plot of Semester 7*

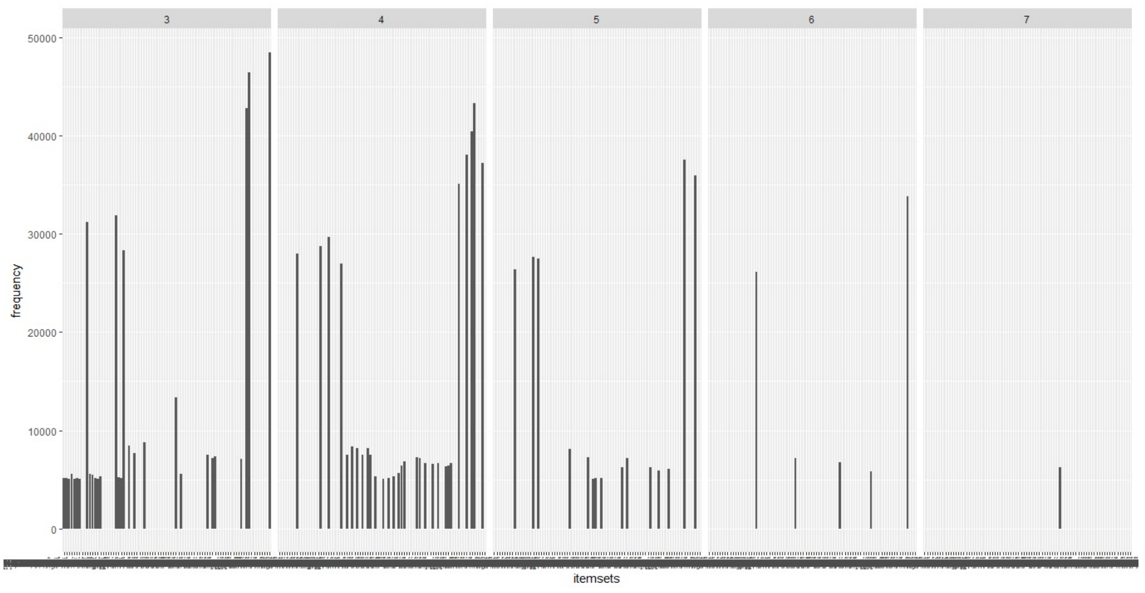

Figure 5.16: *Bar Chart of Frequent Itemsets of Semester 7*

The symmetric matrix in figure 5.15 shows a comprehensive overview of the obtained result from semester seven. The second rectangle shows the itemsets frequency that is ranging from 5000 up to 50000, and it shows that the majority of itemsets have support count between 5000 and 10000. The rectangle beside it shows that itemsets in this category consist of 3-7 books and the frequency of packets is shown in the rectangle under it.

To see the generated itemsets in semester seven and their frequencies, figure 5.16 shows the books for each itemset and its support count. It is clear that itemsets with high frequency distributed between packets with three, four, and five books.

6. CONCLUSION

In this thesis, the primary objective is to implement the frequent pattern mining concept in a data set of Anadolu University Open Education System to discover the most frequent books were taken together by students. An important measure in discovering pattern mining is minimum support, which can be used to identify the importance of patterns or itemsets in term of its frequency in the dataset.

In this thesis, the CHARM algorithm to discover closed frequent itemsets has been implemented. The reason for adopting closed frequent itemset technique and its algorithm CHARM is to avoid generating a large number of duplicates, that is closed frequent itemsets is a compressed representation of the frequent pattern. Moreover, a function to eliminate subsets and duplicates has been performed to reduce the number of discovered itemsets.

Experimental results show that the CHARM algorithm has good performance on datasets of semesters three and seven, and for the dataset of semester five, it could not give an accurate result. Furthermore, a set of frequent itemsets was generated for each semester separately. The result obtained from running CHARM algorithm have matched the summary of datasets obtained by using “arules” package from R program which shows the efficiency of our encoded algorithm.

Finally, the result obtained seem to be beneficial to help decision maker in Anadolu University Open Education System to make decision to package the books in packages to facilitate the distribution process of books.

REFERENCES

[1] Charu C Aggarwal and Jiawei Han. *Frequent pattern mining*. Springer, 2014.

[2] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *Acm sigmod record*, volume 22, pages 207–216. ACM, 1993.

[3] Rakesh Agrawal, Ramakrishnan Srikant, et al. Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499, 1994.

[4] Roberto J Bayardo Jr. Efficiently mining long patterns from databases. *ACM Sigmod Record*, 27(2):85–93, 1998.

[5] Douglas Burdick, Manuel Calimlim, and Johannes Gehrke. Mafia: A maximal frequent itemset algorithm for transactional databases. In *Data Engineering, 2001. Proceedings. 17th International Conference on*, pages 443–452. IEEE, 2001.

[6] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery in databases. *AI Magazine*, 17(3):37–54, 1996.

[7] Paolo Giudici. *Applied data mining: Statistical methods for business and industry*. John Wiley & Sons, 2005.

[8] Bart Goethals. Survey on frequent pattern mining. *Univ. of Helsinki*, 19:840–852, 2003.

[9] Karam Gouda and Mohammed J Zaki. Genmax: An efficient algorithm for mining maximal frequent itemsets. *Data Mining and Knowledge Discovery*, 11(3):223–242, 2005.

[10] Michael Hahsler, Christian Buchta, Bettina Gruen, Kurt Hornik, and Maintainer Michael Hahsler. Package 'arules'. 2018.

[11] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. The Morgan Kaufmann series in data management systems. Elsevier, 2006.

[12] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. In *ACM sigmod record*, volume 29, pages 1–12. ACM, 2000.

[13] David J Hand, Heikki Mannila, and Padhraic Smyth. *Principles of data mining (adaptive computation and machine learning)*. MIT press Cambridge, MA, 2001.

[14] Ruofei He. *Bayesian mixture models for frequent itemset mining*. PhD thesis, University of Manchester, 2012.

[15] Bater Makhabel. *Learning data mining with R*. Packt Publishing Ltd, 2015.

[16] Heikki Mannila, Hannu Toivonen, and A Inkeri Verkamo. E cient algorithms for discovering association rules. In *KDD-94: AAAI workshop on Knowledge Discovery in Databases*, pages 181–192, 1994.

[17] Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal. Discovering frequent closed itemsets for association rules. In *International Conference on Database Theory*, pages 398–416. Springer, 1999.

[18] Jian Pei, Jiawei Han, Runying Mao, et al. Closet: An efficient algorithm for mining frequent closed itemsets. In *ACM SIGMOD workshop on research issues in data mining and knowledge discovery*, volume 4, pages 21–30, 2000.

[19] Xuequn Shang. *SQL based frequent pattern mining*. PhD thesis, Otto-von-Guericke-Universität Magdeburg, Universitätsbibliothek, 2005.

[20] Ramakrishnan Srikant and Rakesh Agrawal. Mining generalized association rules. 1995.

[21] P.N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Pearson Education, Limited, 2014.

[22] Jianyong Wang, Jiawei Han, and Jian Pei. Closet+: Searching for the best strategies for mining frequent closed itemsets. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 236–245. ACM, 2003.

[23] Hadley Wickham and Maintainer Hadley Wickham. Package 'plyr'. *Obtenido de https://cran. rproject. org/web/packages/dplyr/dplyr. pdf*, 2017.

[24] Mohammed J Zaki and Karam Gouda. Fast vertical mining using diffsets. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 326–335. ACM, 2003.

[25] Mohammed J Zaki and Ching-Jui Hsiao. Charm: An efficient algorithm for closed itemset mining. In *Proceedings of the 2002 SIAM international conference on data mining*, pages 457–473. SIAM, 2002.

[26] Mohammed J Zaki, Ching-Jui Hsiao, et al. Charm: An efficient algorithm for closed association rule mining. Technical report, Citeseer, 1999.

[27] Mohammed J Zaki, Wagner Meira Jr, and Wagner Meira. *Data mining and analysis: fundamental concepts and algorithms*. Cambridge University Press, 2014.

[28] Mohammed Javeed Zaki. Scalable algorithms for association mining. *IEEE transactions on knowledge and data engineering*, 12(3):372–390, 2000.

APPENDIX

Table: A.1 *Output File From Semester 3*

itemsets	frequency	books
SAK203U,TÜR201U,SAK201U	5047	3
SAK203U,İST207U,SAK201U	7983	3
SAK201U,TAR201U,TÜR201U	5090	3
SAK201U,TÜR201U,SAK205U	5004	3
SAK201U,İLT203U,İST207U	7891	3
İST207U,ÇEK403U,İLT203U	8110	3
SAK205U,TAR201U,TÜR201U	5352	3
SAK205U,İLT203U,ÇEK403U	7741	3
SOS201U,FEL207U,SOS203U	8306	3
SOS205U,PSİ201U,FEL207U	7975	3
İŞL209U,TAR201U,İŞL205U	5306	3
İŞL209U,İŞL207U,İŞL205U	8107	3
İŞL207U,TAR201U,İŞL205U	5027	3
İŞL207U,TAR201U,HUK211U	5222	3
İŞL207U,İKT203U,İST201U	8518	3
FEL207U,İST203U,PSİ201U	8525	3
ÇEK403U,TÜR201U,İLT203U	5008	3
İŞL205U,TAR201U,İST201U	5031	3
İŞL205U,TAR201U,İKT203U	5120	3
İŞL205U,HUK211U,İKT203U	8193	3
HUK211U,TAR201U,İST201U	5591	3
HUK209U,TAR201U,MLY203U	5715	3
HUK209U,SİY201U,MLY203U	7366	3
İST201U,TAR201U,İKT203U	6013	3
İLT203U,TÜR201U,TAR201U	8274	3

itemsets	frequency	books
SHZ201U,PSİ201U,SHZ203U	15620	3
SHZ203U,ÇMH201U,ÇEK405U	15616	3
ÇMH201U,SOS319U,PSİ201U	15420	3
SİY201U,PSİ201U,İST203U	8160	3
HUK207U,TAR201U,ADL203U	5110	3
HUK207U,HUK203U,HUK205U	26251	3
HUK203U,HUK201U,HUK205U	26337	3
HUK205U,ADL203U,HUK201U	25465	3
ADL203U,HUK201U,ADL201U	26309	3
HUK201U,TAR201U,TÜR201U	5469	3
HUK201U,TÜR201U,ADL201U	5627	3
ADL201U,TÜR201U,TAR201U	6200	3
PSİ201U,TÜR201U,TAR201U	10097	3
İLH2005,İLH2003,İLH2004	32532	3
İLH2003,İLH2001,İLH2002	34737	3
İLH2001,ARA2001,İLH2002	34798	3
İLH2002,TAR201U,ARA2001	24359	3
ARA2001,TÜR201U,TAR201U	23483	3
SAK203U,ÇEK403U,İLT203U,SAK201U	7153	4
SAK203U,İLT203U,İST207U,SAK201U	7270	4
SAK203U,SAK205U,İST207U,SAK201U	7216	4
SAK201U,ÇEK403U,İST207U,İLT203U	7378	4
SAK201U,SAK205U,İLT203U,İST207U	7131	4
İST207U,SAK205U,ÇEK403U,İLT203U	7105	4
SOS201U,İST203U,PSİ201U,SİY201U	7603	4
SOS201U,PSİ201U,FEL207U,SİY201U	7655	4
SOS201U,SİY201U,FEL207U,SOS203U	7312	4
SOS205U,İST203U,PSİ201U,FEL207U	7354	4
SOS205U,SİY201U,FEL207U,PSİ201U	7401	4

itemsets	frequency	books
İŞL209U,HUK211U,İKT203U,İŞL205U	7262	4
İŞL209U,İST201U,İŞL207U,İŞL205U	7264	4
İŞL209U,İKT203U,İŞL207U,İŞL205U	7247	4
SOS203U,İST203U,SİY201U,PSİ201U	7343	4
SOS203U,PSİ201U,FEL207U,SİY201U	7398	4
İŞL207U,İŞL205U,İKT203U,İST201U	7492	4
İŞL207U,HUK211U,İKT203U,İST201U	7849	4
FEL207U,SİY201U,İST203U,PSİ201U	7826	4
İŞL205U,İST201U,HUK211U,İKT203U	7728	4
SHZ201U,TAR201U,SOS319U,TÜR201U	8196	4
SHZ201U,TÜR201U,SHZ203U,SOS319U	8098	4
SHZ201U,ÇEK405U,SHZ203U,ÇMH201U	14573	4
SHZ201U,SOS319U,ÇMH201U,PSİ201U	14826	4
SHZ201U,ÇMH201U,SHZ203U,PSİ201U	14792	4
SHZ203U,TAR201U,ÇMH201U,TÜR201U	7867	4
SHZ203U,TÜR201U,ÇEK405U,ÇMH201U	7962	4
SHZ203U,SOS319U,ÇMH201U,PSİ201U	14048	4
SHZ203U,PSİ201U,ÇEK405U,ÇMH201U	14583	4
ÇEK405U,TAR201U,ÇMH201U,TÜR201U	7685	4
ÇEK405U,TÜR201U,PSİ201U,ÇMH201U	7941	4
ÇEK405U,SOS319U,ÇMH201U,PSİ201U	14259	4
ÇMH201U,TÜR201U,PSİ201U,TAR201U	8119	4
ÇMH201U,TAR201U,SOS319U,PSİ201U	8011	4
SOS319U,TÜR201U,PSİ201U,TAR201U	8069	4
HUK207U,TAR201U,TÜR201U,HUK205U	5073	4
HUK207U,TAR201U,HUK203U,HUK205U	5317	4
HUK207U,TÜR201U,ADL201U,ADL203U	5010	4
HUK207U,TÜR201U,HUK203U,HUK205U	5906	4
HUK207U,ADL201U,HUK201U,ADL203U	24289	4

itemsets	frequency	books
HUK207U,ADL203U,HUK203U,HUK201U	24762	4
HUK207U,HUK201U,HUK203U,HUK205U	25399	4
HUK203U,TAR201U,TÜR201U,HUK201U	5158	4
HUK203U,TAR201U,HUK201U,HUK205U	5396	4
HUK203U,TÜR201U,HUK201U,HUK205U	5797	4
HUK203U,ADL201U,HUK201U,ADL203U	25341	4
HUK203U,ADL203U,HUK205U,HUK201U	25188	4
HUK205U,TAR201U,ADL201U,HUK201U	5015	4
HUK205U,TAR201U,ADL203U,HUK201U	5132	4
HUK205U,TAR201U,HUK201U,TÜR201U	5006	4
HUK205U,TÜR201U,ADL203U,HUK201U	5359	4
HUK205U,ADL201U,HUK201U,ADL203U	24669	4
ADL203U,TÜR201U,TAR201U,HUK201U	5014	4
ADL203U,TÜR201U,HUK201U,ADL201U	5447	4
ADL203U,TAR201U,HUK201U,ADL201U	5462	4
İLH2005,TAR201U,İLH2004,TÜR201U	22633	4
İLH2005,TÜR201U,İLH2003,İLH2004	22800	4
İLH2005,ARA2001,İLH2001,İLH2002	30620	4
İLH2005,İLH2002,İLH2004,İLH2003	31316	4
İLH2005,İLH2001,İLH2003,İLH2004	30931	4
İLH2004,TAR201U,TÜR201U,İLH2003	23154	4
İLH2004,TÜR201U,İLH2002,İLH2003	23308	4
İLH2004,ARA2001,İLH2003,İLH2002	31634	4
İLH2003,TÜR201U,TAR201U,İLH2002	23661	4
İLH2003,TAR201U,İLH2001,İLH2002	23638	4
İLH2003,ARA2001,İLH2001,İLH2002	33204	4
İLH2001,TÜR201U,TAR201U,İLH2002	22744	4
İLH2001,TAR201U,ARA2001,İLH2002	23189	4
İLH2002,TÜR201U,ARA2001,TAR201U	22883	4

itemsets	frequency	books
SAK203U,ÇEK403U,İST207U,İLT203U,SAK201U	6890	5
SAK203U,İLT203U,SAK205U,İST207U,SAK201U	6802	5
SAK201U,ÇEK403U,SAK205U,İST207U,İLT203U	6832	5
SOS201U,İST203U,FEL207U,SİY201U,PSİ201U	7389	5
SOS201U,SOS205U,PSİ201U,SOS203U,SİY201U	7108	5
SOS201U,PSİ201U,SOS203U,FEL207U,SİY201U	7237	5
SOS205U,İST203U,SİY201U,FEL207U,PSİ201U	7103	5
SOS205U,SİY201U,SOS203U,FEL207U,PSİ201U	7017	5
İŞL209U,HUK211U,İST201U,İŞL205U,İKT203U	6905	5
İŞL209U,İST201U,İKT203U,İŞL205U,İŞL207U	6898	5
SOS203U,İST203U,FEL207U,SİY201U,PSİ201U	7161	5
İŞL207U,İŞL205U,HUK211U,İST201U,İKT203U	7179	5
SHZ201U,TAR201U,SHZ203U,SOS319U,TÜR201U	7449	5
SHZ201U,TÜR201U,PSİ201U,SHZ203U,SOS319U	7596	5
SHZ201U,ÇEK405U,PSİ201U,SHZ203U,ÇMH201U	14097	5
SHZ201U,SOS319U,SHZ203U,ÇMH201U,PSİ201U	13885	5
SHZ203U,TAR201U,ÇEK405U,TÜR201U,ÇMH201U	7274	5
SHZ203U,TÜR201U,PSİ201U,ÇEK405U,ÇMH201U	7526	5
ÇEK405U,TAR201U,PSİ201U,TÜR201U,ÇMH201U	7269	5
ÇMH201U,TÜR201U,SOS319U,TAR201U,PSİ201U	7418	5
HUK207U,TAR201U,HUK201U,HUK205U,HUK203U	5037	5
HUK207U,TÜR201U,HUK201U,HUK205U,HUK203U	5460	5
HUK207U,ADL201U,HUK203U,HUK201U,ADL203U	24154	5
HUK207U,ADL203U,HUK205U,HUK203U,HUK201U	24546	5
HUK203U,TAR201U,ADL201U,HUK201U,ADL203U	5164	5
HUK203U,TAR201U,ADL203U,HUK205U,HUK201U	5050	5
HUK203U,TÜR201U,ADL203U,HUK205U,HUK201U	5281	5
HUK203U,ADL201U,HUK205U,HUK201U,ADL203U	24557	5
HUK205U,TÜR201U,ADL201U,HUK201U,ADL203U	5064	5

itemsets	frequency	books
İLH2005,TAR201U,İLH2003,TÜR201U,İLH2004	21930	5
İLH2005,TÜR201U,İLH2002,İLH2004,İLH2003	22019	5
İLH2005,ARA2001,İLH2004,İLH2001,İLH2002	29985	5
İLH2005,İLH2002,İLH2001,İLH2004,İLH2003	30398	5
İLH2004,TAR201U,İLH2002,TÜR201U,İLH2003	22326	5
İLH2004,TÜR201U,İLH2001,İLH2002,İLH2003	22502	5
İLH2004,ARA2001,İLH2001,İLH2003,İLH2002	30993	5
İLH2003,TÜR201U,İLH2001,TAR201U,İLH2002	22296	5
İLH2003,TAR201U,ARA2001,İLH2001,İLH2002	22813	5
İLH2001,TÜR201U,ARA2001,TAR201U,İLH2002	21949	5
SAK203U,ÇEK403U,SAK205U,İST207U,İLT203U,SAK201U	6570	6
SOS201U,İST203U,SOS203U,FEL207U,SİY201U,PSİ201U	7075	6
SOS201U,SOS205U,FEL207U,SOS203U,SİY201U,PSİ201U	6937	6
SOS205U,İST203U,SOS203U,SİY201U,FEL207U,PSİ201U	6866	6
İŞL209U,HUK211U,İŞL207U,İST201U,İŞL205U,İKT203U	6660	6
SHZ201U,TÜR201U,ÇMH201U,SHZ203U,SOS319U,PSİ201U	7234	6
SHZ201U,ÇEK405U,SOS319U,SHZ203U,PSİ201U,ÇMH201U	13475	6
SHZ203U,TAR201U,PSİ201U,ÇEK405U,TÜR201U,ÇMH201U	6962	6
SHZ203U,TÜR201U,SOS319U,ÇEK405U,PSİ201U,ÇMH201U	7041	6
ÇEK405U,TAR201U,SOS319U,TÜR201U,ÇMH201U,PSİ201U	6814	6
HUK207U,TÜR201U,ADL203U,HUK205U,HUK201U,HUK203U	5033	6
HUK207U,ADL201U,HUK205U,HUK201U,ADL203U,HUK203U	23990	6
HUK203U,TÜR201U,ADL201U,HUK205U,HUK201U,ADL203U	5020	6
İLH2005,TAR201U,İLH2002,İLH2004,TÜR201U,İLH2003	21264	6
İLH2005,ARA2001,İLH2003,İLH2001,İLH2004,İLH2002	29704	6
İLH2004,TAR201U,İLH2001,TÜR201U,İLH2002,İLH2003	21663	6
İLH2004,TÜR201U,ARA2001,İLH2001,İLH2003,İLH2002	21973	6
İLH2003,TÜR201U,ARA2001,TAR201U,İLH2001,İLH2002	21659	6
SOS201U,İST203U,SOS205U,SOS203U,FEL207U,SİY201U,PSİ201U	6819	7

itemsets	frequency	books
SHZ201U,TAR201U,ÇMH201U,TÜR201U,SHZ203U,SOS319U,PSİ201U	6740	7
SHZ201U,TÜR201U,ÇEK405U,SOS319U,SHZ203U,PSİ201U,ÇMH201U	6974	7
SHZ203U,TAR201U,SOS319U,ÇEK405U,TÜR201U,ÇMH201U,PSİ201U	6572	7
İLH2005,TAR201U,İLH2001,TÜR201U,İLH2004,İLH2002,İLH2003	20892	7
İLH2005,TÜR201U,ARA2001,İLH2004,İLH2001,İLH2003,İLH2002	21164	7
İLH2004,TAR201U,ARA2001,TÜR201U,İLH2001,İLH2003,İLH2002	21239	7
SHZ201U,TAR201U,ÇEK405U,TÜR201U,SOS319U,SHZ203U,ÇMH201U,PSİ201U	6515	8
İLH2005,TAR201U,ARA2001,TÜR201U,İLH2004,İLH2001,İLH2003,İLH2002	20577	8

***Table:* A.2** *Output File From Semester 5*

itemsets	frequency	books
SOS109U,KOİ301U,KOİ303U	12912	3
KOİ301U,MUH303U,KOİ303U	12905	3
SOS309U,SOS303U,SOS307U	11462	3
SOS311U,SOS301U,SOS305U	11414	3
SOS307U,SOS305U,SOS303U	11163	3
SOS303U,SOS305U,SOS301U	11213	3
HUK301U,MUH103U,ÇEK303U	5747	3
HUK301U,ÇEK301U,ÇEK303U	13348	3
PSİ103U,ÇEK301U,SOS323U	13164	3
SOS323U,MUH103U,ÇEK303U	5746	3
ÇEK303U,MUH103U,ÇEK301U	5747	3
HUK305U,MUH301U,ARY101U	13370	3
SİY303U,HUK101U,SİY301U	8564	3
HUK101U,FİN201U,MUH103U	30273	3
PZL103U,MUH301U,FİN201U	70791	3
FİN201U,MUH103U,MUH301U	30254	3
MUH103U,ÇEK101U,ARY101U	32730	3
MUH103U,ARY101U,MLY301U	36331	3
HİT102U,KYT301U,ARY101U	81090	3
KYT301U,ARY101U,HUK303U	80637	3
ÇEK101U,MLY301U,HUK303U	81304	3
HUK303U,ARY101U,MLY301U	80464	3
MUH301U,ARY101U,MLY301U	13233	3
İKT303U,İKT307U,İKT305U	6771	3
İKT309U,İKT311U,İKT307U	7551	3
İKT307U,İKT311U,İKT305U	7503	3
ULİ305U,ULİ301U,ULİ303U	21827	3

itemsets	frequency	books
ULİ303U,İKT311U,ULİ301U	22070	3
ULİ301U,İKT103U,İKT311U	8564	3
İNG301U,İŞL305U,İŞL301U	5385	3
İKT301U,İŞL107U,İKT103U	9349	3
İŞL303U,İŞL305U,İŞL301U	69045	3
İŞL305U,İKT103U,İŞL301U	29743	3
TAR119U,SÃY303U,SİY301U,HUK101U	8564	4
SOS109U,MUH303U,KOİ303U,KOİ301U	12880	4
SOS309U,SOS301U,SOS307U,SOS303U	10723	4
SOS309U,SOS305U,SOS311U,SOS307U	10796	4
SOS311U,SOS303U,SOS307U,SOS305U	10718	4
SOS307U,SOS301U,SOS305U,SOS303U	10853	4
HUK301U,PSİ103U,ÇEK301U,ÇEK303U	13149	4
PSİ103U,MUH103U,ÇEK303U,ÇEK301U	5735	4
PSİ103U,ÇEK303U,SOS323U,ÇEK301U	13135	4
SOS323U,MUH103U,ÇEK303U,ÇEK301U	5744	4
HUK305U,MLY301U,MUH301U,ARY101U	13203	4
HUK101U,PZL103U,FİN201U,MUH103U	29669	4
HUK101U,MUH301U,MUH103U,FİN201U	29707	4
PZL103U,MUH103U,MUH301U,FİN201U	29857	4
HİT102U,MLY301U,KYT301U,HUK303U	80527	4
HİT102U,HUK303U,KYT301U,ÇEK101U	80731	4
HİT102U,ÇEK101U,KYT301U,ARY101U	80574	4
KYT301U,ÇEK101U,ARY101U,HUK303U	80501	4
KYT301U,MLY301U,ARY101U,HUK303U	80279	4
İKT303U,İKT311U,İKT309U,İKT305U	6653	4
İKT303U,İKT309U,İKT305U,İKT307U	6700	4
İKT309U,İKT305U,İKT311U,İKT307U	7313	4
ULİ305U,İKT311U,ULİ303U,ULİ301U	21625	4

itemsets	frequency	books
ULİ303U,İKT103U,İKT311U,ULİ301U	8564	4
İŞL303U,İKT103U,İŞL305U,İŞL301U	28828	4
SOS309U,SOS301U,SOS305U,SOS303U,SOS307U	10543	5
SOS309U,SOS305U,SOS303U,SOS311U,SOS307U	10418	5
SOS311U,SOS303U,SOS301U,SOS307U,SOS305U	10453	5
HUK301U,MUH103U,ÇEK303U,SOS323U,ÇEK301U	5743	5
HUK301U,PSİ103U,SOS323U,ÇEK301U,ÇEK303U	13104	5
HUK101U,PZL103U,MUH301U,MUH103U,FİN201U	29542	5
MUH103U,HUK303U,HÃT102U,ÇEK101U,ARY101U	32729	5
HİT102U,MLY301U,ÇEK101U,KYT301U,HUK303U	80411	5
HİT102U,HUK303U,ARY101U,KYT301U,ÇEK101U	80387	5
KYT301U,ÇEK101U,MLY301U,ARY101U,HUK303U	80224	5
İKT303U,İKT311U,İKT307U,İKT309U,İKT305U	6620	5
ULİ305U,İKT103U,İKT311U,ULİ301U,ULİ303U	8564	5
SOS309U,SOS301U,SOS311U,SOS305U,SOS307U,SOS303U	10359	6
HUK301U,MUH103U,ÇEK303U,PSİ103U,SOS323U,ÇEK301U	5734	6
HİT102U,MLY301U,ARY101U,KYT301U,HUK303U,ÇEK101U	80145	6
MUH103U,KYT301U,HUK303U,HÃT102U,ÇEK101U,ARY101U,MLY301U	32726	7

Table: A.3 *Output File From Semester 7*

itemsets	frequency	books
ÇEK405U,İŞL293U,ÇEK403U	5144	3
ÇEK405U,SOS319U,ÇEK401U	5013	3
ÇEK405U,ÇEK401U,ÇEK403U	5555	3
HUK407U,MLY401U,HUK405U	5113	3
HUK407U,MLY403U,HUK405U	5064	3
HUK407U,İŞL293U,HUK405U	5277	3
HUK407U,FİN401U,HUK405U	5499	3
MLY401U,HUK405U,İŞL293U	5140	3
MLY401U,FİN401U,İŞL293U	5217	3
ÇEK401U,SOS319U,ÇEK403U	5128	3
ÇEK401U,İŞL293U,ÇEK403U	5168	3
HUK405U,İŞL293U,FİN401U	5522	3
ULİ405U,ULİ401U,ULİ403U	7313	3
ULİ407U,İKT401U,İŞL293U	7114	3
ULİ401U,İKT401U,ULİ403U	7484	3
ULİ403U,İŞL293U,İKT401U	7153	3
SOS317U,SOS321U,SOS315U	8805	3
SOS313U,SOS319U,SOS315U	8475	3
SOS315U,İŞL293U,SOS319U	7647	3
HUK209U,SOS319U,İŞL293U	5057	3
ÇEK403U,SOS319U,İŞL293U	5051	3
TÜR201U,MLY403U,SOS319U	5592	3
TÜR201U,MLY201U,İŞL293U	13341	3
KYT401U,İŞL293U,HUK403U	31894	3
HUK403U,SOS319U,İŞL293U	31190	3
MLY403U,MLY201U,İŞL293U	28312	3
İŞL405U,İŞL401U,İŞL403U	48468	3

itemsets	frequency	books
İŞL403U,İŞL201U,İŞL401U	46442	3
İŞL401U,İŞL293U,İŞL201U	42810	3
ULİ405U,İŞL293U,İKT401U,ULİ403U	6348	4
ULİ405U,ULİ407U,ULİ401U,ULİ403U	6581	4
ULİ405U,İKT401U,ULİ401U,ULİ403U	6629	4
ULİ407U,ULİ403U,İKT401U,ULİ401U	6631	4
ULİ407U,ULİ401U,İŞL293U,İKT401U	6441	4
ULİ401U,İŞL293U,ULİ403U,İKT401U	6620	4
SOS317U,İŞL293U,SOS313U,SOS315U	7518	4
SOS317U,SOS319U,SOS315U,SOS321U	8221	4
SOS317U,SOS313U,SOS321U,SOS315U	8384	4
SOS321U,İŞL293U,SOS313U,SOS315U	7490	4
SOS321U,SOS319U,SOS313U,SOS315U	8167	4
SOS313U,İŞL293U,SOS319U,SOS315U	7520	4
TÜR201U,HUK403U,İŞL293U,SOS319U	5077	4
TÜR201U,HUK403U,KYT401U,SOS319U	5279	4
TÜR201U,KYT401U,İŞL293U,SOS319U	5133	4
TÜR201U,MLY403U,MLY201U,SOS319U	5281	4
TÜR201U,İŞL403U,İŞL405U,İŞL401U	7278	4
TÜR201U,İŞL405U,İŞL401U,İŞL201U	7207	4
TÜR201U,İŞL401U,MLY201U,İŞL201U	6821	4
TÜR201U,İŞL201U,MLY201U,İŞL293U	6371	4
TÜR201U,SOS319U,MLY201U,İŞL293U	5639	4
KYT401U,MLY403U,HUK403U,MLY201U	28778	4
KYT401U,SOS319U,HUK403U,İŞL293U	29661	4
HUK403U,MLY201U,İŞL293U,SOS319U	28004	4
MLY403U,SOS319U,İŞL293U,MLY201U	26991	4
İŞL405U,İŞL293U,İŞL403U,İŞL401U	37214	4
İŞL405U,MLY201U,İŞL401U,İŞL403U	40422	4

itemsets	frequency	books
İŞL405U,İŞL201U,İŞL403U,İŞL401U	43329	4
İŞL403U,İŞL293U,İŞL201U,İŞL401U	38101	4
İŞL401U,MLY201U,İŞL293U,İŞL201U	35133	4
ULİ405U,İŞL293U,ULİ407U,İKT401U,ULİ403U	5926	5
ULİ405U,ULİ407U,İKT401U,ULİ401U,ULİ403U	6253	5
ULİ407U,ULİ403U,İŞL293U,ULİ401U,İKT401U	6106	5
SOS317U,SOS319U,SOS313U,SOS321U,SOS315U	8070	5
SOS321U,İŞL293U,SOS319U,SOS313U,SOS315U	7259	5
TÜR201U,HUK403U,MLY201U,KYT401U,MLY403U	5041	5
TÜR201U,HUK403U,MLY403U,SOS319U,KYT401U	5104	5
TÜR201U,KYT401U,MLY201U,SOS319U,MLY403U	5135	5
TÜR201U,İŞL403U,İŞL201U,İŞL405U,İŞL401U	7152	5
TÜR201U,İŞL401U,İŞL293U,MLY201U,İŞL201U	6262	5
KYT401U,MLY403U,SOS319U,HUK403U,MLY201U	27510	5
KYT401U,MLY201U,İŞL293U,SOS319U,HUK403U	27667	5
HUK403U,MLY403U,İŞL293U,SOS319U,MLY201U	26340	5
İŞL405U,İŞL293U,İŞL201U,İŞL403U,İŞL401U	35917	5
İŞL405U,MLY201U,İŞL201U,İŞL403U,İŞL401U	37579	5
ULİ405U,İŞL293U,ULİ401U,ULİ407U,İKT401U,ULİ403U	5788	6
SOS317U,İŞL293U,SOS319U,SOS321U,SOS313U,SOS315U	7174	6
TÜR201U,İŞL403U,MLY201U,İŞL201U,İŞL405U,İŞL401U	6751	6
KYT401U,MLY403U,İŞL293U,SOS319U,HUK403U,MLY201U	26094	6
İŞL405U,İŞL293U,MLY201U,İŞL201U,İŞL403U,İŞL401U	33787	6
TÜR201U,İŞL403U,İŞL293U,MLY201U,İŞL201U,İŞL405U,İŞL401U	6225	7