A

Project Report

on

DATA CLASSIFICATION USING SEQUENTIAL PATTERN MINING

Submitted in Partial Fulfillment of
the Requirements for the Degree
of

Bachelor of Engineering

in

Computer Engineering

to

North Maharashtra University, Jalgaon

Submitted by

Neha Sonar
Ankit Dixit
Kajal Shelke
Viren Patil
Rahul Patil

Under the Guidance of

Mrs. Dhanashree Tayade

DEPARTMENT OF COMPUTER ENGINEERING
SSBT's COLLEGE OF ENGINEERING AND TECHNOLOGY,
BAMBHORI, JALGAON - 425 001 (MS)

2016 - 2017

SSBT's COLLEGE OF ENGINEERING AND TECHNOLOGY, BAMBHORI, JALGAON - 425 001 (MS)

DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

This is to certify that the project entitled *Data Classification Using Sequential Pattern Mining*, submitted by

Neha Sonar
Ankit Dixit
Kajal Shelke
Viren Patil
Rahul Patil

in partial fulfillment of the degree of *Bachelor of Engineering* in *Computer Engineering* has been satisfactorily carried out under my guidance as per the requirement of North Maharashtra University, Jalgaon.

Date: October 19, 2016
Place: Jalgaon

Mrs. Dhanashree Tayade
Guide

Prof. Dr. Girish K. Patnaik
Head

Prof. Dr. K. S. Wani
Principal

Acknowledgement

First of all we would like to extend our deep gratitude to almighty God, who has enlightened us with power of knowledge. We would like to express our sincere gratitude towards Principal Prof. Dr. K. S. Wani for his encouragement during the work We wish to express our sincere and deep gratitude to Prof. Dr Girish K. Patnaik (Head of Computer Department) for giving us such a great opportunity to develop this project. Inspiration and Guidance are invaluable in all aspects of life especially on the fields of gratitude and obligation and sympathetic attitude which we received from our respected project guide, Mrs. Dhanashree Tayde whose guidance and encouragement contributed greatly to the completion of this project. We would like to thanks to all faculty members of Computer Engineering Department and all friends for their co-operation and supports in making this project successful. We would also like to thanks our parents for supporting us and helping us. We acknowledge our sincere gratitude to all who have directly or indirectly helped us in completing this project successfully.

Neha Sonar
Ankit Dixit
Kajal Shelke
Viren Patil
Rahul Patil

Abbreviations

DFD Data Flow Diagram

IP Interesting Patterns

SCIP Sequence Classification based on Interesting Patterns

UID User Interface Design

UML Unified Modeling Language

Contents

List of Figures

Abstract

In data mining, Sequential pattern mining is used to identify interesting patterns from sequential of data. Sequential pattern mining helps in classification of patterns such as in protein application to identify protein sequence . The existing technique for sequence patterns mining uses a set of rules for classification of data set. These set of rules contains independent rules i.e. x,y are independent of z. Such rules are useless and it is pointless to process them. These rules take unnecessary time for classification. Thus, to prune such rules use of LIFT method. This pruning results in decreasing the amount of time required for processing there rules. Along with decreasing time , the space required to store these rules also decreases.

Keywords: *Sequential Pattern Mining, Interesting Patterns, Classification, LIFT Method.*

Chapter 1

Introduction

Sequential data is often encountered in a number of important settings, such as texts, videos, speech signals, biological structures and web usage logs, where a sequence is generally an ordered list of singletons. Because of a wide range of applications, sequence classification has been an important problem in statistical machine learning and data mining.

In section 1.1 background of the sequential pattern mining is described.Motivation is described in section 1.2. In section 1.3 problem definition is described. Scope is described in section 1.4. In section 1.5 objective is described. The organization of report is described in section 1.6. In section 1.7 Summary is described.

1.1 Background

The sequence classification task can be defined as assigning class labels to new sequences based on the knowledge gained in the training stage. There exist a number of studies integrating pattern mining techniques and classification, such as classification based on association rules (CBA), sequential pattern based sequence classifier , the Classify-By-Sequence(CBS) algorithm, and so on. These combined methods can produce good results as well as provide users with information useful for understanding the characteristics of the dataset. In practice, most datasets used in the sequence classification task can be divided into two main cases. In the first case, the class of a sequence is determined by certain items that co-occur within it, though not always in the same order. In this case, a classifier based on sequential patterns will not work well, as the correct rules will not be discovered, and, with a low enough threshold, the rules that are discovered will be far too specific. In the other case, the class of a sequence is determined by items that occur in the sequence almost always in exactly the same order. At first glance, a sequence based classifier should outperform an itemset based classifier in this situation. However, itemset based classifiers will do better when the pattern sometimes occurs in an order different from the norm. This robustness means that itemset based classifiers can handle cases where small deviations in the subsequences that determine

the class of the sequences occur. Moreover, due to a simpler candidate generation process, itemset based methods are much faster than those based on sequential patterns.

The above observations motivate the proposed research, Sequence Classification based on Interesting Patterns (SCIP). First of all, present algorithms to mine both types of interesting patterns itemsets and subsequences. As a second step, convert the discovered patterns into classification rules, and propose two methods to build classifiers to determine the class to which a new instance belongs. In the first method, select the rules to be applied based on their confidence, while the second uses a novel approach by taking into account how cohesive the occurrence of the pattern that defines the rule is in the new instance. Finally, step away from pattern based classification and evaluate the quality of our pattern miner by using our patterns as features in a variety of feature based classifiers.

When looking for interesting patterns in sequences, a pattern is typically evaluated based on how often patterns occurs (support). However, the proximity of the items that make up the pattern to each other (cohesion) is important, too . If two classes are determined by exactly the same items, traditional pattern based classifiers may struggle. For example, if class A is determined by the occurrence of h with a shortest interval of 2 and class B by the occurrence of h with a shortest interval of 5, pattern h will not be enough to be able to tell the difference, and this will be solved by considering the cohesion information. Therefore, use both cohesion and support to define interesting patterns in a sequence dataset. Finally, utilise these interesting patterns to build classifiers.

Basic Concepts

In data mining,Pattern mining is one of the most important aspects.Although,the patterns mined contain redundant items or subsequences.To eliminate the redundant part various techniques such as frequent pattern mining is used.The drawback of frequent pattern mining is that it introduces monotonicity in the patterns.Thus,lay emphasis on the use of interesting pattern mining.Interesing pattern mining mines the interesting patterns of data pertaining to a particular application.Interesting pattern mining makes use of the Interestingness measure to rate the pattern as to how interesting it is.For finding the interesting measure,the methods support and cohesion are used.Support measures in how many sequences the pattern appears, while cohesion measures how close the items making up the pattern are to each other on average, using the lengths of the shortest intervals containing the pattern in different sequences.Support and Cohesion are elaborated further.

- Support:-Support is an indication of how frequently the item-set appears in the database.

1. Let X be an item-set, X $\rightarrow$ Y an association rule and T a set of transactions of a given database.
2. The support value of X with respect to T is defined as the proportion of transactions in the database which contains the item-set X.

- Cohesion refers to the degree to which the elements of a itemset belong together. Cohesion measures how close the items making up the pattern are to each other on average, using the lengths of the shortest intervals containing the pattern in different sequences.

- Interestingness can be measured using the aforementioned methods,support and cohesion.The product of Support and Cohesion will provide us with Interestingness measure over a class of sequential patterns.

- The items that make up the dataset are converted into Rules after measuring the interestingness. These rules further help in classifying data.The problem is that there are redundant rules,which occupy alot of space and take up CPU time for processing.To avoid such a scenario,pruning of rules is to be done.This pruning of rules is done is using "Lift."

- The lift of a rule is defined as, the ratio of the observed support to that expected if X and Y were independent. If some rule had a lift of 1, it would imply that the probability of occurrence of the antecedent and that of the consequent are independent of each other. When two events are independent of each other, no rule can be drawn involving those two events.If the lift is > 1, that lets us know the degree to which those two occurrences are dependent on one another, and makes those rules potentially useful for predicting the consequent in future data sets.

1.2 Motivation

The pattern mining makes use of interesting patterns. There are numerous ways to find out interesting patterns,but the use of support along with cohesion provides good results with respect to the time factor. The rules are drawn based on these interesting patterns. Confidence is used to prune rules. The problem with Confidence is that it does not account for the dependency of antecedent and consequent of a rule rather,it accounts for the truth value of a rule. So, a rule pertaining to a particular application might be true for another it might not be. In such cases,Lift comes into play. Lift prunes only those rules that are independent of each,thus those rules that are not true for a particular instance but can be true for another are not pruned which is the case when confidence is used.

1.3 Problem Definition

Sequence classification system classifies dataset into sequential pattern. User inputs dataset which is to be classified. The classification involves the measure of interesting pattern in a class of sequence. This involves use of techniques such as support and cohesion. The interesting patterns are then used to build a sequence classifier. The pattern can be both, itemset and subsequences. The support measures in how many sequence the pattern appears. Cohesion measures how close the items are to each on an average. The classifier is built using the classification rules. There may be additional rules which occupy a lot of space. These rules also take time for processing. There is a need to prune such rules . This pruning is done by using a technique known as Lift. Lift is the ratio of observed support to the expected, if x and y were independent . A rule that has independent events are pointless to process. Such rules are eliminated using Lift. The pruned subset classifies the interesting patterns of the dataset into a sequential pattern.

1.4 Scope

Sequence classification uses many techniques such as frequent pattern mining is used. The drawback of frequent pattern mining is that it introduces monotonicity in the patterns. Thus,lay emphasis on the use of interesting pattern mining. There are redundant rules that are drawn for the sole purpose of classification of data. But these redundant rules do not help in any way to classify data.Hence,processing such rules takes up time and space. To prune such rules,Lift is used. Later on,a rule based classifier is built which classifies the data.

1.5 Objective

- To mine patterns in the minimum amount of time.
- Possible interesting Patterns measure which is calculated using support and cohesion methods.
- Prune the redundant classification rules by Lift method.
- Mining data in such a way leads to utilise the time to minimal amount which helps us in attaining our objective.

1.6 Organization of Report

This section specifies the organization of the project.The chapter one of this project contains the introduction about this project which includes the sections like Background, Motivation,Problem Definition, Scope, Objective, Organization of the report and summary. The System Analysis for the project consists of Literature Survey, Proposed System, Feasibility Study(Economical Feasibility, Operational Feasibility and Technical Feasibility), Risk Analysis, Project Scheduling, Effort Allocation and summary which is described in the chapter two. Chapter three constitutes of System Requirement Specification including the sections such as Hardware Requirements, Software Requirements and summary. The sections System Architecture, E-R diagram, Database Design(Schema), Data Flow Diagram, Interface Design(User Interface Design), UML diagrams(USE Case Diagram, Class Diagram, Sequence Diagram, Activity Diagram, Component Diagram, Deployment Diagram, State Diagram) and summary are included in chapter four. Chapter Conclusion and of this project. Thus, the Organization of the report is done in the above manner.

1.7 Summary

In this chapter introduction is described. In the next chapter system analysis is described.

Chapter 2

System Analysis

System analysis is the process of gathering and interpreting facts, diagnosing problems and using the facts to improve the system. System analysis chapter will shows overall system analysis of the concept, description of the system, meaning of the system. System analysis is the study of sets of interacting entities, including computer systems analysis. The development of a computer based information system includes a systems analysis phase which produces or enhances the data model which itself is to creating or enhancing a database. There are a number of different approaches to system analysis. When a computer based information system is developed, systems analysis would constitute the development of a feasibility study, involving determining whether a project is economically, socially, technologically and organizationally feasible. The main purpose behind the development of proposed system is to overcome the drawbacks of existing system.

In the Section 2.1 literature survey is presented. Existing techniques of code clone detection is presented in Section 2.2. In Section 2.3 feasibility study of proposed system is presented. Risk analysis is presented in Section 2.4. In Section 2.5 project scheduling. Effort allocation table is presented in Section 2.6. In Section 2.7 summary is presented.

2.1 Literature Survey

Since the concept of sequential pattern mining was first described by Agrawal and Srikant [1], other sequential pattern mining methods have been developed, such as Generalized Sequential Patterns (GSP) [6], SPADE [9], PrefixSpan [4], and SPAM [3]. PrefixSpan typically shows better performance than GSP and SPADE, but, when dealing with dense databases, the performance of PrefixSpan may be worse than that of SPADE. A number of sequence based classifiers have been based on these methods.

Lesh et al., in [5] has combined sequential pattern mining and a traditional Naive Bayes classification method to classify sequences. They introduced the FeatureMine algorithm

which leveraged existing sequence mining techniques to efficiently select features from a sequence dataset. The experimental results showed that BayesFM (combination of Naive Bayes and FeatureMine) is better than Naive Bayes only. Although a pruning method is used in their algorithm, there was still a large number of sequential patterns used as classification features. As a result, the algorithm could not effectively select discriminative features from a large feature space.

Tseng and Lee in, [8] proposed the Classify-By-Sequence(CBS) algorithm for classifying large sequence datasets. The main methodology of the CBS method is mining classifiable sequential patterns (CSPs) from the sequences and then assigning a score to the new data object for each class by using a scoring function. They proposed a number of alternative scoring functions and tested their performance. The results showed that the length of a CSP is the best attribute for classification scoring.

Exarchos et al. in, [7] has proposed a two-stage methodology for sequence classification based on sequential pattern mining and optimisation. In the first stage, sequential pattern mining is used, and a sequence classification model is built based on the extracted sequential patterns. Then, weights are applied to both sequential patterns and classes. In the second stage, the weights are tuned with an optimisation technique to achieve optimal classification accuracy. However, the optimisation is very time consuming, and the accuracy of the algorithm is similar to that of Feature Mine.

Cheng Zhou et al. in, [2] has proposed in 2016 approach mine the frequent and confident(or discriminative) patterns for building a classifier. the cohesion of a pattern to identify interesting patterns and confident method use for classification rules.Using confident method accuracy of result is increase. But method take a more time for classification and also consume more storage space,hence in proposed system use Lift method instead confident method for better performance.

2.2 Proposed System

The proposed system classifies data using the interesting patterns. Interesting pattern is find by using two methods support and cohesion.

The patterns considered could be both itemsets or subsequences (sequential patterns), which is why provide formal definitions applying to both pattern types below. The support count of a pattern is defined as the number of different sequences in which the pattern occurs; regardless of how many times the pattern occurs in any single sequence. In other

words, when looking for the support count of a pattern alone, user can stop looking at a sequence as soon as user have encountered the first occurrence of the pattern. To determine the interestingness of a pattern, however, it is not enough to know how many times the items making up the pattern occur. also like to know how close they appear to each other. To do this, define interesting patterns in terms of both support and cohesion. Our goal is to first mine interesting patterns in each class of sequences, and then use them to apply a sequence classifier, i.e., a function from the set of sequences S to the set of class labels L.

Classification rules are drawn from interesting patterns. By using Confidence method, find a rule to be true for a particular instances. But the same rule wont be true for another instances. Hence, 'Lift method' is used in proposed system. 'Lift' prunes only those rules in which the antecedent and consequent are independent of each other. The rules which possess dependency are stored in the databases. These rules can be used for classification depending on different instances.

2.3 Feasibility Study

Once scope has been identified, it is reasonable to ask: "Can we build software to meet this scope? Is the project feasible?" All too often, software engineers rush past these questions (or are pushed past them by impatient managers or customers), only to become mired in a project that is doomed from the onset. software feasibility has four solid dimensions: Technology-Is a project technically feasible? Is it within the state of the art? Can defects be reduced to a level matching the application's needs? Finance-Is it financially feasible? Can development be completed at a cost the software organization, its client, or the market can afford?Time-Will the project's time-to- market beat the competition? Resources-Does the organization have the resources needed to succeed?, a project planner must prescribe the time window required for hardware and software and verify that these resources will be available. When a computer-based system (incorporating specialized hardware and software)is to be engineered, the software team may require access to hardware elements being developed by other engineering teams.

2.3.1 Economic Feasibility

This includes an evaluation of all incremental costs and benefits expected if proposed system is implemented. Costs-benefit analysis which is to be done during economical feasibility delineates costs for project development and weighs them against system benefits.The system adds information of colleges and companies for which colleges and companies pays as it provides their information as well as company jobs. So developing this system is economically

feasible. when our project will be complete software which if user is using, does not need to spend extra money for purchasing any type of hardware part hence our project is economic feasible.

2.3.2 Operational Feasibility

Operational feasibility determines whether the proposed system satisfied the user objectives and can be fitted in to current system operation. The system The Prepaid Rationing System can be justified as operationally feasible based on the following Operational feasibility determines if the proposed system satisfied the user objectives and can be fitted in to current system operation. The system Prepaid Rationing System can be justified as operationally feasible based on the following

- The methods of processing and presentation are completely acceptable by the users because they meet all their requirements
- The users have been involved during the preparation of requirement analysis and design process.
- The system will certainly satisfy the user objectives and it will also enhance their capability
- The system will certainly satisfy the user objectives and it will also enhance their effectively.

2.3.3 Technical Feasibility

It is concerned with hardware and software feasibility. In this study, one has to test whether the proposed system can be developed using existing technology or not. As per client requirements the system to be developed should have speed response because of fast exchange of information, reliability, security, scalability, integration and availability. To meet these requirements. We as a developer found Jsp specifications as a right choice because of its features platform independence and reusability. The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system effciently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system. This project is technically feasible as it is developed in Java, which is platform independent.

2.4 Risk Analysis

Uncertainty-the risk may or may not happen; that is, there are no 100 percent probable risks.Loss if the risk becomes a reality, unwanted consequences or losses will occur.When risks are analyzed, it is important to quantify the level of uncertainty and the degree of loss associated with each risk. To accomplish this, different categories of risks are considered. Project risks threaten the project plan. That is, if project risks become real, it is likely that project schedule will slip and that costs will increase. Project risks identify potential budgetary, schedule, personnel (staffing and organization), resource, customer, and requirements problems and their impact on a software project. Project complexity, size, and the degree of structural uncertainty were also defined as project (and estimation) risk factors. Technical risks threaten the quality and timeliness of the software to be produced. If a technical risk becomes a reality, implementation may become difficult or impossible. Technical risks identify potential design, implementation, interface, verification, and maintenance problems. In addition, specification ambiguity, technical uncertainty, technical obsolescence, and "leading edge" technology are also risk factors. Technical risks occur because the problem is harder. Business risks threaten the viability of the software to be built. Business risks often jeopardize the project or the product. Candidates for the top five business risks are

- Building a excellent product or system that no one really wants(market risk)
- Building a product that no longer fits into the overall business strategy for the company(strategic risk)
- building a product that the sales force doesn't understand how to sell
- Losing the support of senior management due to a change in focus or a change in people management risk)
- Losing budgetary or personnel commitment(budget risks)

It is extremely important to note that simple categorization won't always work. Some risks are simply unpredictable in advance. Another general categorization of risks has been proposed by Charette. Known risks are those that can be uncovered after careful evaluation of the project plan, the business and technical environment in which the project is being developed, and other reliable information sources (e.g., unrealistic delivery date, lack of documented requirements or software scope, poor development environment). Predictable risks are extrapolated from past project experience (e.g., staff turnover, poor communication with the customer, dilution of staff effort as ongoing maintenance requests are serviced). Unpredictable risks are the joker in the deck. They can and do occur, but they are extremely difficult to identify in advance.

2.5 Project Scheduling

This section specifies the project scheduling of the project.Software project scheduling is an activity that distributes estimated effort across the planed project duration by allocating the effort to specific software engineering task. In this phase we are identifying all major software engineering activities and the product function to which they are applying. As we have selected the linear sequential model for developing our project we divide the work according to the phases of this model.As we are four partners working on this project and having two months,we schedule the project. If the project has been developed according to the schedule, the project schedule defines the task and milestones that must be tracked and controlled as the project proceed.A Gantt chart helps in scheduling the activities of a project, but it does not help in identifying them. One can begin with the activities identified in the work breakdown structure, as we did for the compiler example. During the scheduling activity, and also during implementation of the project, new activities may be identified that were not envisioned during the initial planning. The manager must then go back and revise the breakdown structure and the schedules to deal with these activities.The fig.2.1 shows the gantt chart of our project, we required 12 weeks to complete our project work, the timeline chart shows the work done in each week and the time required for the same.

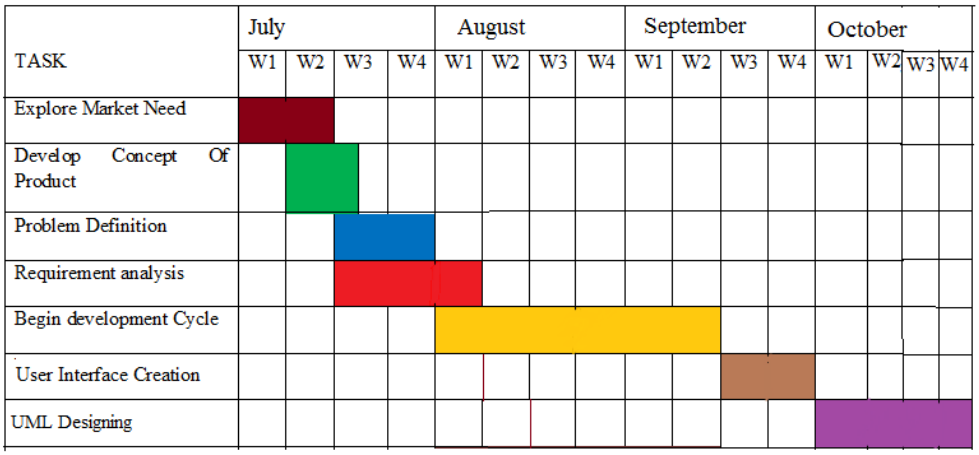

Figure 2.1: Project Scheduling

2.6 Effort Allocation

Each of software project estimation techniques leads o estimate of work units requires completing software development. The characteristics of each project dictate the distribution of efforts. There are four members in this project and there are four phases such as Requirement Gathering, Project planning, Design and Coding, Testing. so below shows effort of each member in this project.

Table 2.1: Effort Allocation

	Neha Sonar	Ankit Dixit	Kajal Shelke	Viren Patil	Rahul Patil
Project Planning	20%	20%	20%	20%	20%
Requirement Gathering	20%	25%	20%	25%	10%
Design	25%	20%	25%	20%	10%

2.7 Summary

In this chapter System analysis is described. In the next chapter System Requirement are described.

Chapter 3

System Requirements

System requirement specifications chapter will provides various requirements of the project such as software and hardware requirements. Understanding user requirements is an integral part of information systems design and is critical to the success of interactive systems.It is now widely understood that successful systems and products begin with an understanding of the needs and requirements of the users. User-centered design begins with a thorough understanding of the needs and requirements of the users. The benefits can include increased productivity, enhanced quality of work, reductions in support and training costs, and improved user satisfaction.

Section 3.1 describes the software requirements of the system. Hardware requirements are described in Section 3.2. Section 3.3 describes summary.

3.1 Hardware Requirements

Hardware requirements give the physical component required for the proposed system.

1. SYSTEM : Pentium IV or above
2. Monitor : 15 VGA Colour.
3. I/O device : Keyboard and mouse
4. Ram : 512 Mb.

3.2 Software Requirements

The Software Requirements Specification is produced at the culmination of the analysis task. The function and performance allocated to software as part of system engineering are refined by establishing a complete information description, a detailed functional description,

a representation of system behavior, an indication of performance requirements and design constraints, appropriate validation criteria, and other information pertinent to requirements.

1. Operating system : LINUX
2. Front end : Java
3. Java version : Jdk1.8

3.3 Functional Requirement

- Provide classification of sequential pattern.
- Find interesting pattern using support and cohesion method.
- Prune the additional classification rules.
- System should be reduce the pattern classification time and increase accuracy

3.4 Non Functional Requirement

Non-functional requirements are requirements that specify criteria that can be used to judge the operation of a system, rather than specific behaviors. This should be contrasted with functional requirements that specify specific behavior or functions. In general, functional requirements define what a system is supposed to do whereas non-functional requirements define how a system is supposed to be. Non-functional requirements are often called qualities of a system. Other terms for non-functional requirements are "constraints", "quality attributes", "quality goals" and "quality of service requirements". Qualities, aka. Non-functional requirements, can be divided into two main categories.

- Execution qualities, such as usability, is observable at run time.
- Evolution qualities, such as testability, maintainability are the static structure of the software system.

3.5 Summary

In this chapter System requirement is described. In the next chapter system design is described.

Chapter 4

System Design

System design is a process of designing the architecture ,components, modules, interfaces and data for a system to satisfy specified requirement. System design could be seen as the application of system theory to product development. In this chapter, various diagrams which are use in developing a system is discussed.

Section 4.1 describes System Architecture. Database Design is described in section 4.2. Section 4.3 describes Data Flow Diagram. Interface Design is described in section 4.4. Section 4.5 describes UML Diagrams. Summary is described in section 4.6.

4.1 System Architecture

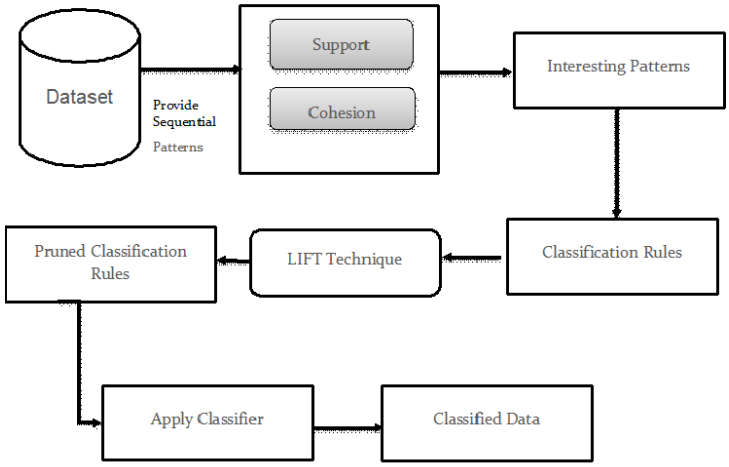

Figure 4.1: System Architecture

The fig.4.1 shows the working of Data Classification System in systematic manner.Syatem Architecture contains four main procedures such as Find Interesting Patterns,Find classification Rules,Prune additional classification rules,Apply Classifier for data classification.

4.2 Database Design

This section specifies the Database design of the project.Database design is the process of producing a detailed data model of a database. This logical data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database.

4.2.1 Schema

A database schema of a database system is its structure described in a formal language supported by the database management system and refers to the organization of data as a blueprint of how a database is constructed. The formal definition of database schema is a set of formulas called integrity constraints imposed on a database. These integrity constraints ensure compatibility between parts of the schema. All constraints are expressible in the same language. A database can be considered a structure in realization of the database language. "A database schema specifies, based on the database administrator's knowledge of possible applications, the facts that can enter the database, or those of interest to the possible end-users." Schemas are generally stored in a data dictionary. Although a schema is defined in text database language, the term is often used to refer to a graphical depiction of the database structure. In other words, schema is the structure of the database that defines the objects in the database.

4.3 Data Flow Diagram

As information moves through software, it is modified by a series of transformations.Data Flow Diagram(DFD) is a graphical representation that depicts information flow and the transforms that are applied as data move from input to output. The basic form of a data flow diagram, also known as a data flow graph or a bubble chart. The data flow diagram may be used to represent a system or software at any level of abstraction. In fact, DFDs may be partitioned into levels that represent increasing information flow and functional detail. Therefore, the DFD provides a mechanism for functional modeling as well as information flow modeling.

The data flow diagram serves two purposes:

- To provide an indication of how data are transform as the moves through the system.
- To depict the function that transforms the data flow.

A level 0 DFD, also called a fundamental system model or a context model. The DFD level 0 shows the abstract of the whole system.

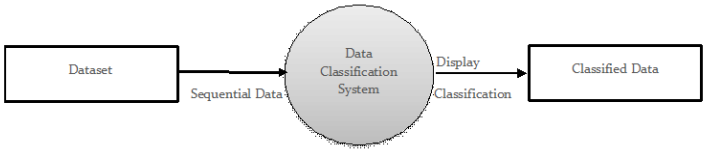

Figure 4.2: Level 0 DFD

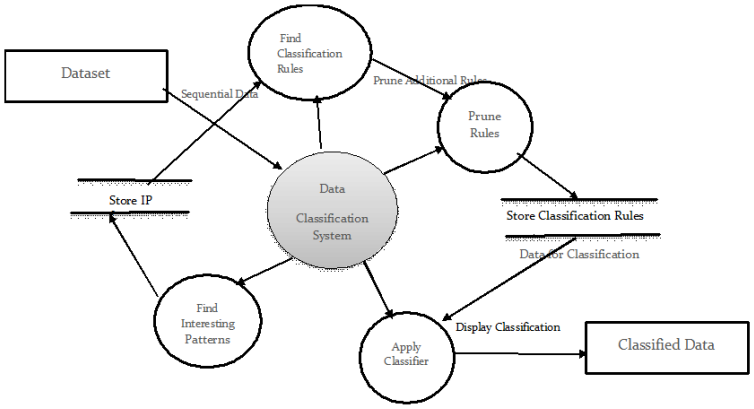

Figure 4.3: Level 1 DFD

The Figure shows the level 1 DFD. The DFD level 1 shows some internal structure of the system also identifies data stores that are used by the major processes.

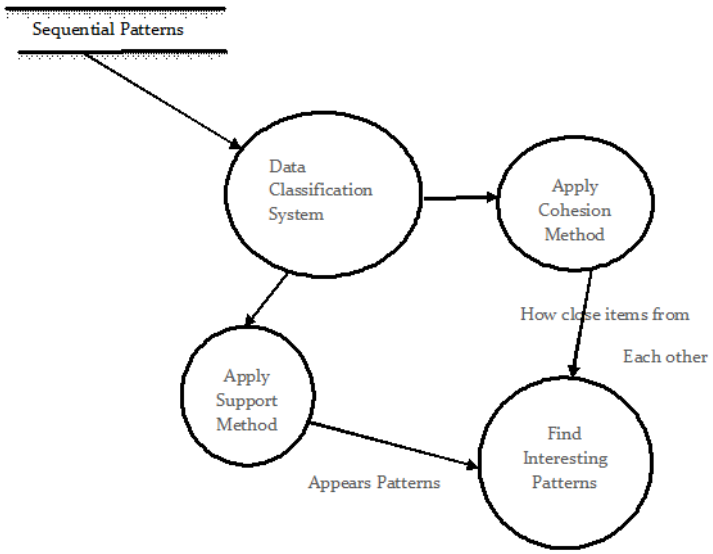

Figure 4.4: Level 2 DFD

Level 2 DFDs aim to give an detailing of the full system. They look at the system in more detail. Major processes are broken down into sub-processes.

4.4 Interface Design

This section specifies the Interface Design of the project.Interface design deals with the process of developing a method for two (or more) modules in a system to connect and communicate. These modules can apply to hardware, software or the interface between a user and a machine. Thus the interface design of the project is stated in this section.

4.4.1 User Interface Design

User Interface Design (UID) or user interface engineering is the design of websites, computers, appliances, machines, mobile communication devices, and software applications with the focus on the user's experience and interaction.

4.5 UML Diagrams

This section illustrates the various UML diagrams of the project.The Unified Modeling Language (UML) is a standard visual modeling language intended to be used for modeling business and similar processes, analysis, design, and implementation of software-based systems.UML is a common language for business analysts, software architects and developers used to describe, specify, design, and document existing or new business processes, structure and behavior of artifacts of software systems. UML is a standard modeling language, not a software development process.

4.5.1 Use Case Diagram

A Use case diagram shows a set of use cases and actors and their relationships. Usecase diagrams address the static use case view of a system. These diagrams are especially important in organizing and modeling the behaviors of a system. The Use Case diagram of the proposed system shows the basic functionality of system such as Provide dataset,measure Interesting PatternsIP,find classification rules,prune the rules,apply the classifier and display classification to user.

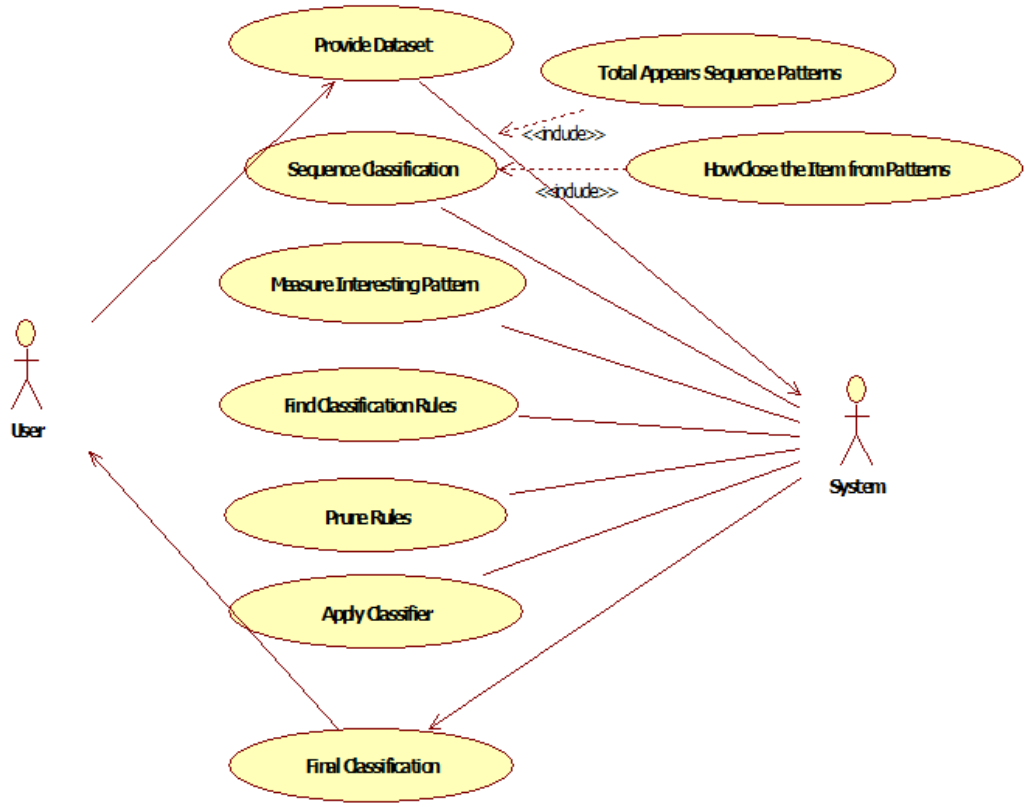

Figure 4.5: UseCase Diagram

4.5.2 Class Diagram

A Class diagram shows a set of classes, interfaces and collaborations and their relationships. These diagrams are the most common diagram found in modeling object-oriented systems. Class diagram address the static design view of a system. Figure 4.6 shows the class diagram of system. Class diagram of the proposed system include various classes with different relationship. Classifier is depend on 'classifier rule' class.

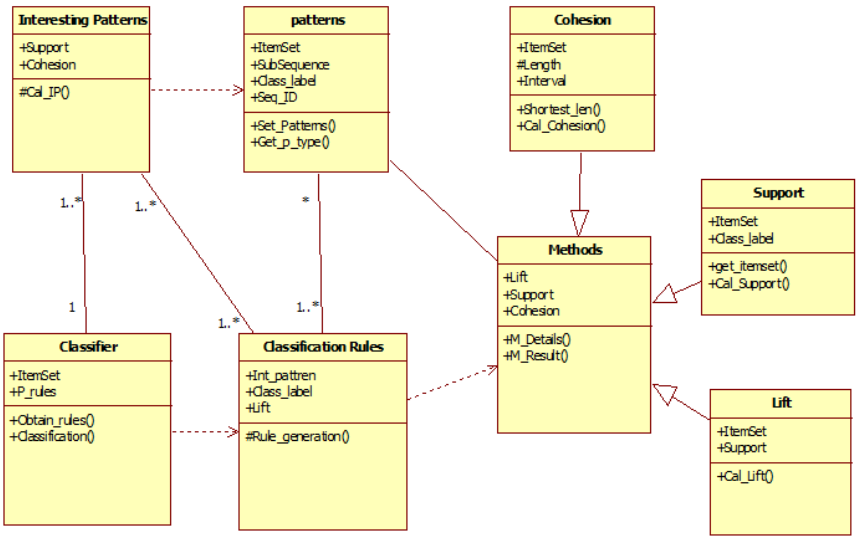

Figure 4.6: Class Diagram

4.5.3 Sequence Diagram

Both sequence diagrams and collaboration diagrams are kinds of interaction diagrams. An shows an interaction, consisting of a set of objects and their relationships, including the messages that may be dispatched among them. Interaction diagrams address the dynamic view of a system. A sequence diagram is an interaction diagram that emphasizes the time ordering of messages; a collaboration diagram is an interaction diagram that emphasizes the structural organization of the objects that send and receive messages. Sequence diagrams and collaboration diagrams are isomorphic, meaning that you can take one and transform it into the other. Sequence diagram shows the Control flow of any system which content object with time-line. figure 4.7, 4.8, 4.9, 4.10 show sequence diagrams of system. All sequence diagrams shows the flow of the various usecases.

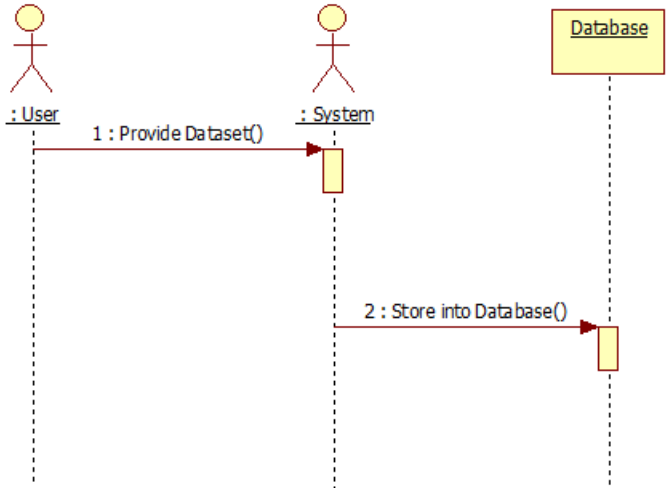

Figure 4.7: Sequence Diagram for UseCase 'Provide Dataset'

4.5.4 Activity Diagram

Activity diagram shows the basic activities between two immediate states of statechart diagram. Activity diagram shows the flow from activity to activity. An activity is an ongoing non atomic execution within a state machine. Activity ultimately results in some action which is made up of executable atomic computation that result in a change in state of system or the return of value. The activity diagram is a collection of vertices and arcs. It also contains FORKING and JOINING as shown in figure. Activity diagram commonly contains activity states and action states, transitions and objects. Figure 4.11 shows the Activity diagram of system.Activity Diagram provide detail between two state such as Sequential patterns and measure interesting patterns.

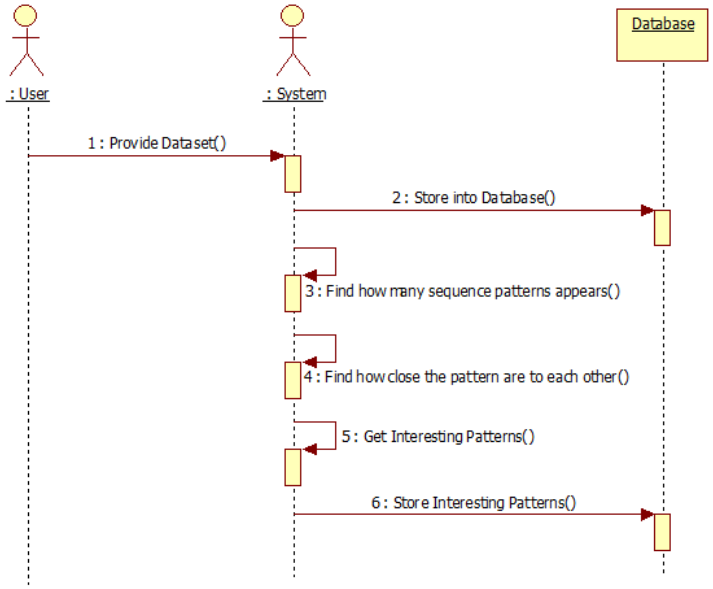

Figure 4.8: Sequence Diagram for UseCase 'Measure Interesting Patterns'

4.5.5 Component Diagram

A component diagram shows the organizations and dependencies among a set of components. Component diagrams address the static implementation view of a system. They are related to class diagrams in that a component typically maps to one or more classes, interfaces, or collaborations. Component Diagram Contains Package Specification,Task specification,Object and dependency relationship amongst the object.Figure 4.12 shows the component diagram of system. proposed system Component diagram shows dependency of modules which are use in system such as system is depend on classifier and dataset again classifier depend upon classification rules and lift method.

4.5.6 Deployement Diagram

A deployment diagram shows the configuration of run-time processing nodes and the components that live on them. Deployment diagrams address the static deployment view of an architecture. They are related to component diagrams in that a node typically encloses one or more components. figure 4.13 shows the deployment diagram of proposed system. Deployment diagram include the physical component of the system such as user and printer.

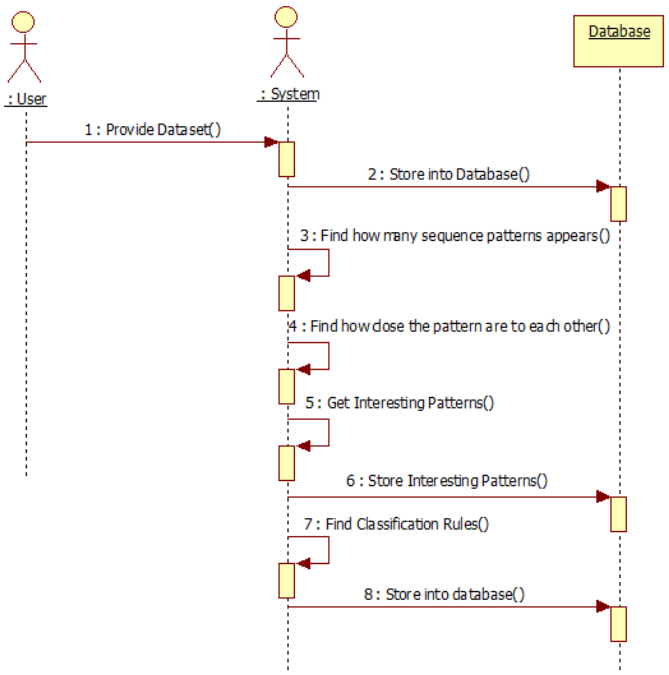

Figure 4.9: Sequence Diagram for UseCase 'Find Classification Rules'

4.5.7 Statechart Diagram

Statechart diagram is use to show behaviour of system in case of some external events. In statechart diagram, a state is a condition during the life of an object or an interaction during which it satisfies some condition, performs some action, or waits for some event. An initial is a kind of pseudo state that represents the starting point in a region of a state machine. It has a single outgoing transition to the default state of the enclosing region, and has no incoming transitions. There can be one (and only one) initial state in any given region of a state machine. It is not itself a state but acts as a marker. A final state represents the last or "final" state of the enclosing composite state. There may be more than one final state at any level signifying that the composite state can end in different ways or conditions. When a final state is reached and there are no other enclosing states it means that the entire state machine has completed its transitions and no more transitions can occur. Figure 4.13 shows the statechart diagram of proposed system. Statechart diagram include various state such as check pattern types,measure interesting pattern,apply classifier etc.

4.6 Summary

In this section UML diagrams are explained which are necessary for understanding the flow of system.

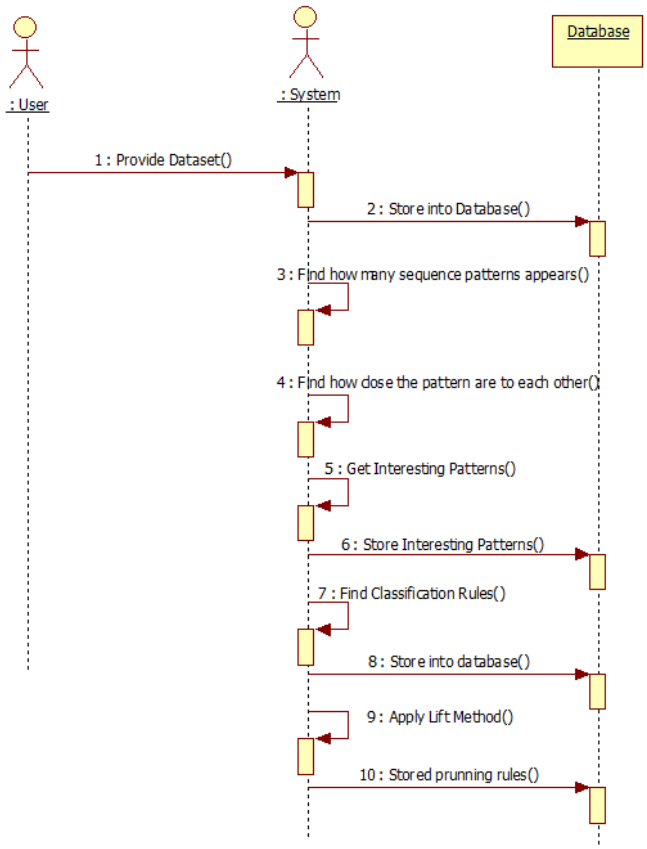

Figure 4.10: Sequence Diagram for UseCase 'Prune Rules'

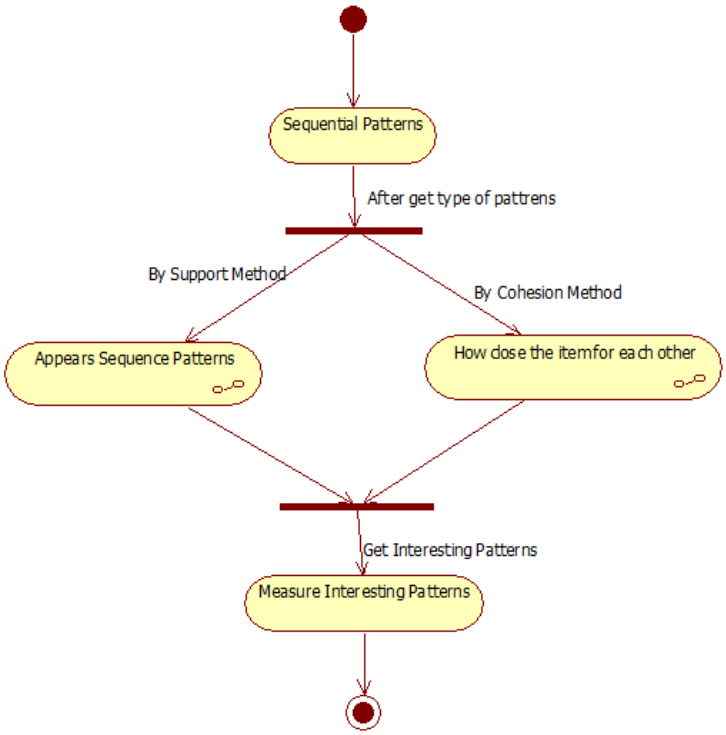

Figure 4.11: Activity Diagram

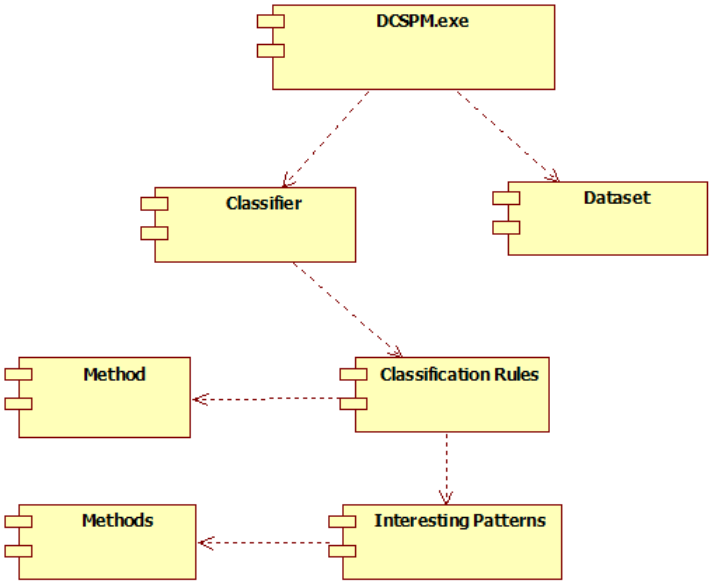

Figure 4.12: Component Diagram

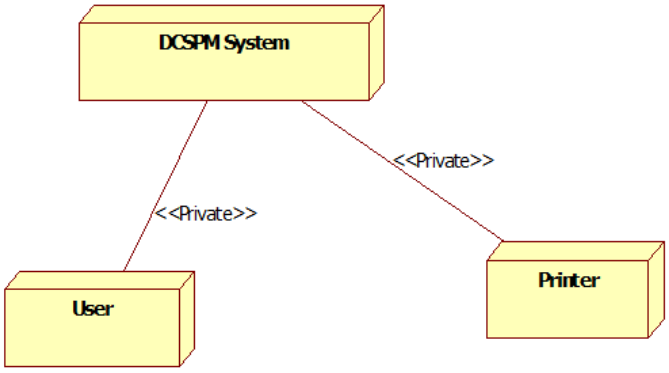

Figure 4.13: Deployment Diagram

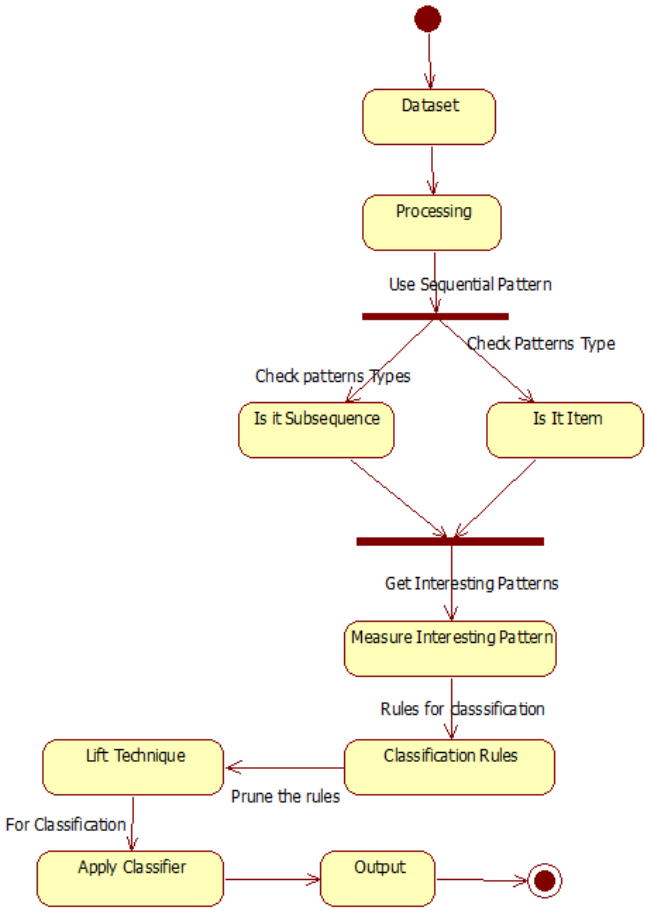

Figure 4.14: StateChart Diagram

Chapter 5

Conclusion

The system classifies data using rules which are drawn from Interesting Pattern. These rules are pruned using Lift instead of Confidence. The proposed algorithm depends mainly on support compared to the algorithm proposed in the paper which uses both support and confidence. This results in simplification of the process as well as the rules that possesses dependency are not pruned. Thus, making the system more versatile.

Bibliography

[1] R. Agrawal and R. Srikant. Mining sequential patterns. *in Proceedings of the 11th International Conference on Data Engineering,*, pages 3–14, 1995.

[2] Boris Cule Cheng Zhou and Bart Goethals. Pattern based sequence classification. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, 2016.

[3] J. Gehrke J. Ayres, J. Flannick and T. Yiu. Sequential pattern mining using a bitmap representation. *in Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM*, pages 429–435, 2002.

[4] B. Mortazavi-Asl J. Wang H. Pinto Q. Chen U. Dayal J. Pei, J. Han and M.-C. Hsu. Mining sequential patterns by patterngrowth: The prefixspan approach. *IEEE Trans. Knowl. Data Eng*, 16(11):1424–1440, 2004.

[5] M. J. Zaki N. Lesh and M. Ogihara. Scalable feature mining for sequential data. *IEEE Intell. Syst*, 15(2):4856, 2000.

[6] R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. *in Proceedings of the 5th International Conference on Extending Database Technology.Springer-Verlag*, pages 3–17, 1996.

[7] C. Papaloukas T. P. Exarchos, M. G. Tsipouras and D. I. Fotiadis. A two-stage methodology for sequence classification based on sequential pattern mining and optimization. *Data and Knowledge Engineering*, 66(3):467–487, 2008.

[8] V. S. Tseng and C.-H. Lee. Effective temporal data classification by integrating sequential pattern mining and probabilistic induction. *Expert Systems with Application*, 36(5):95249532, 2009.

[9] M. J. Zaki. Spade: An efficient algorithm for mining frequent sequences. *Machine Learning*, 42(1-2):31–60, 2001.