

**DOKUZ EYLÜL UNIVERSITY**  
**GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES**

**MACHINE LEARNING APPROACH TO CREATE  
INTELLIGENT DECISION-MAKER SYSTEM**

by  
**Gözde BAKIRLI**

**January, 2016**  
**İZMİR**

# **MACHINE LEARNING APPROACH TO CREATE INTELLIGENT DECISION-MAKER SYSTEM**

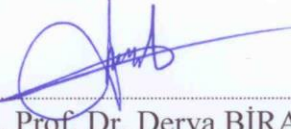
**A Thesis Submitted to the  
Graduate School of Natural and Applied Sciences of Dokuz Eylül University  
In Partial Fulfillment of the Requirements for the Degree of Doctor of  
Philosophy in Computer Engineering**

**by  
Gözde BAKIRLI**


**January, 2016  
İZMİR**

## Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled “**MACHINE LEARNING APPROACH TO CREATE INTELLIGENT DECISION-MAKER SYSTEM**” completed by **GÖZDE BAKIRLI** under supervision of **ASST. PROF. DR. DERYA BİRANT** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

  
Asst. Prof. Dr. Derya BİRANT


Supervisor

  
Prof. Dr. Alp Kut

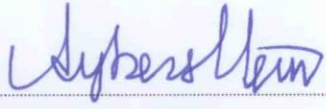
Thesis Committee Member

  
Asst. Prof. Dr. Derya EREN AKYOL

Thesis Committee Member

  
Prof. Dr. Hayri SEVER

Examining Committee Member

  
Doç. Dr. Aybars UĞUR

Examining Committee Member

  
Prof. Dr. Ayşe OKUR

Director

Graduate School of Natural and Applied Sciences

## ACKNOWLEDGEMENTS

I would like to express my special appreciation and thanks to my advisor Asst. Prof. Dr. Derya Birant, you have been a tremendous mentor for me. I would like to thank you for encouraging my research. Your advice on both research as well as on my career have been priceless. I am also highly thankful to Prof. Dr. Alp Kut for his valuable suggestions throughout study. I would also like to thank my committee member, Asst. Prof. Dr. Derya Eren Akyol for serving as my committee member even at hardship and for her brilliant comments and suggestions.

I owe my deepest gratitude to my family. This thesis would not have been possible unless their unflagging love and support. I am indebted to my mother Beyza Bakırlı and my father Mustafa Ali Bakırlı for their care and love. I would also like to thank all of my friends who supported me throughout study.

Special thanks are directed to OLGU Computer Systems for their moral support throughout thesis.

Gözde BAKIRLI

# **MACHINE LEARNING APPROACH TO CREATE INTELLIGENT DECISION-MAKER SYSTEM**

## **ABSTRACT**

Time constraints and growing datasets increase the need for intelligent decision-maker systems. This thesis proposes a novel intelligent decision-maker system for local municipalities using machine learning algorithms. The main purpose of this decision-maker system is to discover intelligent solutions from past data of local municipality services and to estimate future activities. Proposed system includes socio-cultural analyses, income-expense analyses, infrastructure analyses, fraud detection analyses and simplification - verification - similarity analyses.

When analyzing local municipality data, decision tree approach has been used as a data mining technique. We needed to evaluate the similarity of decision trees to compare the knowledge reflected in different trees or datasets. There have been multiple perspectives and multiple calculation techniques to measure the similarity of two decision trees, such as using a simple formula or an entropy measure. Differently from the previous studies, this thesis proposes DTreeSim, a new approach that applies multiple data mining techniques (classification, sequential pattern mining, and k-nearest neighbors) sequentially to identify similarities among decision trees. After the construction of decision trees from different data marts using a classification algorithm, sequential pattern mining was applied to the decision trees to obtain rules, and then k-nearest neighbor algorithm was performed on these rules to compute similarities using two novel measures: General Similarity and Pieced Similarity.

In the experimental studies, novel decision tree similarity measures proposed in this thesis were compared to each other, and also compared with existing approaches. The comparisons indicate that our proposed approach performs better than existing approaches, because it takes into account the values of the branches in the trees through sequential pattern mining.

**Keywords:** Data mining, classification, decision tree, decision support systems, local municipality, machine learning methods

# AKILLI KARAR VERME SİSTEMLERİ OLUŞTURMAK İÇİN MAKİNE ÖĞRENİMİ YAKLAŞIMI

## ÖZ

Zaman kısıtlamaları ve verideki hızlı artış akıllı karar verme sistemlerine olan ihtiyacı arttırmaktadır. Bu tez, makine öğrenme algoritmalarını kullanarak, yerel yönetimler için yeni bir akıllı karar verme sistemi önermektedir. Önerilen karar verme sisteminin temel amacı, yerel yönetimlerdeki geçmiş verilerden akıllı çözümler keşfetmek ve geleceğe yönelik tahminleme yapmaktır. Önerilen sistem sosyo-kültürel analizlerini, gelir-gider analizlerini, altyapı analizlerini, aykırı durum tespit analizlerini ve kolaylaştırma - doğrulama - benzerlik analizlerini içermektedir.

Yerel yönetim verileri analiz edilirken, karar ağacı yaklaşımı bir veri madenciliği tekniği olarak kullanılmıştır. Farklı ağaç veya veri setlerinden yansıyan bilgiyi karşılaştırmak için karar ağaçlarının benzerliğini değerlendirmeye ihtiyaç duyulmuştur. Geçmişte, karar ağaçlarının benzerliğini ölçmek için, basit bir formül veya entropi ölçümü gibi farklı perspektifler ve hesaplama teknikleri kullanılmıştır. Önceki çalışmalardan farklı olarak, bu tezde, karar ağaçları arasındaki benzerliği belirlemek için birden fazla veri madenciliği tekniğini (sınıflandırma, sıralı örüntü madenciliği ve k-en yakın komşu) uygulayan yeni bir yaklaşım, DTreeSim, önerilmektedir. Öncelikle, bir sınıflandırma algoritması kullanılarak farklı veri ambarı alt kümelerinden karar ağaçları oluşturulmakta, ardından kuralları çıkarmak için sıralı örüntü madenciliği karar ağaçları üzerine uygulanmakta, sonrasında da benzerliği hesaplamak amacıyla bu kurallar üzerine Genel Benzerlik ve Parçalı Benzerlik adı verilen iki yeni ölçme tekniği ile birlikte k-en yakın komşu algoritması uygulanmaktadır.

DeneySEL çalışmalarda, tezde önerilen yeni benzerlik hesaplama teknikleri, hem birbirleri ile karşılaştırılmış, hem de mevcut yaklaşımlarla karşılaştırılmıştır. Karşılaştırma sonuçları, tezde önerilen yaklaşımın mevcut yaklaşımlardan daha iyi

sonular rettiđini gstermektedir, nk sıralı rnt madenciliđi ile karar ađalarındaki dalların deđerleri de dikkate alınmaktadır.

**Anahtar kelimeler:** Veri madenciliđi, sınıflandırma, karar ađacı, karar destek sistemleri, yerel ynetim, makine đrenmesi yntemleri



## CONTENTS

	<b>Page</b>
Ph.D. THESIS EXAMINATION RESULT FORM .....	ii
ACKNOWLEDGEMENTS .....	iii
ABSTRACT .....	iv
ÖZ .....	vi
LIST OF FIGURES .....	xii
LIST OF TABLES .....	xiv
<b>CHAPTER ONE - INTRODUCTION .....</b>	<b>1</b>
1.1 General .....	1
1.2 Purpose .....	3
1.3 Novel Contributions of This Thesis .....	5
1.4 Organization of the Thesis .....	6
<b>CHAPTER TWO - MACHINE LEARNING AND DATA MINING .....</b>	<b>7</b>
2.1 Machine Learning .....	7
2.2 Data Mining .....	7
2.2.1 Classification .....	8
2.2.2 Association Rule Mining .....	10
2.2.3 Sequential Pattern Mining .....	13
2.2.4 Clustering .....	14
2.2.5 Outlier Detection .....	15
2.3 Application Areas .....	16
2.4 Challenges of Data Mining in Local Municipalities .....	16
<b>CHAPTER THREE - RELATED WORKS .....</b>	<b>18</b>

3.1 Literature Review .....	18
3.2 Studies for Similarity of DTs .....	18
3.3 Studies for SPM with Other DM Techniques .....	20
3.4 Studies for KNN with New Distance Formulas .....	21
3.5 Studies for Re-Mining.....	21
3.6 Studies for Decision Support .....	21
<b>CHAPTER FOUR - PROPOSED APPROACH: DTREESIM .....</b>	<b>24</b>
4.1 Step 1: Data Marting .....	24
4.2 Step 2: Classification .....	26
4.3 Step 3: Tree-to-sequence Linearization .....	27
4.4 Step 4: Sequential Pattern Mining .....	29
4.5 Step 5: Similarity Calculation and K-Nearest Neighbor Algorithm .....	31
<b>CHAPTER FIVE - EXPERIMENTAL RESULTS .....</b>	<b>35</b>
5.1 Dataset Description .....	35
5.1.1 Data Preparation .....	35
5.1.1.1 Data Reduction.....	35
5.1.1.2 Data Integration.....	35
5.1.1.3 Data Discretization.....	37
5.2 Application of DTreeSim Step by Step.....	39
5.2.1 Application of Data Marting.....	39
5.2.2 Application of Classification .....	40
5.2.3 Application of Tree-to-Sequences Linearization .....	41
5.2.4 Application of Sequential Pattern Mining .....	44
5.2.5 Application of Similarity Calculation.....	47
5.3 General Similarity of Trees .....	49
5.4 Pieced Similarity of Trees .....	52
5.5 Comparison of General Similarity and Pieced Similarity.....	57
5.6 Comparison of DTreeSim with Existing Approaches.....	59

5.7 The Application of DTreeSim to Benchmark Data with Bootstrapping.....	61
---	----

**CHAPTER SIX - DECISION MAKER SYSTEM FOR LOCAL MUNICIPALITIES ..... 63**

6.1 The Scope of the System.....	63
6.2 Scenarios for Local Municipalities .....	64
6.2.1 Socio-Cultural Analyses .....	64
6.2.1.1 Employee Analysis .....	64
6.2.1.2 Citizen Analysis .....	66
6.2.1.3 Staff Analysis.....	68
6.2.2 Income/Expense Analyses .....	70
6.2.2.1 Moveable Material Analysis .....	70
6.2.2.2 Estimation of Wages .....	73
6.2.2.3 Income Operations Analysis .....	75
6.2.3 Infrastructure Analyses .....	77
6.2.3.1 Water Notice Analysis .....	78
6.2.3.2 Electricity Consumption Analysis.....	80
6.2.4 Fraud Detection Analyses.....	81
6.2.4.1 Logs Analysis.....	81
6.2.4.2 Fuel Oil Analysis .....	84
6.2.4.3 Cash Desk Analysis .....	85
6.2.5 Simplification, Verification and Similarity Analyses.....	87
6.2.5.1 User Account Analysis.....	87
6.2.5.2 Accountancy Analysis.....	88

**CHAPTER SEVEN – CONCLUSION AND FUTURE WORK ..... 92**

7.1 Conclusion .....	91
7.2 Future Work .....	92

**REFERENCES..... 92**

**APPENDICES ..... 98**

## LIST OF FIGURES

	<b>Page</b>
Figure 1.1 Proposed decision-maker system architecture.....	4
Figure 2.1 Pseudo code of C4.5 algorithm.....	9
Figure 2.2 Pseudo code of Apriori algorithm .....	12
Figure 2.3 Pseudo code of Prefix-Span algorithm .....	14
Figure 4.1 Schematic showing the DTreSim approach.....	24
Figure 4.2 Data marting .....	25
Figure 4.3 An example tree structure related to municipal data used in this thesis ...	28
Figure 4.4 Outline of the DTreeSim algorithm showing the five main steps .....	33
Figure 4.5 Flowchart of the DTreeSim algorithm.....	34
Figure 5.1 Sample classification output (Weka).....	40
Figure 5.2 TSL pseudo code .....	41
Figure 5.3 Decision tree sequences .....	43
Figure 5.4 Pseudo code of dataset formatting for SPMF.....	44
Figure 5.5 Sample dataset in SPMF form .....	45
Figure 5.6 August 2003 dataset for “Not Paid” class.....	45
Figure 5.7 August 2003 dataset for “Paid” class.....	46
Figure 5.8 FSs of August 2003 dataset for “Paid” class (AUGUST2003Paid.txt)....	46
Figure 5.9 FS of August 2003 dataset for “Not Paid” class (AUGUST2003NotPaid .txt) .....	46
Figure 5.10 Pseudo code of CFS.....	47
Figure 5.11 Result file for “Paid” class.....	48
Figure 5.12 Result file for “Not Paid” class.....	48
Figure 5.13 Pseudo code for similarity calculation.....	49
Figure 5.14 Histogram of general tree similarity based on the common frequent sequences (CFS).....	50
Figure 5.15 General tree similarity based on common frequent sequences (CFS) (top 10).....	51
Figure 5.16 General tree similarity based on common frequent sequences (CFS)....	52

Figure 5.17 Histogram of pieced similarity given by the number of common frequent sequences (CFS).....	53
Figure 5.18 Pieced similarity given by the number of common frequent sequences (CFS) (top 10).....	54
Figure 5.19 Pieced similarity given by the number of common frequent sequences (CFS) .....	54
Figure 5.20 Histogram of pieced similarity given by the percentage values .....	55
Figure 5.21 Pieced similarity given by percentage values (top 10 similarity (%))....	56
Figure 5.22 Average common frequent sequence (ACFS) number used to compare General similarity and Pieced similarity techniques.....	58
Figure 5.23 The number of comparison operations carried out on trees for General similarity and Pieced similarity techniques .....	59
Figure 5.24 Structure for Decision Tree 1 .....	60
Figure 5.25 Structure for Decision Tree 2 .....	60
Figure 6.1 Sample tree for Estimation of Wages scenario.....	75

## LIST OF TABLES

	<b>Page</b>
Table 2.1 An example dataset for Association Rule Mining .....	11
Table 4.1 Data warehouse vs. data mart .....	26
Table 4.2 Sample attribute value pair coding for SPMF.....	29
Table 5.1 Income accrual database table.....	35
Table 5.2 City citizens database table .....	36
Table 5.3 Real estate declaration details database table .....	36
Table 5.4 City streets database table .....	36
Table 5.5 Basic dataset descriptors for local municipality data.....	38
Table 5.6 Sample data mart for August 2003 .....	39
Table 5.7 Similarity values obtained using different approaches .....	61
Table 5.8 Car evaluation dataset descriptors for data from the UCI Machine Learning Repository .....	61
Table 6.1 Employee database table.....	65
Table 6.2 Employee Family Members database table .....	65
Table 6.3 Employee Scoring database table .....	65
Table 6.4 Sample dataset for Employee Analysis.....	65
Table 6.5 Employee Analysis results .....	66
Table 6.6 Citizens database table (reduced form for Citizen Analysis).....	66
Table 6.7 Sample dataset for Citizen Analysis .....	67
Table 6.8 Citizen Analysis results.....	67
Table 6.9 Staff database table .....	68
Table 6.10 Staff Family Members database table .....	69
Table 6.11 Education Status database table .....	69
Table 6.12 Sample dataset for Staff Analysis .....	69
Table 6.13 Sample results of Staff Analysis .....	69
Table 6.14 Receipts database table .....	70
Table 6.15 Receipt Details database table.....	71
Table 6.16 Materials database table .....	71
Table 6.17 Account database table.....	71

Table 6.18 Sample dataset for Moveable Material Analysis .....	71
Table 6.19 Prepared dataset for Moveable Material Analysis .....	72
Table 6.20 Sample results for Moveable Material Analysis .....	72
Table 6.21 Staff database table (reduced form for Estimation of Wages) .....	73
Table 6.22 Staff Movements database table .....	74
Table 6.23 Jobs database table .....	74
Table 6.24 Sample dataset for Estimation of Wages .....	74
Table 6.25 Income Operations database table.....	76
Table 6.26 System Users database table .....	76
Table 6.27 Sample dataset for Income Operations Analysis .....	76
Table 6.28 Sample results of Income Operations Analysis .....	77
Table 6.29 Subscribers database table .....	78
Table 6.30 Notices database table .....	78
Table 6.31 Sample dataset for Water Notice Analysis .....	78
Table 6.32 Prepared dataset for Water Notice Analysis .....	79
Table 6.33 Sample results of Water Notice Analysis .....	80
Table 6.34 Receipts database table .....	80
Table 6.35 Receipt Details database table.....	81
Table 6.36 Sample results of Electricity Consumption Analysis.....	81
Table 6.37 Logs database table .....	82
Table 6.38 Day codes .....	82
Table 6.39 Sample dataset for Logs Analysis .....	83
Table 6.40 Sample results of Logs Analysis .....	83
Table 6.41 Receipts database table (reduced form for Fuel Oil Analysis) .....	84
Table 6.42 Receipt Details database table (reduced form for Fuel Oil Analysis).....	84
Table 6.43 Materials database table .....	85
Table 6.44 Sample dataset for Fuel Oil Analysis.....	85
Table 6.45 Sample results for Fuel Oil Analysis .....	85
Table 6.46 Acquitanes database table .....	85
Table 6.47 Citizens database table (reduced form for Cash Desk Analysis) .....	86
Table 6.48 Collectors database table.....	86
Table 6.49 Sample dataset for Cash Desk Analysis.....	86



Table 6.50 Sample results for Cash Desk Analysis .....	86
Table 6.51 User Accounts database table .....	87
Table 6.52 Sample dataset for User Accounts Analysis .....	87
Table 6.53 Sample results for User Accounts Analysis .....	88
Table 6.54 Accountancy database table .....	89
Table 6.55 Accountancy Details database table.....	89
Table 6.56 Sample dataset for Accountancy Analysis.....	89
Table 6.57 Sample results for Accountancy Analysis .....	90

# **CHAPTER ONE**

## **INTRODUCTION**

### **1.1 General**

Decision maker systems are intelligent systems that supports organizational decision making. These systems ensure fast decision making activities, increase the control and automate managerial processes. There is not universally-accepted taxonomy of decision maker systems. Taxonomy of decision maker systems depends on point of view. For example according to relationship with the user, there are three decision maker system types: passive, active and cooperative (Haettenschwiler, 1999). Passive decision maker systems just collect and reveal data. Active systems process data, analyse data and interpret solutions and cooperative systems allow the user to modify the suggestions provided by the system. Proposed decision maker system in this thesis is active decision maker system and it processes data, shows and interprets solutions for the user. Another taxonomy of decision maker systems is operational taxonomy (Power, 2002). According to operations of decision maker systems, there are five different system types: communication-driven, data-driven, document-driven, knowledge-driven and model-driven. Communication-driven systems use communication and collaboration among people. Unstructured data is used in document-driven systems and model-driven systems use a model as input like statistical and simulation. Data-driven systems use datasets or data warehouses and knowledge-driven decision maker systems use data mining and suggest solutions to users. Proposed decision maker system in this thesis is knowledge-driven and data-driven while data mining is applied on data warehouse. In order to make decision, knowledge-driven decision maker systems utilize data mining techniques which are used to extract hidden patterns from large datasets in human understandable. Data mining has ability to tell unknown important things or what is going to happen next.

Decision maker systems are needed especially in marketing research for different purposes. For example for market segmentation to determine behaviour of customers, for trend analysis or for customer churn. While decision maker systems

are important for business to increase the gain of companies, they are necessary for local municipalities to increase the national economy.

Current municipal management information systems support just execution of operational processes and creation of standard reports. New method and techniques that include data mining should be used to utilize retrospective knowledge, to develop solutions which are close to human intelligence and to make prediction for the future. Data mining techniques simplify the way to give information to the local municipality about current status and to make decision for the future. Data mining solutions for local municipalities provide that local municipalities can discover hidden patterns, relationships, changes, irregularities, and rules from large datasets. Local municipalities can perform their decision making process more rational, more accurate and faster. Increase in the prevalence of use of municipal management information systems and formation of huge data heaps that ordinary analyses are inadequate for, increases business intelligence need at municipal sector.

In local municipalities, it is possible to collect huge amounts of data on citizen transactions, staff usage history, goods purchases, consumption, and service. The data quantity is continuously expanding exponentially, mainly due to increasing ease, availability, and popularity of management information systems.

The advantages of data mining enabled in local municipality are endless. DM can help:

- identify citizen behaviours,
- discover staff related patterns and trends,
- improve the quality of citizen services,
- manage resources effectively,
- achieve better citizen satisfaction,
- reduce consumption ratios,
- analyze records of financial transactions,

- design more effective facilities,
- monitor money movements to detect criminal activities,
- reduce the cost of inactive services.

## **1.2 Purpose**

This thesis proposes an intelligent decision-maker system to give information to the local government about current status and to make decision for the future. It simplifies the way to give information to the local government about current status and to make decision for the future. Additionally, it contributes to the national economy with income/expense, wastage and abuse analyses at local municipalities. Proposed decision-maker system includes socio-cultural analyses, income/expense analyses, infrastructure analyses, fraud detection analyses, simplification/verification and similarity analyses. These analyses include 13 new scenarios which were created to discover intelligent solutions from past data of local municipality and to estimate future activities. Socio-cultural analyses include Employee Analysis, Citizen Analysis and Staff Analysis. Income/Expense analyses are moveable material analysis, estimation of wages and income operation analysis. Another type of analyses is Infrastructure analyses, water notice and electricity consumption analysis belong to infrastructure analyses. Fraud detection analyses include logs analysis, fuel oil analysis and cash desk analysis. User account analysis and accountancy analysis belong to simplification/verification and similarity analyses. The aim of these scenarios is to facilitate decision making for future activities and to give information about current situation of the local municipalities (Bakirli et al., 2012).

Proposed decision-maker system is shown in Figure 1.1. Developed decision-maker system includes five main services: Data Preparation Service, Clustering Service, Classification Service, Association Rule Mining Service and Outlier Detection Service. Data comes from Management Information Service (MIS) database and Data Preparation service prepares this data for data mining algorithms by using four main data preparation techniques: Data Integration, Data Reduction, Data Pre-processing and Data Transformation.

K-Means++ algorithm was preferred for clustering data mining technique. For association rule mining algorithm FP-Growth algorithm was chose and C4.5 algorithm was used for classification technique while Local Outlier Factor (LOF) was preferred for outlier detection data mining technique.

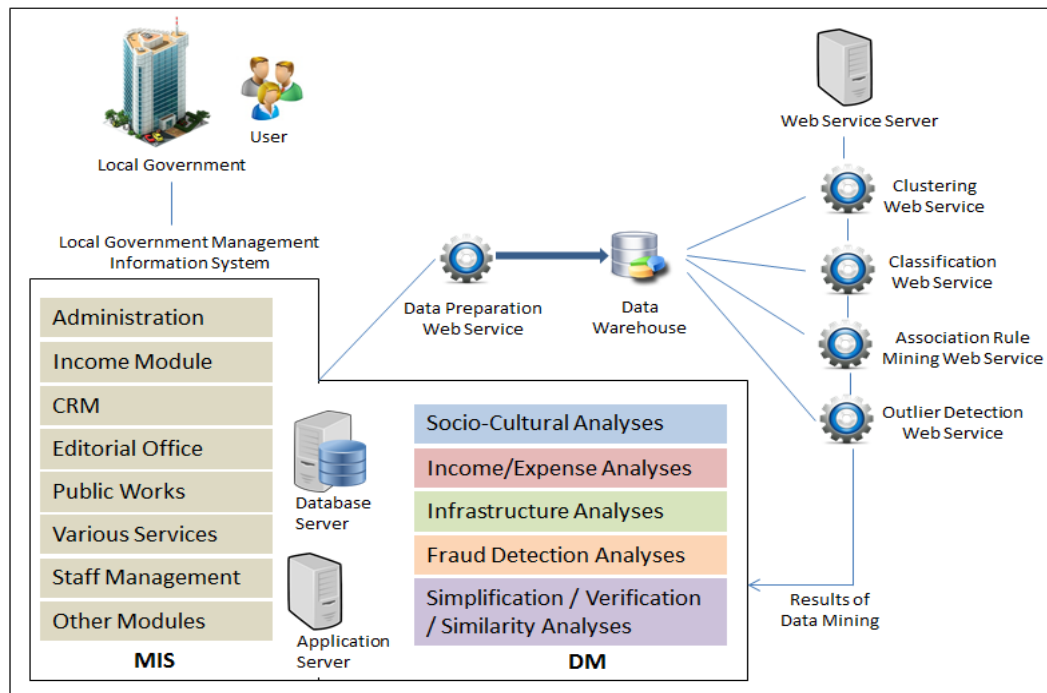


Figure 1.1 Proposed decision-maker system architecture.

This decision-maker system includes a new approach, DTreeSim (Decision Tree Similarity). The main goal of this approach is to identify similarities among DTs by combining three data mining techniques which are described above: (i) classification to construct decision trees (DTs) from a number of data marts, (ii) sequential pattern mining to determine the characteristics of these decision trees, and (iii) k-nearest neighbor method to identify similar decision trees, and consequently similar data marts.

The evaluation of the similarity of DTs can be used for different purposes, e.g., to compare patterns related to different geographical regions or different periods. As an example; the similarity of decision trees in time means that related data marts include

similar observations in the time dimension. In addition, similarity measures can help us compare two models that have been built based on two datasets. Furthermore, the similarities among decision trees are investigated for tree matching problems, interpreting decision trees, and change detection in classification models. A number of recent studies needed to evaluate the similarity of decision trees for different purposes in different application areas such as for privacy preserving (Fletcher & Islam, 2015), agent based modeling systems (Dogra, 2014), the construction of Genetic Algorithm Tree (GATree) (Papagelis & Kalles, 2000) and speech recognition (Telaar & Fuhs, 2013).

### **1.3 Novel Contributions of This Thesis**

This thesis includes five main contributions.

First, proposed decision-maker system and our new approach DTreeSim are applied to novel experimental studies of local municipality data. Using our approach, tax payments can be analyzed and used to predict future tax revenue for the municipality.

Second, a link between DTs and SPM is introduced for the first time, with SPM applied to DTs. To achieve this goal, DTs are linearized and converted into sequences. This is a novel approach to DTs. The SPM provides compact rules for a decision tree to supply fast and correct comparisons.

Third, a new distance function is needed for the KNN algorithm for sequential data, since existing distance functions like the Euclidian or the Manhattan do not meet our requirements. Thus, we defined and compared new distance measures to calculate the similarity of DTs: General Similarity and Pieced Similarity.

Fourth, data mining algorithms are applied sequentially, so a re-mining process is performed on re-mining results for the first time. We have called this operation re<sup>2</sup>mining. Fourth, we apply a classification algorithm to multiple data marts. In this

way, a forest is created from individual DTs for subject, region or time oriented analysis.

Last, DTreeSim, our new approach is used for calculating similarity among DTs. Our comparisons indicate that our proposed approach performs better than existing approaches.

#### **1.4 Organization of the Thesis**

The rest of the thesis is structured as follows:

In Chapter Two, data mining techniques and algorithms which were used in this thesis, are explained.

Chapter Three summarizes the related literature and previous studies.

Chapter Four presents the proposed novel approach, DTreeSim, and explains five steps of this approach. This section also defines a new term, Re<sup>2</sup>Mining, and gives two new concepts: General Similarity and Pieced Similarity.

In Chapter Five, the results obtained from experiments are presented, comparison focusing on an analysis of the different similarity calculation techniques on a real world dataset.

In Chapter Six, intelligent decision-maker system and its analyses and scenarios for local municipalities are explained.

Chapter Seven contains some concluding remarks and possible future studies.

## **CHAPTER TWO**

### **MACHINE LEARNING AND DATA MINING**

#### **2.1 Machine Learning**

Machine learning is the general name of computer algorithms which build models of given problem based on acquired data from the environment of the problem. In other words, machine learning algorithms learn from data and make predictions based on known properties.

Machine learning uses data mining techniques to build models of given problem and to predict future and data mining uses machine learning techniques to mine big datasets. Data mining explains patterns and machine learning predicts with these patterns and models.

#### **2.2 Data Mining**

Data mining is the process of analysis data and extracting hidden useful patterns from large datasets to provide information about a related condition or estimation of a future similar condition. There are different types of data mining techniques, which serve different purposes. The main purposes are classifying new data based on training data, clustering similar instances, finding outliers, and discovering interesting patterns inherent in data. For example, association rule mining is used to find interesting relationships among a large set of data items. For example, outlier detection is used to detect anomalous observations (i.e. fraud) in a sample dataset while clustering is a data mining task to group similar objects together.

The rest of the chapter explains the main tasks of data mining: classification, association rule mining, sequential pattern mining, clustering and outlier detection.



### 2.2.1 Classification

One of the major data mining tasks is classification, assigning items into target categories. Data mining algorithms for classifying include Decision Tree (DT), Naive Bayes, Neural Network, and K-Nearest Neighbors (KNN) algorithms.

Decision tree algorithms build trees from a set of training data based on a heuristic to decide on the importance of the features. In this thesis, in DTreeSim, data warehouse is partitioned into data marts according to specific criteria (time, region... etc.), then decision tree algorithm is applied on each data mart and a forest which includes multiple decision trees, is created. Also decision tree algorithm is applied on local municipality data for '*Estimation of Wages*' scenario in proposed decision maker system in this thesis.

The main decision tree algorithms are ID3 (Iterative Dichotomiser 3), CART (Classification And Regression Tree), Naïve Bayes Tree, Random forests, CHAID (Chi-squared Automatic Interaction Detector), MARS and C4.5 (successor of ID3) algorithms.

The C4.5 algorithm (Quinlan, 1993; 1996) was used as the decision tree algorithm in this thesis. It is the extension of the ID3 algorithm. The C4.5 algorithm can be applied on numeric attributes, can deal with missing values and also can prune to deal with noisy data unlike ID3. The C4.5 algorithm was ranked #1 in the top 10 algorithms for data mining in 2008 (Wu et al., 2008)

The C4.5 algorithm is based on the information gain ratio and constructs the decision tree with a divide and conquer strategy. Pseudo code of the C4.5 algorithm (Kotsiantis, 2007) is given in Figure 2.1.

```

Check for base cases
  For each attribute a
    Find the feature that best divides the training data such as information gain
    from splitting on a
  Let a_best be the attribute with the highest normalized information gain Create a
  decision node node that splits on a_best
    Recurse on the sub-lists obtained by splitting on a_best and add those nodes as
  children of node

```

Figure 2.1 Pseudo code of C4.5 algorithm (Kotsiantis, 2007).

Entropy and Information Gain are calculated to find a\_best. Entropy is a measure of the amount of uncertainty in the data set.

$$Entropy(S) = \sum_{i=1}^n -P(C_i) \times \log_2 P(C_i) \quad (2.1)$$

where S is the dataset, n is the class number in S, C<sub>i</sub> is the i<sup>th</sup> class in S, P(C<sub>i</sub>) is the frequency of C<sub>i</sub> in S.

Information Gain is the measure of the difference in entropy from before to after the set is split on an attribute. It is used to select the attribute that will best separate the samples into individual classes. This attribute is called test attribute.

$$Gain(S, A) = Entropy(S) - \sum_{i=1}^m P(A_i) Entropy(S_{A_i}) \quad (2.2)$$

where Gain(S,A) is the gain of S after a split on attribute A, m is the number of values of attribute A in S, P(A<sub>i</sub>) is the frequency of cases that have A<sub>i</sub> value in S, Entropy(S<sub>A<sub>i</sub></sub>) is the subset of S with items that have A<sub>i</sub> value.

Another popular classification algorithm is k-Nearest Neighbor (KNN). In the KNN algorithm (Dudani, 1976), an object is assigned to the class most common among its *k* nearest neighbors. While identifying the most similar *k* objects, commonly, Euclidian distance function is used to calculate distances for continuous variables and Jaccard distance is used for categorical variables.

Distance functions for continuous variables:

$$\text{Euclidean Distance} = \sqrt{\sum_{i=1}^k (x_i - y_i)^2} \quad (2.3)$$

$$\text{Manhattan Distance} = \sum_{i=1}^k |x_i - y_i| \quad (2.4)$$

$$\text{Minkowski Distance} = (\sum_{i=1}^k (|x_i - y_i|^q))^{1/q} \quad (2.5)$$

Distance functions for categorical variables:

$$\text{Hamming Distance} = \sum_{i=1}^k |x_i - y_i| \quad \begin{cases} x = y \Rightarrow \text{Distance} = 0 \\ x \neq y \Rightarrow \text{Distance} = 1 \end{cases} \quad (2.6)$$

KNN algorithm is used to find k most similar trees in proposed approach DTreeSim in this thesis. Since these functions are not suitable to compute distance among decision trees to find tree similarity, a new distance measure was proposed in this thesis.

### 2.2.2 Association Rule Mining

This method is used to extract association rules. Association rules describe the relationships among the attributes in the dataset. In other words, the main purpose of association rule mining (ARM) is to find rules to predict the occurrence of an item based on the occurrences of other items in a given a set of transactions. This data mining technique was initially used for Market Basket Analysis to find itemsets purchased together by customers, initially.

In this thesis, association rule mining algorithm is applied on local municipality data in proposed decision maker system for different purposes. In ‘*Citizen Analysis*’ and ‘*Staff Analysis*’ scenarios ARM algorithm is applied to profile citizens and staffs according to demographic properties. In ‘*Moveable Material Analysis*’ and ‘*Accountancy Analysis*’ ARM is applied to facilitate and accelerate input operations.

In 'Income Operations Analysis', 'Logs Analysis' and 'User Account Analysis' scenarios ARM is applied to prevent wrong operations.

Association rules are in  $X \rightarrow Y$  [Support, Confidence] format. X is the examined data and Y is a discovered property for this examined data. X and Y are set of items and called itemset. X and Y are disjoint itemsets, i.e.  $X \cap Y = \emptyset$ .  $X \rightarrow Y$  association rule represents the pattern when X occurs, Y also occurs. Support is the percentage of the records satisfying X or Y, confidence is the percentage of the records satisfying both the X and Y to those satisfying only X. In other words, the rule  $X \rightarrow Y$  holds with support s, if s% of transactions in dataset contain  $X \cup Y$ , the rule  $X \rightarrow Y$  holds with the confidence c, if c% of the transactions in dataset that contain X also contain Y. Sample dataset for Association Rule Mining is given in Table 2.1.

$$\text{Support, } s(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N} \quad (2.7)$$

$$\text{Confidence, } c(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)} \quad (2.8)$$

where  $\sigma$  is support count, which is the number of transactions containing a particular itemset,  $N$  is the total number of transactions.

Table 2.1 An example dataset for Association Rule Mining

TID	Items
1	{ A, B }
2	{ A, C, D, E }
3	{ B, C, D, F }
4	{ A, B, C, D }
5	{ A, B, C, F }

Consider  $\{B, C\} \rightarrow \{D\}$  rule.

$$\sigma(B, C, D) = 2, N \text{ is } 5, \text{ so } s(\{B, C\} \rightarrow \{D\}) = \frac{2}{5} = 0.4$$

$$\text{Since } \sigma(B, C) = 3, c(\{B, C\} \rightarrow \{D\}) = \frac{2}{3} = 0.67$$

Well known Association Rule Mining algorithms are: Apriori Algorithm and FP-Growth (Frequent Pattern-Growth) Algorithm. “If an itemset is frequent, then all of its subsets must also be frequent” is the main principle of the Apriori Algorithm. Consider  $\{a, b, c\}$  is a frequent itemset. Any transactions that contains  $\{a, b, c\}$  must also contain its subsets,  $\{a, b\}$ ,  $\{a, c\}$ ,  $\{b, c\}$ ,  $\{a\}$ ,  $\{b\}$ ,  $\{c\}$ . Support of an itemset never exceeds the support of its subsets. This is known as the anti-monotone property of support. Figure 2.2 shows pseudo code of Apriori Algorithm.

1. Scan the transaction database to get the support of  $S$  each 1-itemset
2. Compare  $S$  with  $\text{min\_sup}$ , get a support of 1-itemsets,  $L_1$
3. Use  $L_{k-1}$  join  $L_{k-1}$  to generate a set of candidate  $k$ -itemsets.
4. Use Apriori property to prune the unfrequented  $k$ -itemsets from this set.
5. Scan the transaction database to get the support  $S$  of each candidate  $k$ -itemset in the find set.
6. Compare  $S$  with  $\text{min\_sup}$ , get a set of frequent  $k$ -itemsets  $L_k$ .
7. If candidate set does not equal to null, then go to step 3.
8. For each frequent itemset  $l$ , generate all nonempty subsets of  $l$ .
9. For every nonempty subset  $s$  of  $l$ , output the rule  $s \rightarrow (l - s)$ , if confidence  $C$  of the rule  $\geq \text{min\_conf}$

Figure 2.2 Pseudo code of Apriori algorithm.

The main disadvantage of the Apriori Algorithm is the number of dataset scan because of candidate generation operation. Another ARM algorithm FP-Growth solves this bottleneck by allowing frequent itemset discovery without candidate itemset generation. Since FP-Growth is faster than Apriori Algorithm, FP-Growth (Han, Pei, Yin & Mao, 2004) was selected as ARM algorithm in this thesis. There are two main steps in FP-Growth algorithm:

Step 1: FP-tree Building with two passes over the dataset by compressing a large dataset into compact structure.

First Pass:

Data is scanned to find support for each 1-itemset, then itemsets are sorted according to their supports to be used in building FP-tree.

Second Pass:

1. FP-Growth reads 1 transaction at a time and maps it to a path.
2. Fixed order is used, so paths can overlap when transactions share items.
3. Pointers are maintained between nodes containing the same item, creating singly linked lists.

Step 2: Extracting frequent itemsets from the FP-tree

### ***2.2.3 Sequential Pattern Mining***

Another data mining task, Sequential Pattern Mining (SPM) is applied to discover sequential and recurring structures or patterns. Applications of SPM include analysis of customer shopping sequences, medical treatments, natural disasters, weblog click streams, and DNA sequences. Unlike the previous studies, in this thesis, in DTreeSim, SPM was applied to decision trees, to determine compact rules for a decision tree to supply fast and correct comparisons for a similarity computation. First, sequence transactions were generated through paths from the top to the leaf of a tree, and then a SPM algorithm was applied to these sequence transactions to extract common patterns in the DT. These results (frequent sequences) were used to find similarity among DTs.

The Prefix-Span algorithm (Pei et al., 2001) was selected as the SPM algorithm in this thesis. Since the main idea of the Prefix-Span algorithm is to use the least projections, quickly shrinking the sequence, it is one of the fastest SPM algorithms. It reduces the effort involved in candidate subsequence generation. Prefix-Span is a pattern growth-based approach without candidate generation. Pseudo code of Prefix-Span algorithm (Fournier-Viger, Wu & Tseng, 2013) is shown in Figure 2.3.

```

PrefixSpan(a sequence database SDB, a threshold minsup, a prefix P initially set to
<>)
  Scan SDB once to count the support of each item
  For each item i with a support  $\geq$  minsup
    P' = Concatenate(P, i)
    Output the pattern P'
    SDBi = DatabaseProjection(SDB, i)
    PrefixSpan(SDBi, minsup, P')

```

Figure 2.3 Pseudo code of Prefix-Span algorithm.

Inputs of PrefixSpan algorithm are: sequence database ( $S$ ), minimum support ( $minsup$ ) and a prefix sequence  $P$  (it is empty at first.). Output of the algorithm is frequent sequential pattern set.

Firstly,  $S$  is scanned and each frequent single items ( $i$ ) are found. Then, for each frequent item ( $i$ ) sequential pattern  $\langle\{i\}\rangle$  is created and a projection of the  $S$  by  $i$  is performed to create a projected database  $S_i$ .

#### 2.2.4 Clustering

Clustering is the assignment of a set of observation into clusters so that observations in the same cluster are similar.

In this thesis, clustering technique was applied on local municipality data in proposed decision maker system. Clustering technique is used in '*Employee Analysis*' scenario to cluster employees in local municipality to make common decisions about employee and in '*Water Notice Analysis*' scenario to cluster citizens according to water consumption.

Clustering techniques typically depend on the knowledge of cluster number. If cluster number is not known, then an algorithm without this input parameter (i.e.

DBSCAN) can be applied or cluster validity indices can be utilized to determine the number of clusters in a dataset.

K-means algorithm (MacQueen, 1967) is widely used clustering algorithm and applied to have K clusters of given dataset. Cluster centres are selected randomly. Then distance between each data point and cluster centres are calculated. The data point is assigned to the cluster centre whose distance from the cluster centre whose distance from the cluster centre is minimum of all centres. After this operation, the new cluster centre and the distance between each data point is recalculated. It continuous until no data point is reassigned.

Drawback of K-means algorithm is that it is very sensitive to choice of initial cluster centres. K-means++ algorithm (Arthur & Vassilvitskii, 2007) is a seeding technique for k-means. Only first step (Cluster centres are selected randomly) of standard K-means algorithm changed in K-means++ algorithm. The main idea of K-means++ algorithm is spreading the k initial cluster centres away from each other. For that reason, K-means++ algorithm was selected as clustering algorithm in this thesis.

### ***2.2.5 Outlier Detection***

Outliers are data objects which are inconsistent with other normal data objects and general behaviour in data set. Outlier detection is a data mining technique which is used to find outliers and to detect anomalous observations in dataset.

In this thesis, outlier detection is applied on local municipality data in proposed decision maker system. Outlier detection is used in '*Electricity Consumption Analysis*', '*Fuel Oil Analysis*' scenarios to detect outliers by analysing consumptions and in '*Cash Desk Analysis*' scenario to analyse cash desk operations performed by cashier to find outliers.

Local Outlier Factor (LOF) algorithm [Breunig, Kriegel, Ng & Sander, 2000] is used as outlier detection algorithm in this thesis.



### 2.3 Application Areas

Whereas data mining is used in many areas, such as sales forecasting (Kumari, Prasad & Bala, 2013), stock market analysis (Fiol-Roig, Miro-Julia, & Isern-Deya, 2010), medical treatment (Thangamani, Thangaraj & Bannari, 2013), earthquake prediction (Souza & Wang, 2010), and weather forecasting (Olaiya & Adeyomo, 2012), it has not been extensively used in local municipalities. There are a few studies where data mining has been used for local municipalities (Solomon, Nguyen, Liebowitz & Agresti, 2006; Spielman & Thill, 2007). This thesis addresses this lack by applying a novel approach to large datasets collected by a local municipality to investigate citizens' tax payment behaviors. In this thesis, citizen behavior analysis was carried out to predict future income by analysis monthly-based tax returns for a local municipality in Turkey. Because, the prediction of future tax payments may help local governments to manage risk and budget status.

### 2.4 Challenges of Data Mining in Local Municipalities

Challenges of data mining in local municipalities can be classified into four main titles: Data Preprocessing, Big and Complex Data, Task and Algorithms and finally Post Processing.

- *Data Preprocessing:*

It is an important step to ensure the data quality and to improve the efficiency. Because real-world local municipality data tend to be incomplete, inconsistent and noisy. In addition, it is necessary to convert the local municipality data into suitable forms for mining.

- Data Quality
- Data Ownership and Distribution
- Privacy Preservation

- *Big and Complex Data:*

It is necessary to deal with huge, high dimensionality and complexity of the local municipality data.

- Scalability
- Dimensionality
- Heterogeneous Data

- *Tasks and Algorithms:*

It is the essential steps of knowledge discovery. Generally challenges in tasks and algorithms:

- (a) Determine a purpose
- (b) Determine an appropriate data mining algorithm to get efficient results

- *Post processing:*

It is the process to refine and evaluate the knowledge derived from mining procedure. Generally challenges in post-processing:

- (a) how to evaluate the discovered patterns
- (b) how to present the mining results to the users in a way that is easy to understand and interpret
- (c) how to convert the discovered patterns into knowledge

## **CHAPTER THREE**

### **RELATED WORKS**

#### **3.1 Literature Review**

Related works to our study can be classified into five groups: (i) studies which investigated the similarity of DTs, (ii) studies which combined SPM with other data mining techniques, (iii) studies which proposed new distance formulas for KNN, (iv) studies which used re-mining and (v) studies which include data mining results and decision support for local municipalities.

#### **3.2 Studies for Similarity of DTs**

The first group includes recent studies which have looked at the similarity of DTs from different perspectives. Ntoutsis, Kalousis & Theodoridis (2008) presented a general framework for similarity estimation that includes, as a special case, the estimation of semantic similarity between DTs. The case of their proposed approach is the similarity of two decision trees  $DT_1$  and  $DT_2$  can be measured with respect to their predictions. This is a measure of their semantic similarity, that is how similar are the concepts described by the DTs, and corresponds to the percentage of times that they produce the same predictions on instances drawn from a given attribute space distribution. In other words the decision tree semantic similarity measure proposed by them, depends on the estimation of the distribution that governs the attribute. Zhang & Jiang (2012) developed splitting criteria based on similarity. Their approach supplies similarity based on the decision tree generation algorithm and a pruning methodology. Islam, Barnaghi & Brankovic (2003) created another technique to find the similarity in DTs by comparing their accuracy rates. Syntactic similarity of DTs has been investigated using a graph algorithm for tree matching (Last, Maimon & Minkov, 2002). Another method identifies the number of same/different symbols for the same class, attribute, and attribute-value in DTs (Yoshida, 2008). In some studies (Fletcher & Islam, 2015), the similarity of DTs was measured by the similarity of logic rules associated with the DTs.

We can classify previous studies that tried to find similarity between two different DTs into two sub-categories: semantic and structural. While semantic similarity (Ntoutsi, Kalousis & Theodoridis, 2008) measures the common feature subspace covered by two DTs for particular decision classes, structural similarity (Peahringer & Witten, 1997; Perner, 2011, 2013) compares two DTs structurally, focusing on their nodes, branches, total number of leaves, and tree depth. Dogra (2014) provides a more complete similarity by taking into account both these aspects.

In this thesis, we compared our approach (DTreeSim) with two existing methods: DTSim and  $\text{Sim}_{d1,d2}$ . The first one (DTSim) was proposed by Peahringer & Witten (1997). This function compares the set of all abstract paths, which are the multisets of attributes tested along a specific path from the root to leaf, along with the class assigned to that leaf. They ignored values of attributes and focused only on attribute names. The second method ( $\text{Sim}_{d1,d2}$ ) was proposed by Perner (2011; 2013). He compared trees using decomposition rules and substructures. This method did not include leaf values or class labels.

The work done by Pekerskaya, Pei & Wang (2006) deals with the problem of detecting the difference between two datasets using decision tree comparisons. However, they developed a method to compare two cluster-embedded DTs. Other techniques measure the similarity of an original and a perturbed dataset by evaluating the similarity of DTs (Fletcher & Islam, 2015; Islam, 2008). However, these studies focus on privacy preservation in data mining through noise addition.

Some studies (e.g., (Papagelis & Kalles, 2000)) estimate the similarity of different DTs using a simple formula based only on the differences between the number of nodes and tree levels. For example, in (Papagelis & Kalles, 2001), the similarity of DTs was estimated using the simple formula:  $\text{tree\_diff} = |(\text{levels\_tree1} - \text{levels\_tree2}) + (\text{nodes\_tree1} - \text{nodes\_tree2})|$ . Another study that compares similarities between trees using a simple formula was proposed by Telaar & Fuhs (2013). They measure the fraction of joint entropy that is not mutual:  $\text{DIFF}(T1, T2) = 1 - I(T2; T1)/H(T1, T2)$ , where  $H(T1, T2)$  is the joint entropy of the distribution over all

classes in trees  $T1$  and  $T2$ , and  $I(T2; T1)$  corresponds to the mutual information on  $T1$  and  $T2$ . In contrast to our work, they use an entropy measure to find polyphone decision tree similarity. They also propose using dissimilarity, instead of similarity. In this case, if  $\text{DIFF}(T1, T2) = 0$ , then the trees are identical; while, if  $\text{DIFF}(T1; T2) = 1$ , the trees are not correlated. In contrast to these studies, we chose not to use a simple formula to determine the similarity between two DTs; instead we evaluate it using two data mining algorithms: the Prefix-Span and KNN. Some studies only consider the attributes of the trees during comparison; they ignore the actual values of attributes and their numerical/categorical cut-points. However, the approach proposed in this thesis considers all of these aspects.

A tree is an undirected graph. Zager & Vergheese (2007) outline a class of graph similarity measures. Structural similarity of local neighbourhoods is used in this class to derive pairwise similarity scores for the nodes of two different graphs. Dehmer, Emmert-Streib & Kilian (2006) created an algorithm to measure the structural similarity of generalized trees. They transform graphs into property strings, and graph similarity problem is reduced into a string similarity problem. Based on isomorphic relations, the time complexity of their algorithm is polynomial and so better than classical graph similarity methods. Zhao, Xiao, Lin & Wang (2012) inspired by the q-gram idea for string similarity problem and they extract paths from graphs as features for indexing.

### **3.3 Studies for SPM with Other DM Techniques**

The second related work group includes the studies in which SPM has been combined with other data mining techniques. For example, Ma, Tang, Yang, Wang & Han (2004) combined SPM with clustering to mine moving sequential patterns. Exarchos & Papaloukas (2008) used SPM for the classification of proteins into folds. Another example, Tseng & Lee (2009) integrated SPM with probabilistic induction to discover hidden patterns and classify data using re-mined sequential patterns. D'Silva & Vora (2014) used sequential pattern mining with clustering to create an intelligent recommendation system. Deng, Runger, Tuv & Bannister (2014) used association rules with decision trees and proposed a new tree model, "condition-

based tree (CBT)” and a new algorithm, “condition-based classifier (CBC)”. Unlike these previous studies, we applied SPM to decision trees and we used it for different purpose (to find similarity).

### **3.4 Studies for KNN with New Distance Formulas**

Another group of studies proposed new distance formulas for KNN, although the Euclidean distance formula is usually employed in the KNN method to measure similarity. For example, Ma & Kaban (2013) explored a simple yet effective similarity definition within nearest neighbors for intrusion detection applications. Similarly, our study also proposes a new similarity measure for the k-nearest neighbor algorithm, but for different purpose, to find the k-most similar decision trees to a specific tree.

### **3.5 Studies for Re-Mining**

The other related work group includes the studies which applied re-mining. Re-mining is a technique that applies two data mining techniques, one after another. Liu & Yin (2001) presented an efficient data re-mining technique for updating previously discovered association rules in the light of threshold changes. Demiriz, Ertek, Atan & Kula (2011) proposed a new approach to mine price, time, and domain-related attributes through re-mining of association mining results. The underlying factors behind positive and negative relationships were characterized and described in a second data mining stage. Previous studies that include re-mining used this process for filtering results. In contrast to these studies, we use re-mining twice (re<sup>2</sup>mining), and we use it for different purpose (for determining decision tree similarity).

### **3.6 Studies for Decision Support**

Last group of studies include decision support and data mining solutions for local municipalities. Teixeira et al. (2015) presented a case study and the development of a business intelligence functional solution implemented in local government, providing support and improving the quality of processes in their paper. Adalakun (2012)

explained the business intelligence role in government with a case study of a Swedish municipality contact center. Poles & Margorani (2009) developed a data mining tool which is called modeFrontier for Italian municipalities. Ahmadvand et al. (2010) applied a data mining framework on the database of Tehran municipality. Song & van der Aalst (2008) presented new process mining techniques but also use existing techniques in an innovative manner. Their approach had been implemented in the context of the ProM framework and has been applied in various case studies. They demonstrated the applicability of their techniques by analysing the logs of a municipality in the Netherlands.

## CHAPTER FOUR

### PROPOSED APPROACH: DTREESIM

One of the main purpose of this thesis is to find the most similar trees among hundreds of DTs. The motivation behind this purpose is to analyze factors that make these trees similar and to make predictions for situations with similar factors. To achieve this goal, three data mining techniques, classification, SPM, and the KNN algorithm, are combined.

Figure 4.1 shows the five main steps, involving two re-mining operations used in DTreeSim (Bakirli & Birant, 2015). First, data marts are created by partitioning a data warehouse for specific conditions. Second, a decision tree algorithm is applied to each data mart. Third, the paths of decision trees are linearized into sequences. Fourth, a sequential pattern mining algorithm is applied to each tree sequences to find all frequent sequences. Last, the k-nearest neighbor algorithm is applied to the frequent sequence set to find the most similar trees.

Constraints for determining similarity of decision trees are; common dataset  $D$ , the same attributes  $A = (a_1, a_2, \dots, a_u)$ , each attribute  $a_i$  has domain,  $d(a_i)$  and a class attribute  $C$ , and domain of the class attribute is  $d(C) = (c_1, c_2, \dots, c_m)$ , where  $m$  is the number of classes.



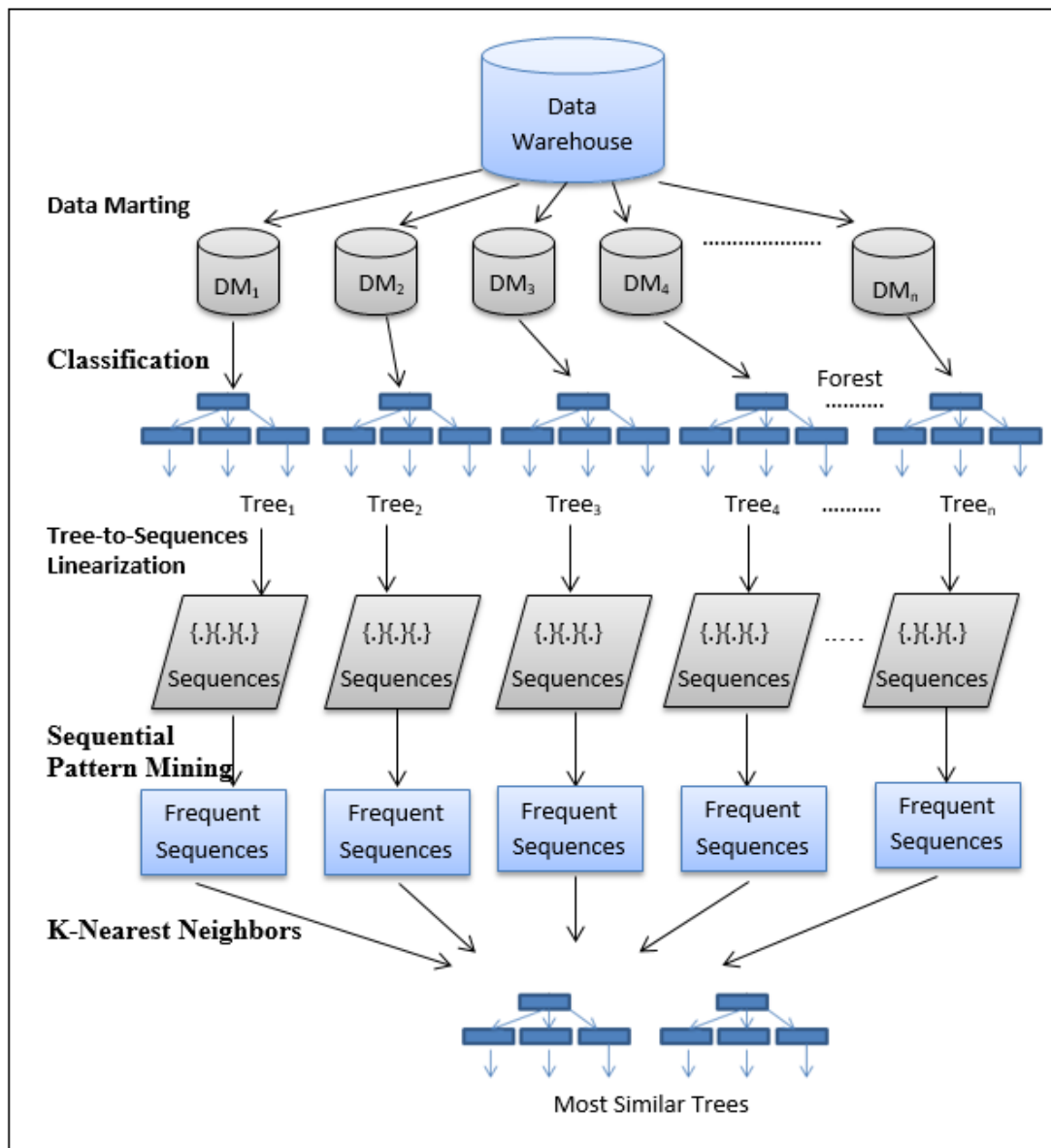


Figure 4.1 Schematic showing the DTreeSim approach.

#### 4.1 Step 1: Data Marting

Data mining has traditionally operated on data in a data warehouse or data mart. A *data warehouse* (DW) is a collection of integrated data from one or more different sources, while a *data mart* (DM) is a subset of the DW created for a specific purpose. In other words, a DW supplies a general view of a problem, whereas a DM provides a specific view for a particular problem.

As shown in Figure 4.2, ETL (Extract, Transform, Load) includes three main processes;

*Extract* : Data is extracted from different data sources (ERP (Enterprise Resource Planning), MIS (Management Information System), CRM (Customer Relationship Management)... etc).

*Transform* : Data is transformed into proper format.

*Load* : Transformed data is loaded into datamart.

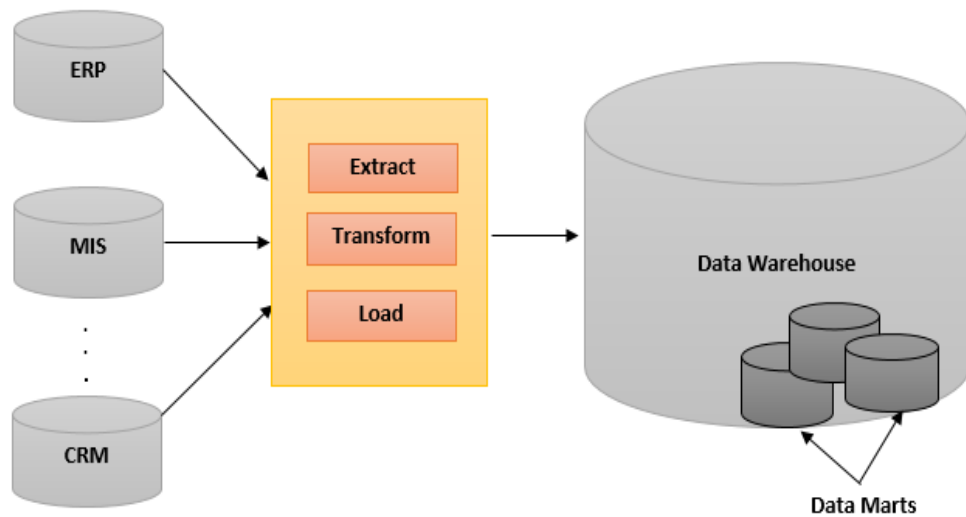


Figure 4.2 Data marting.

It is possible to create subject, region, or time-oriented DMs from a DW. DMs are created by filtering or partitioning a DW for specific conditions. Partitioning criteria, such as subjects, regions or time intervals, should be determined to split a DW into smaller subsets. The comparison of data warehouse and data mart is given in Table 4.1.

Table 4.1 Data warehouse vs. data mart

	<b>Data Warehouse</b>	<b>Data Mart</b>
<b>Scope</b>	Centralized	Decentralized by specific area
<b>Data</b>	Lightly denormalized	Highly denormalized
<b>Subjects</b>	Multiple subjects	One specific subject
<b>Sources</b>	Many sources	Few sources
<b>Other Characteristics</b>	Data-oriented	Subject-oriented
	Long life	Short life
	Large	Small

*Data marting* is the process of partitioning a DW into smaller subsets to provide a specific view of the data. It has the following constraints:

$$DW = \cup_{i=1}^n DM_i \quad (4.1)$$

$$DM \subset DW \quad (4.2)$$

where  $n$  is the total number of DMs, and each DM is a subset of DW for a specific criterion.

## 4.2 Step 2: Classification

The C4.5 decision tree algorithm in Weka is applied to each DM. At the end of this step, a forest (F) with  $n$  DTs has been created.

*Forest* (F) is defined as

$$F = \coprod_{i=1}^n T_i \quad (4.3)$$

where  $n$  is the total number of DMs as well as the number of DTs;  $F$  is a set of  $n$  disjoint trees, and  $T$  is a tree.

### 4.3 Step 3: Tree-to-sequence Linearization

The paths of a decision tree are linearized and converted into sequences of branches. At the end of this step, branch sequences (BSs) are obtained from a DT and tree sequences (TSs) are acquired from various DTs.

A *branch sequence* is a sequence that is constructed along a specific path from the tree root to a given leaf, along with the class assigned to that leaf.

A *tree sequence* is the set of branch sequences related to a tree.

*Tree-to-sequence linearization* (TSL) proceeds as follows:

$$TS = \left\{ \begin{array}{l} BS_1 = \langle ([r] = [v_{r1}])([in_1] = [v_{11}])([in_2] = [v_{21}]) \dots \dots \dots ([in_p] = [v_{p1}])([l_1]) \\ BS_2 = \langle ([r] = [v_{r2}])([in_1] = [v_{12}])([in_2] = [v_{22}]) \dots \dots \dots ([in_p] = [v_{p2}])([l_2]) \\ \vdots \end{array} \right\} \quad (4.4)$$

where [r] is the root of the decision tree; [in<sub>p</sub>] is the p<sup>th</sup> internal node in the related branch; [l] is the leaf node (class); and [v] is the value of the related node (a connection value between the related node and the next node). Each node and value pair ([in<sub>1</sub>] = [v<sub>1</sub>]) is a branch, while each path of a tree is assigned as a sequence of branches. Thus, a TS is constructed from a set of BSs.

Each BS in the TS for related trees starts with [r], and ends with a [l]. There should be only one [r], but there can be multiple [l]s. In this case, depth denotes the number of edges from the tree root node to the leaf node. Thus, there are Depth (l) – 1 internal nodes and value pairs in a BS.

Figure 4.3 gives an example of a tree structure. According to the example, first BS is:

$$BS_1 = \langle ([r] = [v_{r1}])([in_1] = [v_{11}])([in_2] = [v_{21}])([l_1]) \rangle \quad (4.5)$$

where  $[r] = \text{Age}$ ,  $[v_{r1}] = [\leq 35]$ ,  $[in_1] = [\text{Gender}]$ ,  $[v_{11}] = [M]$ ,  $[in_2] = [\text{Tax Type}]$ ,  $[v_{21}] = [\text{Real Estate}]$ ,  $[l_1] = [\text{Paid}]$ . Depth ( $l_1$ ) = 3, hence there are two internal nodes in  $BS_1$ ;  $[\text{Gender}]$  and  $[\text{Tax Type}]$ . There is one root  $[\text{Age}]$  with two values ( $\leq 35$ ,  $> 35$ ) and there are two leaves, class values at the same time;  $[\text{Paid}]$ ,  $[\text{Not Paid}]$ . There is a total of 5 branches:

$$BS_1 = \langle ([\text{Age}] = [\leq 35])([\text{Gender}] = [M])([\text{Tax Type}] = [\text{Real Estate}])([\text{Paid}]) \rangle$$

$$BS_2 = \langle ([\text{Age}] = [\leq 35])([\text{Gender}] = [M])([\text{Tax Type}] = [\text{Envrn.}])([\text{Not Paid}]) \rangle$$

$$BS_3 = \langle ([\text{Age}] = [\leq 35])([\text{Gender}] = [F])([\text{Paid}]) \rangle$$

$$BS_4 = \langle ([\text{Age}] = [> 35])([\text{Gender}] = [F])([\text{Paid}]) \rangle$$

$$BS_5 = \langle ([\text{Age}] = [> 35])([\text{Gender}] = [M])([\text{Not Paid}]) \rangle$$

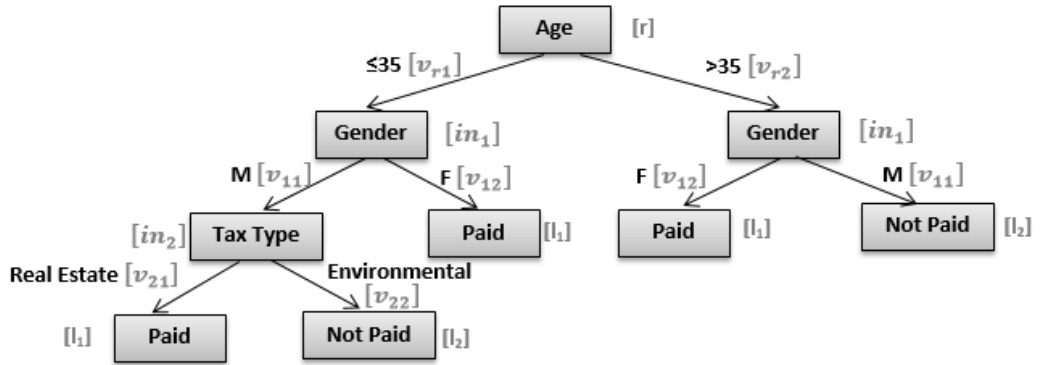


Figure 4.3 An example tree structure related to municipal data used in this thesis.

After this step, each attribute and value pair is coded with a unique integer value to apply the Prefix-Span algorithm in the SPM Framework (SPMF) (Fournier-Viger et al.) to these data. For example, the pair  $([r] = [v])$  is coded as “1” and  $([in_1] = [v_1])$  is coded as “2”. According to tree in Figure 4.3 distinct pairs and their new codes are shown in Table 4.2.

Table 4.2 Sample attribute value pair coding for SPMF

Pair	Code
[Age] = [≤ 35]	1
[Gender] = [M]	2
[Tax Type] = [Real Estate]	3
[Paid]	4
[Tax Type] = [Envrn.]	5
[Not Paid]	6
[Gender] = [F]	7
[Age] = [> 35]	8

After coding operation BSs for tree in Figure 4.3 are as below.

$$BS_1 = \langle (1)(2)(3)(4) \rangle$$

$$BS_2 = \langle (1)(2)(5)(6) \rangle$$

$$BS_3 = \langle (1)(7)(4) \rangle$$

$$BS_4 = \langle (8)(7)(4) \rangle$$

$$BS_5 = \langle (8)(2)(6) \rangle$$

#### 4.4 Step 4: Sequential Pattern Mining

In this step, the Prefix-Span algorithm is applied to each TS to find all frequent sequences (FSs). The SPM's role is a little different in this approach to classical SPM tasks. This step provides a compact rule set, which represents the DT ideally. Frequent sequences (FSs) are created for each TS by the SPM algorithm. Since the SPM is carried out on the results of the previous data mining process, this operation is clearly re-mining. Before applying the SPM, a preprocessing step involving TS clustering is necessary for accurate results.

*Tree sequence clustering* is a preprocessing step before the SPM task, in which the sequences in the TS are grouped according to their class labels. This step is necessary, because the SPM algorithm may ignore class labels in the sequences, leading to errors in identifying similarity.

For example, when there are two TSs:

$$\begin{aligned}
 TS_1 &= \{BS_1 = \langle ([Age] = [\leq 35])([Gender] = [M])([Tax Type] = [Real Estate])([Paid]) \rangle \} \\
 &\quad \dots \\
 TS_2 &= \{BS_1 = \langle ([Gender] = [M])([Tax Type] = [Real Estate])([Not Paid]) \rangle \} \\
 &\quad \dots
 \end{aligned}$$

The SPM can create the following frequent sequences:

$$\begin{aligned}
 & \text{Frequent Sequence for } TS_1 \\
 &= \{ \langle ([Gender] = [M])([Tax Type] = [Real Estate]) \rangle \} \\
 & \text{Frequent Sequence for } TS_2 = \{ \langle ([Gender] = [M])([Tax Type] = [Real Estate]) \rangle \}
 \end{aligned}$$

Hence, if there are multiple BSs in the TSs, which include  $([Gender] = [M])$   $([Tax Type] = [Real Estate])$  items with other distinct items like  $([Not Paid])$ ,  $([Paid])$ ,  $([Age] = [>35])$ , then the SPM eliminates other items and keeps only  $([Gender] = [M])$   $([Tax Type] = [Real Estate])$  as a FS. In this case, these two FSs are equal to each other and their similarity is 100%. However, the actual class labels for these sequences are completely different; one is labelled as  $([Paid])$ , while the other one is  $([Not Paid])$ . It is clear that class labels in the sequences affect the underlying similarity of DTs. In other words, class nodes in a tree have a decisive role, unlike internal nodes and the root. Thus, class labels should be protected during this SPM step. To do this, TSs related to a DT are grouped into  $m$  clusters as follows:

$$TS = \bigcup_{x=1}^m TS_x \quad (4.6)$$

where  $m$  is the number of classes in the dataset. For example, in the above case, there are two class labels: “Paid” and “Not Paid”. Each TS is subdivided into two groups: one of them ( $TS_{\text{paid}}$ ) includes all BSs that end with  $([Paid])$ , while the other

( $TS_{\text{notpaid}}$ ) contains all sequences that end with (*[Not Paid]*). The SPM is applied to each sub-TS, creating sub-FSs. After this step, all sub-FSs should be compared with other sub-FSs with the same class label to calculate their similarity (Pieced Similarity).

*Pieced similarity* is a similarity measure for two FSs that is calculated by adding sub-FS similarities:

$$\text{Sim}(FS_1 + FS_2) = \text{Sim}(FS_{1c_1}, FS_{2c_1}) + \text{Sim}(FS_{1c_2}, FS_{2c_2}) + \dots + \text{Sim}(FS_{1c_m}, FS_{2c_m}) \quad (4.7)$$

where *Sim* stands for similarity and  $m$  is the number of classes, such as  $c_1, c_2, \dots, c_m$ .

#### 4.5 Step 5: Similarity Calculation and K-Nearest Neighbor Algorithm

The KNN algorithm is applied to the FS set to find the most similar  $k$  trees to a specific tree using the similarity measure. The KNN algorithm is executed many times to compare all trees with all others. This step also involves re<sup>2</sup>mining, because the KNN algorithm is applied to the results of the previous re-mining process.

Two main parameters affect the similarity of frequent sets: intersection and the length of the sets.

In pieced similarity:

$$\text{intersection}_{i,j} = \sum_{x=1}^m \#(FS_{ic_x} \cap FS_{jc_x}) \quad (4.8)$$

In general similarity:

$$\text{intersection}_{i,j} = \#(FS_i \cap FS_j) \quad (4.9)$$

where  $m$  is the number of classes. The  $\text{intersection}_{i,j}$  gives the common frequent sets between two FSs ( $FS_i$  and  $FS_j$ ).



Similarity metric to determine intersection<sub>i,j</sub> is *reflexivity* (Theodoridis & Koutroumbas, 2009).

**Reflexivity:**  $S(X, Y) = S_0$  if and only if  $\{X = Y\}$ .  $X$  and  $Y$  are FSs,  $S_0$  is the largest similarity measure between all possible  $X$  and  $Y$ .  $S$  is the similarity measure.

At first glance, the intersection<sub>i,j</sub> seems to be adequate to identify similarity. However, the length of the FS sets affects similarity as well. Thus, the similarity of two sets is directly proportional to their intersection<sub>i,j</sub> and inversely proportional to the maximum set length.

Given  $FS_1$  and  $FS_2$ , where the intersection<sub>1,2</sub> = 25, as well as  $FS_3$  and  $FS_4$ , where the intersection<sub>3,4</sub> = 15, it may appear that  $FS_1$  and  $FS_2$  are more similar than  $FS_3$  and  $FS_4$ . However, if  $\text{Max}(|FS_1|, |FS_2|) = 75$  and  $\text{Max}(|FS_3|, |FS_4|) = 25$ , it is clear that  $FS_3$  and  $FS_4$  are more similar. Thus, we defined similarity as follows:

$$Sim(FS_i, FS_j) = 100 \times \frac{\sum_{x=1}^m \frac{\#(FS_{ic_x} \cap FS_{jc_x})}{\text{Max}(|FS_{ic_x}|, |FS_{jc_x}|)}}{m} \quad (4.10)$$

where  $FS_i$  is the  $i^{\text{th}}$  FS,  $FS_j$  is the  $j^{\text{th}}$  FS, and  $m$  is the number of classes.

Sim is a measure in the range [0,100].

$$\begin{cases} Sim(FS_i, FS_j) = 100 \text{ if } FS_i = FS_j \\ Sim(FS_i, FS_j) = 0 \text{ if } FS_i \cap FS_j = 0 \end{cases}$$

Figure 4.4 shows the pseudo code and Figure 4.5 shows flowchart for our proposed algorithm DTreeSim, consisting of the five steps described above. To ensure the correctness of the algorithm, we tested it on two identical decision trees ( $DTreeSim(DT_1, DT_1) = 100\%$ ) and two completely different decision trees ( $DTreeSim(DT_1, DT_2) = 0\%$ ). Our proposed method has a maximum time complexity of  $O(n \times \log n)$  based on the complexities of the C4.5 ( $n \times a \times \log n$ ), PrefixSpan  $O(n \times L)$  and the KNN with kdtree ( $n \times \log n$ ) algorithms, where  $a$  is the number of

attributes,  $n$  is the number of instances, and  $L$  is the maximum length of all transactions.

```

// Input: Data Warehouse (DW)
// Output: The most similar  $k$  tree pairs
Determine partitioning criteria (subject, region or time) for DW
// STEP1: Data Marting
For each transaction T in DW
    Add T to the related Data Mart (DM)
End for
// STEP2: Classification
For each DM in DW
    Apply C4.5 Classification, create a decision tree DT from DM
    Add DT to Forest F
End for
// STEP3: Tree-to-Sequence Linearization (TSL)
For each decision tree DT in F
    For each path P in DT
        Create a Branch Sequence (BS) from P
        Add BS to Tree Sequence (TS) of DT
    End for
End for
// STEP4: Sequential Pattern Mining
For each TS
    For each BS in TS
        Convert BS into SPMF form
    End for
    Apply Prefix-Span Algorithm to TS, create frequent sequences (FS)
    Add frequent sequences into FS set
End for
// STEP5: K-Nearest Neighbors
Determine k value
array[] maxSimilarityValues{};
array[,] similarPairs {{},{}}
For i = 1 to n // n is the number of trees
    For j = i+1 to n
        similarity = Calculate similarity between FSi, FSj
        If (maxSimilarityValues.Length < k)
            push similarity in maxSimilarityValues
            push {FSi,FSj} in similarPairs
        Else If (similarity > min(maxSimilarityValues) then
            maxSimilarityValues[index of min(maxSimilarityValues)] = similarity
            similarPairs[index of min(maxSimilarityValues)] = {FSi,FSj}
        End if
    End for
End for
Return similarPairs

```

Figure 4.4 Outline of the DTreeSim algorithm showing the five main steps.

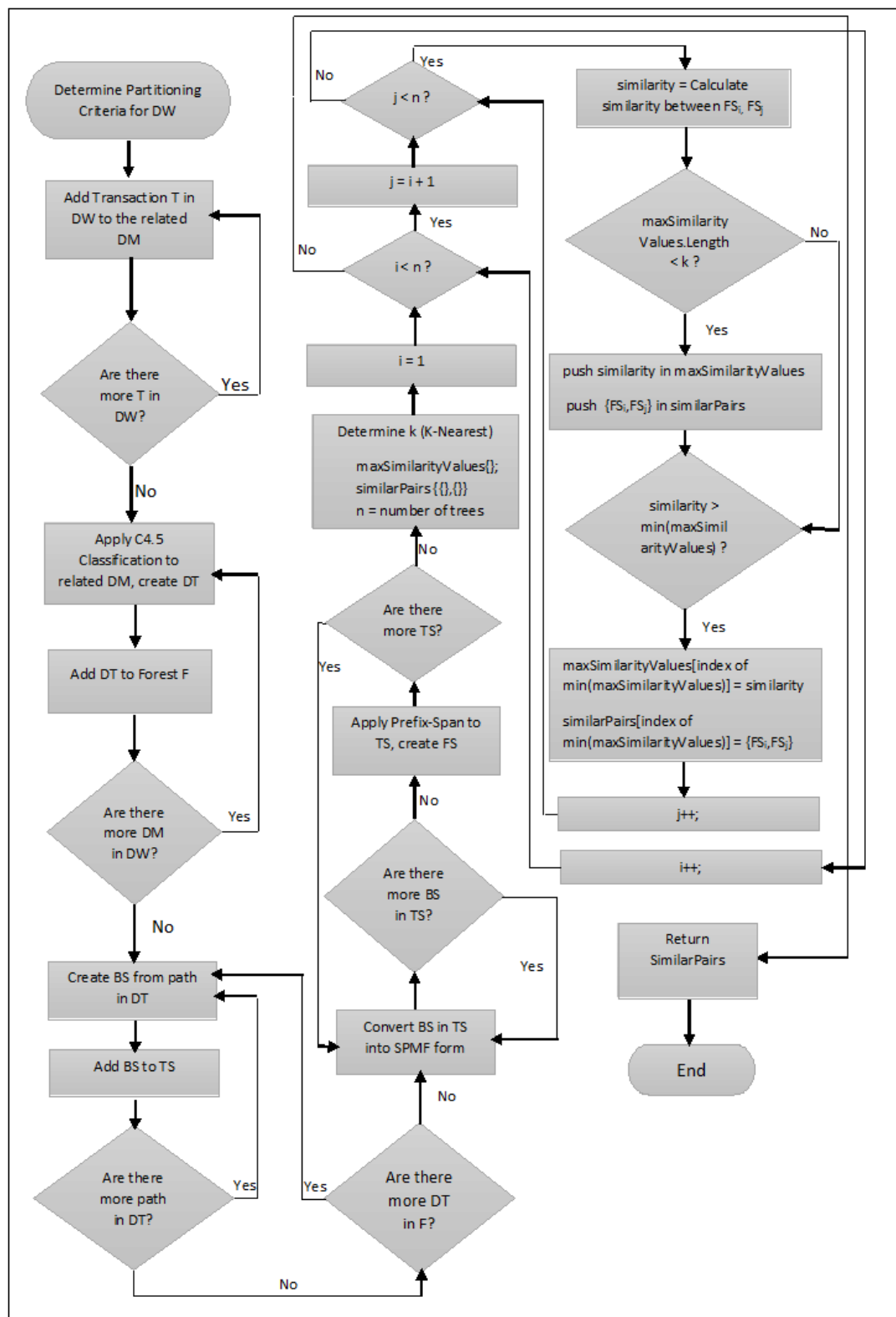


Figure 4.5 Flowchart of the DTreeSim algorithm.

## CHAPTER FIVE

### EXPERIMENTAL RESULTS

#### 5.1 Dataset Description

For the purpose of testing the accuracy of DTreeSim, the tax payments for citizens of a municipality in Turkey were analyzed over 10 years, between 2003 and 2013. Data preparation operations were applied to have dataset for the purpose.

##### 5.1.1 Data Preparation

The purpose of data preparation operations is to have the final suitable dataset for modelling. Data preparation operations included data reduction, integration and discretization.

###### 5.1.1.1 Data Reduction

Data reduction is a process which is used to reduce the volume of data. Data reduction is applied on four database tables for the proposed scenario: (i) income accrual, (ii) city citizens, (iii) real estate declaration details and (iv) city streets.

###### 5.1.1.2 Data Integration

Data integration is a process used to combine data from disparate sources into valuable information. In this thesis, four tables from three different schemas were integrated to have dataset; Accrual Table (Table 5.1) in Income Schema, Citizens Table (Table 5.2) in City Schema, Declaration Details Table (Table 5.3) in Real Estate Schema and Streets Table (Table 5.4) in Street Shema.

Table 5.1 Income accrual database table

<b>Income Accrual</b>
<b>Late_Payment_Penalties</b>
<b>Citizen_Id</b>

Table 5.1 Income accrual database table (Cont.)

<b>Income Accrual</b>
<b>Declaration_Id</b>
<b>Declaration_Detail_Id</b>
<b>Canceled_Id</b>
<b>Income_Type_Code</b>
<b>Year</b>
<b>Term</b>
<b>Expiry_Date</b>
<b>Status_Code</b>

Table 5.2 City citizens database table

<b>City Citizens</b>
<b>Id</b>
<b>Date_Of_Birth</b>
<b>Gender_Code</b>
<b>Marital_Status_Code</b>
<b>Type_Code</b>
<b>Region_Id</b>
<b>Year</b>
<b>Term</b>

Table 5.3 Real estate declaration details database table

<b>Real Estate Declaration Details</b>
<b>Id</b>
<b>Declaration_Id</b>
<b>Street_Id</b>

Table 5.4 City streets database table

<b>City Streets</b>
<b>Id</b>
<b>Description</b>

The following Structure Query Language (SQL) was created to have suitable dataset.

```

SELECT C.Gender_Code, C.Marital_Status_Code, C.Region_Id, M.Description,
T.Income_Type_Code, Year (CURRENT Date)-Year (DOGUM_TARIHI) AS Age,
K.Job_Id, K.Children_Number
      SUM ( CASE WHEN  Acquittance_Id=0 THEN 1 ELSE CASE WHEN
Late_Payment_Penalties>0  THEN  1  ELSE  0  END  END) AS
Delayed_Payments_Number,
      SUM (CASE WHEN  Acquittance_Id =0 THEN 0 ELSE CASE WHEN
Acquittance_Id >0 THEN 0 ELSE  1 END END) AS OnTime_Payments_Number
FROM ((Income.Accruals T INNER JOIN City.Citizens C ON T.Citizen_Id =
C.Id)
INNER JOIN RealEstate.Declaration_Details BD ON (T.Declaration_Detail_Id =
BD.Id) AND (T.Declaration_Id = BD.Declaration_Id)) INNER JOIN City.Streets M
ON BD.Street_Id = M.Id
WHERE T.Status_Code <> 'P' AND T.Canceled_Id = 0 AND
T.Expiry_Date<CURRENT Date AND C.Type_Code='R' AND C.Date_Of_Birth
Is Not Null
GROUP BY  C.Gender_Code,  C.Marital_Status_Code,  C.Region_Id,
M.Description,  T.Income_Type_Code,  Year  (CURRENT  Date)  -  Year
(Date_Of_Birth), C.Job_Id, C.Children_Number;

```

Delayed\_Payments\_Number and OnTime\_Payments\_Number are obtained after executing SQL query. The proportions of these numbers are used to have “Paid” and “Not Paid” class values.

### *5.1.1.3 Data Discretization*

This operation is used to make discretize continuous numeric values to have categorical values. Age attribute had numeric values, to make this attribute usable in

decision tree algorithm, discretization is applied and five different categorical values were obtained. Border points were determined to supply normal distribution.

$$\leq 20, (20 - 35], (35 - 50], (50 - 65], > 65$$

After data reduction, integration and discretization steps, a data warehouse is constructed from a huge database. Several database tables were joined to form the citizens' tax payments data, which include nearly 720,000 instances, with nine attributes, as shown in Table 5.5.

Table 5.5 Basic dataset descriptors for local municipality data

<b>Attribute</b>	<b>Attribute Values</b>
<b>Gender</b>	male, female, indefinite
<b>Marital Status</b>	single, married, widowed, indefinite
<b>Region (in identity card)</b>	Aegean, Central Anatolia, Marmara, Black Sea, Eastern Anatolia, Southern Anatolia, Mediterranean, Abroad
<b>District</b>	D1, D2, D3
<b>Income Type</b>	environmental sanitation tax, real estate tax, water consumption, entertainment tax, etc.
<b>Age</b>	$\leq 20, (20 - 35], (35 - 50], (50 - 65], > 65$
<b>Job</b>	public, private sector, unemployed, retired
<b>Children Number</b>	0,1,2,3,4+
<b>Class</b>	Paid, Not Paid

In this experimental study, time-oriented DMs were created to investigate similarities through time. Citizens' tax payments for the period 2003 to 2013 were partitioned by month, and each month was stored as a DM. After partitioning the DW, 12 DMs were created for each year, yielding 120 DMs for the 10-year study

period. Thus, the DW stores data for all 10 years, while the DM contains only monthly data.

The purpose of this experimental study was to predict the potential income of the local municipality in advance. For example, if citizens' tax payment behaviors are similar for the months of January and July each year, then we can predict the income for the next July by the end of this January. The same analysis can be applied on region-oriented DMs. In this situation, we can find most similar k regions for citizen behaviors and we can analyze factors that make similar these regions. For example, if the citizens live in similar regions make their tax payments on time, the municipality can reward these regions by serving more services. If these regions are similar since the citizens in these regions don't pay their tax payments then the municipality can make campaign in these regions to encourage them to pay their tax payments on time.

## 5.2 Application of DTreeSim Step by Step

### 5.2.1 Application of Data Marting

Data warehouse was partitioned according to year and month values, to have data marts. Sample data mart for related dataset (local municipality data) is shown in Table 5.6.

Table 5.6 Sample data mart for August 2003

Gender	Marital Status	Region	District	Income Type	Age	Job	Chi Id#	C
M	M	Mediterranean	D1	EST	(35–50]	U	2	Paid
F	M	Aegean	D1	RET	≤ 20	P	0	Not Paid
F	S	Aegean	D2	EST	(20–35]	PR	1	Paid



Abbreviations in dataset:

Gender = M (Male), F (Female)

Marital Status= M (Married), S (Single)

Income Type = EST (environmental sanitation tax), RET (real estate tax)

Job = U (unemployed), P (public), PR (private sector)

### 5.2.2 Application of Classification

C4.5 Algorithm in Weka Tool was used for classification to create decision trees of data marts. Sample output of the classification operation in Weka is shown in Figure 5.1. The constructed trees are saved into text files, i.e. August2003.txt file.

```
=== Run information ===

Scheme:   weka.classifiers.trees.J48 -C 0.25 -M 2
Relation: income
Instances: 5913
Attributes: 9
           gender
           marital_status
           region
           district
           income_type
           class
           age
           job
           children number
Test mode: 10-fold cross-validation

=== Classifier model (full training set) ===

J48 pruned tree
-----

income_type = EST: NOT_PAID (1284.0/195.0)
income_type = RET: NOT_PAID (2961.0/252.0)
income_type = water_consumption
| marital_status = S: PAID (204.0/42.0)
```

Figure 5.1 Sample classification output (Weka).

```

| marital_status = W: PAID (54.0/6.0)
| marital_status = E
| | region = mediterranean
| | | job = p
| | | | gender = I: PAID (0.0)
| | | | gender = M: NOT_PAID (15.0/6.0)
| | | | gender = F: PAID (9.0/3.0)
| | | job = pr
| | | | gender = I: NOT_PAID (0.0)
| | | | gender = M: NOT_PAID (20.0/6.0)
| | ...

```

Figure 5.1 Sample classification output (Weka). (Cont.)

### 5.2.3 Application of Tree-to-Sequences Linearization

Tree-to-Sequences Linearization (TSL) is applied on the trees, were constructed in the previous step. Pseudo code of TSL is shown in Figure 5.2.

```

array[] months =
{"january","february","march","april","may","june","july","august","september","october","nov
ember","december"}
for k = 2003 to 2013
  for i = 0 to 12
    file = months[i] + k + ".txt";
    newFile = linear/months[i] + k + ".txt";

    open newFile for writing;
    open file for reading;
    seek 22th line in file;

    array[] headers = {};
    while not end of file
      str = read line from file;

      firstCharacter = str[0];
      temp="";
      if (firstCharacter != "|")
        s = str.Split(' ');
        h = 0;
        foreach (item in s)

```

Figure 5.2 TSL pseudo code.

```

    firstCharacter = item[0];
    if (firstCharacter != "(")
        if (h != 0 & h % 3 == 0)
            temp = temp + " ";
            temp = temp + item;
        End if
        h++;
    End Foreach
    Write temp in newFile
    control = true;
    if (s[s.Length-1].[s[s.Length - 1].Length - 1] != ")")
        For j=0 to headers.count
            array[] tempStr = headers[j].split(' ');
            array[] strStr = temp.Split(' ');
            if (tempStr.Length == strStr.Length)
                headers[j] = temp;
                control = false;
            End if
        End for
        if (control==true)
            Headers.Add(temp);
        End if
    End if
    Else
        s = str.split(' ');
        int lcount = 0;
        for p=0 to s.Length
            if (s[p] == "|")
                lcount++;
        End for
        temp = temp + headers[lcount-1];
        if (lcount != headers.count)
            remove headers[headers.Count-1]

        h=0;
        foreach item in s
            if (item!="")
                firstCharacter = item[0];
                if (firstCharacter != "" & firstCharacter != "(" & firstCharacter != "|")
                    if (h!=0 & h%3 == 0)
                        temp = temp + " ";
                        temp = temp + item;
                    End if
                    h++;
                End foreach
            Write temp in newFile;
            control = false;

```

Figure 5.2 TSL pseudo code. (Cont.)

```

if (s[s.Length-1].[ s[s.Length - 1].Length - 1] != ")")

    for j=headers.Count-1 to j=0; j--;
        array[] stri = headers[j].split(' ');
        array[] tempi = temp.split(' ');
        if (stri.Length == tempi.Length)

            headers[j] = temp;
            control = true;
            break;
        End if
    End for
    if (control == false)
        headers.add(temp);
    End if
End While
Close file;
Close newFile;
EndFor
EndFor

```

Figure 5.2 TSL pseudo code. (Cont.)

After linearization operation, tree given in Figure 5.1 converted into linear form, as shown in Figure 5.3.

```

income_type=EST: NOT_PAID
income_type=RET: NOT_PAID
income_type=water_consumption
income_type=water_consumption marital_status=S: PAID
income_type=water_consumption marital_status=W: PAID
income_type=water_consumption marital_status=M
income_type=water_consumption marital_status=M region=mediterranean
income_type=water_consumption marital_status=M region=mediterranean job=p
income_type=water_consumption marital_status=M region=mediterranean job=p gender=I: PAID
income_type=water_consumption marital_status=M region=mediterranean job=p gender=M:
NOT_PAID
income_type=water_consumption marital_status=M region=mediterranean job=p gender=F:
PAID
income_type=water_consumption marital_status=M region=mediterranean job=pr

```

Figure 5.3 Decision tree sequences.

```

income_type=water_consumption marital_status=M region=mediterranean job=pr gender=I:
NOT_PAID
income_type=water_consumption marital_status=E region=mediterranean job=pr gender=M:
NOT_PAID
...

```

Figure 5.3 Decision tree sequences. (Cont.)

### 5.2.4 Application of Sequential Pattern Mining

In order to apply sequential pattern mining using SPMF, tree sequences should be converted into proper format for SPMF. Pseudo code in Figure 5.4 is applied on TS.

```

array[] months =
{"january", "february", "march", "april", "may", "june", "july", "august", "september", "october", "novem
ber", "december"}
array[,] values = {};
counter = 1;
for k = 2003 to 2013
  for i = 0 to 12
    file = months[i] + k + ".txt";
    newFile = spmf/months[i] + k + ".txt";

    open newFile for writing;
    open file for reading;

    line = "";
    array[] specialChars = {':'}
    temp = "";

    while not end of file
      str = read line from file;
      array[] s = str.Split(' ');

      foreach (item in s)
        temp = item;
        if (item.EndsWith(":"))
          temp = temp.substring(0, temp.length - 1);
          if (there is not temp in values)
            push {temp, counter} into values

            line = line + counter;
            counter ++;
        End if

```

Figure 5.4 Pseudo code of dataset formatting for SPMF.

```

        Else
            Line = line + values[temp];
        if (temp!=s[s.length-1])
            line = line + " -1 ";
        Else
            line = line + "-2";
        End foreach
        Write line into newFile;
        line = "";
    End While
    Close newFile
End for
End for

```

Figure 5.4 Pseudo code of dataset formatting for SPMF. (Cont.)

Figure 5.5 shows sample dataset in SPMF form.

```

1 -1 4 -2
2 -1 4 -2
5 -2
5 -1 3 -1 6 -2
5 -1 12 -1 6 -2
5 -1 7 -2
5 -1 7 -1 2 -2
5 -1 7 -1 2 -1 11 -2
5 -1 7 -1 2 -1 11 -1 8 -1 6 -2
5 -1 7 -1 2 -1 11 -1 9 -1 4 -2
5 -1 7 -1 2 -1 11 -1 26 -1 6 -2
5 -1 7 -1 2 -1 59 -2
5 -1 7 -1 2 -1 59 -1 8 -1 4 -2
5 -1 7 -1 2 -1 59 -1 9 -1 4 -2
...

```

Figure 5.5 Sample dataset in SPMF form.

Data marts in SPMF form were partitioned according to class values. Figure 5.6 shows sample dataset for August 2003 for “Not Paid” class.

```

income_type=EST: NOT_PAID
income_type=RET: NOT_PAID
income_type=water_consumption marital_status=E region=mediterranean job=p gender=M:
NOT_PAID

```

Figure 5.6 August 2003 dataset for “Not Paid” class.

```
income_type=water_consumption marital_status=E region=mediterranean job=pr gender=l:
NOT_PAID
```

Figure 5.6 August 2003 dataset for “Not Paid” class. (Cont.)

Figure 5.7 shows sample dataset for August 2003 for “Paid” class.

```
income_type=water_consumption marital_status=S: PAID
income_type=water_consumption marital_status=W: PAID
income_type=water_consumption marital_status=E region=mediterranean job=p gender=l: PAID
income_type=water_consumption marital_status=E region=mediterranean job=p gender=F: PAID
...
```

Figure 5.7 August 2003 dataset for “Paid” class.

After partitioning operation, prefix-span algorithm was applied on two TSs and frequent sequences (FSs) of TSs were obtained. Sample FSs for August 2003 dataset for “Paid” class are shown in Figure 5.8.

```
...
marital_status=M region=eastern children_number=0 #SUP: 2
marital_status=M region=eastern gender=M #SUP: 10
marital_status=M region= eastern gender=M children_number=0 #SUP: 2
marital_status=M region= eastern gender=M age=50-65 #SUP: 2
marital_status=M region= eastern gender=M age=20-35 #SUP: 3
marital_status=M region= eastern gender=M children_number=2 #SUP: 5
marital_status=M region= eastern gender=M children_number=2 age=50-65 #SUP: 2
marital_status=M region= eastern age=50-65 #SUP: 2
region=marmara gender=M age=>65 #SUP: 3
...
```

Figure 5.8 FSs of August 2003 dataset for “Paid” class (AUGUST2003Paid.txt).

Sample FSs for August 2003 dataset for “Not Paid” class are shown in Figure 5.9.

```
...
marital_status =M region= eastern gender=M children_number=0 #SUP: 3
marital_status =M region= eastern gender=M age=>65 #SUP: 2
marital_status =M region= eastern gender=M children_number=1 #SUP: 3
marital_status =M region= eastern age=>65 #SUP: 2
region=marmara gender=M age=50-65 #SUP: 2
```

Figure 5.9 FSs of August 2003 dataset for “Not Paid” class (AUGUST2003NotPaid.txt).

### 5.2.5 Application of Similarity Calculation

For each TS partition, FSs were obtained using prefix-span algorithm via SPMF as shown in Figure 5.8 and Figure 5.9. After this operation, each FS was compared with other FSs to find CFS. Pseudo code for calculating CFS is shown in Figure 5.10. The support values of FSs are ignored since number of common FSs are not important.

```
for i=0 to fileNames.Count
  array[] FS1 = {};
  str = read line from fileNames[i];
  while not end of file
    array[] sequences = str.split(' ');
    temp = "";
    for p=0 to sequences.length - 2
      if (sequences[p] != "")
        temp = temp + " " + sequences[p];
    End for
    Push temp into FS1;
    str = read line from fileNames[i];
  End While
  Close fileNames[i];
  for j=i+1 to fileNames.Count
    array[] FS2 = {};
    str = read line from fileNames[i];
    while not end of file
      array[] sequences = str.split(' ');
      temp = "";
      for p=0 to sequences.Length - 2
        if (sequences[p]!="")
          temp = temp + " " + sequences[p];
        End for
        Push temp into FS2
        str = read line from fileNames[i];
      End while
      Close fileNames[j];
      frequentSequenceCounter = 0;
      for f=0 to FS1.Count
        if (FS2 contain FS1[f])
          frequentSequenceCounter++;
        End for
        Write fileNames[i]+ " " + fileNames[j] + " " + frequentSequenceCounter + " " +
Max(FS1.Count, FS2.Count) into ResultFile;
      End for
    End for
  End for
  Close ResultFile;
```

Figure 5.10 Pseudo code of CFS.



Result of the pseudo code given in Figure 5.10 is saved as file as shown in Figure 5.11 and Figure 5.12.

```
JANUARY2003Paid.txt FEBRUARY2003Paid.txt 266 1801
JANUARY2003Paid.txt MARCH2003Paid.txt 297 1836
JANUARY2003Paid.txt APRIL2003Paid.txt 266 1505
JANUARY2003Paid.txt MAY2003Paid.txt 137 1516
JANUARY2003Paid.txt JUNE2003Paid.txt 31 735
JANUARY2003Paid.txt JULY2003Paid.txt 69 735
JANUARY2003Paid.txt AUGUST2003Paid.txt 64 754
JANUARY2003Paid.txt SEPTEMBER2003Paid.txt 43 735
JANUARY2003Paid.txt OCTOBER2003Paid.txt 47 735
JANUARY2003Paid.txt NOVEMBER2003Paid.txt 115 735
JANUARY2003Paid.txt DECEMBER2003Paid.txt 30 735
JANUARY2003Paid.txt JANUARY2004Paid.txt 160 735
JANUARY2003Paid.txt FEBRUARY2004Paid.txt 67 735
JANUARY2003Paid.txt MARCH2004Paid.txt 130 735
JANUARY2003Paid.txt APRIL2004Paid.txt 110 735
JANUARY2003Paid.txt MAY2004Paid.txt 96 735
JANUARY2003Paid.txt JUNE2004Paid.txt 33 735
.....
```

Figure 5.11 Result file for “Paid” class.

For example; the line “JANUARY2003Paid.txt FEBRUARY2003Paid.txt 266 1801” means that CFS is 266 and maximum FS number is 1801.

```
JANUARY2003NotPaid.txt FEBRUARY2003NotPaid.txt 158 1117
JANUARY2003NotPaid.txt MARCH2003NotPaid.txt 188 1065
JANUARY2003NotPaid.txt APRIL2003NotPaid.txt 174 1442
JANUARY2003NotPaid.txt MAY2003NotPaid.txt 107 1736
JANUARY2003NotPaid.txt JUNE2003NotPaid.txt 25 413
JANUARY2003NotPaid.txt JULY2003NotPaid.txt 38 413
JANUARY2003NotPaid.txt AUGUST2003NotPaid.txt 48 734
JANUARY2003NotPaid.txt SEPTEMBER2003NotPaid.txt 32 413
JANUARY2003NotPaid.txt OCTOBER2003NotPaid.txt 35 413
JANUARY2003NotPaid.txt NOVEMBER2003NotPaid.txt 81 559
JANUARY2003NotPaid.txt DECEMBER2003NotPaid.txt 19 413
JANUARY2003NotPaid.txt JANUARY2004NotPaid.txt 79 413
JANUARY2003NotPaid.txt FEBRUARY2004NotPaid.txt 58 413
JANUARY2003NotPaid.txt MARCH2004NotPaid.txt 99 683
JANUARY2003NotPaid.txt APRIL2004NotPaid.txt 80 413
JANUARY2003NotPaid.txt MAY2004NotPaid.txt 100 1180
JANUARY2003NotPaid.txt JUNE2004NotPaid.txt 22 413
....
```

Figure 5.12 Result file for “Not Paid” class.

Similarity is calculated as shown in Figure 5.13. *filePaidResults* represents file given in Figure 5.11 and *fileNotPaidResults* represents file given in Figure 5.12.

```

filePaidResults, fileNotPaidResults;
array[,] paid;
array[,] paidMax;

str = read line from filePaidResults;
while (str is not null)
    array[] s = str.split(' ');
    push s[0]+"-"+s[1], s[2] into paid;
    push s[0]+"-"+s[1],s[3] into paidMax;
    str = read line from filePaidResults;
End While

array[] notPaid;
array[] notPaidMax;

str = read line from fileNotPaidResults;
while (str is not null)
    array[] s = str.Split(' ');
    push s[0]+"-"+s[1],s[2] into notPaid;
    push s[0]+"-"+s[1],s[3] into notPaidMax;
    str = read line from fileNotPaidResults;
End While

max=0;
maxPair = "";
for i=0 to paid.count
    resultValue = (100)*(( paid[i].Value / paidMax[i].Value) + (notPaid[i].Value /
notPaidMax[i].Value)) / 2;
    if (resultValue>max)
        max = resultValue;
        maxPair = paid[i].Key;
    End if
End for
Return max,MaxPair

```

Figure 5.13 Pseudo code for similarity calculation.

### 5.3 General Similarity of Trees

Similarities between the trees in Forest  $F$  created at Step 2 (Classification) are calculated 7140 times ( $1 + 2 + \dots + 199$ ) for 120 trees by applying Step 5 (Similarity Calculation and K-Nearest Neighbor Algorithm), after performing Step 3 (TSL) and Step 4 SPM (with minimum support = 0.001).

*Common frequent sequences (CFSs)* are given by the cardinality of the intersection of two frequent sets. They are calculated for pieced and general similarity respectively.

The histogram of general tree similarity based on the CFS is shown in Figure 5.14, to illustrate their general similarity. While the  $y$ -axis of this graph shows the number of tree pairs, the  $x$ -axis includes CFSs between two trees. Each  $t^{\text{th}}$  tree is compared with all other  $(120 - t)$  trees in  $F$ . The average CFS number was 15 for all tree pairs in  $F$ , while the maximum CFS number was nearly 80, for data between March 2003 and April 2003. Figure 5.15 displays top 10 tree pairs which have maximum CFSs.

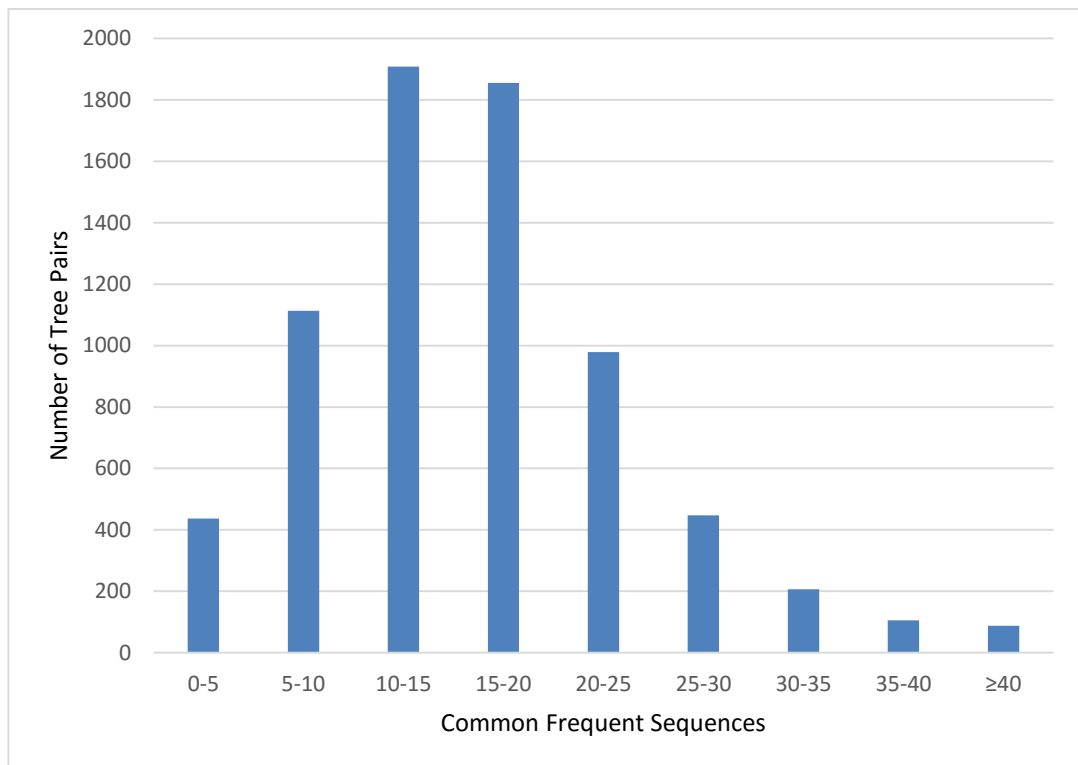


Figure 5.14 Histogram of general tree similarity based on the common frequent sequences (CFS).

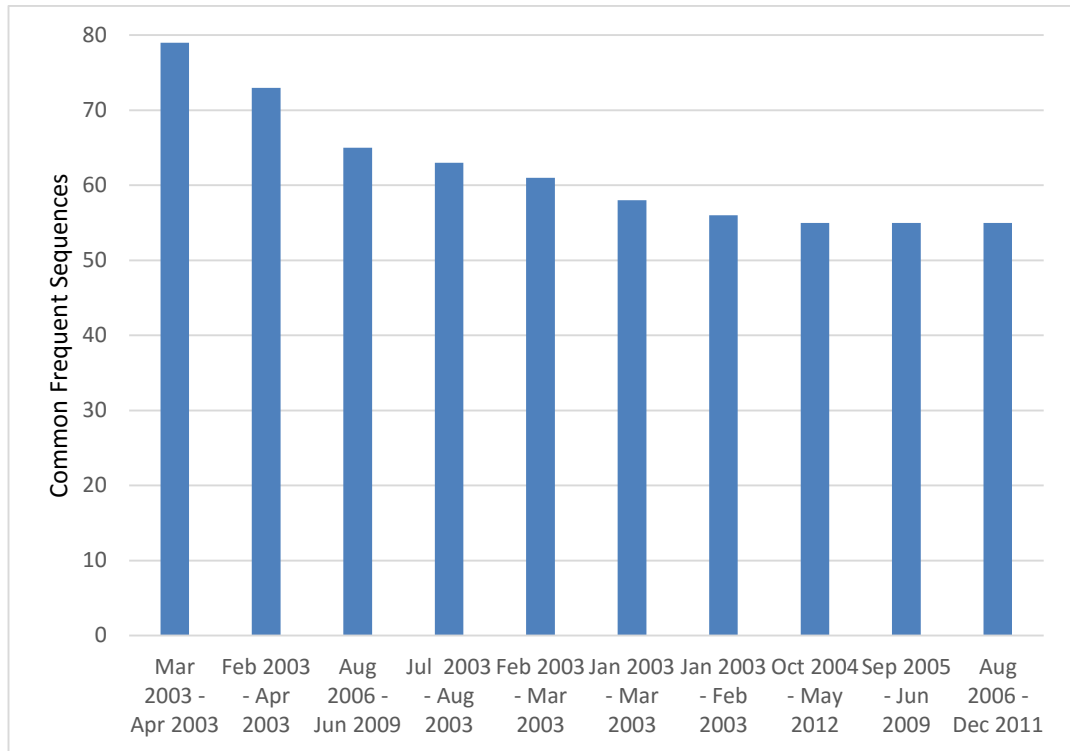


Figure 5.15 General tree similarity based on the common frequent sequences (CFS) (top 10).

Figure 5.16 displays CFS of all 7140 tree pairs ordered according to their CFS. First 10 tree pairs are; March 2003 – April 2003 (79), February 2003 – April 2003 (73), August 2006 – June 2009 (65), February 2003 – August 2003 (63), February 2003 – March 2003 (61), January 2003 – March 2003 (58), January 2003 – February 2003 (56), October 2004 – May 2012 (55) and September 2005 – January 2009 (55).

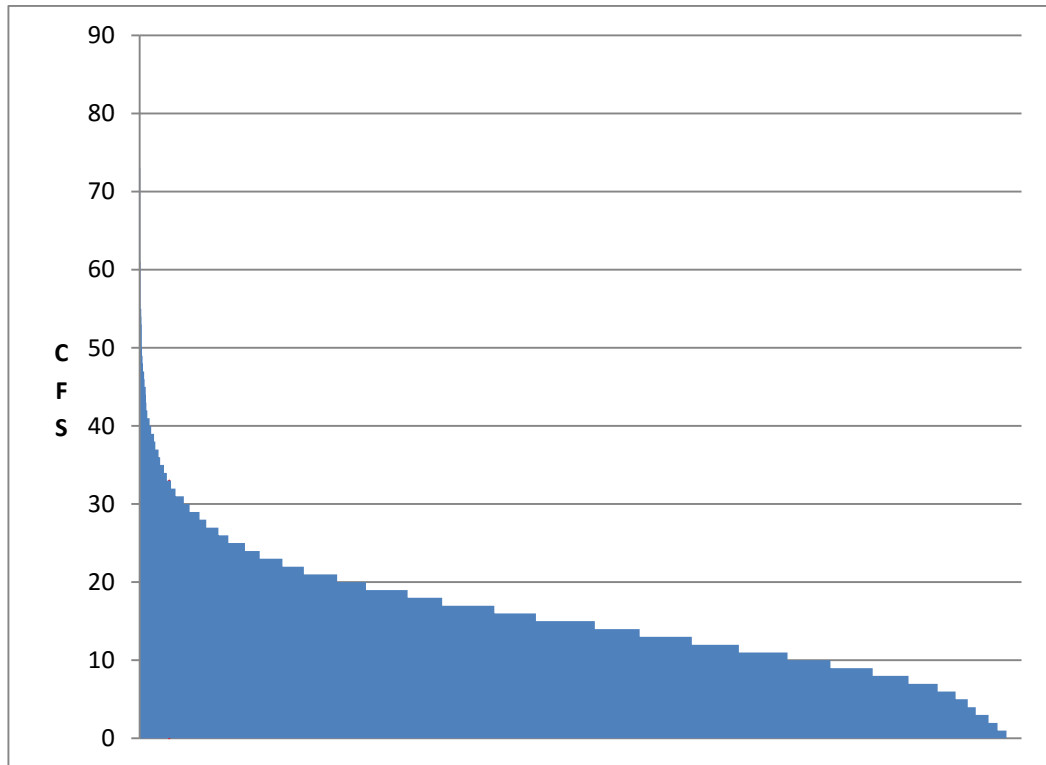


Figure 5.16 General tree similarity based on the common frequent sequences (CFS).

#### 5.4 Pieced Similarity of Trees

Pieced similarity (Def. 7) was used to divide all 120 TSs into two sub-TSs ( $TS_{\text{paid}}$ ,  $TS_{\text{notpaid}}$ ), after the TSL step. The SPM step was applied to both sub-TSs to create two sub-FSs ( $FS_{\text{paid}}$ ,  $FS_{\text{notpaid}}$ ). Each sub-FS was compared with other sub-FSs, which included the same class label. For example, January 2012  $FS_{\text{paid}}$  was compared with February 2012  $FS_{\text{paid}}$ , and January 2012  $FS_{\text{notpaid}}$  with February 2012  $FS_{\text{notpaid}}$ . Similarity was calculated using the formulas given in Equation 4.10. Figure 6 shows the histogram of pieced similarity given by the number of CFSs.

As shown in histogram in Figure 5.17, 1640 tree pairs have CFS between 30 - 40 while only 272 tree pairs have CFS between 70 – 80.

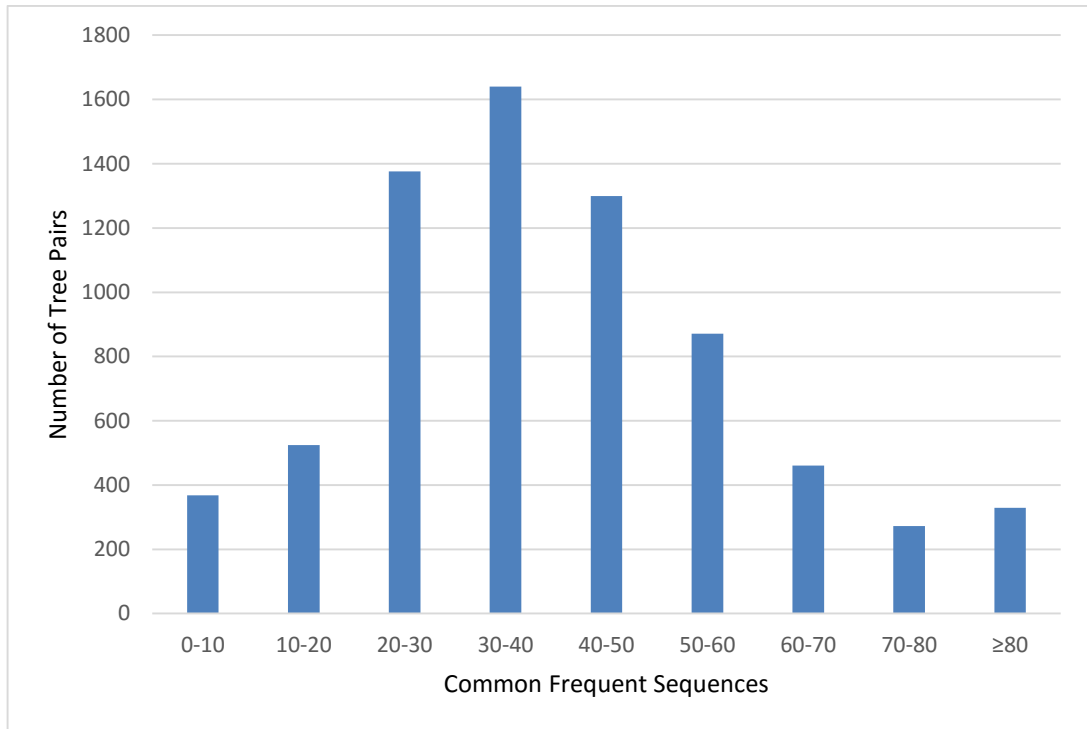


Figure 5.17 Histogram of pieced similarity given by the number of common frequent sequences (CFS).

Our results given in Figure 5.18 and Figure 5.19 show that the average CFS number was 40, while the maximum CFS was 200 (for March 2003 and April 2003).

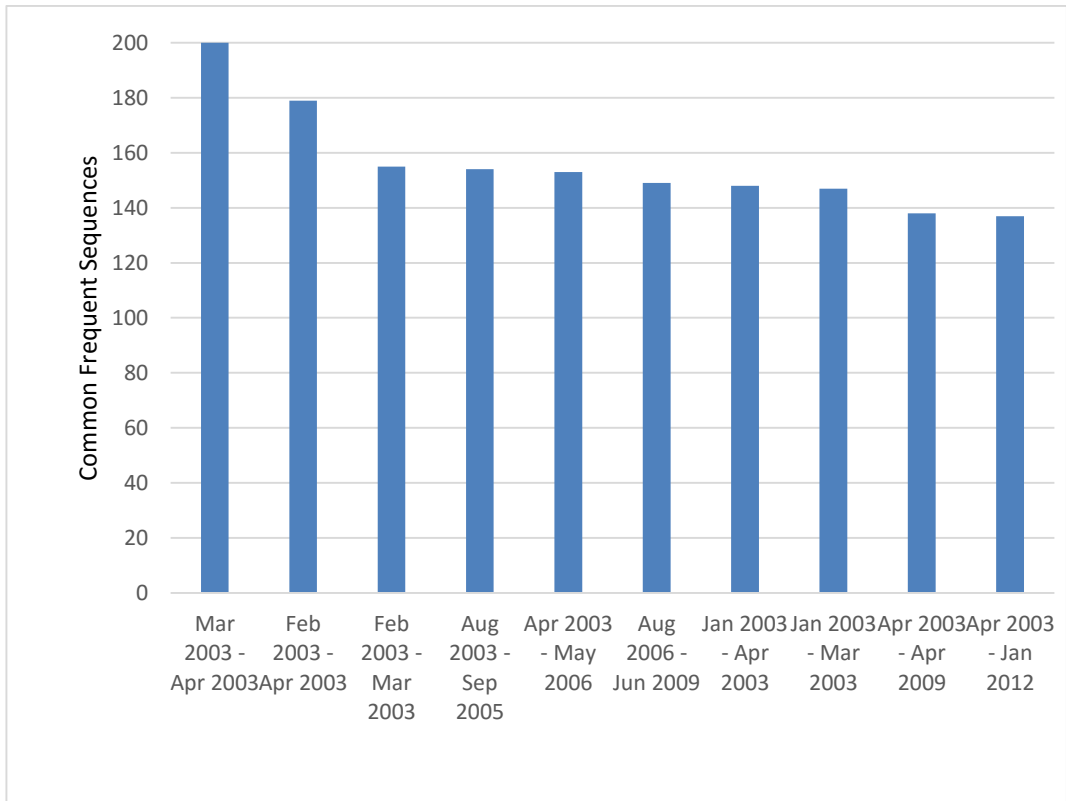


Figure 5.18 Pieced similarity given by the number of common frequent sequences (CFS) (top 10).

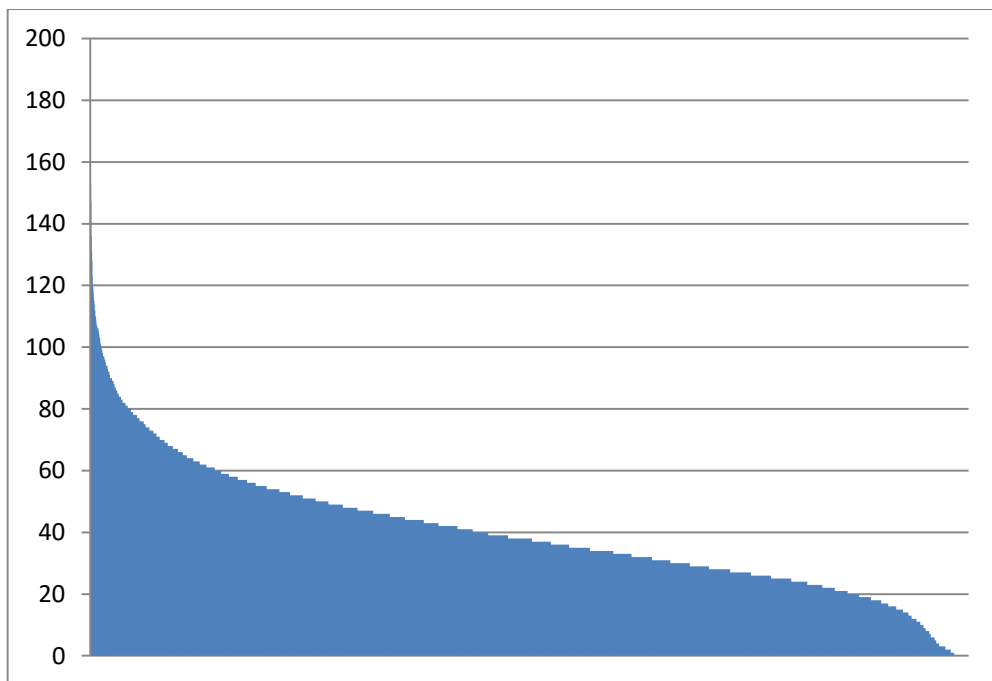


Figure 5.19 Pieced similarity given by the number of common frequent sequences (CFS).

However, the maximum CFS number does not imply the most similarity, since the maximum number of sequences in the pairs also affects similarity. Figure 5.20 displays the histogram of pieced similarity (%). When we considered the number of sequences, the most similar (64%) tree pairs were the DTs constructed for September 2010 and September 2011, as shown in Figure 5.21. According to the results, the DTs constructed for September and June were found similar for the six years. So we can say that local municipality can predict tax revenue of September at the end of June.

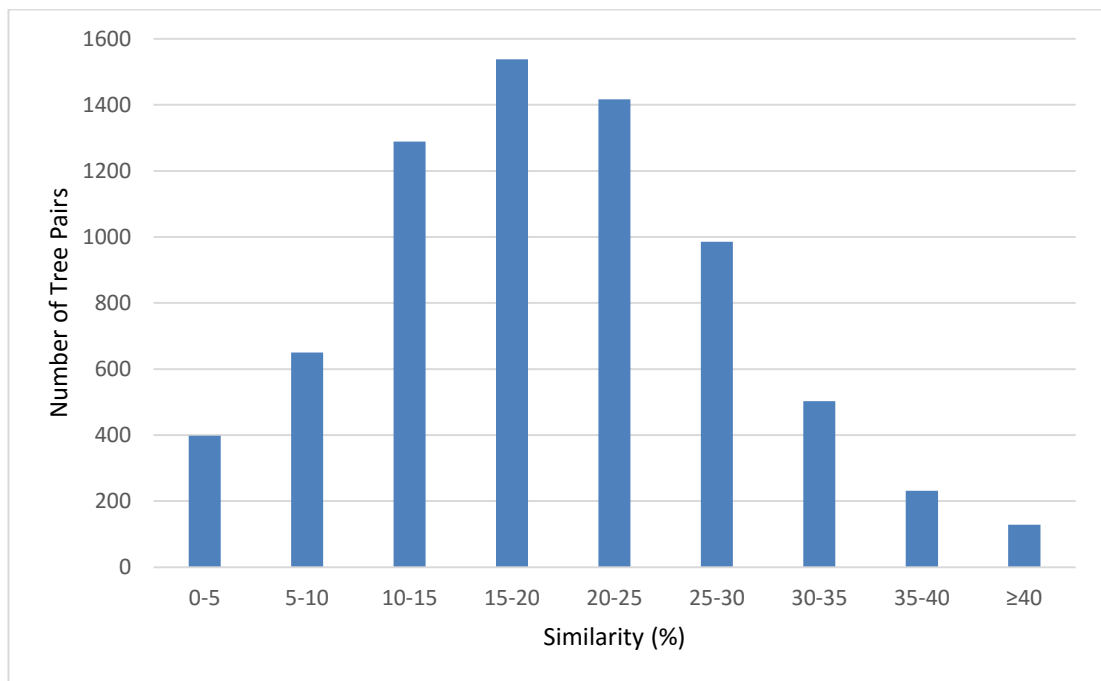


Figure 5.20 Histogram of pieced similarity given by percentage values.



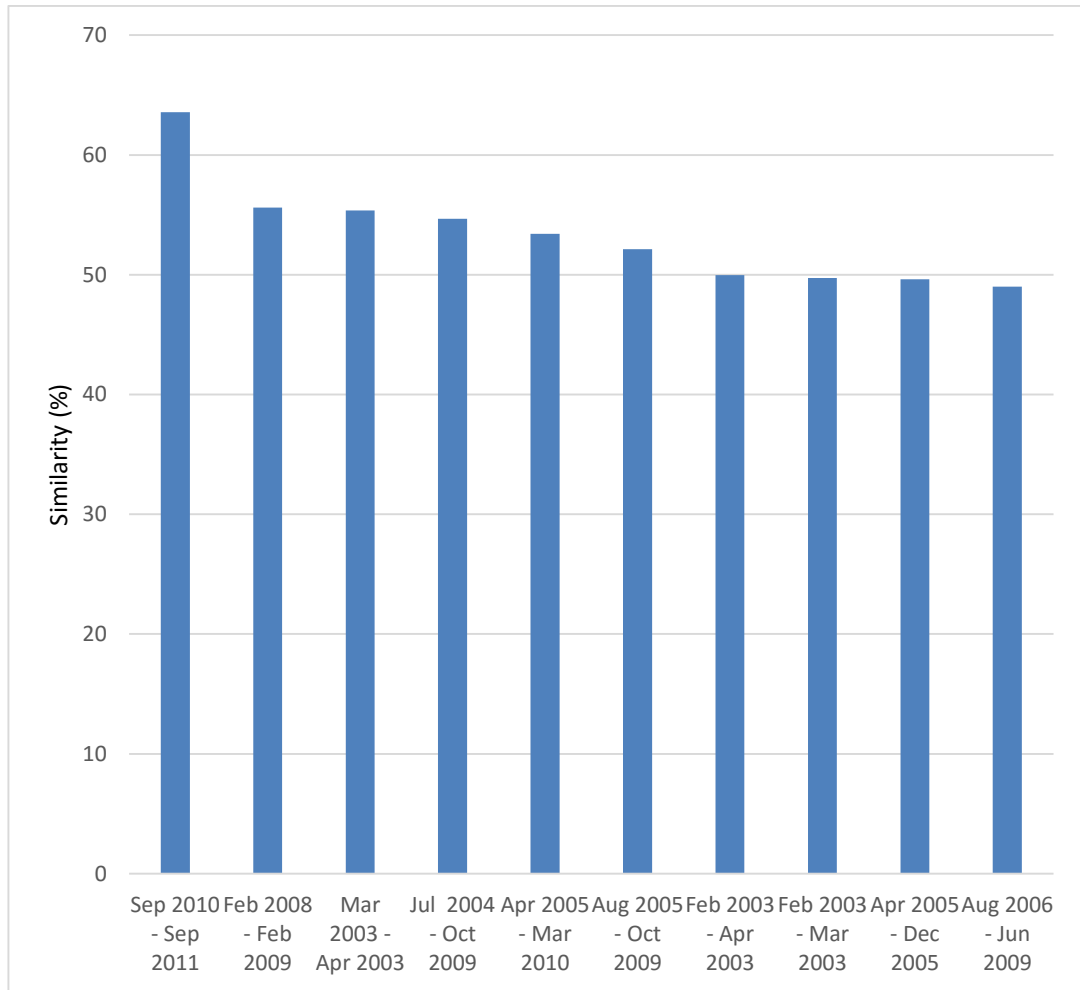


Figure 5.21 Pieced similarity given by percentage values (top 10 similarity (%)).

In another example, the decision tree with 62 sequences constructed for September 2010 was compared with the tree with 63 sequences for September 2011. The number of common sequences for these trees was 37, so the similarity for the *paid* class can be calculated as  $37/\text{Max}\{62,63\}$ , equal to 58%. When the same operation is repeated for the *notpaid* class, the result was 68%. The final similarity (64%) was calculated by averaging the results for each class.

$$\#(\text{September 2010 } FS_{\text{paid}}) = 62$$

$$\#(\text{September 2011 } FS_{\text{paid}}) = 63$$

$$\#(\text{September 2010 } FS_{\text{paid}} \cap \text{September 2011 } FS_{\text{paid}}) = 37$$

$$Sim(\text{September 2010 } FS_{paid}, \text{September 2011 } FS_{paid}) = \frac{37}{Max(62,63)} \cong 58\%$$

$$\#(\text{September 2010 } FS_{notpaid}) = 54$$

$$\#(\text{September 2011 } FS_{notpaid}) = 57$$

$$\#(\text{September 2010 } FS_{notpaid} \cap \text{September 2011 } FS_{notpaid}) = 39$$

$$Sim(\text{September 2010 } FS_{paid}, \text{September 2011 } FS_{paid}) = \frac{39}{Max(54,57)} \cong 68\%$$

$$Sim(\text{September 2010 } FS, \text{September 2011 } FS) = \frac{0.58 + 0.68}{2} \cong 64\%$$

## 5.5 Comparison of General Similarity and Pieced Similarity

The *average common frequent sequences (ACFSs)* is given by the average cardinality of the intersection of two frequent sets for all trees.

$$ACFS = \frac{\sum_{x=1}^n CFS_x}{n} \quad (5.1)$$

where  $n$  is the total number of trees.

The ACFSs obtained using both similarity approaches are shown in Figure 5.22. According to our results, pieced similarity supplies 40 ACFSs, which is more than twice the ACFSs generated by general similarity.

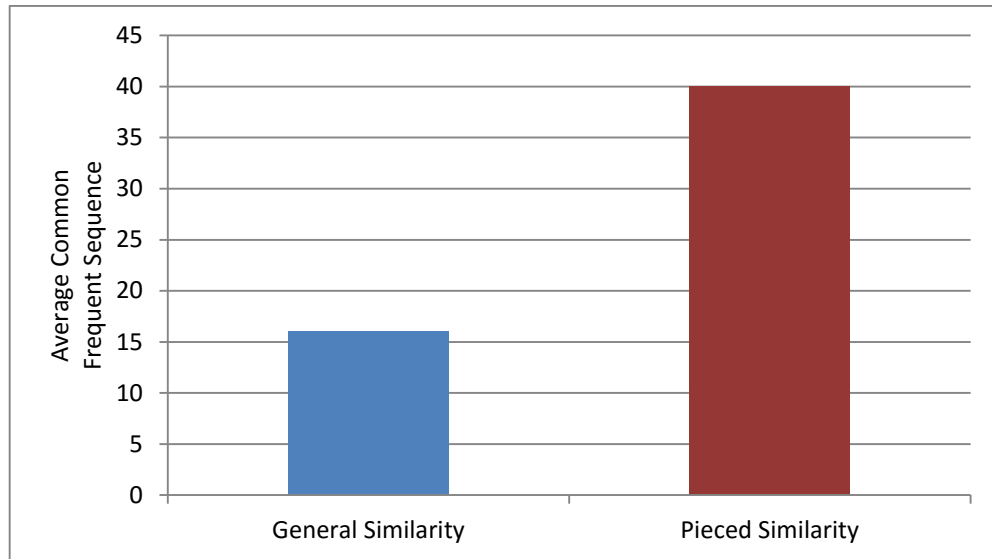


Figure 5.22 Average common frequent sequence (ACFS) numbers used to compare General similarity and Pieced similarity techniques.

*Comparison number* is the number of comparison operations carried out on trees to have the most similar trees.

Based on Figure 5.22, we can say that the correlation among the General Similarity based CFS results and Pieced Similarity based CFS results is very high. Since the sizes of the trees are smaller and more similar BSs are expected to be grouped under different class labels. However, the main disadvantage of using pieced similarity is the number of comparisons it generates, since the number of sequence sets doubles compared with general similarity, as shown in Figure 5.23.

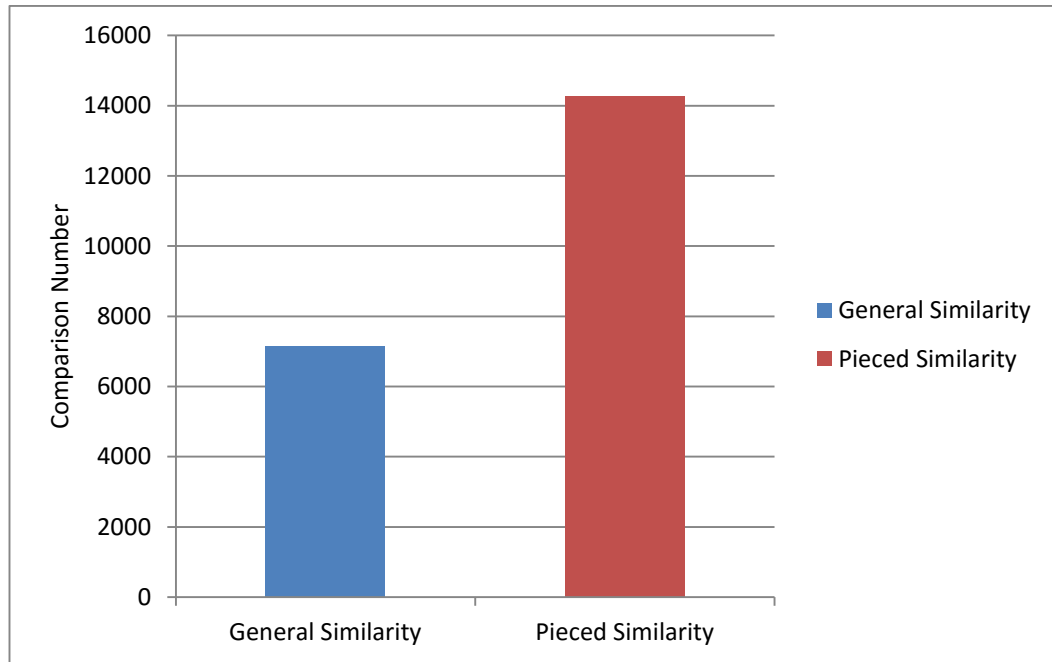


Figure 5.23 The number of comparison operations carried out on trees for General similarity and Pieced similarity techniques.

### 5.6 Comparison of DTreeSim with Existing Approaches

DTreeSim was compared with existing two approaches: DTSim (Peahringer & Witten, 1997) and  $Sim_{d1,d2}$  (Perner, 2011, 2013). Sample small trees used to compare similarity approaches are shown in Figures 5.24 and 5.25. The same trees, except with different attribute values (branch values), were used in (Perner, 2011, 2013). We also needed to add the attribute values to calculate similarity using our approach.

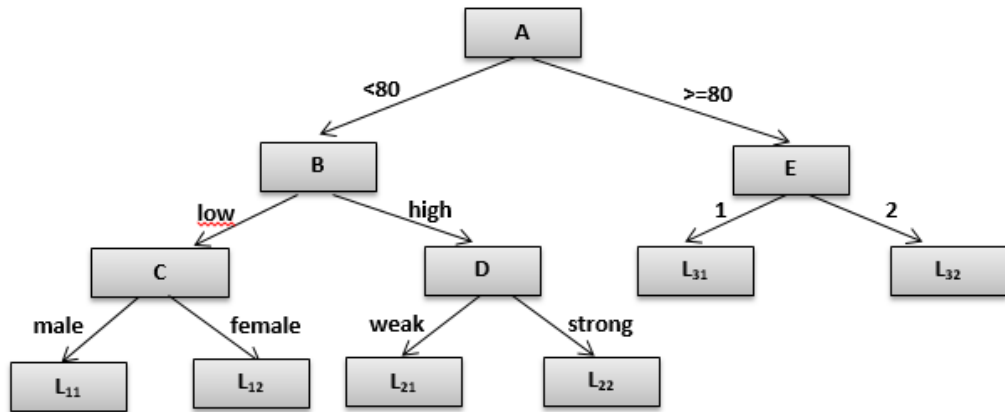


Figure 5.24 - Structure for Decision Tree 1 (Perner, 2011; 2013).

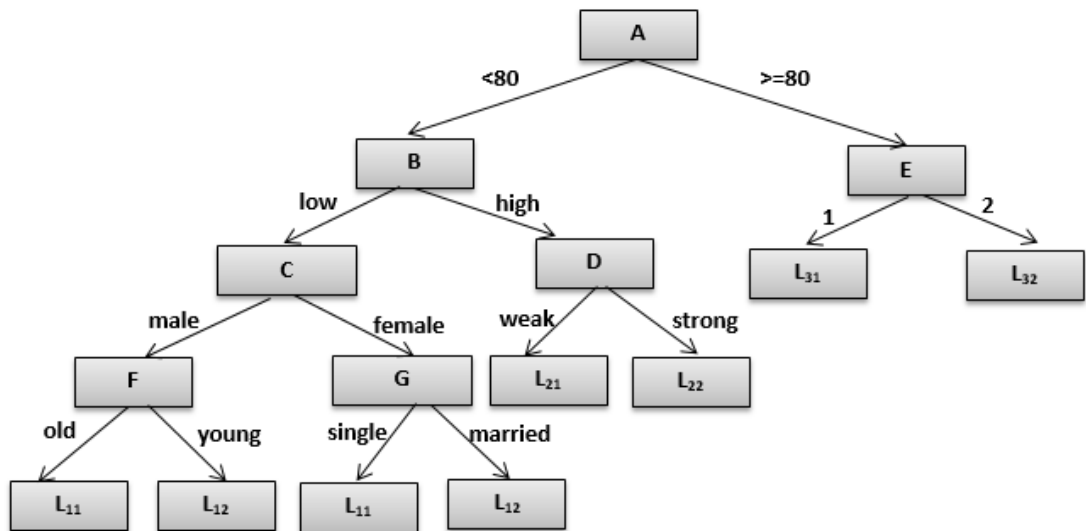


Figure 5.25 Structure for Decision Tree 2 (Perner, 2011; 2013).

It is possible to classify the degree of similarity in five types; Very Similar ( $\geq 80$ ), Similar ( $< 80$  and  $\geq 60$ ), Neither Similar nor Dissimilar ( $< 60$  and  $\geq 40$ ), Dissimilar ( $< 40$  and  $\geq 20$ ) and Very Dissimilar ( $< 20$ ). Similarity values and degrees between Tree 1 and Tree2 are shown in Table 5.7. Maximum similarity was calculated using  $Sim_{d1,d2}$  (Perner, 2011; 2013), giving 91.66%. This reflects the fact that Perner (Perner, 2011; 2013) ignores leaf (class) and attribute values. DTSim (Peahringer & Witten, 1997) had the minimum similarity value, with

66%. This is because DTSim compares branches one-to-one. In this algorithm, if there is one different node on a branch, then these branches are deemed different, even though nodes are in the same order. Since our approach finds FSs for each tree and compares these FSs rather than full branches, our similarity is greater than DTSim (Peahringer & Witten, 1997), but smaller than  $Sim_{d1,d2}$  (Perner, 2011; 2013), because we consider both leaf (class values) and attribute values. So, the proposed approach produces more reliable results with respect to the existing approaches.

Table 5.7 Similarity values obtained using different approaches

Approach	Similarity Values (%)	The degree of similarity
DTreeSim	78.30	Similar
DTSim (Peahringer & Witten, 1997)	66	Similar
$Sim_{d1,d2}$ (Perner, 2011, 2013)	91.66	Very Similar

### 5.7 The Application of DTreeSim to Benchmark Data with Bootstrapping

Since decision trees are known to be unstable and small perturbations in the dataset might yield significant differences in the resulting tree, we also used an Ensemble Learning technique (Seni & Elder, 2010). Although there is a framework inTrees (interpretable trees) (Deng, 2014) that creates random forest with multiple trees, we applied bootstrapping (Ishwaran & Rao, 2009) with the fraction  $(1 - 1/e \approx 63.2\%)$ . We constructed randomly sampled DMs for each partition, obtained a different DT for each sample and measured the similarities among DTs. In this case, DTreeSim was applied to a dataset “Car Evaluation” from the UCI Machine Learning Repository (<http://archive.ics.uci.edu/ml/>). This dataset includes 1728 instances, with six attributes, as shown in Table 5.8.

Table 5.8 Car evaluation dataset descriptors for data from the UCI Machine Learning Repository

Attribute	Attribute Values
Buying	vhigh, high, med, low
Maint	vhigh, high, med, low
Doors	2, 3, 4, 5, more
Persons	2, 4, more
Lug-Boot	small, med, big

Table 5.8 Car evaluation dataset descriptors for data from the UCI Machine Learning Repository (Cont.)

<b>Attribute</b>	<b>Attribute Values</b>
<b>Safety</b>	low, med, high
<b>Class</b>	unacc, acc, good, vgood

Bootstrapping was applied to the dataset to form five different DMs, and also different DTs. Using DTreeSim, we found that the 3<sup>rd</sup> and 5<sup>th</sup> trees were most similar. When we evaluated our result by checking classification accuracy for each tree, we saw that the 3<sup>rd</sup> and 5<sup>th</sup> trees were in fact closest to each other. This result supports the reliability of the proposed approach.

## **CHAPTER SIX**

### **DECISION-MAKER SYSTEM FOR LOCAL MUNICIPALITIES**

#### **6.1 The Scope of the System**

Developed system includes 13 scenarios under 5 main categories:

- **Socio-Cultural Analyses**
  - Employee Analysis
  - Citizen Analysis
  - Staff Analysis
  
- **Income/Expense Analyses**
  - Moveable Material Analysis
  - Estimation of Wages
  - Income Operations Analysis
  
- **Infrastructure Analyses**
  - Water Notice Analysis
  - Electricity Consumption Analysis
  
- **Fraud Detection Analyses**
  - Logs Analysis
  - Fuel Oil Analysis
  - Cash Desk Analysis
  
- **Simplification, Verification and Similarity Analyses**
  - User Account Analysis
  - Accountancy Analysis



## 6.2 Scenarios for Local Municipalities

### 6.2.1 Socio-Cultural Analyses

These kinds of analyses were created to analyze citizens, corporate foundations like bank, company, cooperative and municipal employee.

#### 6.2.1.1 Employee Analysis

The purpose of this scenario is to cluster local government employees according to number of their children, age, duration of working, the number of days that employee didn't work and education status.

This scenario is important to make common decisions about municipal employees. For example, municipal or department manager may want to warn employees who have big number of non-working days by sending e-mails to this cluster or manager may want to make advance payments to employees whose duration of working bigger than 10 years and number of children bigger than 3 also have a few non-working days. In this situation, manager is able to find the related cluster easily. Also correlation between employee's attributes can be found via this scenario.

Clustering service is used for this scenario. First, data reduction is applied on database tables which are shown in Table 6.1, 6.2 and 6.3. Second, data integration is performed by joining these tables to have proper dataset. Last, data normalization is applied to dataset using *Min-Max Normalization* technique.

$$v' = \frac{v - \min A}{\max A - \min A} (\text{new\_max}A - \text{new\_min}A) + \text{new\_min}A \quad (6.1)$$

where  $v$  is the current value of attribute  $A$ ,  $\min A$  is the minimum value of the attribute  $A$  and  $\max A$  is the maximum value of the attribute  $A$ .  $\text{new\_max}A$  is the new maximum value and  $\text{new\_min}A$  is the new minimum value of attribute  $A$ . *Min-Max Normalization* maps  $v$  to  $v'$  in the range  $\{\text{new\_min}A, \text{new\_max}A\}$

Table 6.1 Employee database table

<b>Employee</b>
<b>Id</b>
<b>Date_of_Birth</b>
<b>Date_of_Start</b>
<b>Education_Status_Id</b>
<b>Date_of_Leaving</b>
<b>Type_Code</b>

Table 6.2 Employee Family Members database table

<b>Employee Family Members</b>
<b>Employee_Id</b>
<b>Relationship_Code</b>

Table 6.3 Employee Scoring database table

<b>Employee Scoring</b>
<b>Employee_Id</b>
<b>Property_of_Day_Code</b>
<b>Working_Status_Code</b>

In order to have proper data, a SQL query is executed and data preparation service is used initially. Sample dataset after data preparation operation is shown in Table 6.4.

Table 6.4 Sample dataset for Employee Analysis

<b>Id</b>	<b>Children#</b>	<b>Age</b>	<b>Work Year</b>	<b>Off Day#</b>	<b>Education_Status_Code</b>
31013	0	30	2	0	8
31014	2	34	2	4	9
31015	3	51	3	5	9
30939	0	29	3	2	3
31002	1	25	1	1	3

When the number of clusters ( $k$  parameter) was selected 4 for K-Means++ algorithm then sample results obtained from the analysis are shown in Table 6.5.

Table 6.5 Employee Analysis results

Cluster 1	Cluster 2	Cluster 3	Cluster 4
30935 0 31 2 0 9	30703 1 50 25 0 3	30938 1 36 2 0 3	30735 1 49 21 38 2
30940 1 41 2 0 9	30626 2 54 20 0 3	30939 0 29 3 1 3	30758 0 47 21 20 2
30943 0 27 2 0 12	30706 1 57 25 0 2	30861 1 39 12 0 2	30770 1 45 20 28 3
30950 2 43 1 0 8	30722 2 46 23 0 2	30869 0 37 4 0 3	30855 1 40 13 46 2
.....	.....	.....	.....

### 6.2.1.2 Citizen Analysis

Citizens can be profiled according to their demographic properties, so local management decisions can be made according to these profiles.

Citizen profiles can be useful for election campaigns, promotions and charity events. For example, according to our sample analyses, age of 66.9% citizens are between 35 - 66. In this situation, it is possible to see that promotions which focus on adults should be done in this area or this area is not suitable for education charity events. If the mayor wants to perform election campaign then mayor should create strategies for majority to satisfy citizens. For example, new hiking trails may be more suitable than new tennis courts for these citizens.

“Citizens” database table in reduced form (Table 6.6) is used for citizen profiling analysis. There are 113558 records in dataset.

Table 6.6 Citizens database table (reduced form for Citizen Analysis)

Citizens
Date_of_Birth
Gender_Code
Marital_Status_Code
City

Table 6.6 Citizens database table (reduced form for Citizen Analysis) (Cont.)

Citizens
Nationality_Id

Initially, a SQL query is executed to calculate the ages of citizens using Date\_of\_Birth field, then discretization method is applied on the related age column to have categorical data. Sample dataset is shown in Table 6.7.

Table 6.7 Sample dataset for Citizen Analysis

Age	Gender	Marital Status	City	County	Nationality
15-35	F	Married	AYDIN	Y County	206
35-65	M	Married	AYDIN	Y County	206
15-35	M	Single	IZMIR	X County	206
65-80	M	Married	IZMIR	X County	206
...	...	...	...	...	...

FP-Growth algorithm as association rule mining technique is applied for this scenario with 3% minimum support value. Sample results are shown in Table 6.8.

Table 6.8 Citizen Analysis results

Support-Count-Rule	
<b>Single Rules</b>	%10 - 11404 - 15-35 %16.8 - 19086 - Married %18.5 - 20988 - 65-80 %41.7 - 47338 - F %66.9 - 76015 - 35-65 ....
<b>Double Rules</b>	%17.8 - 20207 - 65-80 ; X County of IZMIR %27.7 - 31503 - F ; 35-65 %28.1 - 31879 - F ; Married ....
<b>Trio Rules</b>	%21.3 - 24236 - F ; 35-65 ; Married %25.6 - 29123 - F ; 35-65 ; X County of IZMIR %26 - 29508 - F ; Married; X County of IZMIR ....
<b>Quadruple Rules</b>	%4.7 - 22352 - F ; 35-65 ; Married; X County of IZMIR ....

### 6.2.1.3 Staff Analysis

Local government staff can be profiled according to their demographic properties, so local management decisions can be made according to these profiles.

Three database tables (reduced form) given in tables 6.9, 6.10 and 6.11, were joined to have proper dataset for FP-Growth algorithm.

Table 6.9 Staff database table

<b>Staff</b>
<b>Id</b>
<b>Date_of_Birth</b>
<b>Start_Date</b>
<b>Education_Status_Id</b>
<b>Date_of_Leaving</b>
<b>City</b>
<b>Marital_Status_Code</b>

Table 6.10 Staff Family Members database table

<b>Staff Family Members</b>
<b>Personel_Id</b>
<b>Relationship_Code</b>

Table 6.11 Education Status database table

<b>Education Status</b>
<b>Code</b>
<b>Description</b>

Sample data set is shown in Table 6.12. The values in the age column are calculated by using Date\_of\_Birth field in a SQL query.

Table 6.12 Sample dataset for Staff Analysis

Children#	Age	Work Year	City	Marital_Status	Education_Status
0	40	3	IST	Married	BSc
0	28	3	ANK	Single	High School
2	51	26	KARS	Married	MSc
1	29	2	RIZE	Single	BSc
1	32	1	SIIRT	Single	High School

Age and Work Time columns are converted to categorical data using discretization data preparation technique. Sample results using 10% minimum support are shown in Table 6.13.

Table 6.13 Sample results of Staff Analysis

Support-Count-Rule	
<b>Single Rules</b>	%10.3 - 38 - 15-20 (Work Year) %15.7 - 58 - MSc %21.9 - 81 - BSc %24.6 - 91 - High School %22.7 - 84 - Married %33.8 - 125 - 1-5 (Work Year) %42.2 - 156 - 20-47 (Age) ...
<b>Double Rules</b>	%10 - 37 - BSc ; Married %11.4 - 42 - High School ; 45-55 (age) %13.5 - 50 - High School ; 20-47 (work) %11,6 - 43 - 2 ; 20-47 (work) %19.2 - 71 - 2 ; Married %31.9 - 118 - 45-55 (age) ; 20-47 (work) ....
<b>Trio Rules</b>	%10.8 - 40- High School; 45-55 (age); 20-47 (work) ; Married %13.2 - 49 - High School; 20-47 (work); Married %10.5 - 39 - Single ; 23-35 (age); 0 %10.3 - 38 - 23-35 (age); 0 ; 1-5 (work) %13 - 48 - 1 ; 45-55 (age); 20-47 (work) %13.5 - 50 - 1 ; 45-55 (age); Married %13.2 - 49 - 1 ; IZMIR ; Married %15.7 - 58 - 1 ; 20-47(work); Married %11.4 - 42 - 1-5 (work) ; IZMIR ; Married %11.4 - 42 - 1-5 (work); IZMIR ; 0 ...

Table 6.13 Sample results of Staff Analysis (Cont.)

Support-Count-Rule	
<b>Quadruple</b>	% 12.2 - 45 - 1 ; 45-55 (age); 20-47 (work); Married
<b>Rules</b>	% 15.4 - 57 - 45-55 (age); IZMIR ; 20-47 (work); Married
	...

### 6.2.2 Income/Expense Analyses

These analyses include tax payments like real estate, environmental and expense of moveable material logs analyses.

#### 6.2.2.1 Moveable Material Analysis

The aim of this scenario is to find material-year-supplier-amount relations. This scenario may be used to facilitate moveable material input process. For example, according to analysis, if 20.5% of logs 150.01.04.01 and 150.01.01.03 moveable codes are used together then, when user enters 150.01.04.01 code, 150.01.01.03 code will be showed on the screen automatically. This scenario will facilitate and accelerate input operations.

Four database tables (reduced form) which are shown in Table 6.14, 6.15, 6.16 and 6.17, were joined to have dataset for FP-Growth algorithm.

Table 6.14 Receipts database table

Receipts
<b>Id</b>
<b>Person_Id</b>
<b>Total_Amount</b>
<b>Process_Type_Code</b>
<b>Status_Code</b>
<b>Type_Code</b>
<b>Year</b>

Table 6.15 Receipt Details database table

<b>Receipt Details</b>
<b>Receipt_Id</b>
<b>Material_Id</b>

Table 6.16 Materials database table

<b>Materials</b>
<b>Id</b>
<b>Moveable_Material_Account_Code</b>

Table 6.17 Accounts database table

<b>Accounts</b>
<b>Code</b>

Sample dataset is shown in Table 6.18.

Table 6.18 Sample dataset for Moveable Material Analysis

<b>Receipt Id</b>	<b>Moveable Material Account Code</b>	<b>Year</b>	<b>Person Id</b>	<b>Total Amount</b>
90	150.03.04	2007	454587	626.4
97	150.01.01.04	2007	345642	150
97	150.01.03.01.08	2007	345642	150
97	150.01.03.02.05	2007	345642	150
97	150.01.05.02.01	2007	345642	150
102	254.01.05.09	2007	358473	258733.28
103	254.01.05.09	2007	358473	258733.28
106	150.12.02.02.01	2007	454519	590.6
106	150.12.03.02.03	2007	454519	590.6
106	150.12.03.05.04	2007	454519	590.6
109	150.05.02.01.10	2007	428313	19249.5
109	150.05.02.02.02	2007	428313	19249.5

Discretization operation is applied on ‘Total Amount’ field in order to have categorical data. In addition, the rows are joined according to Receipt\_Id. A sample dataset obtained after data preparation step is shown in Table 6.19.



Table 6.19 Prepared dataset for Moveable Material Analysis

Receipt Id	Moveable Material Account Code	Year	Person Id	Total Amount
90	150.03.04	2007	454587	0.01-1000
97	150.01.01.04, 150.01.03.01.08, 150.01.03.02.05, 150.01.05.02.01	2007	345642	0.01-1000
102	254.01.05.09	2007	358473	30000-388633
103	254.01.05.09	2007	358473	30000-388633
106	150.12.02.02.01, 150.12.03.02.03, 150.12.03.05.04	2007	454519	0.01-1000
109	150.05.02.01.10 150.05.02.02.02 253.03.01.99.01 253.03.01.99.02	2007	428313	1000-5000

FP-Growth algorithm is applied on prepared dataset with 2% minimum support. Sample results are shown in Table 6.20.

Table 6.20 Sample results for Moveable Material Analysis

Support-Count-Rule	
<b>Single Rules</b>	%2 - 88 - 150.01.05.01.03 %2.1 - 90 - 150.01.01.06 %2.1 - 93 - 150.01.01.07 %2.2 - 96 - 150.12.03.01.01 %2.2 - 98 - 422923 %2.3 - 101 - 410732 %18.6 - 814 - 2007 %24.2 - 1020 - 150.01.04.01 %25.3 - 1043 - 150.01.01.03 %42.7 - 1871 - 2008 %29.4 - 1287 - 2009 %9.3 - 407 - 2010 %14.3 - 627 - 1000-5000 %16.8 - 736 - 5000-30000 %64.4 - 2821 - 0,01-1000 ...
<b>Double Rules</b>	%6.4 - 282 - 150.01.01.03 ; 0,01-1000 %20.5 - 902 - 150.01.04.01 ; 150.01.01.03 %20.9 - 915 - 2009 ; 0,01-1000 ...

Table 6.20 Sample results for Moveable Material Analysis (Cont.)

<b>Support-Count-Rule</b>	
<b>Trio Rules</b>	%2.4 - 107 - 150.01.04.01 ; 150.01.01.03 ; 150.01.03.01.03 %2.5 - 110 - 457235 ; 150.01.01.03 ; 150.01.03.01.03 %5 - 221 - 150.01.01.03 ; 150.01.03.01.03 ; 0,01-1000 ...
<b>Quadruple Rules</b>	%2.4 - 107 - 457235 ; 150.01.01.03 ; 150.01.03.01.03 ; 0,01-1000 %2.4 - 106 - 150.01.04.01 ; 150.01.01.03 ; 150.01.03.01.03 ; 0,01-1000 ...

### 6.2.2.2 Estimation of Wages

Users can estimate the wages of employees by analyzing previous employee records through this scenario. To make prediction, the ages of employees, the duration of working as years, education level, job and wage are used. This scenario is useful to provide fairness. Because there will be concrete assessment criteria to determine wages.

Three database tables (reduced form) given in Table 6.21, 6.22 and 6.23 were joined to create proper dataset for related scenario.

Table 6.21 Staff database table (reduced form for Estimation of Wages)

<b>Staff</b>
<b>Id</b>
<b>Date_of_Birth</b>
<b>Start_Date</b>
<b>Education_Status_Id</b>
<b>Date_of_Leaving</b>
<b>Type_Code</b>

Table 6.22 Staff Movements database table

<b>Staff Movements</b>
<b>Personel_Id</b>
<b>Wage</b>
<b>Date</b>
<b>Job_Id</b>
<b>Type_Code</b>

Table 6.23 Jobs database table

<b>Jobs</b>
<b>Id</b>
<b>Description</b>

Sample dataset is shown in Table 6.24.

Table 6.24 Sample dataset for Estimation of Wages

<b>Age</b>	<b>Work Year</b>	<b>Education Status</b>	<b>Job</b>	<b>Wage</b>
37	14	9	Worker	54.97
52	28	2	Worker	64.79
54	20	3	Worker	59.85
43	20	3	Worker	59.85
45	21	2	Driver	60.92
43	21	2	Worker	60.92
45	21	2	Driver	60.92
48	25	3	Worker	63.78
57	25	2	Worker	63.03
50	25	2	Worker	63.43
34	4	3	Worker	48.44

Discretization is applied on Age and Wage columns. C4.5 algorithm is applied on sample dataset. Sample result is shown in Figure 6.1. Target attribute is Wage.

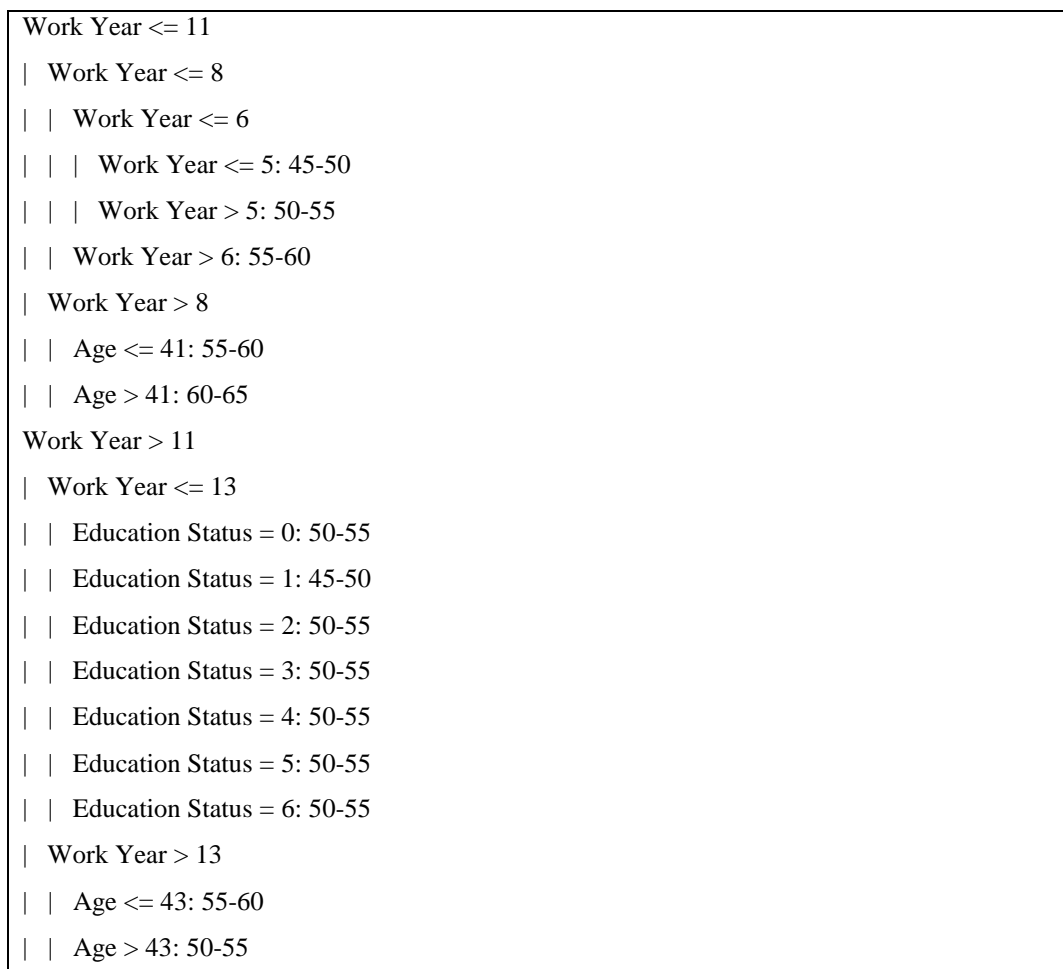


Figure 6.1 Sample decision tree for Estimation of Wages scenario.

### 6.2.2.3 Income Operations Analysis

The main aim of this scenario is to explore which types (MS, BIL, A, D, etc.) of which income operations (real estate, water, environmental cleaning etc.) are done frequently by which users on which months. This scenario is useful to prevent wrong operations. For example, according to analysis, User Name="Gözde Bakırlı" and Income Type Code = "EML" rule has 27% support value. If this user does any operation which Income Type Code="Sanitation Tax" then the system will warn him about operation. By this way, it is possible to prevent wrong operations.

Two database tables from two different schemas were used for this scenario. Database tables are shown in Table 6.25 and 6.26.

Table 6.25 Income Operations database table

<b>Income Operations</b>
<b>User_Id</b>
<b>Income_Type_Code</b>
<b>Type_Code</b>
<b>System_Date</b>

Table 6.26 System Users database table

<b>System Users</b>
<b>Id</b>
<b>User_Name</b>

Sample dataset for this scenario is shown in Table 6.27.

Table 6.27 Sample dataset for Income Operations Analysis

<b>Income Type Code</b>	<b>Type Code</b>	<b>Month</b>	<b>User Name</b>
EML	BEY	4	DERYA BİRANT
HU	TAH	2	GÖZDE BAKIRLI
HU	TAH	2	GÖZDE BAKIRLI
HU	TAH	2	GÖZDE BAKIRLI
HU	TAH	2	GÖZDE BAKIRLI
HU	TAH	2	GÖZDE BAKIRLI
HKP	TBG	2	FEVZİYE KAYHANLIGİL
HKP	TAH	2	FEVZİYE KAYHANLIGİL
HKP	TBG	2	PEYAMİ ÇOŞKUNLU
HKP	TAH	2	PEYAMİ ÇOŞKUNLU
HU	TAH	2	TURABİ REMZİOĞLU

FP-Growth algorithm with 2% minimum support value is applied on proper dataset to perform scenario. Sample results of this scenario are shown in Table 6.28.

Table 6.28 Sample results of Income Operations Analysis

Support-Count-Rule	
<b>Single Rules</b>	%2 - 147 - "MİRAY DEMİRCİTA"
	%2.1 - 156 - "ÖEA"
	%2.3 - 168 - "G"
	%3.5 - 255 - "BURÇAK SEDEREN"
	%3.7 - 268 - "YEKNUR PASPASLI"
	%5.1 - 369 - "MAZLUMCAN YAPRAKOĞLU"
	%4.2 - 302 - "A "
	%6.8 - 492 - "D "
	%7.5 - 542 - "GOİ"
	%7.8 - 569 - 7
	%7.9 - 574 - 8
	%8.2 - 594 - 9
	%8.2 - 596 - 11
	%8.3 - 602 - 10
	%9 - 656 - 3
	%13.8 - 1001 - "MS "
	%24.8 - 1802 - "ÇTV"
	%30.2 - 2306 - "GÖZDE BAKIRLI"
	%49.6 - 3604 - "EML"
	.....
<b>Double Rules</b>	%3.2 - 234 - "MAH" ; "GOİ"
	%2.4 - 178 - "BURÇAK SEDEREN " ; "EML"
	%2.3 - 169 - "YEKNUR PASPASLI " ; "EML"
	%3.1 - 224 - "TAH" ; "HÜ "
	%2.4 - 172 - "MSİ" ; "EML"
	%4.2 - 302 - "A " ; "EML"
	%2.1 - 156 - "ÖEA" ; "GOİ"
	%4.5 - 328 - "BİL" ; "EML"
	%4.5 - 329 - "BİL" ; "ÇTV"
	%2.1 - 152 - 1 ; "ÇTV"
	%4.3 - 314 - 1 ; "EML"
	%6.1 - 442 - "MS " ; "ÇTV"
	%6.9 - 498 - "MS " ; "EML"
	%5.2 - 376 - "S " ; "ÇTV"
	%6.5 - 472 - "S " ; "EML"
	%27 - 1905 - "GÖZDE BAKIRLI " ; "EML"
	....

### 6.2.3 Infrastructure Analyses

These kinds of analyses are created to analyze water and electricity consumptions and distribution of corporate foundation according to their depts.

### 6.2.3.1 Water Notice Analysis

The purpose of this scenario is to cluster citizens according to their water consumption. This scenario can be useful when municipal authority wants to warn citizens who consume water higher than limit.

Two database tables were joined for this scenario. These tables are shown in reduced form in Table 6.29 and 6.30.

Table 6.29 Subscribers database table

<b>Subscribers</b>
<b>Id</b>
<b>Fitment_Id</b>

Table 6.30 Notices database table

<b>Notices</b>
<b>Subscriber_Id</b>
<b>First_Year</b>
<b>First_Month</b>
<b>Last_Year</b>
<b>Last_Month</b>
<b>Additional_Consumption</b>
<b>Total_Consumption</b>

Sample dataset is shown in Table 6.31.

Table 6.31 Sample dataset for Water Notice Analysis

<b>Subscriber Id</b>	<b>Total Month</b>	<b>First Year</b>	<b>First Month</b>	<b>Last Year</b>	<b>Last Month</b>	<b>Total Consumption</b>
2	18	2009	2	2010	7	4
4	19	2009	1	2010	7	5
5	14	2009	2	2010	3	4
5	14	2009	2	2010	3	4
5	14	2009	2	2010	3	4
5	14	2009	2	2010	3	4
5	14	2009	2	2010	3	42

Table 6.31 Sample dataset for Water Notice Analysis (Cont.)

Subscriber Id	Total Month	First Year	First Month	Last Year	Last Month	Total Consumption
6	16	2009	2	2010	5	10
6	16	2009	2	2010	5	5
8	19	2009	1	2010	7	5
9	18	2009	2	2010	7	3
9	18	2009	2	2010	7	3

Total month is calculated as follows:

*Total Month =*

$$12 \times (Last\_Year - First\_Year) + (Last\_Month - First\_Month) + 1 \quad (6.1)$$

Rows are grouped according to Subscriber\_Id and also total amount is shared out the years by the total month. Last form of the dataset is shown in Table 6.32.

Table 6.32 Prepared dataset for Water Notice Analysis

Subscriber Id	2001	2002	2003	2004	2005	2006
342	0	0	0	0	0	170
22	0	7850	0	7000	7495	13742
12	0	267	0	0	31	136
123	0	0	0	128	76	65
554	85	0	0	0	0	0
67	0	53	0	0	147	128
74	0	0	0	0	0	0
23	0	1791	1999	2546	2141	2154
90	0	0	25	4	124	109
24	0	0	92	58	71	51
567	0	0	0	12	3	16
476	0	0	61	41	62	43

K-means++ algorithm is applied to dataset. Data is clustered into 7 clusters. Sample results are shown in Table 6.33.



Table 6.33 Sample results of Water Notice Analysis

	2001	2002	2003	2004	2005	2006
<b>Cluster 1</b>	0	0	0	0	0	170
	0	267	0	0	31	136
	0	0	0	128	76	65
	0	53	0	0	147	128
	....					
<b>Cluster 2</b>	66	38	86	84	88	84
	88	90	119	120	100	121
	101	113	133	130	90	79
	72	0	0	0	120	30
	...					
<b>Cluster 3</b>	0	7850	0	7000	7495	13742
	0	5994	6277	6442	9261	7074
	0	41	14977	10911	8894	9017
	0	0	0	11120	9624	11099
	...					
<b>Cluster 4</b>	0	1270	6437	6807	3065	1879
	0	5180	5426	5269	3843	4968
	0	4717	4276	3538	3298	4261
	0	4483	8841	4358	5645	3355
	...					
<b>Cluster 5</b>	0	1791	1999	2546	2141	2154
	0	1728	2117	3257	3106	2780
<b>Cluster 5</b>	0	29	73	0	9933	0
	0	0	4540	0	2248	2237
	...					

### 6.2.3.2 Electricity Consumption Analysis

The aim of this scenario is to detect outliers by analyzing electrical consumptions.

Two database tables given in Table 6.34 and 6.35 (in reduced form) were joined for this scenario.

Table 6.34 Receipts database table

Receipts
<b>Id</b>
<b>Date</b>

Table 6.34 Receipts database table (Cont.)

<b>Receipts</b>
<b>Amount_To_Be_Paid</b>
<b>Status_Code</b>

Table 6.35 Receipt Details database table

<b>Receipt Details</b>
<b>Receipt_Id</b>
<b>Account_Code</b>
<b>Economic_Code</b>
<b>Corporate_Code</b>

Example analysis result is shown in Table 6.36. It is possible to see that the highest consumption was done in August in 2007, while the least consumption was done in May in 2009. Reasons of these situations can be investigated to find out their reasons.

Table 6.36 Sample results of Electricity Consumption Analysis

<b>Month</b>	<b>Year</b>	<b>Total</b>
5	2009	35.94414
1	2009	1981.184
12	2009	3620.022
8	2007	4075.097

#### **6.2.4 Fraud Detection Analyses**

Cancellation operations at cash desks, personnel performances and fuel oil analyses belong to fraud detection analyses.

##### *6.2.4.1 Logs Analysis*

The main aim of this scenario is to determine who performs which transactions on which computer, on which days on which tables. This scenario can be useful for table update or transportation operations. For example if the support value of “Table Name=’Income’, Time=’08.00-13.00’ and Day=’Wednesday’” rule is 46% then we

can realize that on Wednesday at 08.00-12.00 critical operations should not be applied on “Income” table, this can cause a big problem. Because, generally personnel work on this table on Wednesday between 08.00 and 12.00.

Possible abnormalities can be caught by using pre-created general user profile information.

One database table given in reduced form in Table 6.37 is used to perform this scenario.

Table 6.37 Logs database table

<b>Logs</b>
<b>Date</b>
<b>Hour</b>
<b>Process</b>
<b>Computer</b>
<b>User_Id</b>
<b>Table_Name</b>

Hour column was set into categorical using discretization operation. Coding of Day columns is shown in Table 6.38.

Table 6.38 Day codes

<b>Code</b>	<b>Day</b>
<b>0</b>	Sunday
<b>1</b>	Monday
<b>2</b>	Tuesday
<b>3</b>	Wednesday
<b>4</b>	Thursday
<b>5</b>	Friday
<b>6</b>	Saturday

Sample dataset (100000 records) for the scenario is shown in Table 6.39.

Table 6.39 Sample dataset for Logs Analysis

Day	Hour	Process	Computer	User_Id	Table_Name
4	1-8	U	HESAP01	16	Deliveries
4	8-13	I	TAHSILAT01	23	Acquitances
0	13-19	I	HESAP01	16	Permits
5	1-8	U	TAHSILAT01	23	Accruals
5	1-8	I	HESAP01	16	Permits
6	1-8	D	HESAP01	16	Accruals
...	...	...	...	...	...

FP-Growth algorithm is applied on dataset with 2% minimum support. Sample results are shown in Table 6.40.

Table 6.40 Sample results of Logs Analysis

Support-Count-Rule	
<b>Single Rules</b>	%2.2 - 2215 - 368
	%2.2 - 2226 - 312
	%2.3 - 2265 - 218
	%2.8 - 2833 - "VEZNE0001"
	%2.8 - 2800 - "VEZNE6"
	%2.3 - 2278 - 254
	%3 - 3036 - "VEZNE2"
	%3.9 - 3875 - "Documents"
	%8.3 - 8293 - Accruals"
	%9 - 9030 - "Permits"
	%23.2 - 23201 - 13-19
	%33.5 - 33539 - 3
	%43.3 - 43320 - "I"
	%50.1 - 50085 - "U"
	%61.3 - 61316 - 8-13
.....	
<b>Double Rules</b>	%3 - 2965 - "Acquitantes" ; "I"
	%2.4 - 2378 - "Documents" ; 8-13
	%2.4 - 2425 - "Documents" ; 2
	%3.3 - 3342 - "Documents" ; "I"
	%5.1 - 5133 - "Accruals" ; 8-13
	%2.4 - 2379 - "Permits" ; 13-19
	%26.4 - 26381 - "I" ; 8-13
	....
<b>Trio Rules</b>	%2 - 2049 - "Documents" ; 8-13 ; "I"
	%2.1 - 2052 - "Documents" ; 2 ; "I"

Table 6.40 Sample results of Logs Analysis (Cont.)

<b>Support-Count-Rule</b>	
<b>Trio Rules</b>	%2.1 - 2142 - 0 ; 8-13 ; "Permits" %2.3 - 2284 - "Accruals" ; "U" ; 8-13 %9.2 - 9250 - 13-19 ; "I" ; 2 %12 - 12025 - 13-19 ; "U" ; 2 %11.3 - 11300 - 3 ; "I" ; 8-13 %12.8 - 12785 - 3 ; "U" ; 8-13 %14.5 - 14451 - "I" ; 8-13 ; 2 %17.1 - 17140 - "U" ; 8-13 ; 2
<b>Quadruple Rules</b>	%2.1 - 2139 - 0 ; 8-13 ; "Permits" ; "I" %2.3 - 2295 - 0 ; 2 ; "Permits"

#### 6.2.4.2 Fuel Oil Analysis

Outliers can be detected by analyzing fuel oil consumptions through this scenario. The benefit of this scenario is similar to “Electricity Consumption Analysis” scenario.

Three database tables given in Tables 6.41, 6.42 and 6.43 are used for this scenario.

Table 6.41 Receipts database table (reduced form for Fuel Oil Analysis)

<b>Receipts</b>
<b>Id</b>
<b>Date</b>
<b>Type_Code</b>
<b>Year</b>

Table 6.42 Receipt Details database table (reduced form for Fuel Oil Analysis)

<b>Receipt Details</b>
<b>Receipt_Id</b>
<b>Material_Id</b>
<b>Amount</b>

Table 6.43 Materials database table

<b>Materials</b>
<b>Id</b>
<b>Moveable_Material_Account_Code</b>
<b>Description</b>

Sample dataset is shown in Table 6.44.

Table 6.44 Sample dataset for Fuel Oil Analysis

<b>Month</b>	<b>Year</b>	<b>Amount</b>	<b>Moveable Material Account Code</b>
7	2007	68230.61	150.04.02.02
7	2007	18940	150.04.02.04
8	2007	29147	150.04.02.02
8	2007	21440	150.04.02.04
9	2007	87696	150.04.02.02
9	2007	20420	150.04.02.04
10	2007	7906	150.04.02.01

Local Outlier Factor (LOF) algorithm is applied as Outlier Detection algorithm to find outliers. Sample results are shown in Table 6.45.

Table 6.45 Sample results of Fuel Oil Analysis

<b>Month</b>	<b>Year</b>	<b>Amount (TL)</b>	<b>Moveable Material Account Code</b>
11	2007	131577.978	150.04.02.02

#### 6.2.4.3 Cash Desk Analysis

The goal of this scenario is to analyze cash desk operations that are performed by cashier and to detect outliers.

Three database tables from two different schemas were joined to have proper dataset for this scenario. Tables are shown in Table 6.46, 6.47 and 6.48.

Table 6.46 Acquittances database table

<b>Acquittances</b>
<b>Person_Id</b>
<b>Collector_Id</b>

Table 6.46 Acquittances database table (Cont.)

<b>Acquittances</b>
<b>Document_Type_Code</b>
<b>Status_Code</b>

Table 6.47 Citizens database table (reduced form for Cash Desk Analysis)

<b>Citizens</b>
<b>Id</b>
<b>Name</b>
<b>Surname</b>

Table 6.48 Collectors database table

<b>Collectors</b>
<b>Id</b>
<b>Person_Id</b>

Sample dataset for Cash Desk Analysis scenario is shown in Table 6.49.

Table 6.49 Sample dataset for Cash Desk Analysis

<b>Person Id</b>	<b>Collector Id</b>	<b>Number of Deletion</b>
99569	309	3
104211	289	3
154340	25	3
159584	25	4
168454	132	3
178427	289	4
...	...	...

LOF algorithm is applied to find outliers. Table 6.50 shows sample result of analysis. As shown below, the collector whose ID is 13, deletes records that belong to person whose ID is 22, 73 times. This is not normal. This situation should be asked collector.

Table 6.50 Sample results for Cash Desk Analysis

<b>Person Id</b>	<b>Collector ID</b>	<b>Number of Deletion</b>
22	13	73

## 6.2.5 Simplification, Verification and Similarity Analyses

These analyses include User Account and Accountancy Analyses.

### 6.2.5.1 User Account Analysis

The purpose of this scenario is to profile user accounts. According to this scenario answer of “which user generally when studies” question can be found. For example, “User Name=”Bakirli” Start Time=”08.00-13.00” rule is found with 24% support value. In this situation for example we can conclude that extra works should not be demanded from “Bakirli” between 08.00 and 12.00. Because, she works hard at these hours.

The database table shown in Table 6.51 is used for this scenario.

Table 6.51 User Accounts database table

<b>User Accounts</b>
<b>User_Name</b>
<b>Start_Date</b>
<b>Start_Time</b>
<b>Computer_Name</b>
<b>End_Date</b>
<b>End_Time</b>

Sample dataset for this scenario is shown in Table 6.52.

Table 6.52 Sample dataset for User Accounts Analysis

<b>User Name</b>	<b>Day</b>	<b>Start Date</b>	<b>Computer Name</b>	<b>End Date</b>	<b>End Time</b>
Bakirli	4	8-13	Accrual02	8-13	<5 min
Bakirli	4	8-13	Accrual02	8-13	<3 hour
Birant	4	8-13	Receiving01	8-13	<5 min
Birant	3	8-13	Receiving01	13-19	>4 hour
Bakirli	4	13-19	Receiving01	13-19	<1 hour
Birant	1	13-19	Receiving01	18-19	<1 hour
...	...	...	...	...	...



FP-Growth algorithm is applied for this scenario. Sample results are shown in Table 6.53.

Table 6.53 Sample results for User Accounts Analysis

<b>Support-Count-Rule</b>	
<b>Single Rules</b>	%2.1 - 1785 - Devim %2.1 - 1799 - Biran % 7 - 6002 - Accrual04 %7.1 - 6109 - 8 % 8.4 - 6201 - Accrual03 %10.2 - 8853 - 2 %12.1 - 10469 - 4 %13.9 - 12024 - 3 %14.9 - 12856 - 1 % 17.2 - 18004 - Receiving01 %26 - 28024 - Bakırlı %51.1 - 44176 - 8-13 ...
<b>Double Rules</b>	%2.9 - 2512 - 5 ; 8-13 %3.3 - 2886 - 7 ; 8-13 %6.1 - 5255 - 9 ; 0-8 %2.9 - 2495 - Bakırlı ; 13-19 %3.8 - 3281 - Birant; 8-13 %4.8 - 4185 - Birant ; 0 %3.5 - 3024 - 8 ; 0-8 %3.6 - 3084 - 8 ; 8-13 %3.8 - 3249 - 2 ; 13-19 %5.1 - 4388 - 2 ; 8-13 %3.3 - 2896 - 4 ; 0-8 %8.6 - 7419 - 4 ; 8-13 %2.1 - 1784 - 3 ; 13-19 %5.7 - 4909 - 3 ; 8-13 %6.2 - 5329 - 3 ; 0-8 %24 - 2495 - Bakirli ; 8-13

#### 6.2.5.2 Accountancy Analysis

The main aim of this scenario is to detect which accounts are entered together. Thus, accounts will be viewed on the screen automatically. This provides easy information entrance.

Two database tables were joined for this scenario. Database tables are shown in Table 6.54 and 6.55.

Table 6.54 Accountancy database table

<b>Accountancy</b>
<b>Id</b>
<b>Status_Code</b>
<b>Process_Type_Code</b>

Table 6.55 Accountancy Details database table

<b>Accountancy Details</b>
<b>Accountancy_Id</b>
<b>Account_Code</b>
<b>Type_Code</b>

Sample dataset (392426 lines) for this scenario is shown in Table 6.56.

Table 6.56 Sample dataset for Accountancy Analysis

<b>Dataset for Accountancy</b>
102B 333A
160B 103A
835A 150A 150B 905B 630B 103A 360A 830B 900A
102B 800A 805B 600A
905B 900A 102B 835A 630B 630B 830B 830B 160A 905B 900A
900A 630B 630B 630B 630B 630B 835A 905B 900A 630B 630B 905B 333A
830B 830B
360A 900A 830B 830B 830B 630B 835A 905B 360A 630B 630B 103A
905B 900A 830B 830B 360A 360A 103A 630B 630B 835A
360A 905B 900A 900A 630B 630B 630B 630B 630B 830B 830B 830B 830B
830B 905B

FP-Growth algorithm with 25% minimum support is applied on dataset. Sample results are shown in Table 6.57.

Table 6.57 Sample results for Accountancy Analysis

<b>Support-Count-Rule</b>	
<b>Single Rules</b>	%26.9 - 21145 - 805B %27.2 - 21412 - 800A %28.5 - 22437 - 600A %31.7 - 24986 - 360A %31.9 - 25107 - 102B %31.9 - 25107 - 102B %32.6 - 25661 - 630B %35.1 - 27614 - 835A %35.1 - 27635 - 900A %35.1 - 27635 - 905B %35.1 - 27641 - 830B %45.3 - 35633 - 103A ...
<b>Double Rules</b>	%26 - 20461 - 805B ; 600A %26.9 - 21137 - 805B ; 800A %26.2 - 20640 - 800A ; 600A %30.4 - 23955 - 630B ; 835A %30.5 - 23971 - 630B ; 900A %30.5 - 23971 - 630B ; 905B %30.5 - 23976 - 630B ; 830B %35.1 - 27613 - 835A ; 900A %35.1 - 27613 - 835A ; 905B %35.1 - 27635 - 900A ; 830B %35.1 - 27635 - 900A ; 905B %35.1 - 27635 - 905B ; 830B ...
<b>Trio Rules</b>	%35.1 - 27613 - 835A ; 900A ; 905B %35.1 - 27613 - 835A ; 900A ; 830B %35.1 - 27613 - 835A ; 905B ; 830B %35.1 - 27635 - 900A ; 830B ; 905B %26 - 20458 - 805B ; 600A ; 800A %30.4 - 23954 - 630B ; 835A ; 900A %30.4 - 23954 - 630B ; 835A ; 905B %30.5 - 23971 - 630B ; 900A ; 830B %30.5 - 23971 - 630B ; 900A ; 905B %30.5 - 23971 - 630B ; 905B ; 830B ...
<b>Quadruple Rules</b>	%30.5 - 23971 - 630B ; 900A ; 830B ; 905B %30.4 - 23954 - 630B ; 835A ; 900A ; 830B %30.4 - 23954 - 630B ; 835A ; 905B ; 830B ...

## **CHAPTER SEVEN**

### **CONCLUSION AND FUTURE WORK**

#### **7.1 Conclusion**

This thesis proposes a new decision-maker system for local municipalities. This decision-maker system includes five different analyses; socio-cultural analyses, income/expense analyses, infrastructure analyses, fraud detection analyses and simplification, verification and similarity analyses with multiple scenarios for local municipalities. This system provides that local municipalities can discover hidden patterns, relationships, changes, irregularities, and rules from large datasets. Local municipalities can perform their decision making process more rational, more accurate and faster through proposed decision-maker system.

This thesis proposes DTreeSim as a new approach to evaluate the similarity of DTs and to find the k-most similar trees. To achieve this purpose, three data mining techniques (classification, SPM, and KNN), involving five steps, were proposed. In contrast to previous studies, SPM is used to find compact and pruned forms of frequent branches in DTs. With this new approach, the paths of DTs are considered as sequences, and their branches are considered as sequence items. In the final step, the frequent sequences extracted by the SPM are used by the KNN algorithm with a new distance measure to find similarities. While the SPM technique is applied to classified data, the KNN algorithm is carried out on the results of the SPM, such that a re-mining process is performed twice. We have called this re<sup>2</sup>mining.

The applicability of our new approach is shown using data that have been collected on citizens' tax payments by a local municipality in Turkey over a 10-year period. Two novel similarity techniques (general and pieced) were applied to find the k-most similar trees. Experimental results show that our approach can be effectively used to find similarity in DTs. It could clearly be applied to different problems in different areas.

## **7.2 Future Work**

In the future, other SPM algorithms (e.g., SPADE, GSP) can be applied to find frequent sequences. In addition, an incremental approach will be developed to enable results to be updated, as new data become available. Furthermore, new scenarios can be added into decision-maker system.

## REFERENCES

- Adelakun, O. (2012). *The role of business intelligence in government: A case study of a Swedish municipality contact center*. M.Sc. Thesis, University West, Trollhättan.
- Ahmadvand, A. M., Bidgoli, B. M., & Akhondzadeh, E. (2010). A hybrid data mining model for effective citizen relationship management: a case study on Tehran municipality. *e-Education, e-Business, e-Management, and e-Learning, 2010 IC4E '10 International Conference on*. 277 – 281.
- Arthur, D., & Vassilvitskii, S. (2007). K-means++: The advantages of careful seeding. *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. 1027 - 1035
- Bakirli, G., Birant, D., Mutlu, E., Kut, A., Denктаş, L., & Çetin, D. (2012). Data mining solutions for local municipalities. *The Electronic Journal of e-Government, 10* (2), 97 – 106.
- Bakirli, G., & Birant, D. (2015) DTreeSim: A new approach to compute decision tree similarity using re-mining. *Turkish Journal of Electrical Engineering & Computer Sciences, (article in press)*
- Breunig, M. M., Kriegel, H. P., Ng, R. T., & Sander, J. (2000). LOF: Identifying density-based local outliers. *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*. 93 – 104.
- Dehmer, M., Emmert-Streib, F., & Kilian, J. (2006). A similarity measure for graphs with low computational complexity. *Applied Mathematics and Computation, 182*, 447 – 459.

- Demiriz, A., Ertek, G., Atan, T., & Kula, U. (2011). Re-mining item associations: methodology and a case study in apparel retailing. *Decision Support Systems*, 52 (1), 284 - 293.
- Deng, H. (2014). *Interpreting tree ensembles with inTrees*. Technical Report, arXiv:1408.5456.
- Deng, H., Runger, G., Tuv, E., & Bannister, W. (2014). CBC: An associative classifier with a small number of rules. *Decision Support Systems*, 59, 163–170.
- Dogra, I. S. (2014). *Improving and maintaining prediction accuracy in agent based modeling systems under dynamic environment*. M.Sc. Thesis, University of Windsor, Windsor.
- D'Silva, M.R., & Vora D. (2014). Intelligent recommendation system using clustering and closed sequential pattern mining. *International Journal of Computer Science Engineering and Information Technology Research*, 4 (1), 133 - 140.
- Dudani, S.A. (1976). The distance-weighted k-nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, SMC-6 (4), 325 - 327.
- Exarchos, T.P., Papaloukas C., Lampros, C., & Fotiadis, D.I. (2008). Mining sequential patterns for protein fold recognition. *Journal of Biomedical Informatics*, 41 (1), 165 - 179.
- Fiol-Roig, G., Miro-Julia, M., & Isern-Deya A.P. (2010). Applying data mining techniques to stock market analysis. *Advances in Intelligent and Soft Computing*, 71 519 - 527.
- Fletcher, S., & Islam, M.Z. (2015). An anonymization technique using intersected decision trees. *Journal of King Saud University – Computer and Information Sciences*, 27 (3), 297-304.

- Fournier-Viger, P., Wu, C., & Tseng, V.S. (2013). Mining maximal sequential patterns without candidate maintenance. *Lecture Notes in Computer Science*, 8346, 169 - 180.
- Fournier-Viger, P., Gomariz, A., Gueniche, T., Soltani, A., Wu, C., & Tseng, V.S. (2014). SPMF: A java open-source pattern mining library. *Journal of Machine Learning Research*, 15, 3389 - 3393.
- Haettenschwiler, P. (1999). Neues anwenderfreundliches konzept der entscheidungsunterstützung. *Gutes Entscheiden in Wirtschaft, Politik und Gesellschaft*. Zurich: vdf Hochschulverlag AG 189-208
- Han, J., Pei, J., Yin, Y., & Mao, R. (2004). Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining and Knowledge Discovery*, 8, 53 – 87.
- Ishwaran, H., & Rao, J. (2009). Decision trees, advanced techniques in constructing. In M. Kattan (Ed.), *Encyclopedia of medical decision making*, (328-332). Thousand Oaks, CA: SAGE Publications.
- Islam, M.Z. (2008). *Privacy preservation in data mining through noise addition*. Ph.D. Thesis, University of Newcastle, Newcastle.
- Islam, M.Z., Barnaghi, P.M., & Brankovic, L. (2003). Measuring data quality: predictive accuracy vs. similarity of decision trees. *6th International Conference on Computer & Information Technology*, 457 - 462.
- Kotsiantis, S.B. (2007). Supervised machine learning: A review of classification techniques. *Informatica*, 31, 249 - 268.



- Kumari, A., Prasad, U., & Bala P.K. (2013). Retail forecasting using neural network and data mining technique: A review and reflection. *International Journal of Emerging Trends & Technology in Computer Science*, 2 (6), 266 - 269.
- Last, M., Maimon, O., & Minkov, E. (2002). Improving stability of decision trees. *International Journal of Pattern Recognition and Artificial Intelligence*, 16 (2), 145 - 159.
- Liu, J., & Yin, J. (2001). Towards Efficient data re-mining (DRM). *Lecture Notes in Computer Science*, 2035, 406 - 412.
- Ma, Z., & Kaban, A. (2013). K-Nearest neighbours with a novel similarity measure for intrusion detection. *Computational Intelligence (UKCI) 13th UK Workshop on*, 266 - 271.
- Ma, S., Tang, S., Yang, D., Wang, T., & Han, T. (2004). Combining clustering with moving sequential pattern mining: a novel and efficient technique. *Lecture Notes in Computer Science*, 3056, 419-423.
- MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of 5<sup>th</sup> Berkeley Symposium on Mathematical Statistics and Probability*, 281 – 297
- Ntoutsi, I., Kalousis, A., & Theodoridis, Y. A. (2008). A general framework for estimating similarity of datasets and decision trees: exploring semantic similarity of decision trees. *SIAM International Conference on Data Mining*, 810 - 821.
- Olaiya, F., & Adeyomo, A.B. (2012). Application of data mining techniques in weather prediction and climate change studies. *International Journal of Information Engineering and Electronic Business*, 4 (1), 51 - 59.

- Papagelis, A., & Kalles, D. (2001). Breeding decision trees using evolutionary techniques. *18th International Conference on Machine Learning*, 393 - 400.
- Papagelis, A., & Kalles, D. (2000). GATree: Genetically evolved decision trees. *Proceedings of the 12th IEEE International Conference on Tools with Artificial Intelligence*, 203–206.
- Peahringer, B., & Witten, I.H. (1997). *Improving bagging performance by increasing decision tree diversity*, Kluwer Academic Publishers. Boston.
- Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U. et al. (2001). PrefixSpan: Mining sequential patterns by prefix-projected growth. *Proceedings of the 17th International Conference on Data Engineering*, 215 - 224.
- Pekerskaya, I., Pei, J., & Wang, K. (2006). Mining changing regions from access-constrained snapshots: A cluster-embedded decision tree approach. *Journal of Intelligent Information Systems*, 27 (3), 215-242.
- Perner, P. (2013). How to compare and interpret two learnt decision trees from the same domain? *27th International Conference on Advanced Information Networking and Applications Workshops*, 318 - 322.
- Perner, P. (2011). How to interpret decision trees? *Lecture Notes in Computer Science*, 6870, 40-55.
- Poles, S., & Margonari, M. (2009). A modeFRONTIER application: data mining the Italian municipalities, *Newsletter EnginSoft*, 6 (2), 31 – 37.
- Power, D. J. (2002). *Decision support systems: Concepts and resources for managers*. Quorum Books: Westport, Connecticut.

- Seni, G., & Elder, J. F. (2010). *Ensemble methods in data mining: Improving accuracy through combining predictions*. Morgan and Claypool Publishers.
- Solomon, S., Nguyen, H., Liebowitz, J., Agresti, W. (2006). Using data mining to improve traffic safety programs. *Industrial Management & Data Systems*, 106 (5), 621 - 643.
- Souza, F.T., & Wang, Z. (2010). A data mining approach to predict mass movements induced by seismic events in Sichuan, China. *Natural Computation (ICNC), 2010 Sixth International Conference on*, 1172 - 1177.
- Song, M., & van der Aalst, W. M. P. (2008). Towards comprehensive support for organizational mining. *Decision Support Systems*, 46 (1), 300 – 317.
- Spielman, S.E., & Thill, J. (2007). Social area analysis, data mining and GIS. *Computers, Environment and Urban Systems*, 32 (2) 110 - 122.
- Teixeira, R., Afonso, F., Oliveira, B., Machado, J. Abelha, A. Santos, M. F., et al. (2015). Decision support in e-government – a pervasive business intelligence approach. *Advances in Intelligent Systems and Computing*, 354, 155-166
- Telaar, D., & Fuhs, M. C. (2013). Accent- and speaker-specific polyphone decision trees for non-native speech recognition. *Proceedings of the 14th Annual Conference of the International Speech Communication Association*, 3313-3316.
- Thangamani, M., Thangaraj, P., & Bannari (2013). Automatic medical disease treatment system using datamining. *Information Communication and Embedded Systems (ICICES), 2013 International Conference on*, 120 - 125.
- Theodoridis, S., & Koutroumbas, K. (2009). *Pattern recognition*. (4th ed.). Elsevier Inc.

- Tseng, V.S., & Lee C.H. (2009). Effective temporal data classification by integrating sequential pattern mining and probabilistic induction. *Expert System with Applications*, 36 (5), 9524 - 9532.
- Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Fransisco, CA, USA: Morgan Kaufmann Publishers.
- Quinlan, J. R. (1996). Improved use of continuous attributes in C4.5. *Journal of Artificial Intelligence Research*, 4 (1), 77 - 90.
- Wu, X., Kumar, V., Quinlan, J.R., Ghosh, J., Yang, Q., Motoda, H., et al. (2008). Top 10 algorithms in data mining. *Knowledge and Information System*, 14 (1), 1 - 37.
- Yoshida, T. (2008). Term usage difference in a single dataset through collaborative registration. In A. M. Columbus, (Ed.). *Advances in psychology research*. (153-170). Nova Science Publishers: Hauppauge, NY, USA.
- Zager, L. A., & Verghese, G. C. (2008). Graph similarity scoring and matching. *Applied Mathematics Letters*, 24 (1), 86 – 94.
- Zhang, X., & Jiang, S. (2012). A splitting criteria based on similarity in decision tree learning. *Journal of Software* 7 (8), 1775 - 1782.
- Zhao, X., Xiao, C., Lin, X., & Wang, W. (2012). Efficient graph similarity joins with edit distance constraints. *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*, 834 – 845.

## APPENDICES

### APPENDIX-1: ABBREVIATIONS

<b>Abbreviation</b>	<b>Description</b>
DW	Data Warehouse
DM	Data Mart
DT	Decision Tree
F	Forest
TS	Tree Sequence
BS	Branch Sequence
FS	Frequent Sequence
DT	Decision Tree
DTreeSim	Decision Tree Similarity
GATree	Genetic Algorithm Tree
SPM	Sequential Pattern Mining
KNN	K-Nearest Neighbor
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
FP-Growth	Frequent Pattern-Growth
FP-tree	Frequent Pattern Tree
ID3	Iterative Dichotomiser 3
CART	Classification And Regression Tree
CHAID	Chi-squared Automatic Interaction Detector
ETL	Extract, Transform, Load
TSL	Tree-to-Sequence Linearization
ERP	Enterprise Resource Planning
MIS	Management Information System
CRM	Customer Relationship Management
SPMF	Sequential Pattern Mining Framework