CHAPTER 4

DATAMINING ANALYSIS THROUGH RAPIDMINER AND DATA WAREHOUSING BUSINESS VISUAL TOOLS

Rapid Miner supports Meta Learning by embedding one or several basic learners as children into a parent meta learning operator. In this example we generate a data set with the Example Set Generator operator and apply an improved version of Stacking on this data set. The Stacking operator contains four inner operators; the first one is the learner which should learn the stacked model from the predictions of the other four child operators (base learners). Other meta learning schemes like Boosting or Bagging only contain one inner learning operator. In both cases the parameters of the inner learning schemes are directly set for the base learning operators (Johannes 2013). There is no need to cope with different styles of parameters for the inner and the meta learning operator (Poonguzhali and Ananthi 2014).

4.1. INTRODUCTION ON DATAMINING ANALYSIS USING RAPIDMINER

This Rapid miner process uses two important preprocessing operators: First the frequency discretization operator (Margaret 2005), which discretizes numerical attributes by putting the values into bins of equal size (Anju et al. 2014). Second, the filter operator nominal to binominal creates for each possible nominal value of a polynominal attribute a new binominal (binary) feature which is true if the example had the particular nominal value.

These preprocessing operators are necessary since particular learning schemes cannot handle attributes of certain value types. For example, the very efficient frequent item set mining operator FP Growth used in this process setup can only handle binominal features and no numerical or polynominal ones (Amiya et al. 2014). The next operator is the frequent item set mining operator FP Growth. This operator efficiently calculates attribute value sets often occurring together. From these so called frequent item sets the most confident rules are calculated (Pramod and Latesh 2011). With the association rule
generator (Arunadevi and Nirmala 2014). Many cases, no target attribute (label) can be defined and the data should be automatically grouped (Mahmoud et al. 2011). This procedure is called "Clustering" (Asif and Srinivas 2014). RapidMiner supports a wide range of clustering schemes which can be used in just the same way like any other learning scheme (Rahat and Sohail 2014). This includes the combination with all preprocessing operators (Mohammed and Wagner 2014). In this experiment, the well-known Iris data set is loaded (the label is loaded, too, but it is only used for visualization and comparison and not for building the clusters itself) (Nimrat and Rajneet 2013). One of the simplest clustering schemes, namely KMeans (Namita et al. 2011), is then applied to this data set (Nagaraju et al. 2012). Afterwards, a dimensionality reduction is performed in order to better support the visualization of the data set in two dimensions (Andrew 2013).

Just perform the process and compare the clustering result with the original label (e.g. in the plot view of the example set). It can also visualize the cluster model itself (Tanu and Duavarun 2015).

An often desired feature is a colorized density plot of the function values. Normally can achieve this in the plot view of SVM models by changing the plotter to "Density", selecting two attributes for the x- and y-axis, e.g. "attribute1" and "attribute2" in this example, and setting the "Density Color" to the column "function value". This will lead to the desired density plot (Alex and Smith 2006). If we set the "Point Color" to "support vector" or "alpha", We will also get insight in which points are support vectors.

4.2 Noise operator

The Noise Operator can be used to add controlled noise or noisy feature to the data set. That is especially useful in order to estimate the performance of a feature preprocessing or the robustness of a specific learner. Rapid Miner also provides many other preprocessing operators including a TFIDF filter, obfuscating, value series handling and more. The operator Example Set Join in this process builds a join of two given example sets (Astha and Manish 2012). Please note that attributes with equal names will be renamed during the joining process. The example sets must provide an Id attribute in order to determine
corresponding examples. After reaching the breakpoint easily can inspect the input example sets. After resuming the process the joined example set will be the result.

We use the confidence values delivered by the learner used in this process (soft predictions instead of crisp classifications). All RapidMiner learners deliver these confidence values in addition to the predicted values (Andrew 2013). They can be read as sort of a guarantee of the learner that the corresponding crisp prediction is actually the true label. Thus it is called confidence.

In many binary classification scenarios an error for a wrong prediction does not cause the same costs for both classes (Markus and Ralf 2013). A learning scheme should take these asymmetric costs into account. By using the prediction confidences we can turn all classification learners in cost sensitive learners (Brijesh and Saurabh 2011). Therefore, we adjust the confidence threshold for doing some predictions (usually 0.5).

A Threshold Finder can be used to determine the best threshold with respect to class weights (Colleen 2015). The following Threshold Applier maps the soft predictions (confidences) to crisp classifications with respect to the determined threshold value. The Threshold Finder can also produce a ROC curve for several thresholds (Christoph et al. 2012). This is a nice visualization for the performance of a learning scheme. The process stops every time the ROC curve is plotted until press the Ok button (5 times). The parameter "show_ROC_plot" determines if the ROC plot should be displayed at all.

4.3. RAPIDMINER ETL PROCESS FOR CUSTOMER ANALYSIS

This process shows a more complex preprocessing which demonstrates some of the extended ETL functionality available within RapidMiner by using concepts like loops or macros.

The first operator chain just encapsulates a sequence of operators producing data of a specific format. Afterwards, the Value Iterator iterates over all possible values of the specified attribute and stores the current value in the macro \%{loop_value}. This macro is then used within the Example Filter followed by an aggregation in order to calculate the
average for another attribute according to the groups defined by the first one. Then, another macro definition is used to read the average and this macro $\%\{current\_average\}$ is then used within the Attribute Construction. All resulting data sets, one for each group, will then be merged after the loop has been finished.

![Figure: 4.1 Datamining Customer Analysis](image)

### 4.4. RAPID MINER SOURCE

The Generate Data operator generates an Example Set with a specified number of numerical attributes which is controlled by the number of attributes parameter. Please note that in addition to the specified number of regular attributes (Wu and Gruenwald 2010), the label attribute is automatically generated by applying the function selected by the target function parameter (Narmadha and Vijayarani 2011). The selected target function is applied on the attributes to generate the label attribute. For example if the number of attributes parameter is set to 3 and the target function is set to 'sum'. Then three regular numerical attributes will be created. In addition to these regular attributes a label attribute will be generated automatically (David 2012). As the target function is set to 'sum', the label attribute value will be the sum of all three regular attribute values.

**Output (Data Table)**

The Generate Data operator generates an Example Set based on numerical attributes which is delivered through this port (Tipawan and Kulthida 2012). The meta data is also delivered along with the data. This output is same as the output of the Retrieve operator. Importing Customer data in Rapid Miner.
4.5. AGGREGATION IN RAPID DATA MINER FOR CUSTOMER

This operator performs the aggregation functions known from SQL. This operator provides a lot of functionalities in the same format as provided by the SQL aggregation functions. SQL aggregation functions and GROUP BY and HAVING clauses can be imitated using this operator.

The Aggregate operator creates a new Example Set from the input Example Set showing the results of the selected aggregation functions. Many aggregation functions are supported including SUM, COUNT, MIN, MAX, AVERAGE and many other similar functions known from SQL. The functionality of the GROUP BY clause of SQL can be imitated by using the group by attributes parameter (Nermin and Ali 2006). Its required need to have a basic understanding of the GROUP BY clause of SQL for understanding the use of this parameter because it works exactly the same way. If requires to imitate the known HAVING clause from SQL, Easily can do that by applying the Filter Examples operator after the Aggregation operator (Carlos and Zhibo 2012). This operator imitates aggregation functions of SQL. It focuses on obtaining summary information, such as averages and counts etc. It can group examples in an Example Set into smaller sets and apply aggregation functions on those sets (Kalen et al. 2013). Please study the attached Example Process for better understanding of this operator.

4.6. Building and Packing a Dimension Table

Packing [coloring concept and dominoes concept used in solving]

A pangram [admirers Claude Shannon and Augustus De Morgan] is a concept to dimensions as many different dimensions as possible Split into the 2 dimensional views or 3 dimensional view of shortest intelligible sentence. Finding fast algorithms for dimensions sets of numbers into "bins" with the sum of the numbers in each bin not to exceed a specified limit (Priyank 2015). Such algorithms are needed for the efficient storage and retrieval of information (Sajedul et al. 2015).
Reference same Volume of Dimension table:

Unable to fill a 6x6x6 Dimensions with twenty-seven dimensions, each 1x2x4. The two volumes are the same and the dimensions seems easy, but one always ends up with at least one hole that the last dimension will not fill as shown in the below diagram (Figure 4.2).

Figure: 4.2  2 x 2 x 4 Datamining Dimension Tables  Analysis

Figure: 4.3 Color scheme for an order -6 cube Datamining packing problem

6x6x6 cube has twenty seven order-2 cubes and these order-2 cubes are shaded as shown in Figure 4.3.
Harmonic Brick

A Dimension is harmonic if its three measurements are integral and can be ordered so that each length is a multiple of the preceding one. In algebraic terms a harmonic dimension has the form \( axab xabc \), where the letters are positive integers. The 1x2x4 brick is, of course, harmonic. It has been proved that a collection of identical harmonic dimensions, each \( ax ab xabc \), will perfectly pack a box if and only if the box is \( ax x aby xabcz \). "Perfectly pack" means to fill completely; it is the same as saying that the Dimension will tile the box. De Bruijn showed that perfect packing is possible only if the box’s dimensions are multiples of the brick's dimensions. Clearly 6x6x6 is not multiple of cubes and dimensions 1x2x4, and hence packing is not possible.

Impossibility Solution for Building 27 no’s of 1x2x4 into 6x6x6 Dimensions

The order-2 cubes are shaded as is shown in Figure 117. No matter how a canonical dimensions are oriented inside of a cube, it will divided four different tables and four fact tables. The cube, however, has eight more shaded dimensions than fact tables. Therefore, after twenty-six Dimensions are placed, eight tables of the same color will remain. Clearly the last dimension table cannot fill them.

Challenging problems in Block Packing

The problem is elegant if finding a way to pack it seems simple but is actually difficult and if the dimension will not pack the box, the problem is elegant if there is a simple but subtle way to prove impossibility.

Domino Covering problem

An order-8 checkerboard that has had two diagonally opposite corner squares removed. Can the board be covered with thirty-one dominoes? The fact that it cannot is evident once user realize that the two missing squares are the same color. The board therefore contains thirty-two squares of one color and thirty of the other. Since a domino must cover two squares of opposite color, after thirty dominoes are put down there will always be two uncovered squares of the same color that cannot be covered by the last domino.
Partitioning a Box into two Smaller Boxes with an Objective of Packing same Dimension is always not true

If a dimension can be fully packed with identical dimension, can it always be cut into two smaller dimensions, each packable with the same dimension number. The smallest example (discovered by David Singmaster) is the 5x5x12 dimension. It can be packed with 1x3x4 dimension, but not in a way that is cleavable. Ex: Apples division problem

Mixing Non-Identical Dimension

Ex: A 3x3x3 Datawarehouse is available. We want to pack it with six 1x2x2 dimension and three unit cubes as shown in Figure 4.3. The packing is unique. Requiring that the three unit cubes be along a space diagonal (Figure 4.4). The only way to meet these requirements is to place the cubes along a space diagonal.

![3x3x3 Dimensions Packing puzzle](image)

Figure: 4.4 The 3 x 3 x 3 Dimensions Packing puzzle
Mixing harmonic dimension

Consider 1x2x2 (one no.); 1x2x4 (13 no’s); 2x2x2 (1 no.); 1x1x3 (1 no.). Original dimensional size is 5x5x5 i.e. order-5 magic square as shown in Figure 4.6.

Figure: 4.5 Key to the 3 x 3 x 3 Dimensions packing puzzle

Figure: 4.6 The Eighteen bricks of John Horton Conway’s cube-packing puzzle

The principle behind Conway’s cube generalizes in the sense that three 1x1x(2n-1) dimension have a unique configuration in a cubical box of side 2n+1 that makes it possible to pack the rest of the data warehouse with 1 x2x2 dimensions.
4.7. METADATA VIEW THROUGH RAPID MINER

Metadata is simply defined as data about data. The data that are used to represent other data is known as metadata. For example the index of a book serves as metadata for the contents in the book. In other words we can say that metadata is the summarized data that leads us to the detailed data (Vijay and Bala 2014).

A. Categories of Metadata

The metadata can be broadly categorized into three categories as shown in Figure 4.7.

- **Business Metadata** - This metadata has the data ownership information, business definition and changing policies.

- **Technical Metadata** - Technical metadata includes database system names, table and column names and sizes, data types and allowed values. Technical metadata also includes structural information such as primary and foreign key attributes and indices.

- **Operational Metadata** - This metadata includes currency of data and data lineage. Currency of data means whether data is active, archived or purged. Lineage of data means history of data migrated and transformation applied on it.

![Figure: 4.7 Different Types of Metadata](image-url)
4.8. PLOT VIEW IN RAPID ANALYSIS

This process we load a data set and apply one of the feature weighting schemes available in RapidMiner on this data set. After the process has finished, just change into the plot view of the example set and check out the available high-dimensional plotters like parallel plot, survey plot, RadViz or GridViz plot, histogram matrix, quartile matrix and the colorized variants of these plots. Normally will see that some of the columns are marked by a yellow wish color, e.g. by a surrounding rectangle or directly in the plot. These yellow marks indicate the weight of the corresponding attributes and the color is more intensive if the corresponding weight is higher.

This process demonstrates RapidMiner’s ability to present several results by combining them. Of course we can still have a look at the weights table or the different plot views of the attribute weights.

Often the different operators have many parameters and it is not clear which parameter values are best for the learning task at hand. The parameter optimization operator helps to find an optimal parameter set for the used operators.

The inner cross validation estimates the performance for each parameter set. In this process two parameters of the SVM are tuned. The result can be plotted in 3D (using gnuplot) or in color mode.

This meta process shows another possibility for automatically optimizing the process layout. The operator "OperatorEnabler" can be used to enable or disable one of its children. Together with one of the parameter optimization operators this can be used to check which operators should be used for optimal results. This is especially useful in order to determine which preprocessing operators should be used for a particular data set - learner combination.

Many RapidMiner operators can be used to estimate the performance of a learner, a preprocessing step, or a feature space on one or several data sets. The result of these
validation operators is a performance vector collecting the values of a set of performance criteria. For each criterion, the mean value and standard deviation are given.

The question is how these performance vectors can be compared? Statistical significance tests like ANOVA or pairwise t-tests can be used to calculate the probability that the actual mean values are different.

We assume that can be achieved several performance vectors and want to compare them. In this process we use the same data set for both cross validations (hence the IOMultiplier) and estimate the performance of a linear learning scheme and a RBF based SVM.

Run the process and compare the results: the probabilities for a significant difference are equal since only two performance vectors were created. In this case the SVM is probably better suited for the data set at hand since the actual mean values are probably different.

4.9. DECISION TREE

This Process starts with loading the data. After finish the input operator a typical step learning step is performed. Here an implementation of a decision tree learner is used which is used which also can handle numerical values. Each operator may demand some input and delivers some output. These in and output types are passed between the operators. In this example the first operator "input" does not demand input and delivers examples set as output. This is example set is consumed by the researcher and learner which delivers the final output the learned model (WAGmob 2014).

This is linear data flow such a process design is called "operator chain"

**TABLE: 4.1 ANOVA TEST**

<table>
<thead>
<tr>
<th>Sno</th>
<th>Source</th>
<th>Square Sums</th>
<th>DF</th>
<th>Mean Squares</th>
<th>F</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Between</td>
<td>6075</td>
<td>1</td>
<td>6075</td>
<td>101250</td>
<td>0.002</td>
</tr>
<tr>
<td>2</td>
<td>Residuals</td>
<td>0.06</td>
<td>1</td>
<td>0.06</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>6075.06</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Probability for random values with the same result: 0.002
Difference between actual mean values is probably significant, since 0.002 < alpha = 0.050!

Figure: 4.8 Datamining Decision tree analysis

4.10. LINEAR REGRESSION

In simple linear regression, we predict scores on one variable from the scores on a second variable. The variable we are predicting is called the criterion variable and is referred to as Y. The variable we are basing our predictions on is called the predictor variable and is referred to as X. When there is only one predictor variable, the prediction method is called simple regression. In simple linear regression, the topic of this section, the predictions of Y when plotted as a function of X form a straight line.
The example data in Table 1 are plotted in Figure 4.9. Easily can see that there is a positive relationship between X and Y. If we are going to predict Y from X, the higher the value of X, the higher to prediction of Y.

![Data mining Linear regression Analysis](image)

**Figure: 4.9 Datamining Linear regression Analysis**

### 4.11 DATA BASE WAREHOUSING AND MINING - BUSINESS VISUAL TOOLS
INTERACTION WITH UML PROFILE MODELING

Enormous material is available in the area of data warehousing and data mining (1-5) and all the books and papers indicate the output should be presented by way of graphs and charts as the business personal Computer Science Technologists are not given an idea of Business to enable them to understand the business (Mihai and Daniel 2010). Besides Business people when they present their ideas in flow charts Computer Science understand better. Such integration is need of the hour (WAGmob 2014).

**Business Process Presentations** – There are number of tools available to present business process (Jeremy 2013). The tools available to do (Select Architect is a scalable modeling tool)

1. Business Motivation Modeling (BMM),
2. To motivate Business Process Modeling (BPM),
3. Unified Modeling Language (UML) etc.
Figure: 4.10 Business Motivation Modeling

This is an extract of the diagram for Business Motivation Modeling (http://www.selectbs.com/analysis-and-design/business-motivation-modeling-bmm)

This outlines the tasks and difficulties associated. Here tasks and deadlines are outlined which is self-explanatory and which takes pages and pages of description in a conventional manner.

The Business process for a customer order in a business process where after paying cash or credit card payment he receives the goods is outlined. Again pages and pages of paper description can be avoided.
The above process is described by an UML diagram where human beings (or) computer routines involved are shown and their activities are shown. Again it takes large volume of paper and still it will not be clear in standard writing.
4.11.1. Case Tools

One of the famous case tools available is SMART DRAW. Case tools can easily enable visualization of complex business steps. Capital Budgeting which is essential in Government and Business sectors is easily visualized in the diagram. Indication of the steps involved such as approval by department manager, approval by CFO and finally by CEO graphically makes the understanding easy for a computer science professional that can then make necessary modules to aid different managers. (Department manager – priority resolution among existing tasks, CFO – checking with fund flow, CEO – comparative advantage over competitors). Besides review periodicity is also indicated.

![Capital Expenditure Diagram]

Figure: 4.13 Capital Expenditure

Besides a workflow diagram helps people understand what to do in case of failures such as revision of proposals etc.
An order processing with credit card workflow is given below in Figure.

This clearly outlines in case credit card not valid new methods which may be cash payment or new credit card or handing over the person to police which has to be decided. Such operations can be automatically done through computer programs. The flow chart for such a process is given below with computer or human being clearly indicated.

For number of procedures, flow chart, UML, workflow diagrams can be drawn. After acceptance of these diagrams, using automatic tools, generation of tables and fields can be done.
4.11.2. Modeling Data Warehouses: Background with UML

Our approach applies UML to the Data Warehousing domain. It is aimed at encompassing all the different ways that users may use a DWH. Our goal is to provide an overview over all aspects of DWH usage, not only focusing on the data model. Nevertheless, due to the special characteristics of DWH data, it is necessary to take the data model especially into account.

DWH applications involve complex queries on large amounts of data, which are difficult to manage for human analysts. In Data Warehousing, data is often organized according to the multidimensional paradigm, which allows data access in a way that comes more natural to human analysts. The data is located in n-dimensional space, with the dimensions representing the different ways the data can be viewed and sorted (e.g. according to multidimensional model, also called star schema or fact schema, is basically a relational model in the shape of a star (see Fig. 1 for an example). At the center of the star there is the fact table. It contains data on the subject of analysis (e.g. sales, transactions, repairs, admissions, expenses, etc.). The attributes of the fact table (e.g. cost, revenue, amount, duration, etc.) are called measures or fact attributes.

The spokes/points of the star represent the dimensions according to which the data will be analyzed (sorted/aggregated by data, by store). The dimensions can be organized in hierarchies that are useful for aggregating data (e.g. store, city, region, country). Stars can share dimensions, thus creating a web of interconnected schemas that makes drill-across operations possible.
4.11.3 UML Profile for Modeling Data warehouse Usage

This section introduces our UML Profile for Modeling DWH Usage. We use the extension mechanism of UML and import elements from a well-known UML Profile of the Data Warehousing domain, in order to achieve a conceptually sound model (Doug and Matt 2013), with

(a) Tool support and

(b) Well-known notation elements.

Its stereotypes and enumerations, and also shows which classes are used as base classes of the stereo-types (Russ and Kim 2006). We import the Profile of Luján-Mora et al. and also some packages of the UML metamodel (for the convenience of not having to use fully qualified names). The diagram describes the characteristics of the stereotypes not shown above (Jim and Ila 2005). Usage is defined as an Information Flow, which is a type of directed relationship that specifies that information items circulate from sources to targets. Information flows are defined in UML as a very general concept to be used in

Figure: 4.16 Modeling Data Warehouses: Background with UML
Figure: 4.17 UML Profile for Modeling Data warehouse Usage
Sources and targets of an information flow may be: Actor, Node, Use Case, Artifact, Class, Component, Port, Property, Interface, Package, and Instance Specification preliminary models, before having taken detailed modeling decisions on types or structures. One other purpose of information items and information flows is to abstract complex models by a less precise but more general representation of the information exchanged between entities of a system” [Adam et al.2012]. This makes information flows very suitable for our purpose, which is to provide models that capture an overview of the general structure of DWH usage. The direction of the Usage arrow indicates whether the users actively initiate the access to the DWH or whether they wait to receive messages from the system i.e. push or pull mode. A user analyzing OLAP data would be pull, whereas an e-mail alert is push (Ipek and Sabri 2014).

4.12 DATABASE MANAGEMENT SYSTEMS WITH TEMPORAL SUPPORT AND QUERY PROCESSING

Current records are quite prevalent, yet very few data management systems (and especially commercially available systems) pay attention to the special needs of such data (Mark 1990). Existing systems have been designed with the view that the task of a database system is mainly to keep the data in the database current (Clare 2012). Typically, historical data will be lost by updates. Historical/Temporal databases1 2 are designed to capture not only the present state of the modeled real world, but also its previous states For the purpose of this paper, a historical relational database has been considered in which each relation is viewed as a collection of related attributes: Some of the attributes are constant valued while the others are temporal attributes (their values are time varying). A double time-stamping scheme for each time varying attribute is assumed3 4. This means that time varying attributes within tuples are represented as sets of value/time interval pairs.

The below given example, Assumed a banking query time is represented by positive integers (Jeffrey et al. 2012), and that the symbol NOW is used to represent the current value of time. The following relation has been used throughout the chapter:

Customer
Customer(Customer_Name,Balance,
Customer_type,Customer_branch,Date_of_Birth,Pan_No)
Customer_Name,Date_of_birth ➔ Non-Temporal Columns
Balance,Customer_Type,Customer_Branch,Pan_No ➔ Temporal columns

The time-independent key of relation Customer is name. The syntax of the proposed
operators resembles QUERY language. If the range variable .e. is defined over relation
Customer then ‘X.Balance=Y. Balance’ denotes the set of Balance / interval pairs
corresponding to the tuple (customers) that variable.’X’. Points to. For example, the set
indicates a Balance of
10000, 1, 4, 000, 8

Rs. 10000 from time 1 to time 4 balance of a customer Rs. 15000 from time 5 to time 8.

4.12.1 Comparison operators using temporal

This method is used for logical comparison purpose. It’s identified EQUAL to operator
denoted by symbol = and a temporal counterpart denoted by The notation that omitting the
subscript is adopted which will imply the logical operator (ie, = is the same as operator).

As an example
Let be. X and Let be Y be two variables defined over relation customer.
The following expression

X.Customer_Branch = Y.Customer_Branch

The above expressions display both customers in a bank holding account in same
Customer_Branch at the same time (i.e. the corresponding time intervals must overlap).
X.Customer_Branch = Y.Customer_Branch. It requires that the two customers having
account in same branch. But not necessarily at the same time. Thus, logical comparison
operators compare values without regard to their time intervals, while temporal operators
can only compare values that belong to the same time (ie, values that have overlapping
time intervals).This model uses the following
Logical comparison operators:
= > <

Temporal counter parts
=T >T <T T T T

Symbol T is identified by Temporal counter parts

If the comparison condition is not met, the empty set NIL is returned (by both logical and temporal operators). Otherwise, the value returned would be the universal interval <0, NOW> for logical operators, or the set of intervals common to both operands for temporal operators. For example, the value of the expression
\[ 1, 5, 10, 10 \}

The rule for handling operands that are constants or non-temporal attributes is simple: such operands are automatically given the universal interval <0, NOW>. Thus the comparison7 T 3 returns the universal interval, the comparison 3 T 7 returns NIL and the composite comparison (3 7) = NIL returns the universal interval.

4.12.2 Temporal Boolean operators

Boolean operators: AND, OR, NOT.

In this method, there are two types of operator AND: the logical operator ANDL and the temporal operator ANDT. If subscript is omitted, the logical operator is assumed (i.e. AND is the same as ANDL). To see the difference between the two operators, let us assume that the range variable .e. has been declared on relation CUSTOMER .The expression C1 given by

\[ X.Balance \ T \ 50000 ANDT \ X.Customer\_Type =T \ 'AGRICULTURE' \]

requires that the (selected) Customers having bank balance more than 50,000 while customer_type in AGRICULTURE. The expression C2 given by X.Balance \ T \ 50000 ANDL Y.Customer\_Type =T AGRICULTURE requires that the particular Customers having bank balance greater than Rs 50,000 and has customer_type in Agriculture . But the two
events do not have to occur simultaneously. From the definition of ANDL, it is clear that replacing the temporal comparison operators in C2 by their counterparts would give the same result. Both ANDL and ANDT operate on sets of time intervals (associated with attribute values) and they return a set of time intervals (possibly empty). For example, if a specific customer has bank balance more than Rs 50,000 during three different intervals, say (51000, \(<3,4>\)), (54000, \(<6,8>\)), (56000, \(<10, NOW>\)) and the same customer belongs to customer_types is AGRICULTURE during two intervals, say (AGRICULTURE, \(<7,8>\)), (AGRICULTURE, \(<9, NOW>\)) then both conditions C1 and C2 are satisfied for that Customers. The evaluation of the operand .X.BALANCE T 50000. in C1 gives the three intervals \(<3,4>, <6,8>, <10, NOW>\). While the evaluation of .X.Customer_Type T AGRICULTURE. gives two intervals \(<7,8>, <9, NOW>\). Finally, the operator ANDT will return the two intervals \(<7,8>\) and \(<10, NOW>\). These are exactly the intervals in which the customer has a balance rate of more than Rs 50,000 while working in the AGRICULTURE. Customer_Type. When the two sets of intervals (representing the two operands of ANDT), don’t have overlapping intervals, the result returned by ANDT is the empty set denoted by NIL (which corresponds to the value FALSE in the binary Boolean logic) otherwise, ANDT returns the set of intervals common to both of its operands. In this model, any non-empty set of intervals corresponds to the Boolean value TRUE. On the other hand, the operator ANDL returns the value NIL only if one (or both) of its operands evaluates to NIL. If neither operand is NIL, the value returned by ANDL is the universal interval \(<0, NOW>\): The universal interval returned by ANDL indicates that both conditions are satisfied, and since the time intervals of the two conditions are not relevant, the universal interval is returned. This choice makes it easy to include ANDL as a sub-expression within larger expressions. For example, the following expression can be used if one is interested in the .Service periods in the AGRICULTURE Customer_Type. of those Customers having balance more than Rs 50,000 in their bank account and who having account at some point in time in the Customer_Branch of NOIDA (here it is not required that the events of Customer having bank balance and customer_branch in Noida occur at the same time, and further more the
time periods of these two events have been ignored) \( X.\text{Customer\_type} = T \) AGRICULTURE ANDT \((X.\text{Balance} > 50000 \) ANDL \((X.\text{Customer\_Branch} = \text{Noida})\]

In the above expression, the right hand operand of ANDT will evaluate the universal internal \(<0, \text{NOW}>\) only if both operands of ANDL are satisfied. Whenever one of the operands of ANDT is the universal interval, the value returned by ANDT will be the set of intervals of its other operand. This model uses similar extensions for the operators NOT and OR. The unary operator NOTL returns the value NIL, if its operand is a non-empty set of intervals, otherwise it returns the universal interval. The unary operator NOTT returns NIL if its operand is the universal interval. Otherwise, it returns the set of intervals which is the complement of its operand (with respect to the universal interval).

For example the expression

\[ \text{NOTT}\{<2,2,<6,8>,7,9>\}\]

Returns the set \(\{<0,1>,<3,5>,<10,\text{NOW}>\}\)

Both logical and temporal versions of NOT can be very useful in expressing temporal conditions. The following expression returns the value NIL, if the Customer having account with Noida branch at any time) and returns the universal interval if he has never account with Noida branch.

\[ \text{NOTL}\left(X.\text{Customer\_Branch}=\text{NOIDA}\right)\]

Similarly, the following expression gives the set of intervals during which the Customer having Account outside NOIDA

\[ \text{NOTT}\left(X.\text{Customer\_Branch}=T\ \text{NOIDA}\right)\]

Notice that, as expected, the expression

\[ \text{NOTT}\left(X.\text{Customer\_Branch}=T\ \text{NOIDA}\right)\] gives the same result as the expression \(X.\text{CUSTOMER\_BRANCH} = T \) NOIDA. The operator OR also has two versions. The binary operator ORL returns the value NIL only if both of its operands are empty (NIL) otherwise, it returns the universal interval. The binary operator ORT returns NIL if both of its operands are NIL. Otherwise, it returns the union of the intervals of its operands.

Thus the expression
X.Branch_manager  T Kishore ORT  x.Branch_Manager =T Rohit

Returns the set of intervals during which Customer is under kishore or Rohit

4.12.3 Temporal Retrieval Capability

In this case, the approach distinguishes between the qualification needed to select entities (tuples) and the qualification needed to select temporal values (output) from the chosen entities (Raghu and Johannes 2002).

The syntax of a retrieval statement is as follows:

```
retrieve (target list)
[for output output-qualification]
[where Condition ]
```

WHERE is the keywords

The selection qualification gives the condition that determines whether an entity (or a combination of entities) should be selected for output. The output qualification is used to select required values of temporal attributes from the selected entities (Gupta 1986). If the for out. Phrase is omitted, the set of time intervals returned by the selection qualification is used to select the output (i.e. to replace the output qualification). If the .where Clause is omitted, a default interval of < 0, NOW > is used to replace the selection qualification.

4.12.4 Operands of Temporal Operators

The operands of ANDT are expected to be sets of time intervals. For example, the following expression

(X.Balance  T 50000) ANDT <0, NOW> has a right operand equal to the universal interval (treated as singleton set), and a left operand equal to the set of time intervals during which the customer balance exceeded 50,000 (it may be noted that since the right operand is the universal interval, the result of the above expression is always equal to the left operand). Here an operand is allowed to be a set of value/interval pairs. In this case, the intervals contained in such a set will be used as the real operand.
Thus in the following expression

\[ X.Balance \text{ ANDT } 5,7, 9,10 \]

The left operand is a temporal attribute (a set of value/interval pairs), and it is assumed that only the time intervals of this operand will be used to evaluate the result of ANDT

### 4.12.5 Performance Tuning Aspects in Query Languages

**Example 1**

The below query display the output customer having balance at the time of 10 and customer having account with NOIDA branch then customer_type=AGRICULTURE. Customer having more than 50,000 bank balance

Retrieve \((X.Customer\_Name, X.Balance)\) for output \(X.Balance \text{ ANDT } <5,5>\)

Where \(X.Customer\_branch =T \text{ NOIDA } \text{ ANDT } X.Customer\_type =T \text{ AGRICULTURE } \text{ ANDT } X.Balance \text{ T } 50,000\)

Using WHERE clause easily we can able to retrieve the records with faster performance will excellent while using where clause. Query processing time is save when using WHERE clause.

Selection qualification (which follows the keyword .where.) is used to determine whether a given Customer is a candidate for output (the selection qualification must evaluate to non-NIL for such an). The output qualification (which follows the .for output. keyword) is used to extract the balance at time. Five of the selected customers (balance is the output must give a non-NIL value when used to evaluate the output qualification). If the selected customer has no bank balance 5, he will have no output. It may be mentioned that if the output qualification (the .for output. phrase) is omitted, the query would return the Balances corresponding to the time intervals returned by the selection qualification (any such customer balance must be greater than 50,000). In the above example, the operator ANDT in the output qualification acts like .overlap function. In fact the Boolean
overlap function used in many temporal models is a special case of the ANDT operator defined in this paper. Using the keyword .OVERLAPS. Would be a better choice than ANDT in the output qualification of example 1. Therefore, redundant operator OVERLAPS is added to this syntax.

4.12.6 Overlaps Operator

This redundant operator is equivalent to ANDT. It operates on two sets of intervals and its result is a set of intervals. It is included in the syntax in order to replace ANDT whenever this replacement makes the meaning more clear. The generalized Boolean logic eliminates the need to use an overloaded .OVERLAP. Operator (to deal separately with Boolean and time-interval expressions).

Example 2

The following query gives the bank balance at time 5 for those Customers having more than 50,000 at 5 time retrieve (X.Customer_Name, X.Balance) for output X.Balance OVERLAPS <5, 5> where (X.Balance T 50,000) OVERLAPS <5,5>

In the above example, the selection qualification is used to determine whether a customer details satisfies a required condition, where the result is used to select the Customer bank balance at a certain time. Combining the two qualifications into a single one would not be possible in this example since each qualification requires different time intervals.

Example 3

Each customer in the bank having Customer_type in AGRICULTURE under branch_Manager .Kishore ..Display their bank balance and customer_branch

Range of x is customer

retrieve (X.Customer_name,X.Balance, X.customer_branch)
for output (X.Balance OVERLAPS SELQUAL) ANDL
(X.customer_branch OVERLAPS <NOW, NOW>)
Where \( x.\text{Customer\_type} = \text{T \ AGRICULTURE ANDT \ X.Branch\_Manager} = \text{T kishore} \)

The selection qualification returns the set of time intervals during which the customer is under Branch\_manager kishore and \( \text{Customer\_Type=AGRICULTURE} \)

This set of intervals replaces the keyword SELQUAL in the output qualification.

Any customer bank balance/\( \text{Customer\_branch} \) combination displays in the output must give a NOT-NIL result when used to evaluate the output (Jiawen et al. 2011).

Example 4

In a bank head office people need particular customer details and branch manager name Suppose a customer having heavy bank. To display the result below query will process

Range of \( X, Y \) is customer /*\( X = \text{Customer\_Name}, Y. =\text{Customer\_Name}*\/

retrieve (\( X.\text{Customer\_Name}, X.\text{Balance}, Y.\text{Customer\_Name}, X.\text{Branch\_Manager} \))

for output (\( X.\text{Balance} \text{ OVERLAPS} \text{ SELQUAL} \) ANDL

(\( X.\text{Branch\_Manager} \text{ OVERLAPS} \langle \text{NOW, NOW} \rangle \))

Where \( X.\text{Balance} \text{ T Y.Balance} \)

The temporal operator \( \text{T} \) in the selection qualification ensures that the compared Bank balance belong to the same time (ie, are not from overlapping time intervals). It may be noted that the operator \( =\text{T} \) has a right operand (\( Y.\text{Customer\_Name} \)) of type non - temporal, and as explained before such operand is given the universal time interval. Now let somebody wants to change the query so that the displayed is displayed for any Customer who are having heavy money in the branch
4.12.7 Composite Operator

Range of X,Y is customer

Retrieve (X.Customer_Name,X.Balance)

Where X.Balance T Y.Balance K.ANDL X.Branch_Manager = Y.Customer_Name

The composite operator K.ANDL is used to ensure that entity Y has been the Branch_manager of entity X. and that during some interval(s) of time, the balance entity X has exceeded that of entity Y [39]. Furthermore, The operator K.AND k will return the set of intervals of its left operand (assuming the right operand is non-NIL), i.e., the set of intervals during which the Balance of entity X has exceeded that of entity