An Oracle White Paper
February 2011

Oracle Data Mining 11g Release 2

Mining Star Schemas

A Telco Churn Case Study

ORACLE

Table of Contents

Executive Overview

Oracle Data Mining (ODM) is a priced option to the Oracle Database 11g Enterprise Edition. Mining runs directly in the database, removing issues associated with data movement, data duplication, security, and scalability. Moreover, ODM has been designed to fit well with the SQL language. While other data mining products need to utilize complex and non-performant operations when trying to mine data in star schemas – such as transposing from transactional form to a flat, relational form - ODM is designed to leave the data in its natural form and mine it directly. Such an approach yields simplicity, power, and improved performance.

Section 1: Introduction

A churn analysis case study [CACS] performed by Telecom Italia Lab presents a real-world solution for mining a star schema to identify customer churn in the telecommunications industry. [CACS] provides some background on the churn problem, and a detailed methodology describing the data processing steps required to predict churn. [CACS] is a natural fit for Oracle Data Mining and serves to show the power and simplicity of using ODM to mine a star schema. This white paper is heavily based on [CACS], and though this white paper can be read stand-alone, it is recommended that the reader become acquainted with the original case study.

Steps for implementing the methodology using the Oracle Data Mining API, together with additional SQL, are available in a blog three-post series [AOB]. This white paper is intended for an audience that uses Oracle Data Miner, the graphical user interface for ODM, which is available from within SQL Developer 3.0.

This white paper has two goals:

- Demonstrate a telco churn case study using Oracle Data Miner
- Explain the case study workflow methodology

Sections 2, 3, and 4 demonstrate the case study by preparing a database schema and importing a pre-made Oracle Data Miner workflow. At the conclusion of Section 4, a fully functional implementation of the churn methodology is made available. Section 5 describes the workflow in detail, relating the operations performed in the workflow to the methodology described in [CACS]. Sections 6 and 7 capture final comments and references.

Section 2: Preparing the Schema

Step 1: Create the User

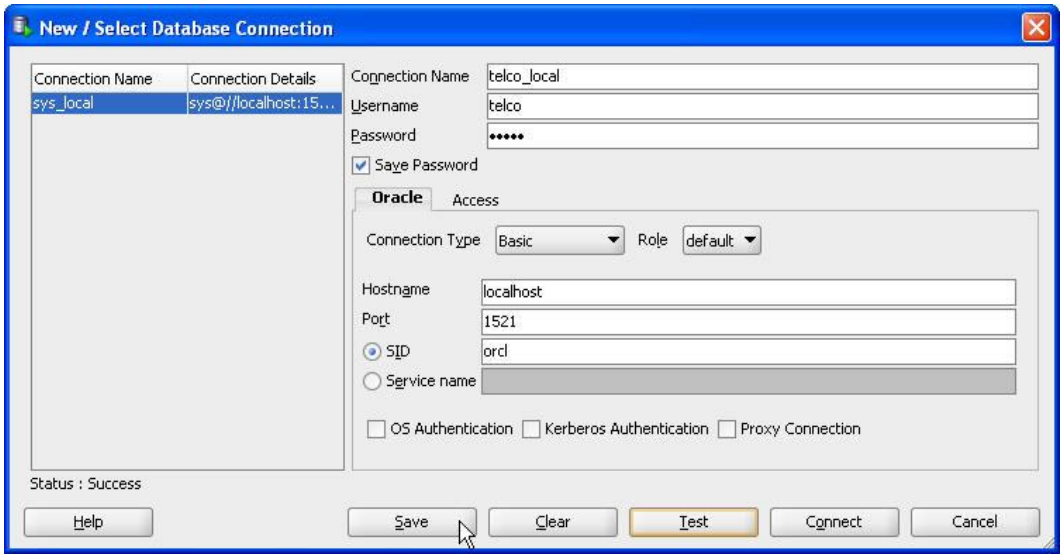
A user who performs data mining must be granted a few database privileges. The **create mining model** privilege is necessary in order to train a data mining model using ODM. In addition, the user needs the **create table** and **create view** privileges to perform the necessary operations associated with training mining models. For the purpose of this white paper example, the user will need the **create procedure** privilege so that the data generation procedure can be installed in the user's schema. Finally, the user must have the **create session** privilege and be granted **tablespace** to hold the contents of the tables and mining models.

A sample statement to create a data mining user is provided below. Such a statement must be issued by a privileged user. Contact your DBA if you need assistance with this step.

```
grant create session, create table, create view,
create mining model, create procedure,
unlimited tablespace
to telco identified by telco;
```

For simplicity, future references to the data mining user's schema will assume that the schema is named **telco**, as in the example above.

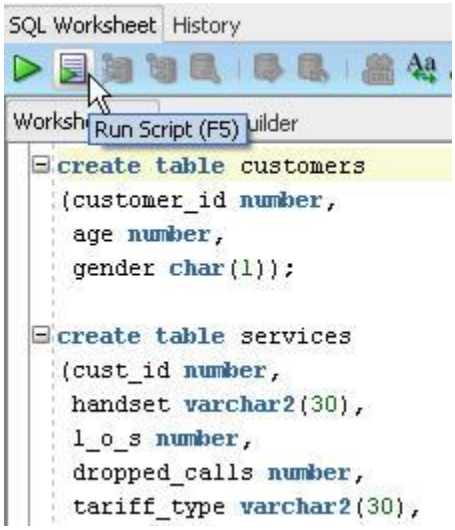
Once the user is created, you can create a database connection for this user from within the SQL Developer connection navigator.



Step 2: Create and Populate the Tables

The example in this white paper involves five database tables. Four of them map, column for column, to the tables described in [CACS]: **CUSTOMERS**, **SERVICES**, **CDR_T**, and **REVENUES**. The fifth table, **CHURN_RISK**, is included to facilitate deployment of results. In addition, a procedure named **telco_load** is used to generate customer data.

The ODMtelcosetup.sql script should be run from within the telco schema. This script will create the five tables, create the procedure for data generation, and invoke the procedure to generate data corresponding to 10,000 customers. The ODMtelcosetup.sql script can be loaded into the SQL Worksheet of SQL Developer and invoked from there. Make sure that the script is run from within the telco schema.



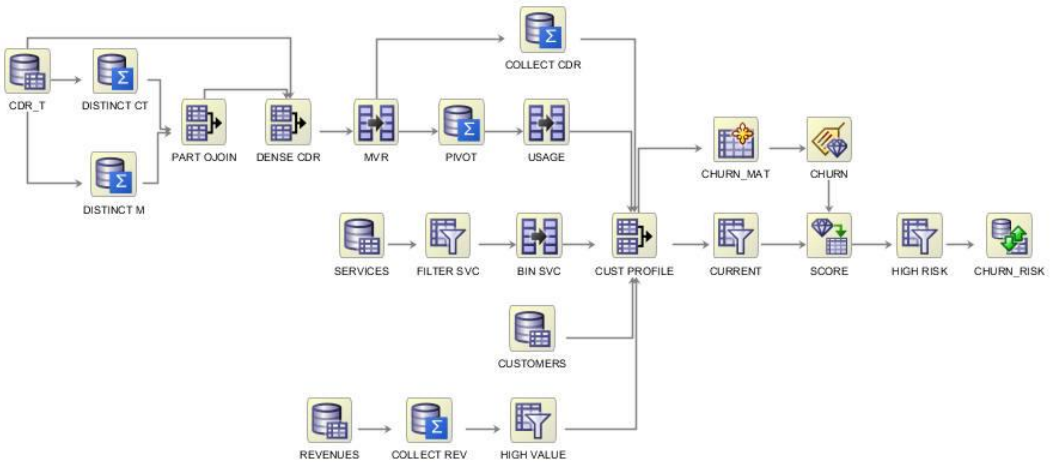
Section 3: Importing the Workflow

The Oracle Data Miner interface provides a mechanism for exporting and importing workflows that capture complex data mining methodologies. The next step in this example is to create a project and import a pre-defined workflow into the telco schema. The workflow, ODMTelcoChurn.xml, contains all of the logic necessary to train a model to predict churn and score the model to identify high-valued customers who are at-risk for churn.

NOTE: When you first try to connect to telco from the Data Miner navigator panel (available in the SQL Developer Tools menu), you will need to perform some setup. If it is the first time you are ever connecting to the database using a Data Miner navigator connection, you will be guided through installation of the Data Miner repository. If it is the first time you are connecting as a particular user, Data Miner will guide you through the steps of granting necessary privileges and, optionally, installing demo data. You will need the password for the SYS account to proceed, so contact your DBA if you need assistance with these steps.

To create the project and import the workflow, right-click in the Data Miner navigator as shown in the two images below. The first step will create a new project, and the second will perform the import of the pre-defined workflow, ODMTelcoChurn.xml.

Once imported, the workflow should look like the picture below:

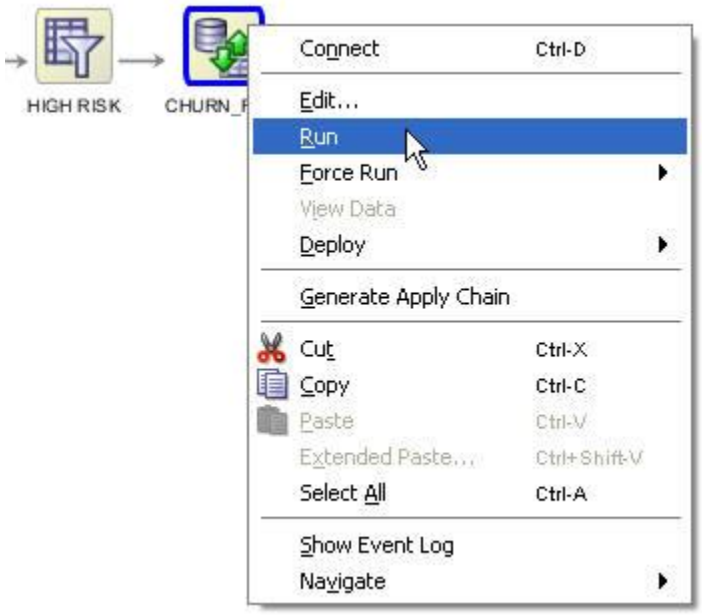


.

Section 4: Running the Workflow

Now that the underlying tables are populated and the workflow is loaded, it is time to mine the customer data and identify high-valued, at-risk customers. This section jumps right into running the workflow. The following section will go step-by-step through the different components of the workflow to explain exactly what is being performed, and why.

Right-click the CHURN_RISK node (far right node in the workflow) and select Run from the menu.



This action will start a job in the database that will:

1. Prepare the customer data
2. Train and test a classification model named CHURN
3. Score current, high-valued customers with the CHURN model
4. Identify those current, high-valued customers who are at-risk for churn
5. Store the at-risk customers in the CHURN_RISK table

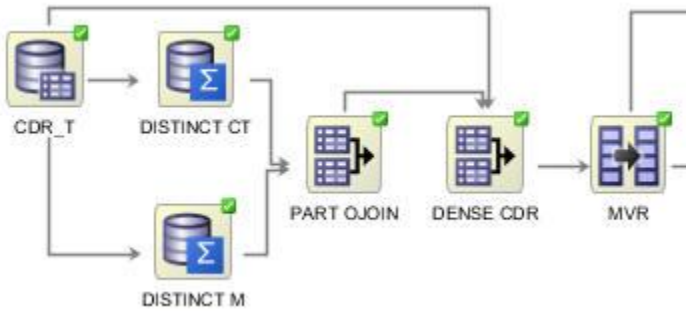
The contents of this table can be viewed through any desired tool, such as OBIEE, SQL Developer, etc., or can be processed as part of other database operations. To

yield a more dynamic result, instead of persisting results in the table, the workflow can be modified to create a view leveraging the CHURN model. Whenever the view is queried, the model will be scored against up-to-date customer data, enabling real-time identification of churn risk.

Section 5: Understanding the Workflow

The methodology as captured in the ODMTelcoChurn.xml workflow mirrors much of the work done in [CACS]. Most of [CACS] revolves around data preparation, and some of it is quite complex due to the need to transform star schema data into a pivoted, flat view for mining. The corresponding steps using ODM are much simpler since ODM is capable of mining the star schema data in its natural form. [CACS] leaves a few details open to interpretation, and in those cases we take a best guess as to the intended operation.

Step 1: Missing value replacement in call data records

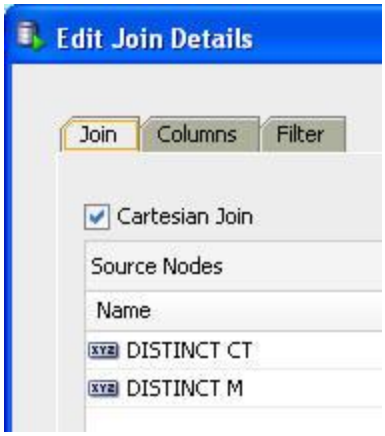


Call data records, otherwise known as call detail records, often comprise the largest amount of data in a telecommunications database. A separate record is captured for each phone call. The records contain length and time of call, which can be used to identify a call category, such as weekend, peak time, etc. In [CACS], the call category is represented by the tariff column, and the data is pre-aggregated to the month level per customer-tariff combination. Only the most recent five months per customer-tariff combination are retained.

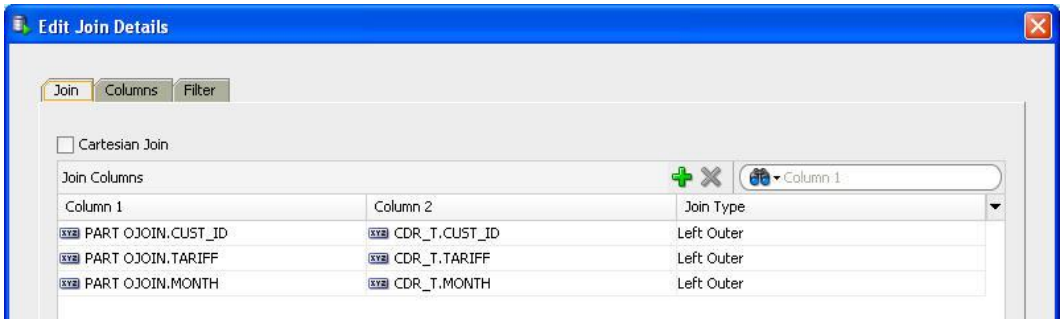
When performing data mining, intelligent treatment of missing values can lead to models with improved accuracy. Missing data problems may arise from data acquisition issues, data merging issues, cleansing problems, bad data entry, etc.

In [CACS], the missing value replacement step requires generating new rows corresponding to customer-tariff combinations that have fewer than five months recorded. The missing months will be populated with the average value of usage minutes from the recorded months in each customer-tariff combination.

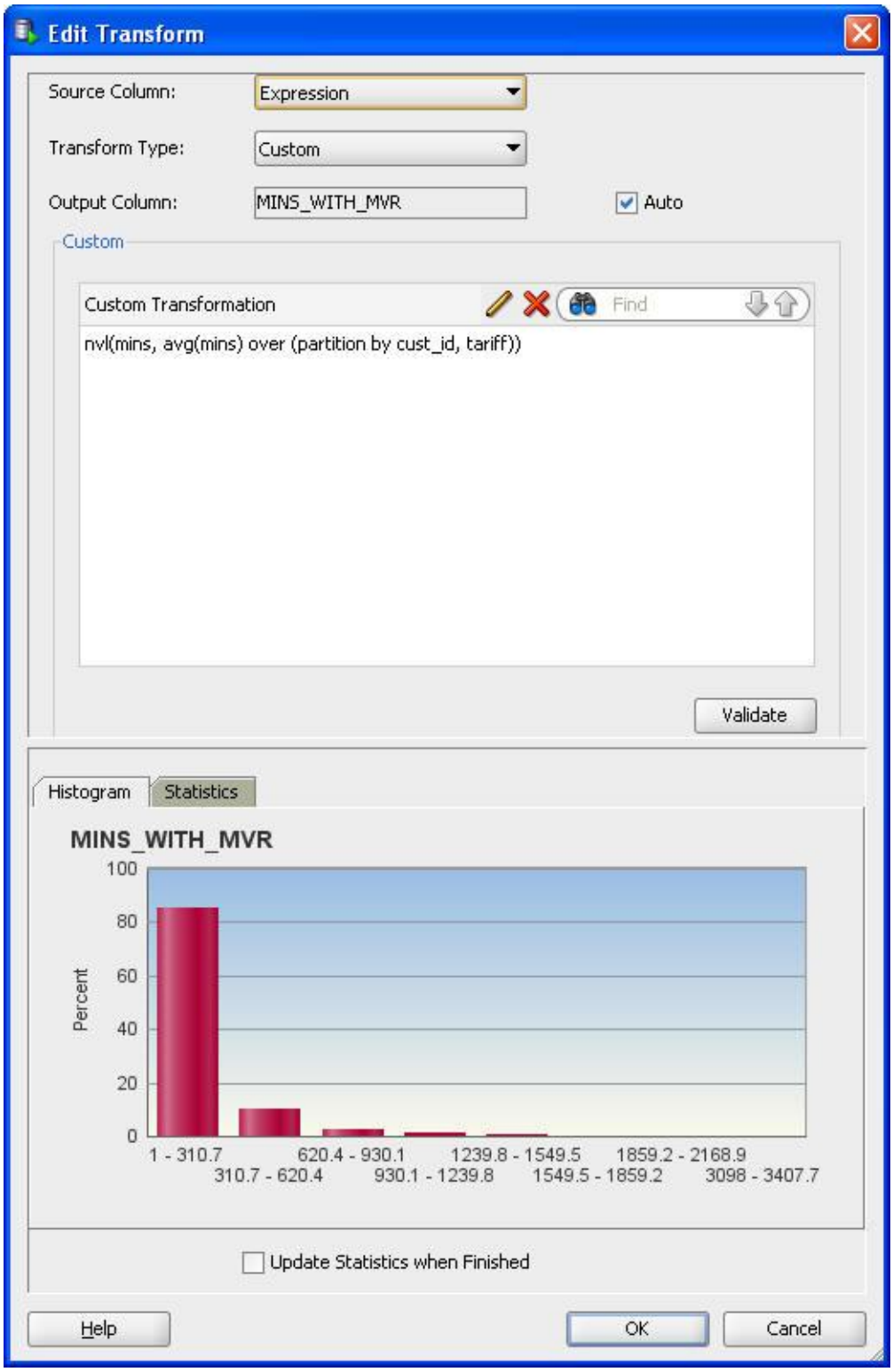
Generating all applicable customer-tariff-month combinations can be achieved by performing a Cartesian product (join with no join condition) between the distinct set of months and the distinct set of customer-tariff combinations. By performing a distinct on the customer-tariff combinations, we ensure that only valid, existing customer-tariff combinations are present in the result set. This step is represented in the workflow by three nodes: DISTINCT CT, DISTINCT M, and PART OJOIN.



Once the Cartesian product is complete, we can perform an outer join with the original CDR_T dataset. The outer join will enable us to produce all desired combinations while retaining the actual phone minute usage values from CDR_T where rows exist, and producing NULLs for the rows which were missing. This step is captured in the workflow by the DENSE CDR node.



At this point, we have generated rows to represent all the data, but we have NULLs in places where there was no row present in the CDR_T table for a particular customer-tariff-month combination. Oracle's SQL syntax for window functions is a perfect fit for the next step. This step is captured in the workflow by the MVR node.


Edit Transform
Source Column:
Expression
Transform Type:
Custom
Output Column:
MINS_WITH_MVR
Auto
Custom
Custom Transformation
Find
nvl(mins, avg(mins) over (partition by cust_id, tariff))
Validate
Histogram
Statistics
MINS_WITH_MVR
Percent
100
80
60
40
20
0
1 - 310.7
310.7 - 620.4
620.4 - 930.1
930.1 - 1239.8
1239.8 - 1549.5
1549.5 - 1859.2
1859.2 - 2168.9
3098 - 3407.7
Update Statistics when Finished
Help
OK
Cancel

The MVR node is a transformation workflow node with a custom expression. The custom expression is as follows:

nvl(mins, avg(mins) over (partition by cust_id, tariff))

The partition by clause will segment the data into separate groups, one group per customer-tariff combination. Within each of those groups, the average number of minutes will be calculated based on the rows present in the CDR_T table. Finally, the nvl function is used to replace the mins (phone usage minutes) column with this average whenever the mins column is NULL - it will be NULL when it came from the PART OJOIN node as opposed to the original CDR_T table.

At the conclusion of the MVR node processing, we will have filled in missing rows corresponding to all valid customer-tariff combinations using an appropriate, specialized computation.

Step 2: Call data record processing

This step gets to the root of how Oracle Data Mining is able to mine a star schema - and why the ODM in-database approach is so simple and powerful. This step requires only 1 node with a very small specification, yet it is sufficiently powerful to capture the impact of telephone usage at different levels in the call data records hierarchy. The corresponding step as presented in [CACS] requires **37** nodes (see the diagram from step 3.2.3 in [CACS], and note that some of the flow in that diagram is omitted for brevity).

In [CACS], the first step is to pivot data from transactional to relational form. Separate columns are generated for each tariff-month combination. With five months and four tariff types, that yields twenty new columns. Those values are also rolled-up across months, yielding four more columns at a coarser level of granularity. Specifying and representing this transformation is burdensome, and there is significant performance impact associated with moving data from one representation to another. Further, this approach does not scale as the number of categories, and category combinations, increases.

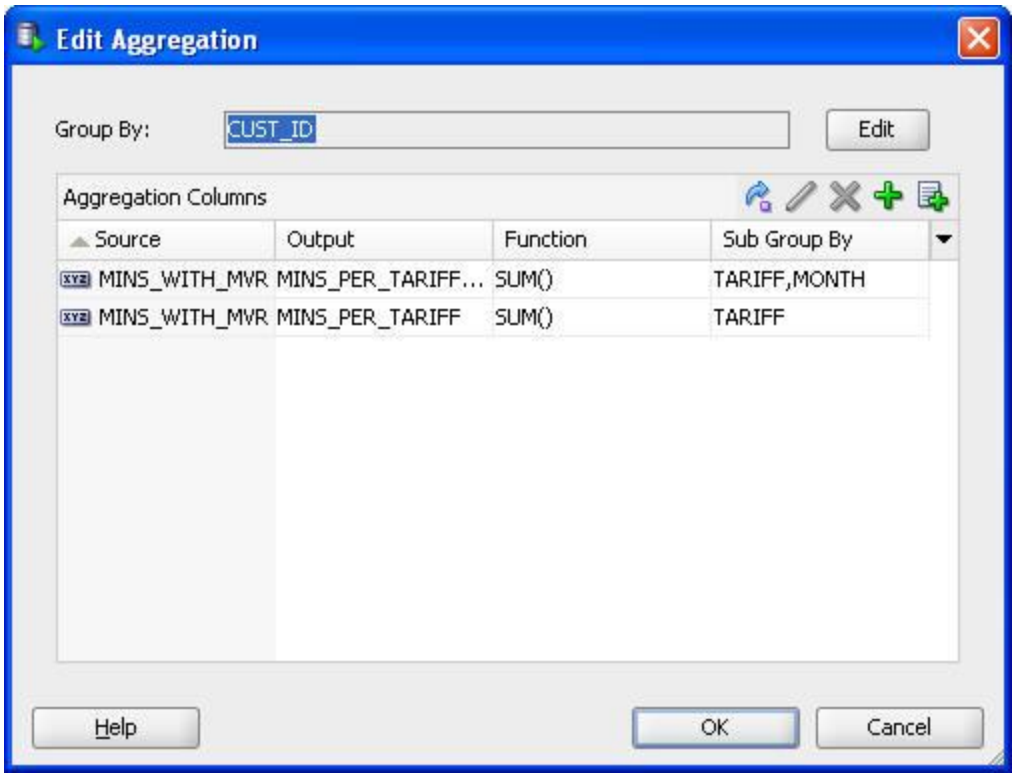
The database stores the CDR_T table in transactional form. This is natural, and is a good representation for data that may be sparse. ODM enables mining such data without having to transpose the data out of transactional form.

The Data Miner workflow aggregation node can be used to roll the CDR_T data up to the customer level, which is necessary when performing mining on a customer basis. The simplest roll-up is to aggregate all the way to a single, scalar value per customer, such as total number of minutes of usage for that customer. While such information is useful, a finer level of granularity may be much more valuable for identifying churn. For example, it is possible that the number of peak minutes used in the most recent month is a key indicator of churn. In order to let the data mining algorithm process data at a finer level of granularity, we must produce information that is not aggregated to the top. This can be done using the sub-group by feature of the Oracle Data Miner aggregation workflow node.

In the specification below, the COLLECT CDR node will generate two new nested columns:

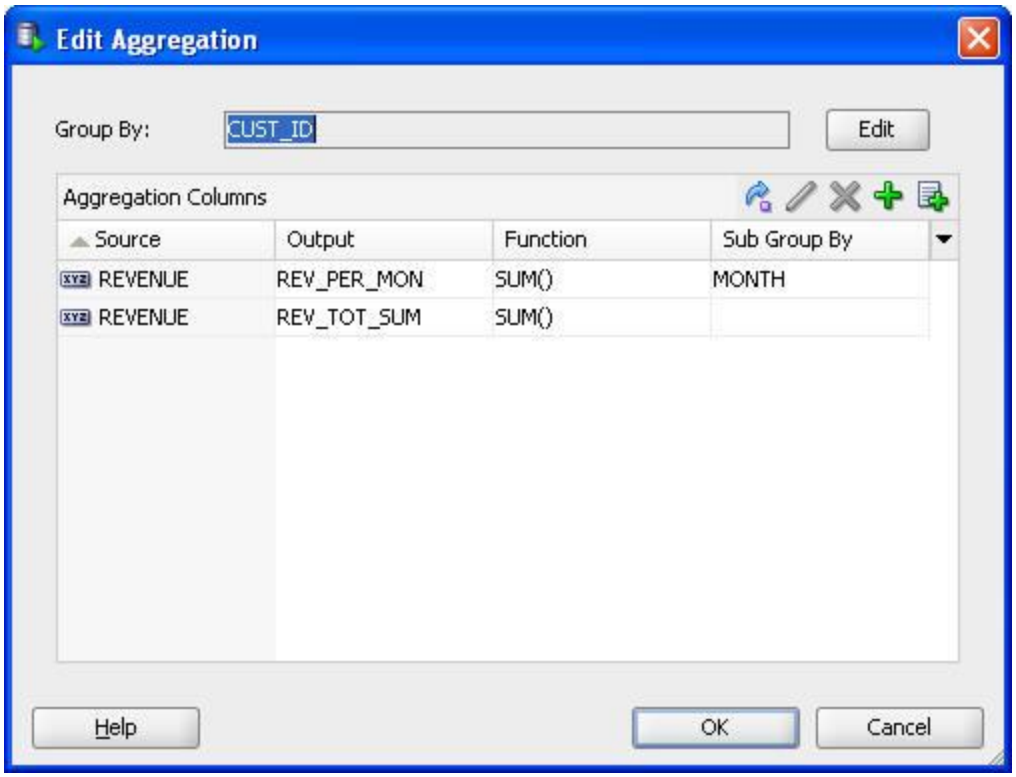
- Number of minutes per customer, broken out by tariff-month combination
- Number of minutes per customer, rolled up to the tariff level

This straightforward specification can represent data at the customer level while retaining a fine level of detail which may yield rich information for mining. It is straightforward to specify, and the evaluation of this step does not suffer from the cost of transposing data.



Step 3: Revenue processing

Once again, ODM's approach is greatly simplified as compared to traditional tools. [CACS] uses 8 nodes (see the diagram from step 3.2.4 in [CACS]), but all of that work is collapsed into the single COLLECT REV node. This node will compute the revenue per month for each customer, which will be fed to the mining algorithm. In addition, the COLLECT REV node does more – it also rolls up the total revenue per customer, which is the basis for identifying high-valued customers. That corresponding operation occurs in a separate step (3.2.5) in [CACS].

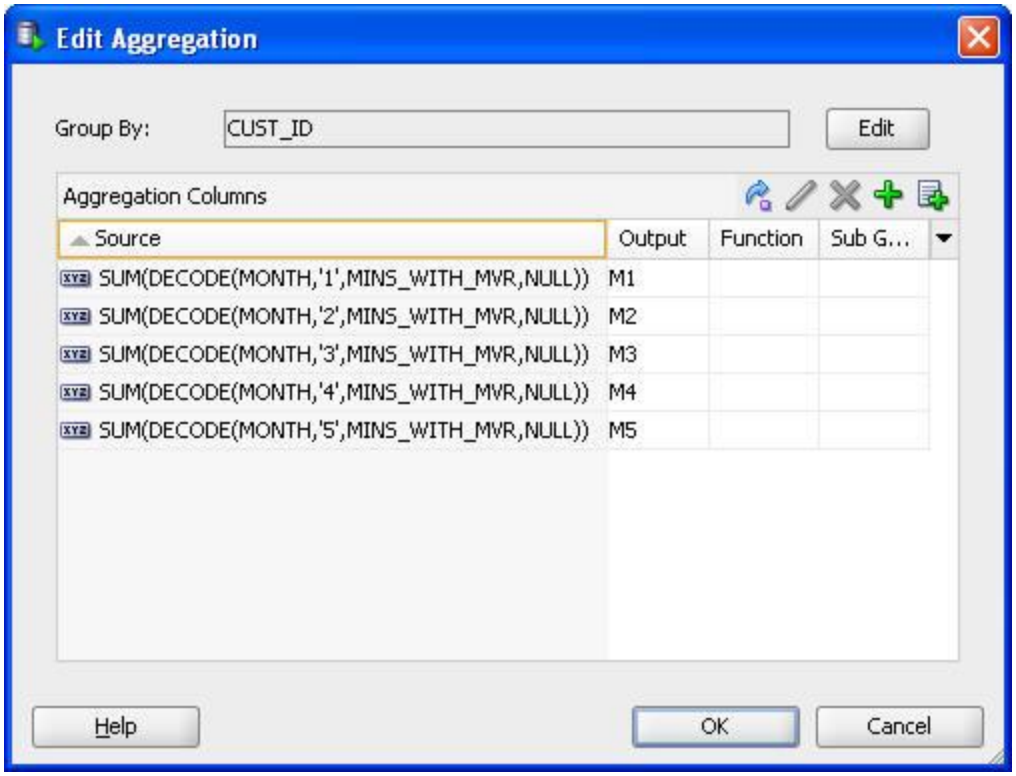


Following the COLLECT REV node is a row filter workflow node, HIGH VALUE, to retrieve only the high-valued customers. This step is a divergence from [CACS] to enable discussion of a simple, meaningful deployment scenario. The same replicated operations performed in [CACS] for modeling customers at different value levels can be achieved with Oracle Data Miner, with approximately the same additional amount of complexity.

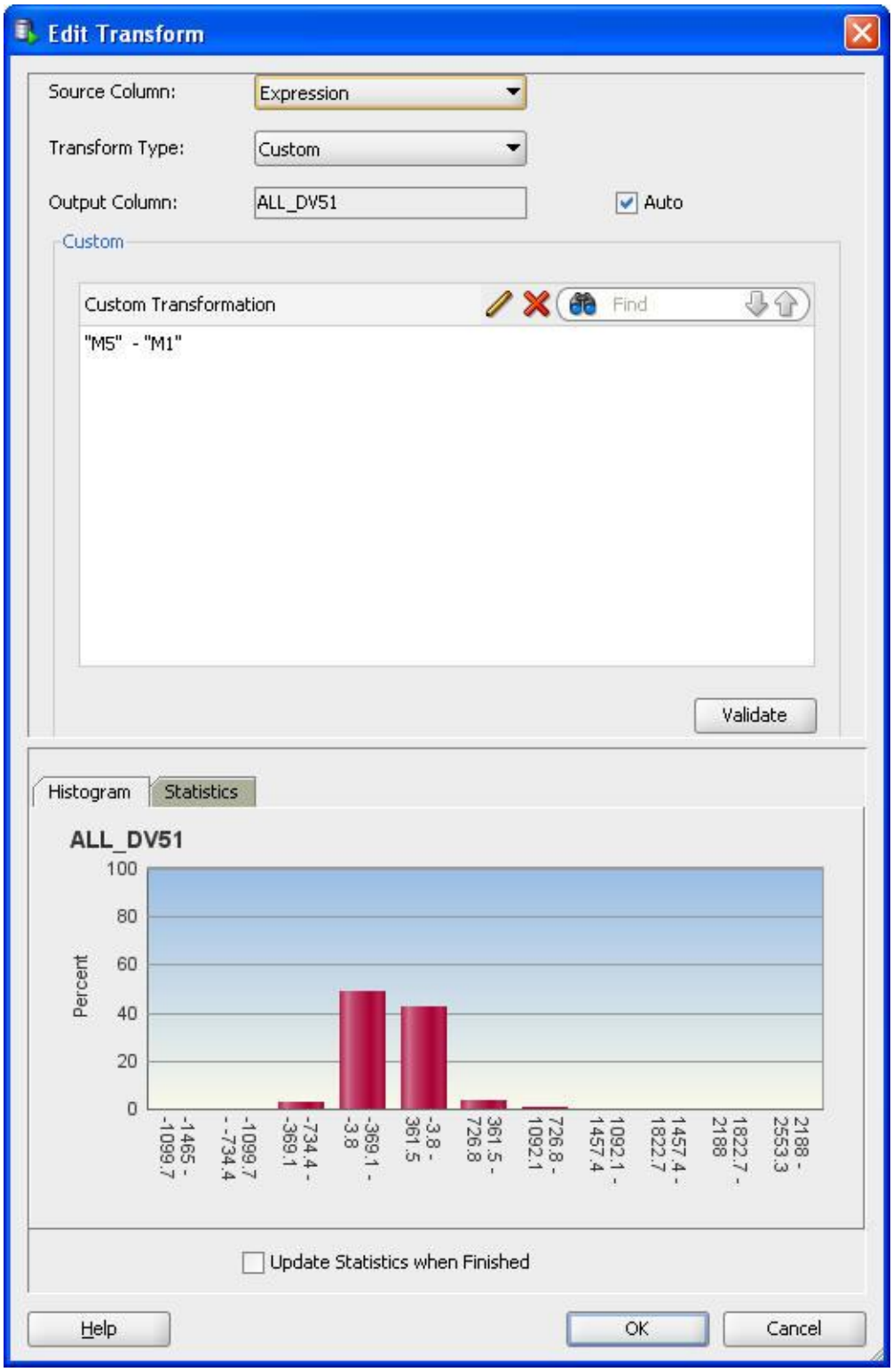
Step 4: Create usage measures

While Oracle Data Mining is designed to analyze data in its native, transactional form, in some cases it is simpler to express new attributes using pivoted data. The usage measures from [CACS] provide such a scenario.

The PIVOT node aggregates to the customer level and generates five new columns. The columns are computed by conditional summing. The first column, M1, is computed by summing the minutes for each customer where the month column is 1. The other four months are computed in a similar fashion. As a result of the PIVOT node, five new columns are generated.



The USAGE workflow node takes the output of the PIVOT node and generates five new values that are to be fed into the mining algorithm. One such value, ALL_DV51, is computed by taking the difference between the minutes used in month 5 and the minutes used in month 1, on a per-customer basis. The other new columns are computed in a similar manner using the custom expression type of the transformation workflow node.

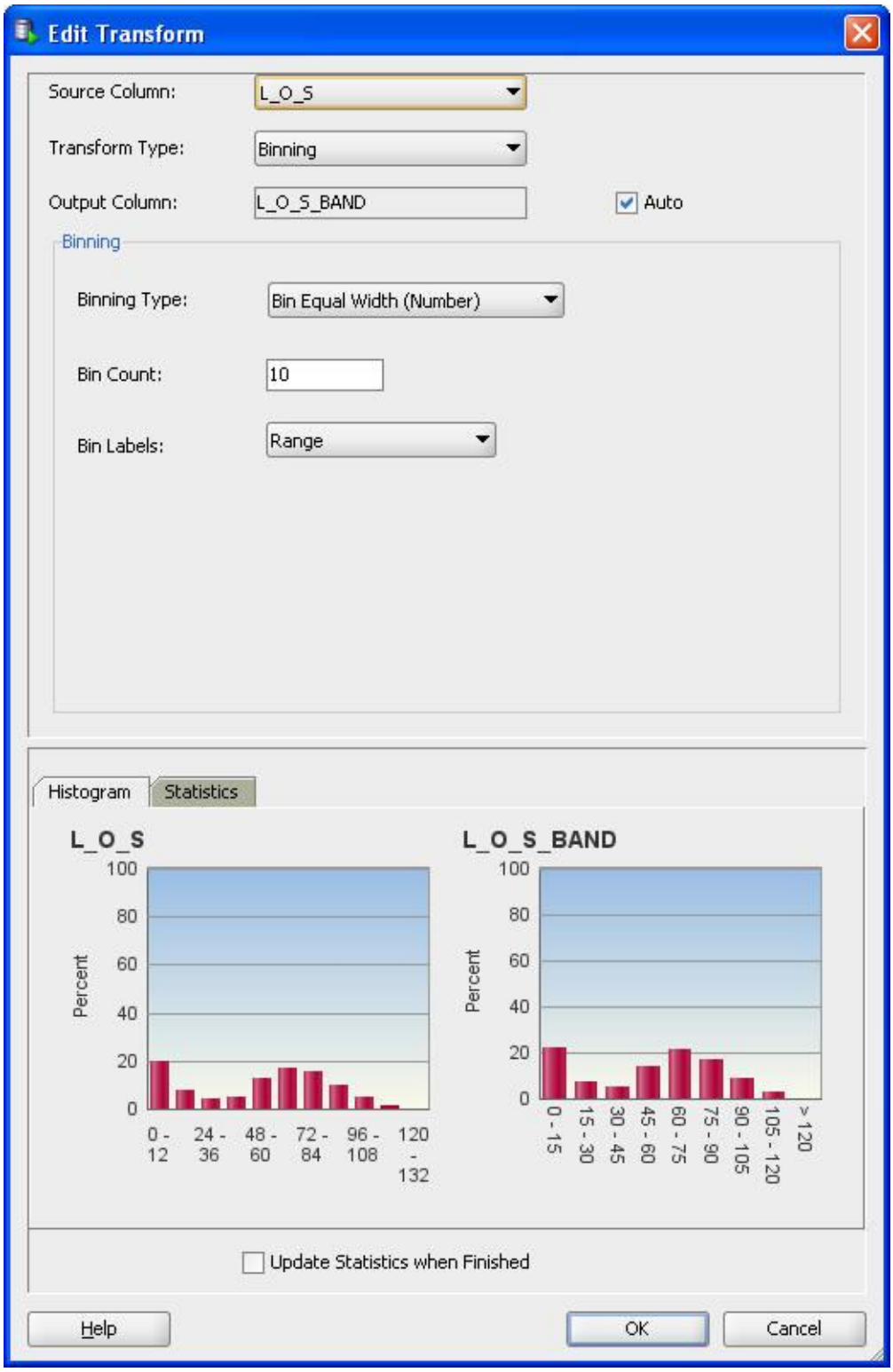

Edit Transform
Source Column:
Expression
Transform Type:
Custom
Output Column:
ALL_DV51
Auto
Custom
Custom Transformation
Find
"M5" - "M1"
Validate
Histogram
Statistics
ALL_DV51
Percent
100
80
60
40
20
0
-1465 - -1099.7
-1099.7 - -734.4
-734.4 - -369.1
-369.1 - -3.8
-3.8 - 361.5
361.5 - 726.8
726.8 - 1092.1
1092.1 - 1457.4
1457.4 - 1822.7
1822.7 - 2188
2188 - 2553.3
Update Statistics when Finished
Help
OK
Cancel

Step 5: Process Services

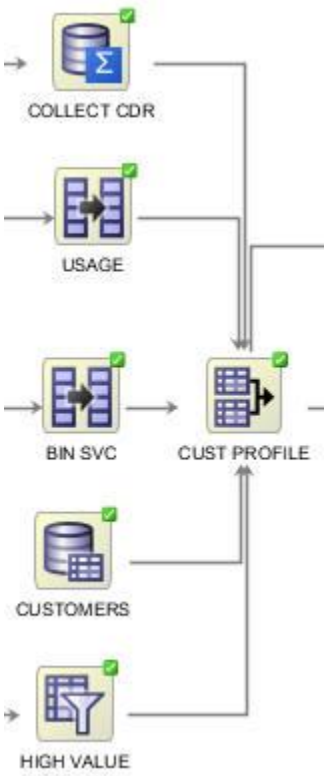
The SERVICES table contains information about the services subscribed to by each customer. There is a 1-to-1 relationship between the rows in the SERVICES table and the rows in the CUSTOMERS table.

[CACS] restricts mining to the 'CAT' and 'PLAY' tariff plans. This step is easily performed by FILTER SVC, a workflow row filter node. By providing a simple filter predicate, undesirable rows can be easily removed.

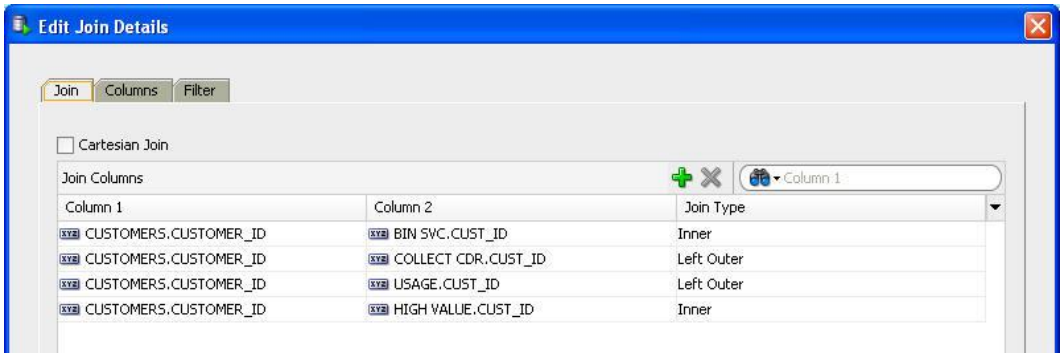
[CACS] also bins the length of service and quality of service (dropped calls) before feeding the data to the mining algorithm. This step is performed by the BIN SVC workflow transformation node. The transformation node has built-in support for common transformation, such as equi-width binning, making them easy to specify.


Edit Transform
Source Column:
L_O_S
Transform Type:
Binning
Output Column:
L_O_S_BAND
Auto
Binning
Binning Type:
Bin Equal Width (Number)
Bin Count:
10
Bin Labels:
Range
Histogram
Statistics
L_O_S
Percent
100
80
60
40
20
0
0 - 12
24 - 36
48 - 60
72 - 84
96 - 108
120 - 132
L_O_S_BAND
Percent
100
80
60
40
20
0
0 - 15
15 - 30
30 - 45
45 - 60
60 - 75
75 - 90
90 - 105
105 - 120
> 120
Update Statistics when Finished
Help
OK
Cancel

Step 6: Bring it together into a complete customer profile

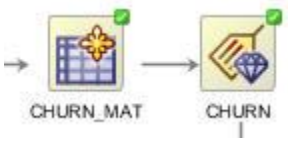


Finally, all of the data is brought together into a single place, including customer demographics, services, usage patterns, and revenue. This is the full customer profile. The join between CUSTOMERS and SERVICES will remove all customers that do not satisfy the FILTER SVC node. The outer join with CDR_T (both the usage information and the granular aggregate information) ensures that CUSTOMERS will not be eliminated if they happen to have no data available in CDR_T – in this case, other attributes can still be used to predict churn. Finally, the join to the REVENUES data will reduce the data for mining to only the high valued customers.



At this point, all of the data is brought together, and there is one row in the CUST PROFILE data source corresponding to each customer of interest. The aggregate information from the CDR_T and REVENUES tables are carried along, but they remain in their natural, sparse, transactional form. The difficult part – data transformation - is now complete.

Step 7: Modeling

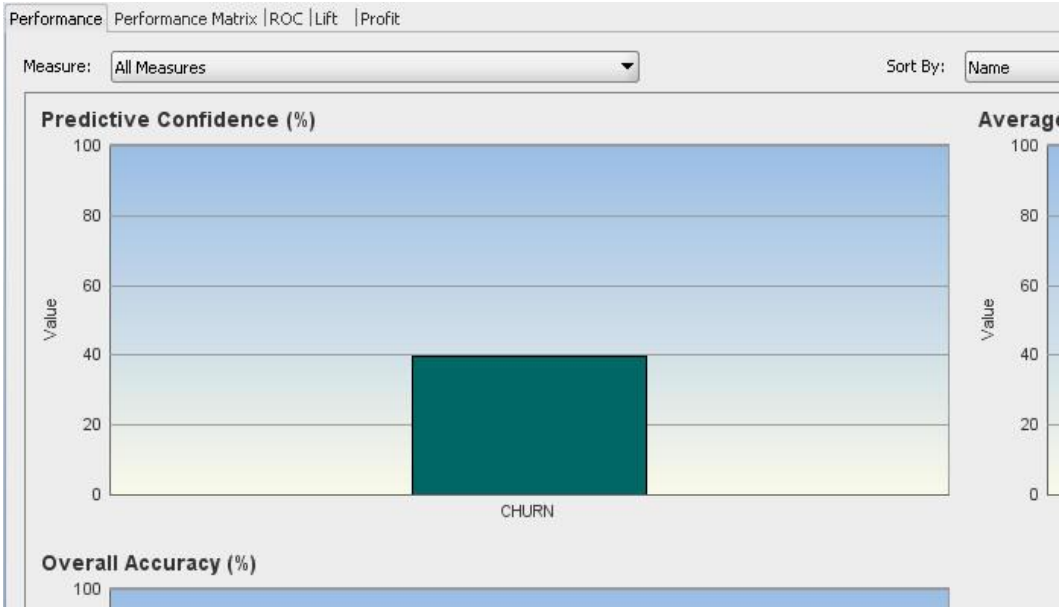


The Oracle Data Miner user interface provides a model build node that is simple and powerful. With one node, analysts can build multiple models, specify different algorithms and settings, vary the selected input, separate the data into train and test samples for accuracy testing, and compare the results across models to identify the best model for the given problem. This white paper example builds a single model, a model to predict churn, using the Generalized Linear Model algorithm with default settings.

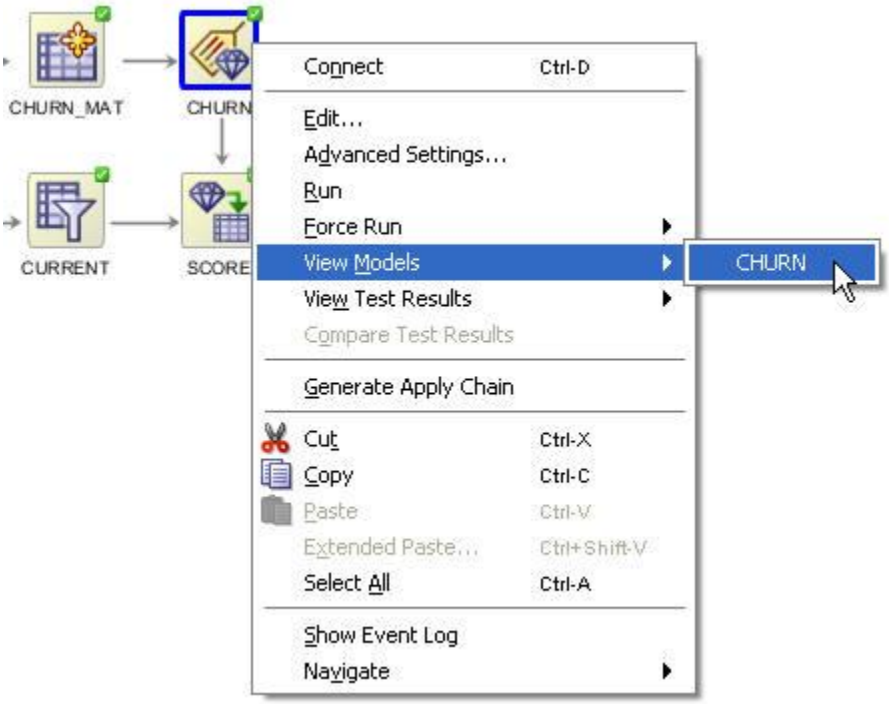
The CHURN_MAT materialization step is not strictly necessary, but it can make iterative model training runs more performant. When training and analyzing models, it is often the case that the analyst may want to tweak settings and rerun, so materialization can be helpful for this common scenario.

After building a model, it is necessary to assess how well the model addresses our goals. Oracle Data Miner's default behavior of splitting the input data into train and test datasets is used in this example so as to enable that assessment. Part of the data

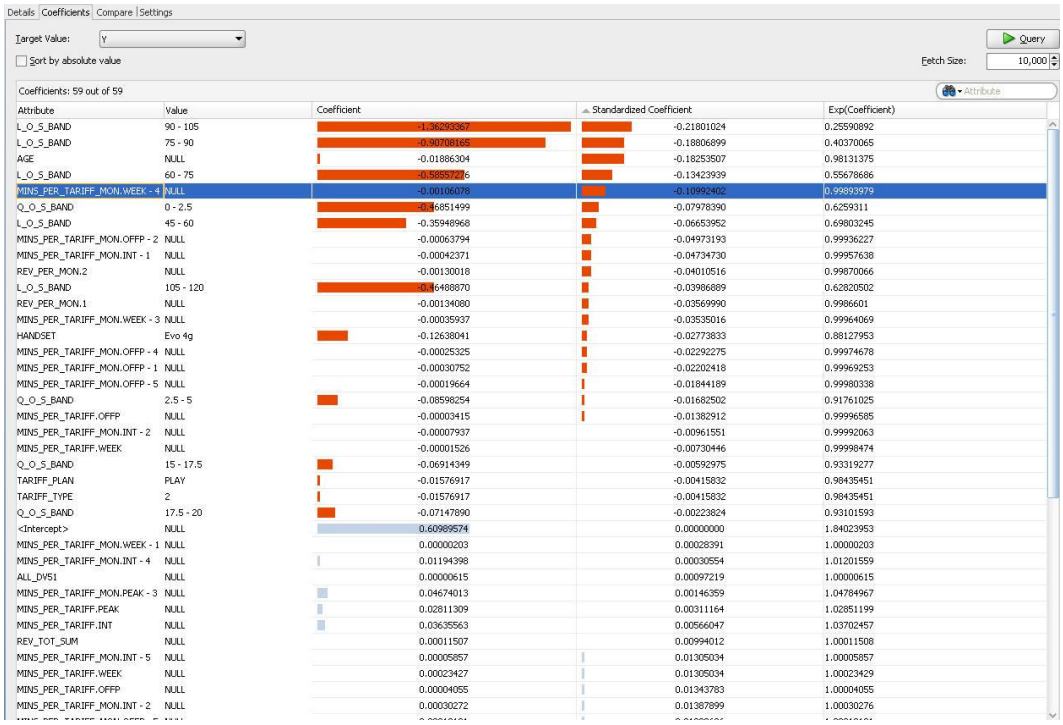
is used to train the model, and then the remaining, held-aside data is passed through the model and scored. The model's predictions are compared to the actual, known values of churn, resulting in a measure of model accuracy. There are many measures of accuracy (overall accuracy, average accuracy, lift, etc.). One measure of accuracy highlighted by Oracle Data Miner is the predictive confidence value. This measure compares the model's accuracy against a naïve model. For a model which has no improvement over the naïve model, the value for predictive confidence will be zero. Anything larger indicates that the model has some incremental benefit.



In this case, the Generalized Linear Model has a very clear benefit over the naïve model, so it definitely has predictive power. To gain more insight into the reasons for churn, you can inspect the contents of the model.



You can identify the attributes which have the strongest contribution towards churn (target value 'Y') by looking at the standardized coefficients stored in the Generalized Linear Model. Length of service and age are both important indicators. Furthermore, the number of weekend usage minutes in a recent month is also a strong indicator of churn. Recall that the minutes of usage per tariff-month combination comes from the CDR_T transactional table in the star schema, which is joined into the customer profile to enrich the data.

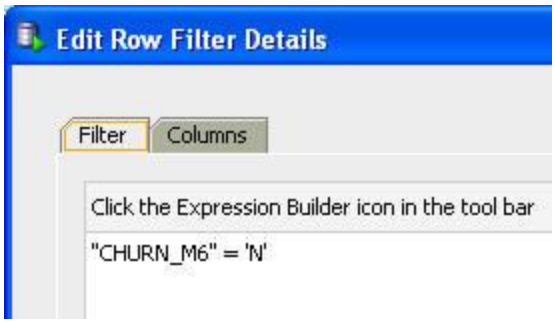


Step 8: Deployment

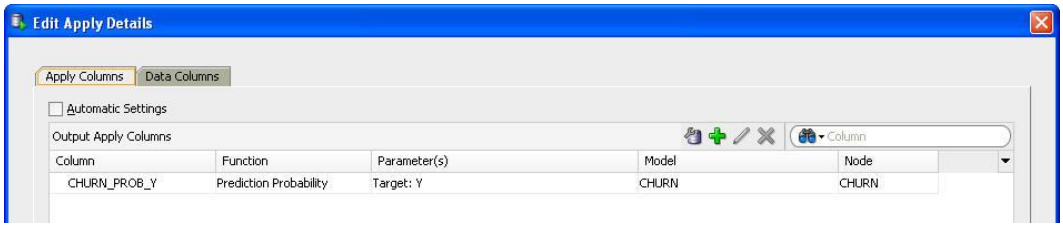
Once a model is built, most of its value is realized through deployment. This step enables the results of the churn methodology to be leveraged by the entire organization through integration with business processes.

Above and beyond all of the simplicity described in the previous steps, it is the deployment strategy that separates Oracle Data Mining from its competitors. The models are in the database. Scoring occurs in the database, alongside workloads for on-line transaction processing, data warehousing, business intelligence, vertical applications, and all other forms of production database usage. Mining results can be integrated into any and all of these environments with minimal effort on the IT side. Scoring models can be performed in parallel, both on SMPs and RAC, as well as pushed to the storage tier processing as part of Exadata Smart Scan. Real-time what-if analysis can be embedded into applications, e.g., to produce immediate churn risk updates as new information is conveyed to a telco CSR by its customer.

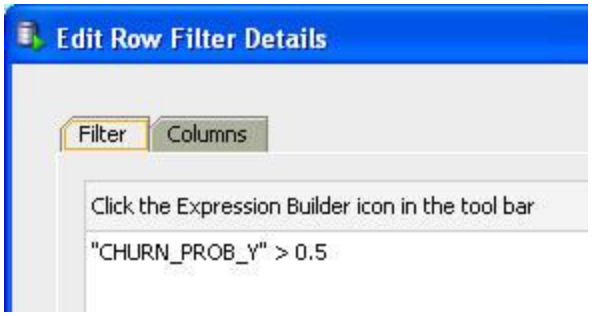
The deployment scenario in this white paper involves identifying high-valued customers that are at-risk for churn. After the model has been trained using both past and current customers, the model can be scored against current customers. The CURRENT row filter node retrieves only current customers for scoring.



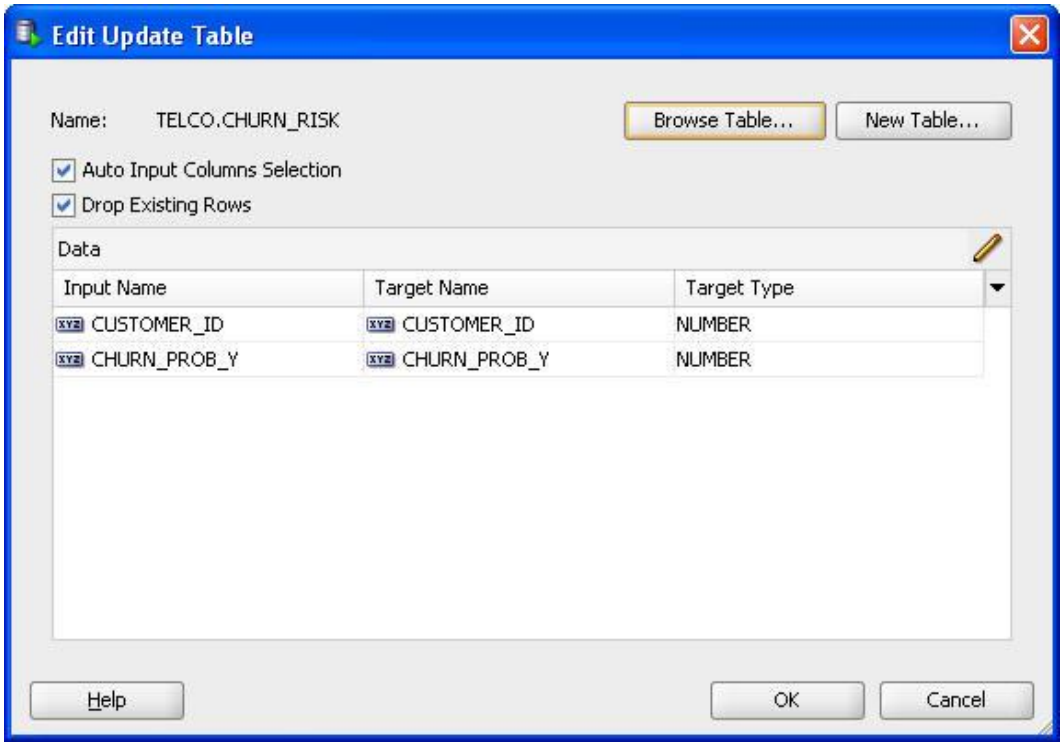
The next step is to apply, or score, the model against the customers of interest. Oracle Data Mining has built-in SQL functions for scoring models, and those are exposed through the Oracle Data Miner apply workflow node. The probability of a specific outcome – in this case churn, identified by a value of 'Y' – can be computed when scoring a model.



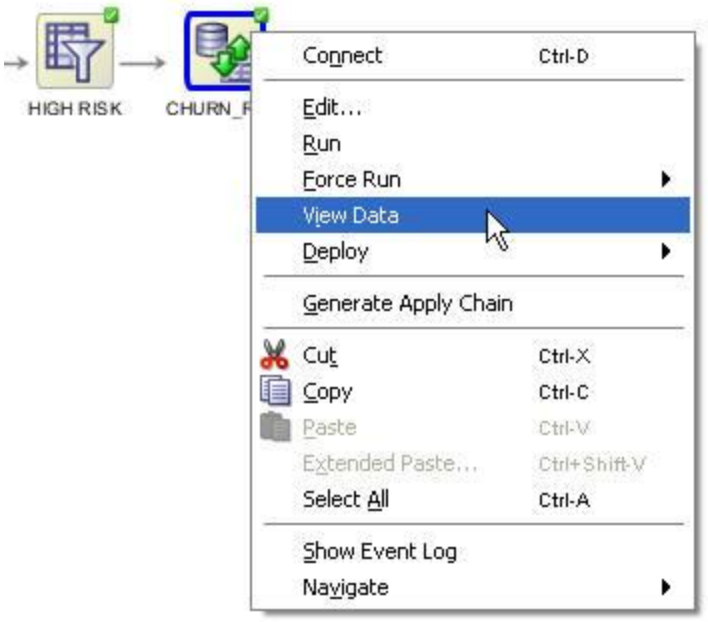
The next step in this deployment scenario is to filter the scored customers so as to track only those customers that are most at-risk.



Finally, the list of high-valued, at-risk customers can be fed into any arbitrary process. In this white paper deployment scenario, the high-valued, at-risk customers are inserted into a table called CHURN_RISK. Placing the results in a table makes them available to a host of database tools, such as OBIEE and SQL Developer, as well as enabling the information to be combined with other data in the database in arbitrary and powerful ways.



The contents of this table can be inspected from within the workflow as well.



The CHURN_DATA table is populated with the customer identifier and the associated probability of churn.

Data | Columns | SQL

View: Actual Data | Sort... | Filter: Enter Where Clause

	CUSTOMER_ID	CHURN_PROB_Y
1	3,868	0.9168
2	2,401	0.8756
3	1,141	0.841
4	2,593	0.8356
5	3,129	0.817
6	6,516	0.81
7	977	0.8025
8	4,145	0.7991
9	3,137	0.7983
10	331	0.7968
11	4,786	0.7903
12	1,496	0.7848
13	7,976	0.783

Section 6: Conclusions

Oracle Data Mining (ODM) is a simple, yet powerful, feature of the Oracle Database. ODM enables analysts to build models against data that is in a star schema, while leaving the data in its natural form. This database-centric representation yields simplicity of specification as well as performance benefits. Deployment of resulting models and methodologies is instantaneous, and can be combined with production database scenarios with ease.

Section 7: References

[CACS] Richeldi, Marco and Perrucci, Alessandro, Telecom Italia Lab, *Churn Analysis Case Study*, http://www-ai.cs.uni-dortmund.de/DOKUMENTE/richeldi_perrucci_2002b.pdf, December 17, 2002

[AOB] Mozes, Ari, *Mining a Star Schema: Telco Churn Case Study*, http://amozes-oracle.blogspot.com/2010/12/mining-star-schema-telco-churn-case.html, December 8, 2010

Oracle White Paper— Oracle Data Mining 11g Release 2: Mining Star Schemas, A Telco Churn Case Study
February 2011
Author:Ari Mozes

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

Oracle is committed to developing practices and products that help protect the environment

0109