



Deductor

АНАЛИТИЧЕСКАЯ ПЛАТФОРМА

для эффективных бизнес решений

Руководство аналитика

© BaseGroup Labs 1998-2006

www.basegroup.ru

© BaseGroup Labs 1998-2006

В руководстве описана аналитическая платформа Deductor 4.4: идеология анализа данных, реализованные механизмы, составные части, архитектура. Демонстрируются типовые задачи анализа бизнес данных и способы их решения при помощи Deductor. Книга предназначена для аналитиков, руководителей подразделений и других специалистов, которым необходимо применение в работе современных методов анализа. Специальных знаний в области анализа данных не требуется, но предполагается, что читатель знаком с вводным курсом статистики и является квалифицированным пользователем компьютера.

Содержание

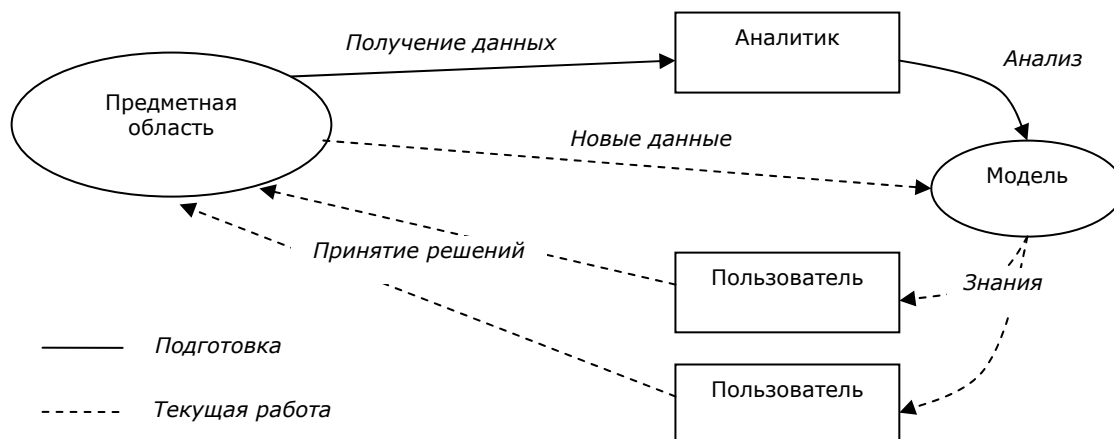
Содержание	3
Введение	5
Анализ данных – основные принципы.....	7
Два подхода к анализу данных.....	7
Базовые методы анализа	8
OLAP - On Line Analytical Processing	8
KDD - Knowledge Discovery in Databases	10
DM - Data Mining.....	11
Состав и назначение аналитической платформы Deductor	13
Deductor Studio и Deductor Warehouse	13
Поддержка процесса от разведочного анализа до отображения данных	13
Тиражирование знаний	14
Архитектура Deductor Studio – аналитическое приложение.....	17
Основные модули.....	17
Подготовка сценариев.....	17
Визуализация данных.....	21
Работа с отчетами	22
Пакетная обработка	23
Архитектура Deductor Warehouse – многомерное хранилище данных	25
Многомерное представление данных.....	25
Создание хранилища данных.....	26
Подключение к Deductor Warehouse	27
Создание структуры хранилища с помощью Редактора хранилища	27
Загрузка данных в хранилище	28
Процессы	28
Измерения	29
Автоматическая загрузка данных в хранилище	32
Импорт данных из хранилища	32
Импорт процесса	32
Импорт измерения.....	34
Работа с OLAP-кубом	37
Кросс-таблица.....	37
Размещение измерений	37
Способы агрегации и отображения фактов	40
Селектор – фильтрация данных кросс-таблицы	41
Кросс-диаграмма	44
Описание аналитических алгоритмов	47
Очистка данных	48
Парциальная обработка	48
Редактирование аномалий.....	48
Заполнение пропусков	49
Сглаживание.....	50
Очистка от шумов	51
Факторный анализ.....	52
Корреляционный анализ.....	53
Обнаружение дубликатов и противоречий.....	54
Фильтрация.....	55
Трансформация данных.....	57
Настройка набора данных	57
Настройка полей.....	57
Кэширование данных	58
Скользящее окно	59
Преобразование даты.....	60
Квантование значений	61
Сортировка	63
Слияние	63
Замена данных.....	64

Группировка.....	65
Разгруппировка	65
Data Mining.....	67
Автокорреляция	67
Нейронные сети	68
Линейная регрессия	73
Прогнозирование	75
Деревья решений.....	76
Карты Кохонена	78
Ассоциативные правила	80
Пользовательские модели	85
Вспомогательные методы обработки	87
Скрипт	87
Калькулятор.....	88
Интерпретация результатов	89
Анализ «что-если»	91
Таблица «что-если»	91
Диаграмма «что-если»	92
Подготовка данных для анализа.....	94
Выдвижение гипотез	94
Формализация и сбор данных.....	94
Представление и минимальные объемы необходимых данных	95
Построение моделей – анализ	97
Оптимизация работы и создания сценариев	99
Какие источники использовать	99
Кэширование	99
Динамические фильтры.....	100
Быстрая подготовка сценариев (скрипты).....	101
Перенастройка узла	103
Пример создания законченного аналитического решения.....	104
Создание хранилища данных.....	104
Прогнозирование объемов продаж	105
Поиск оптимальной наценки.....	109
Анализ потребительской корзины.....	110
Аналитическая отчетность.....	111
Создание отчетности	112
Что делать при возникновении ошибок	114
Заключение	116
Список литературы	117
Ссылки в Internet	117

Введение

Анализ информации является неотъемлемой частью ведения бизнеса и одним из важных факторов повышения его конкурентоспособности. При этом в подавляющем большинстве случаев все сводится к применению одних и тех же базовых механизмов анализа. Они являются универсальными и применимы к любой предметной области, благодаря чему имеется возможность создания унифицированной программной платформы, в которой реализованы основные механизмы анализа, такой как Deductor.

Обычно анализ производят аналитики и эксперты предметной области предприятия. Они подготавливают данные к пригодному для анализа виду, применяют к ним различные методы анализа, приводят результаты к легко воспринимаемому виду. Результаты анализа необходимы лицам предприятия, принимающим решения, например, руководителям отделов, менеджерам. Они могут совершенно не разбираться в методах анализа, но у них есть потребность в их результатах. Таким образом, требуется с одной стороны выделить и формализовать знание эксперта о предметной области, с другой обеспечить возможность использовать эти знания человеком, не разбирающимся в особенностях использования механизмов анализа, т.е. решить проблему тиражирования знаний (см. рисунок).



Deductor 4 предназначен для эффективного решения проблемы тиражирования знаний. Deductor – это аналитическая платформа, основа для создания законченных прикладных решений в области анализа данных. Реализованные в Deductor технологии позволяют на базе единой архитектуры пройти все этапы построения аналитической системы от создания хранилища данных до автоматического подбора моделей и визуализации полученных результатов. Deductor 4 состоит из трех частей: Deductor Warehouse – хранилища данных, консолидирующего информацию из разных источников, Deductor Studio – аналитического приложения, позволяющего пройти все этапы построения прикладного решения и Deductor Viewer – средства тиражирования знаний. Deductor 4 содержит большое количество методов подготовки, трансформации, обработки и визуализации данных.

В этом Руководстве речь пойдет о применении Deductor при решении задач анализа данных. Руководство рассчитано на аналитика, занимающегося практическими вопросами анализа информации на основе платформы Deductor. Оно требует от читателя владения лишь базовыми основами анализа и используемых в его процессе математических методов. Руководство имеет следующую структуру.

В первой части «Анализ данных – основные принципы» описываются общие вопросы анализа данных, базовые подходы и методики проведения анализа, рассматривается место аналитической системы в анализе данных.

В главе «Состав и назначение аналитической платформы Deductor» рассматриваются основные возможности, область применения и задачи, решаемые с использованием платформы Deductor.

Архитектура двух основных составных частей платформы – аналитического приложения Deductor Studio и хранилища данных Deductor Warehouse – описываются в разделах «Архитектура Deductor

Studio – аналитическое приложение» и «Архитектура Deductor Warehouse – многомерное хранилище данных».

Одним из важных методов представления данных и проведения оперативного анализа является технология OLAP. Ее основы и реализация в программах Deductor рассматриваются в главе «Работа с OLAP-кубом».

Большое внимание в Руководстве уделено описанию разнообразных алгоритмов анализа, реализованных в Deductor. Для каждого алгоритма описаны принцип работы, исходные данные и получаемые результаты, доступные настройки и, кроме того, приводятся примеры их практического использования. Вся эта информация сгруппирована в главе «Описание аналитических алгоритмов».

Одним из важнейших аспектов анализа является сбор исходных данных. Вопросы определения важности данных для анализа, объемов выборки и представления данных рассматриваются в главе «Подготовка данных для анализа».

Наконец, в последней главе «Пример создания законченного аналитического решения» рассматривается процесс создания решения на базе платформы Deductor. В ней на конкретном примере подробно описываются все этапы, которые требуется пройти, начиная от постановки задачи и заканчивая подготовкой системы отчетности.

Читатель Руководства будет ознакомлен со всеми аспектами применения платформы Deductor в анализе данных и сможет самостоятельно приступить к разработке аналитических моделей и готовых решений на их базе, а также их использованию на практике.

В конце Руководства приведен список литературы, наиболее активно использовавшейся в ходе разработки платформы Deductor. В ней рассматриваются как общие вопросы Data Mining и Knowledge Discovery, так и более узкие темы – нейронные сети, статистические методы анализа, генетические и нечеткие алгоритмы, подходы к решению отдельных проблем анализа (прогнозирование, кластеризация, факторный, корреляционный и другие виды анализа). Кроме того, к списку литературы добавлены несколько полезных ссылок на сайты в Internet, имеющие непосредственное отношение к анализу данных.

Анализ данных – основные принципы

Два подхода к анализу данных

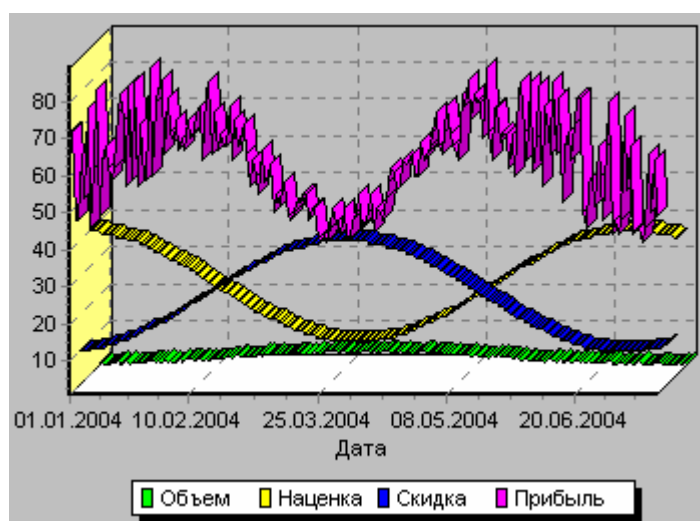
Любая организация в процессе своей деятельности стремится повысить прибыль и уменьшить расходы. В этом ей помогают новые компьютерные технологии, использование разнообразных программ автоматизации бизнес-процессов. Это учетные, бухгалтерские и складские системы, системы управленческого учета и многие другие. Чем аккуратнее и полнее ведется сбор и систематизация информации, тем полнее будет представление о процессах в организации. Современные носители информации позволяют хранить десятки и сотни гигабайт информации. Без использования специальных средств хранения и анализа накопленной информации такие носители превращаются просто в свалку мусора. Очень часто принятие правильного решения затруднено из-за того, что хотя данные и имеются, они являются неполными, или наоборот, избыточными, замусорены информацией, которая вообще не имеет отношения к делу, несистематизированными или систематизированными неверно. Тогда прибегают к помощи программных средств, которые позволяют привести информацию к виду, который позволяет с достаточной степенью достоверности оценить содержащиеся в ней факты и повысить вероятность принятия оптимального решения.

Есть два подхода к анализу данных с помощью информационных систем.

В первом варианте программа используется для *визуализации* информации - извлечения данных из источников и предоставления их человеку для самостоятельного анализа и принятия решений. Обычно данные, предоставляемые программой, являются простой таблицей, и в таком виде их очень сложно анализировать. В этом случае зачастую используются разнообразные средства визуализации информации: многомерный анализ данных, диаграммы и гистограммы.

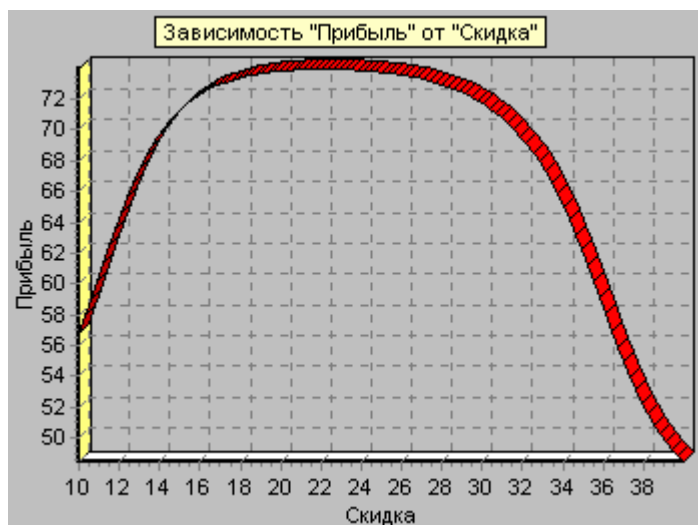
Второй вариант использования программы – это построение моделей. Модель имитирует некоторый процесс, например, изменение объемов продаж некоторого товара, поведение клиентов и другое. Для построения модели необходимо сделать предобработку данных и далее к ним применять математические методы анализа: кластеризацию, классификацию, регрессию и т. д. Построенную модель можно использовать либо непосредственно как результат анализа, либо для принятия решений. В последнем случае модель используется для оценки влияния различных факторов на результат.

Рассмотрим пример. Предоставление скидки покупателям является стимулом для увеличения объемов закупок. Чем больше продается некоторого товара, тем больше прибыль. С другой стороны, чем больше предоставляется скидка, тем меньше наценка на товар, и тем меньше прибыли приносят продажи этого товара. Пусть есть история продаж, представленная таблицей со столбцами: дата, объем продаж, скидка в процентах, наценка и прибыль. При проведении анализа «вручную» можно рассмотреть диаграмму.



На диаграмме видно, что при максимальной скидке прибыль минимальна. Максимум же прибыли достигается где-то посередине между минимальной и максимальной скидкой. Точный ответ об

оптимальной скидке по этой диаграмме получить довольно трудно. Для этого можно воспользоваться другим подходом к анализу. Например, построить модель зависимости прибыли от скидки.



По этой зависимости легко определить, что максимум прибыли достигается при скидке 22 процента.

Как визуализация, так и построение моделей осуществляются путем применения к данным базовых методов анализа. Это достаточно известные методы и они используются в самых разнообразных сферах деятельности.

Базовые методы анализа

OLAP - On Line Analytical Processing

Любая система поддержки принятия решений, прежде всего, должна обладать средствами отбора и представления пользователю данных в удобной для восприятия и анализа форме. Как правило, наиболее удобными для анализа являются многомерные данные, описывающие сразу многие предметную область сразу с нескольких точек зрения. Для описания таких наборов данных вводится понятие многомерных кубов (гиперкубов, метакубов). По осям такого куба размещаются параметры, а в ячейках – зависящие от них данные. Вдоль каждой оси представлены различные уровни детализации данных. Использование такой модели данных позволяет повысить эффективность работы с ними – генерировать сложные запросы, создавать отчеты, выделять подмножества данных и т.д. Технология комплексного многомерного анализа данных и представления результатов этого анализа в удобной для использования форме получила название OLAP.

OLAP - OnLine Analytical Processing – оперативная аналитическая обработка данных. OLAP дает возможность в реальном времени генерировать описательные и сравнительные сводки данных и получать ответы на различные другие аналитические запросы. OLAP-кубы представляют собой проекцию исходного куба данных на куб данных меньшей размерности. При этом значения ячеек агрегируются, то есть объединяются с применением функции агрегации – сумма, среднее, количество, минимум, максимум. Такие проекции или срезы исходного куба представляются на экране в виде кросс-таблицы.

Проиллюстрируем идею OLAP-куба на простом примере. Пусть, информацию, хранящуюся в базе данных или подмножество данных, получаемое в результате выполнения запроса можно представить в виде таблицы:

№ п/п	Город	Годовой объем продаж (руб.)			Итого
		Товар 1	Товар 2	Товар 3	
1	Москва	525000	234000	325000	1084000
2	Тула	224000		12560	236560
3	Владимир		123000	425000	548000
4	Самара	78000	234000	45000	357000

5	Тверь		45000	78000	123000
---	-------	--	-------	-------	--------

Основываясь на данных из таблицы, можно дать ответы на несколько вопросов, которые могут возникнуть при анализе объемов продаж. Например.

Каков объем продаж каждого товара по одному из городов?

	Владимир
Товар 1	
Товар 2	123000
Товар 3	425000

Данную выборку можно интерпретировать как одномерную, поскольку объемы продаж расположены только вдоль одного измерения с наименованием товара.

Каков объем продаж каждого из товаров по городам?

Город	Товар 1	Товар 2	Товар 3
Москва	525000	234000	325000
Тула	224000		12560
Владимир		123000	425000
Самара	78000	234000	45000
Тверь		45000	78000

Данная выборка является двумерной и может быть представлена в виде «плоского куба», в котором 3-е измерение «отключено».

Москва	525000	234000	325000
Тула	224000		12560
Владимир		123000	425000
Самара	78000	234000	45000
Тверь		45000	78000
	Товар 1	Товар 2	Товар 3

Чтобы представленный куб превратился в 3-х мерный необходимо добавить еще одно измерение данных, т.е., практически, привлечь дополнительную информацию по анализируемым параметрам. Например, если в базе данных предусмотрена информация о квартальном объеме продаж каждого товара по всем городам, то номер квартала может стать этим дополнительным измерением.

Город	Поквартальный объем продаж (тыс. руб.)											
	Товар 1				Товар 2				Товар 3			
	I	II	III	IV	I	II	III	IV	I	II	III	IV
Москва	125,0	125,0	150,0	125,0	85,0	70,0	45,0	34,0	75,0	75,0	75,0	100,0
Тула	45,0	55,0	100,0	24,0					5,0	5,0		2,56
Владимир					45,0	40,0	25,0	13,0	10,0	10,0	10,0	12,5
Самара	12,5	15,5	32,0	18,0	85,0	65,0	50,0	34,0	15,0	5,0	17,0	13,0
Тверь					10,0	10,0	10,0	15,0	20,0	15,0	25,0	18,0

На основе подобной выборки можно построить 3-х мерный куб.

	I	II	III	IV
Москва	125000	85000	75000	
Тула	45000		5000	
Владимир		45000	10000	
Самара	12500	85000	15000	
Тверь		10000	20000	
	Товар1	Товар2	Товар3	

Такая модель представления данных позволяет получать нужную информацию, производя соответствующие сечения (срезы) OLAP-куба.

Нет необходимости пытаться произвести геометрическую интерпретацию OLAP-куба с размерностью более 3-х. Действительно, поскольку человеческое сознание «приспособлено» к восприятию 3-х мерной действительности, все прочие представления сложны для восприятия. Тем более что речь идет не о реальном, а об информационном пространстве, а само понятие «многомерный куб» есть не что иное, как служебный термин, используемый для описания метода.

В принципе, используемое число измерений может быть любым. Однако следует отметить, что задача с большим числом измерений, во-первых, является трудоемкой с точки зрения ее выполнения на ПК и, во-вторых, ее осмысление и интерпретация результатов аналитиком могут быть затруднены и даже приводить к ошибочным решениям. Поэтому с методической точки зрения, сложные задачи, требующие анализа данных большой размерности, следует по возможности сводить к нескольким более простым. Аналогично, сложные таблицы, которые содержат большое количество полей и записей, являющихся трудными для чтения, восприятия и анализа, можно разбить на несколько более простых таблиц. Это сделает работу с ними намного более удобной.

KDD - Knowledge Discovery in Databases

KDD - Knowledge Discovery in Databases – извлечение знаний из баз данных. Это процесс поиска полезных знаний в «сырых данных». KDD включает в себя вопросы подготовки данных, выбора информативных признаков, очистки данных, применения методов Data Mining (DM), постобработки данных, интерпретации полученных результатов.

Привлекательность этого подхода заключается в том, что, вне зависимости от предметной области, мы применяем одни и те же операции:

1. Подготовка исходного набора данных. Этот этап заключается в создании набора данных, в том числе из различных источников, выбора обучающей выборки для обучения нейросетей и т.д. Для этого должны существовать развитые инструменты доступа к различным источникам данных: файлам разных форматов, базам данных, учетным системам.
2. Предобработка данных. Для того чтобы эффективно применять методы анализа, следует обратить серьезное внимание на вопросы предобработки данных. Данные могут содержать пропуски, шумы, аномальные значения и т.д. Кроме того, данные могут быть избыточны, недостаточны и т.д. В некоторых задачах требуется дополнить данные некоторой априорной информацией. Наивно предполагать, что если подать данные на вход системы в существующем виде, то на выходе получим полезные знания. Данные должны быть качественны и корректны с точки зрения используемого метода анализа. Более того, иногда размерность исходного пространства может быть очень большой, и тогда желательно применение специальных алгоритмов понижения размерности: отбор наиболее значимых признаков и отображение данных в пространство меньшей размерности.

3. Трансформация данных. Для различных методов анализа требуются данные, подготовленные в специальном виде. Например, где-то в качестве входов может использоваться только цифровая информация.
4. Data Mining. На этом шаге применяются различные алгоритмы для нахождения знаний. Это нейронные сети, деревья решений, алгоритмы кластеризации и установления ассоциаций и т.д.
5. Постобработка данных. Интерпретация результатов и практическое применение полученных знаний в бизнесе.

Описанный процесс повторяется итеративно, а реализация этих этапов позволяет автоматизировать процесс извлечения знаний.

Например, нужно сделать прогноз объемов продаж на следующий месяц. Есть сеть магазинов розничной торговли. Первым шагом будет сбор истории продаж в каждом магазине и объединение ее в общую выборку данных. Следующим шагом будет предобработка собранных данных. Например, их группировка по месяцам, сглаживание кривой продаж, устранение факторов, слабо влияющих на объемы продаж. Далее следует построить модель зависимости объемов продаж от выбранных факторов. Это можно сделать с помощью линейной регрессии или нейронных сетей. Имея такую модель, можно получить прогноз, подав на вход модели нашу историю продаж. Зная прогнозное значение, его можно использовать, например, для оптимизации размещения товара на складе.

DM - Data Mining

DM - Data Mining – добыча данных. Это метод обнаружения в «сырых» данных ранее неизвестных, нетривиальных, практически полезных и доступных для интерпретации знаний, необходимых для принятия решений в различных сферах человеческой деятельности. DM обеспечивает решение всего шести задач — классификация, кластеризация, регрессия, ассоциация, последовательность и отклонения.

1. *Классификация* — это отнесение объектов (наблюдений, событий) к одному из заранее известных классов.
2. *Кластеризация* — это группировка объектов (наблюдений, событий) на основе данных (свойств), описывающих сущность объектов. Объекты внутри кластера должны быть «похожими» друг на друга и отличаться от объектов, вошедших в другие кластеры. Чем больше похожи объекты внутри кластера и чем больше отличий между кластерами, тем точнее кластеризация.
3. *Регрессия*, в том числе задачи *прогнозирования*. Установление функциональной зависимости между входными и *непрерывными* выходными переменными.
4. *Ассоциация* — выявление зависимостей между связанными событиями, указывающих, что из события X следует событие Y. Такие правила называются ассоциативными. Впервые эта задача была предложена для нахождения типичных шаблонов покупок, совершаемых в супермаркетах, поэтому иногда ее еще называют анализом потребительской корзины (*market basket analysis*).
5. *Последовательные шаблоны* — установление закономерностей между связанными во времени событиями. Например, после события X через определенное время произойдет событие Y.
6. *Анализ отклонений* — выявление наиболее нехарактерных шаблонов.

Примеры задач, где применяются эти методы.

Классификация используется в случае, если заранее известны классы отнесения объектов. Например, отнесение нового товара к той или иной товарной группе, отнесение клиента к какой-либо категории. При кредитовании это может быть, например, отнесение клиента по каким-то признакам к одной из групп риска.

Кластеризация может использоваться для сегментирования и построения профилей клиентов (покупателей). При достаточно большом количестве клиентов становится трудно подходить к каждому индивидуально. Поэтому клиентов удобно объединить в группы – сегменты со сходными признаками. Выделять сегменты клиентов можно по нескольким группам признаков. Это могут быть сегменты по сфере деятельности, по географическому расположению. После сегментации можно узнать, какие именно сегменты являются наиболее активными, какие приносят наибольшую

прибыль, выделить характерные для них признаки. Эффективность работы с клиентами повышается за счет учета их персональных предпочтений.

Регрессия чаще всего используется при прогнозировании объемов продаж, в этом случае зависимой величиной являются объемы продаж, а факторами, влияющими на эту величину, могут быть предыдущие объемы продаж, изменение курса валют, активность конкурентов и т.д. Или, например, при кредитовании физических лиц вероятность возврата кредита зависит от личных характеристик человека, сферы его деятельности, наличия имущества.

Ассоциации помогают выявлять совместно приобретаемые товары. Это может быть полезно для более удобного размещения товара на прилавках, стимулирования продаж. Тогда человек, купивший пачку спагетти, не забудет купить к ним бутылочку соуса.

Последовательные шаблоны могут быть использованы, например, при планировании продаж или предоставлении услуг. Например, если человек приобрел фотопленку, то через неделю он отдаст ее на проявку и закажет печать фотографий.

Для анализа отклонений необходимо сначала построить шаблон типичного поведения изучаемого объекта. Например, поведение человека при использовании кредитных карт. Тогда будет известно, что клиент (покупатель) использует карту регулярно два раза в месяц и приобретает товар в пределах определенной суммы. Отклонением будет, например, не запланированное приобретение товара по данной карте на большую сумму. Это может говорить об ее использовании другим лицом, то есть о факте мошенничества.

Перечисленные выше базовые методы анализа данных используются для создания аналитических систем. Причем, под такой системой понимается не только какая-то одна программа. Некоторые механизмы анализа могут быть реализованы на бумаге, некоторые на компьютере с использованием электронных таблиц, баз данных и других приложений. Однако, такой подход при частом использовании не эффективен. Намного лучшие результаты даст применение единого хранилища данных и единой программы, содержащей в себе всю функциональность, необходимую для реализации концепции KDD.

Состав и назначение аналитической платформы Deductor

Deductor Studio и Deductor Warehouse

Deductor состоит из трех компонентов: аналитического приложения Deductor Studio, многомерного хранилища данных Deductor Warehouse и средства тиражирования знаний Deductor Viewer.

Deductor Warehouse – многомерное хранилище данных, аккумулирующее всю необходимую для анализа предметной области информацию. Использование единого хранилища позволяет обеспечить непротиворечивость данных, их централизованное хранение и автоматически обеспечивает всю необходимую поддержку процесса анализа данных. Deductor Warehouse оптимизирован для решения именно аналитических задач, что положительно сказывается на скорости доступа к данным.

Deductor Studio – это программа, предназначенная для анализа информации из различных источников данных. Она реализует функции импорта, обработки, визуализации и экспорта данных. Deductor Studio может функционировать и без хранилища данных, получая информацию из любых других источников, но наиболее оптимальным является их совместное использование.

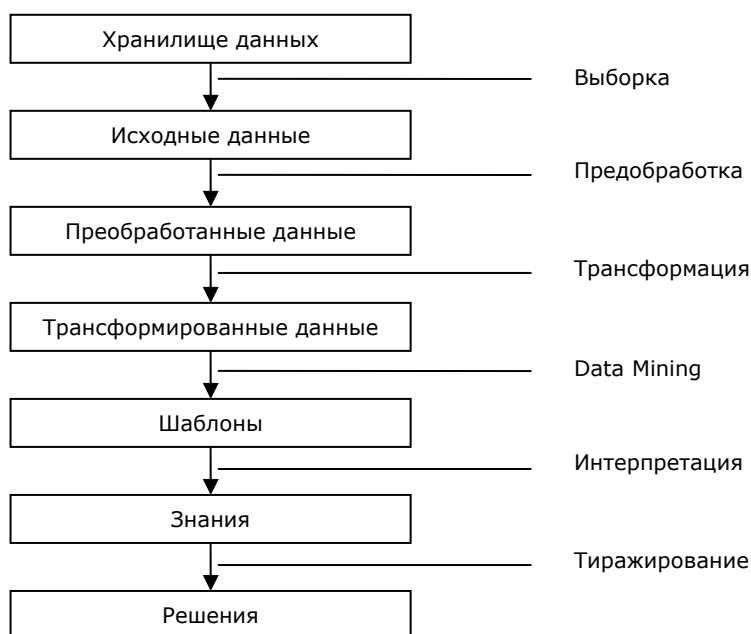
Deductor Viewer – это облегченная версия Deductor Studio, предназначенная для отображения построенных в Deductor Studio отчетов. Она не включает в себя механизмов создания сценариев, но обладает полноценными возможностями по их выполнению и визуализации результатов. Deductor Viewer является средством тиражирования знаний для конечных пользователей, которым не требуется знать механику получения результатов или изменять способы их получения.

Studio и Viewer базируются на одной архитектуре, поэтому все дальнейшее описание, касающееся визуализации и интерпретации результатов в равной мере относится к обоим приложениям.

Поддержка процесса от разведочного анализа до отображения данных

Deductor Studio позволяет пройти все этапы KDD, перечисленные выше.

Схема на рисунке отображает процесс извлечения знаний из данных.



На начальном этапе в программу импортируются данные из какого-либо источника. Хранилище данных Deductor Warehouse также является одним из источников данных. Обычно в программу загружаются не все данные, а какая-то выборка, необходимая для дальнейшего анализа. После получения выборки можно получить подробную статистику по ней, просмотреть, как выглядят данные на диаграммах и гистограммах. Следующим шагом является принятие о необходимости предобработки данных. Например, если выясняется, что в выборке есть пустые значения (пропуски данных), можно применить фильтрацию для их устранения. Преобработанные данные далее

подвергаются трансформации. Например, нечисловые данные преобразуются в числовые, что является необходимым условием для некоторых алгоритмов. Непрерывные данные могут быть разбиты на интервалы, то есть производится их квантование. К трансформированным данным применяются методы более глубокого анализа. На этом этапе выявляются скрытые зависимости и закономерности в данных, на основании которых строятся различные модели. Модель представляет собой шаблон, который содержит в себе формализованные знания. Следующий этап – интерпретация – предназначен для того, чтобы из формализованных знаний получить знания на языке предметной области. Наконец, последним этапом является тиражирование знаний – предоставление людям, принимающим решения, возможности практического применения построенных моделей.

Тиражирование знаний

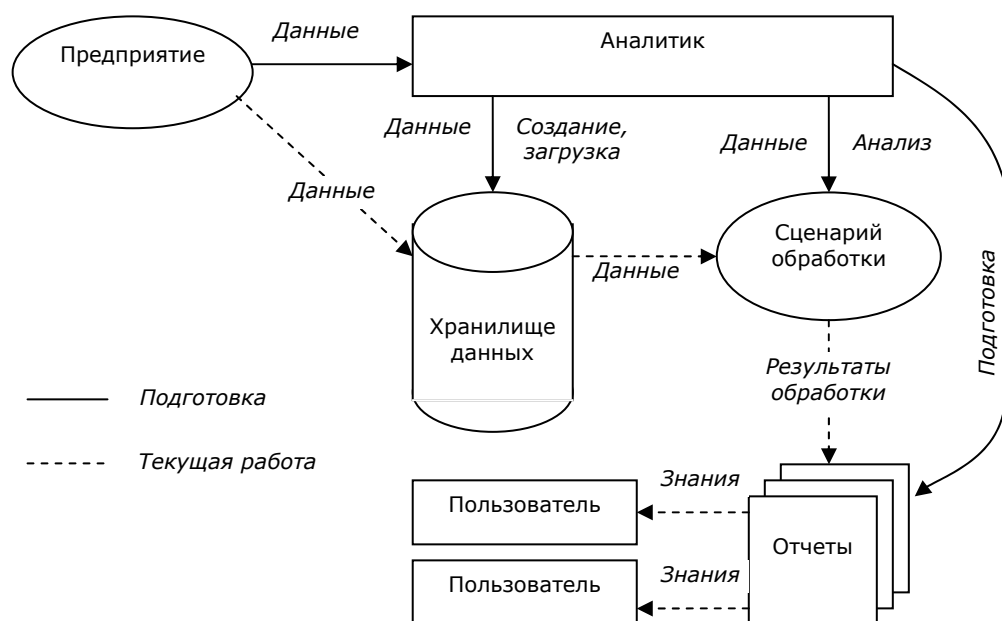
Одной из наиболее важных функций любой аналитической системы является поддержка процесса тиражирования знаний, т. е. обеспечение возможности сотрудникам, не разбирающимся в методиках анализа и способах получения того или иного результата, получать ответ на основе моделей, подготовленных экспертом.

Например, сотрудник, оформляющий кредиты, должен внести данные по потребителю, а система автоматически выдать ответ, на какую сумму кредита данных потребитель может рассчитывать. Либо сотрудник отдела закупок при оформлении заказа должен получить автоматически рассчитанный, рекомендуемый объем закупки каждого товара.

Потребность в тиражировании знаний является объективной, так как, с одной стороны, интеллектуальная составляющая бизнеса становится все более значимой, с другой стороны, невозможно требовать от каждого специалиста, чтобы он разбирался в механизмах анализа. Создание моделей, поиск зависимостей и прочие задачи анализа нетривиальны. Подготовить систему, которая могла бы на всех данных гарантированно давать качественные результаты (прогнозы, рекомендации и прочее) не представляется возможным. Слишком широк спектр решаемых задач и слишком отличается логика бизнеса в различных компаниях. Но использование аналитической платформы Deductor позволяет по-другому подойти к решению задачи построения и использования моделей. Эксперт готовит сценарии обработки (модели) с учетом особенностей конкретного бизнеса, а остальные пользователи просто используют уже готовые модели, получая отчеты, прогнозы, правила не задумываясь о том, как эти модели работают.

Делается это следующим образом (см. рисунок):

1. Создается хранилище данных – Deductor Warehouse, консолидирующее всю необходимую для анализа информацию. Настраиваются механизмы автоматического обновления сведений в хранилище. Данная операция гарантирует оперативное получение актуальной, непротиворечивой и целостной информации для анализа.
2. Эксперт настраивает сценарии обработки, т.е. определяет последовательность шагов, которую необходимо провести для получения нужного результата. Почти всегда результат нельзя получить за одну операцию. Обычно это целая цепочка различного рода обработок. Например, для получения качественного прогноза необходимо получить сгруппированные нужным образом данные, провести очистку их от выбросов, сгладить, построить модель и «прогнать» через эту модель новые данные. Подготовка сценариев наиболее сложная часть работы, требующая серьезных знаний в предметной области и понимание методов анализа, но эту работу может провести один человек в компании.
3. Вывести на панель «Отчеты» ту информацию, которую необходимо получить пользователю системы, при этом, сгруппировав ее в папки в зависимости от решаемой задачи.
4. Настроить способы визуализации полученных данных. Подобрать наиболее оптимальные методы отображения.



В результате пользователю будут доступны результаты обработки в наиболее удобном для анализа виде. При выборе того или иного отчета система автоматически проведет все необходимые операции и предоставит результат. Данный механизм позволяет отделить процедуру подготовки сценариев, требующую определенных знаний и собственно получение результатов с использованием готовых сценариев. Таким образом, решается задача тиражирования знаний.

Для того чтобы предоставлять конечному пользователю только необходимую ему информацию и не затруднять его пониманием используемых при анализе средств и методов, к перечисленным выше этапам тиражирования знаний можно добавить еще один – тиражирование с использованием Deductor Viewer. Deductor Viewer не включает в свой состав лишних для обычного пользователя инструментов, а лишь необходимую ему систему визуализации.

При использовании Deductor Viewer может быть решен такой важный вопрос, как разграничение прав пользователей на доступ к информации. Для того чтобы обеспечить аналитическую поддержку всех подразделений предприятия, в хранилище данных нужно загружать большое количество разнообразной информации, например, о финансовом положении фирмы, отношениях с поставщиками и клиентами, товарной, ценовой и конкурентной политике и т.д. В результате человек, имеющий доступ к хранилищу, получает в свое распоряжение большую часть информации о деятельности компании и неизвестно, каким образом он захочет ею распорядиться. Если каждому пользователю аналитической информации установить Deductor Studio, он сможет извлечь из хранилища не только те данные, которые ему нужны для работы, но и другую конфиденциальную информацию, доступа к которой он иметь не должен. Например, сотруднику склада, которому нужны отчеты по остаткам товаров и срокам годности, совершенно нет необходимости знать о финансовых результатах деятельности компании за прошедший год. Deductor Viewer позволяет ограничить доступ различных категорий пользователей к информации из хранилища или других источников данных.

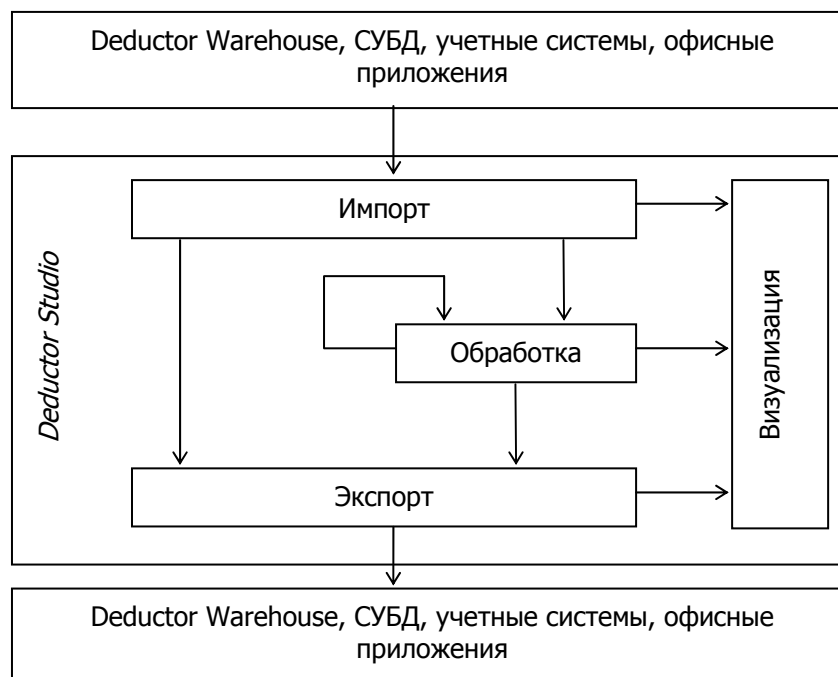
Так как Deductor Viewer не включает в свой состав средств для самостоятельного извлечения или анализа данных, то можно предоставить каждому пользователю отчеты, содержащие только нужную ему для работы информацию. Доступа к другим данным из хранилища и баз данных этот пользователь не получит.

Архитектура Deductor Studio – аналитическое приложение

Основные модули

Вся работа по анализу данных в Deductor Studio базируется на выполнении следующих действий:

- импорт данных;
- обработка данных;
- визуализация;
- экспорт данных.





На рисунке показана схема функционирования Deductor Studio. Отправной точкой для анализа всегда является процедура импорта данных. Полученный набор данных может быть обработан любым из доступных способов. Результатом обработки также является набор данных, который в свою очередь опять может быть обработан. Импортированный набор данных, а также данные, полученные на каждом этапе обработки, могут быть экспортированы. Результаты каждого действия можно отобразить различными способами. Способ возможных отображений зависит от выбранного метода обработки данных. Например, нейросеть содержит визуализатор «Граф нейросети», специфичный только для нее. Есть способы визуализации, пригодные почти для всех методов обработки, например, в виде таблицы, диаграммы или гистограммы

Последовательность действий, которые необходимо провести для анализа данных называется **сценарием**. Сценарий можно автоматически выполнять на любых данных.

Подготовка сценариев

Перечисленные выше действия реализуются с помощью четырех мастеров: импорта, обработки, визуализации и экспорта. Для построения сценария достаточно использовать только эти мастера и ничего более.

Сценарий отображается на панели сценариев. Показать или скрыть эту панель можно, выбрав пункт «Сценарии» меню «Вид» или нажав на кнопку  на панели инструментов. Сверху на панели сценариев расположены кнопки для вызова мастеров.

Построение сценария начинается с вызова мастера импорта. *Мастер импорта* предназначен для автоматизации получения данных из любого источника, предусмотренного в системе. Чтобы вызвать это действие, достаточно воспользоваться кнопкой  «Мастер импорта» в верхней части панели или выбрать соответствующую команду из контекстного меню, вызываемого щелчком правой кнопки мыши в любом месте панели «Сценарии». На первом шаге мастера импорта открывается список всех предусмотренных в системе типов источников данных. Среди них следует

выбрать нужный тип источника и, для перехода на следующий шаг, щелкнуть по кнопке «Далее». Число шагов мастера импорта, а также набор настраиваемых параметров отличается для разных типов источников.

Например, если исходные данные хранятся в текстовом файле с разделителями, нужно выбрать источник данных Text (Direct). Direct – означает прямой доступ к текстовому файлу, что увеличивает скорость работы с ним. Важным шагом в мастере импорта является настройка полей. Каждому полю источника данных можно присвоить метку столбца, которая будет использоваться для дальнейшей работы в программе. Например, если в источнике данных поле имеет имя «Name», ему можно задать метку «Наименование», что гораздо удобнее при дальнейшем отображении этого поля в таблицах или диаграммах.


Далее каждому полю нужно указать тип:

- логический - данные в поле могут принимать только два значения - 0 или 1 (ложь или истина);
- дата/время - поле содержит данные типа дата/время;
- вещественный - значения поля - числа с плавающей точкой;
- целый - данные в поле представляют собой целые числа;
- строковый - данные в столбце представляют собой строки символов.

Затем указывается вид данных:


- непрерывный - значения в столбце могут принимать любое значение в рамках своего типа. Как правило, непрерывными являются только числовые данные;
- дискретный - данные в столбце могут принимать ограниченное число значений. Как правило, дискретный характер несут строковые данные.

В зависимости от содержимого поля «Тип данных», на выбор вида данных накладываются ограничения – например, строковые данные не могут быть непрерывными. К выбору типа и вида данных нужно относиться серьезно, так как это влияет на возможность дальнейшего использования этого поля. Неправильное указание типа данных может также привести к потере информации. Например, если в поле хранятся дробные числа с плавающей точкой, а вид этого поля мы указали «целый», то в полученной после импорта таблице все значения этого поля окажутся пустыми, а в логе появятся сообщения об ошибках преобразования типов.


Мастер обработки предназначен для настройки всех параметров выбранного алгоритма. Для вызова мастера обработки, достаточно воспользоваться кнопкой  «Мастер обработки» в верхней части панели или выбрать соответствующую команду из контекстного меню, вызываемого щелчком правой кнопки мыши в любом месте панели «Сценарии». В окне первого шага мастера обработки представлены все доступные в системе методы обработки данных. Как правило, на следующем шаге мастера обработки производится настройка назначений полей. В зависимости от выбранного алгоритма предлагается выбрать некоторые из перечисленных назначений:

- непригодное - данные в поле не пригодны для данного способа обработки (программа автоматически указывает полю это назначение). Например, для преобразования даты поле должно иметь тип «Дата/время». Если оно будет иметь, например, строковый тип, то программа автоматически укажет для него назначение «Непригодное».
- неиспользуемое - запрещает использование поля в обработке данных и исключает его из выходного набора. В отличие от непригодного поля, такие поля в принципе могут использоваться, просто в этом нет необходимости;
- первичный ключ - поле будет использоваться в качестве первичного ключа;
- входное - поле таблицы, построенное на основе столбца, будет являться входным полем обработчика (нейронной сети, дерева решений и т.д.).
- выходное - поле таблицы, построенное на основе столбца, будет являться выходным полем обработчика (например, целевым полем для обучения нейронной сети).
- информационное - поле содержит вспомогательную информацию, которую часто полезно отображать, но не следует использовать при обработке;
- измерение - поле будет использоваться в качестве измерения в многомерной модели данных;
- факт - значения поля будут использованы в качестве фактов в многомерной модели данных.
- свойство - поле содержит описание свойств или параметров некоторого объекта;

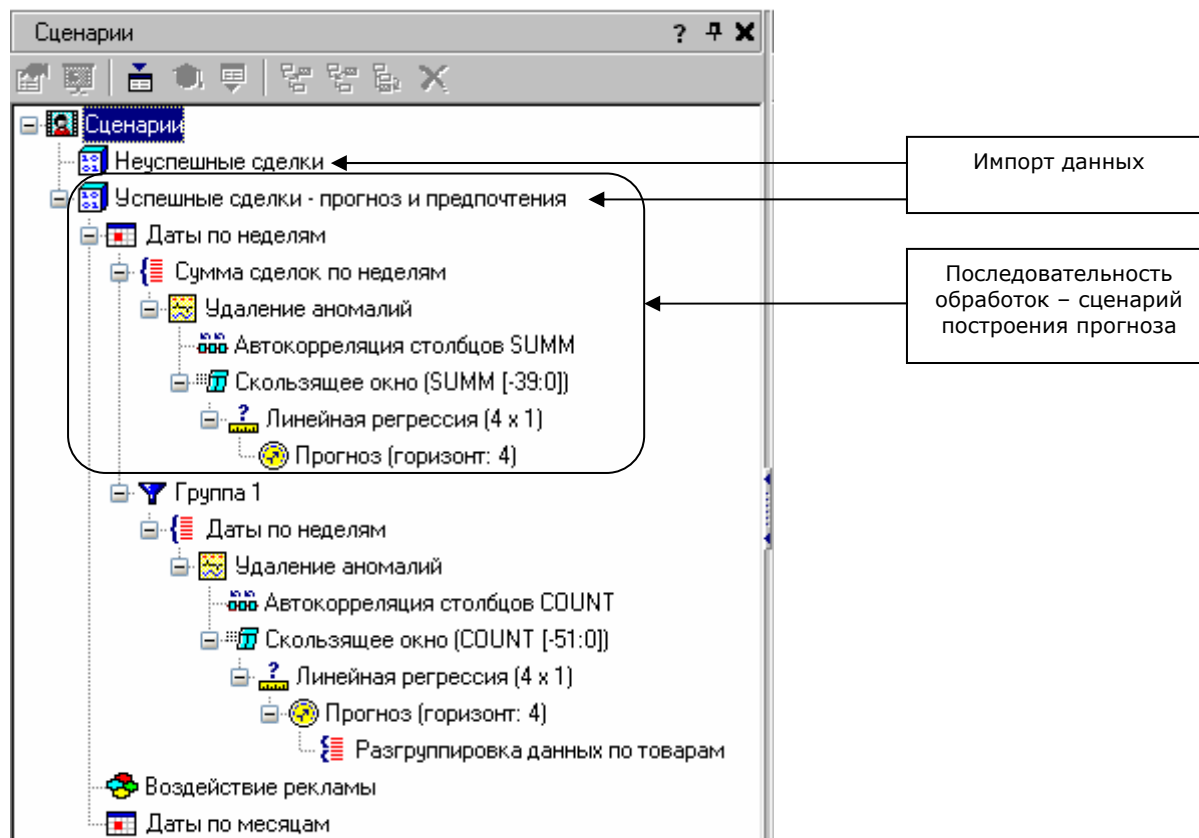
- транзакция – поле, содержащее идентификатор событий, происходящих совместно (одновременно). Например, номер чека, по которому приобретены товары. Тогда покупка товара – это событие, а их совместное приобретение по одному чеку - транзакция;
- элемент – поле, содержащее элемент транзакции (событие).

Мастер отображений позволяет в пошаговом режиме выбрать и настроить наиболее удобный способ представления данных. В зависимости от выбранного способа представления будут настраиваться различные параметры, а Мастер, соответственно, будет содержать различное число шагов. Для вызова мастера отображений можно воспользоваться кнопкой  «Мастер визуализации» на панели сценариев, предварительно выделив нужную ветвь в сценарии обработки, или выбрав соответствующую команду из контекстного меню, вызываемого для данной ветви сценария. В зависимости от метода обработки, в результате которого была получена ветвь сценария обработки, список доступных для нее видов отображений будет различным. Например, после построения деревьев решений их можно отобразить с помощью визуализаторов «Деревья решений» и «Правила». Эти способы отображения не доступны для других обработок. Одновременно может быть выбрано несколько способов визуализации, при этом каждое из них будет открыто на отдельной закладке. Если одновременно выбрано несколько способов отображения данных, то все соответствующие шаги будут последовательно включены в общую процедуру настройки. Например, если выбраны и диаграмма и гистограмма, то в Мастера отображений будут последовательно включены отдельные шаги для настройки диаграммы и гистограммы.

Мастер экспорта позволяет в пошаговом режиме выполнить экспорт данных в файлы наиболее распространенных форматов, различные базы данных и хранилище данных Deductor Warehouse.


Для вызова мастера экспорта можно воспользоваться кнопкой  «Мастер экспорта» на панели сценариев. На первом шаге мастера экспорта представлен список всех форматов, в которые может быть выполнен экспорт данных. Среди них следует выбрать нужный и далее следовать шагам Мастера. В результате набор данных будет выгружен в выбранный приемник в виде обычной таблицы.

С помощью последовательного применения мастеров импорта, обработки и экспорта можно построить сценарий. Пример сценария на рисунке.







Выполнение сценария – это последовательное выполнение обработок, начиная от узла импорта данных, до выбранного узла сценария. Для выполнения сценария необходимо выбрать нужный узел двойным щелчком мыши. Например, если дважды щелкнуть мышью по узлу «Прогноз (горизонт: 4)» (см. рисунок), то выполнится сценарий: «Успешные сделки – прогноз и предпочтения» (импорт данных), «Даты по неделям» (преобразование даты), «Сумма сделок по неделям» (группировка данных), «Удаление аномалий», «Скользящее окно», «Линейная регрессия» (построение модели), «Прогноз (горизонт: 4)». В результате будут получены прогнозные значения объемов продаж, для которых, например, будет построена диаграмма прогноза.

Судить о том, выполнен узел сценария или нет, можно по его иконке – у активного узла она цветная, у неактивного – серая. Отключать однажды исполненный узел может иметь смысл для экономии памяти или для пересчета построенной модели, например, после того, как были изменены исходные данные. Это можно сделать, вызвав всплывающее меню щелчком правой кнопки мыши на нужном узле и выбрав пункт «Активный». В результате выполненный узел деактивируется. При повторном нажатии узел будет снова выполнен с новыми данными или настройками.

Для каждого узла сценария существует возможность изменения настроек. Для этого из всплывающего меню или панели инструментов выбирается команда  «Настроить...», в результате чего появляется окно Мастера обработки, импорта или экспорта, в зависимости от типа узла. В нем можно изменить любые параметры узла. Внесение изменений может быть отменено на любом шаге Мастера. Изменения принимаются на последнем этапе (ввод заголовка окна), после чего текущий и все подчиненные узлы деактивируются. При повторном выполнении сценария все внесенные изменения уже будут учтены.

При разработке сценария можно изменять порядок следования узлов одного родителя, а значит, и последовательность их выполнения. Для этого выделенный узел перемещается вверх-вниз по дереву в пределах подчинения своему родителю с помощью комбинации клавиш <Ctrl-Вверх> и <Ctrl-Вниз>.

На панели инструментов и в контекстном меню окна сценариев дополнительно доступны следующие команды управления деревом проекта:

-  Вставить узел – вставляет новый узел перед текущим узлом и вызывает для него Мастер обработки. Вставить узел перед узлом импорта данных нельзя.
-  Вырезать узел – удаляет текущий узел из дерева. Все его потомки при этом перемещаются на один уровень вверх и начинают подчиняться родителю удаленного узла.
-  Копировать ветвь – копирует ветвь сценария, начиная с текущего узла. Родителем новой ветви станет родительский узел оригинальной ветви.
-  Удалить ветвь – удаляет ветвь сценария, начиная с текущего узла.
- Сохранить ветвь – сохраняет ветвь сценария, начиная с текущего узла в файл ветви или файл проекта для последующего использования.
- Загрузить ветвь – загружает из файла проекта или файла ветви сценарий обработки. Если сценарий не включает узла импорта данных, то его родителем станет текущий узел. В противном случае его родителем будет корневой узел дерева сценариев.

При выполнении операций вставки и удаления узла могут возникнуть проблемы с последующим выполнением сценария. Если узел выполнял важную для дальнейшего выполнения обработку и был удален из дерева, то его потомки могут стать неработоспособными. Аналогично, вставка узла, который изменяет столбцы данных, например, удаляет или переименовывает их, может помешать выполнению узлов-потомков.

После операции загрузки новая ветвь сценария сможет выполняться только в том случае, если параметры столбцов данных совпадают с настройками ее корневого узла.

Перечисленные операции позволяют легко вносить изменения в дерево сценариев, а значит изменять порядок и свойства обработки узлов в ходе выполнения. Однако результат обработки сам по себе может оказаться не столь значимым. Ведь он должен быть представлен аналитику в удобном и доступном виде, поэтому в следующем разделе речь пойдет о важной части платформы – визуализации данных.

Визуализация данных


На любом этапе обработки можно визуализировать данные. Система самостоятельно определяет, каким способом она может это сделать, например, если будет обучена нейронная сеть, то помимо таблиц и диаграмм можно просмотреть граф нейросети. Пользователю необходимо выбрать нужный вариант из списка и настроить несколько параметров.


Возможные способы визуализации данных:


- **OLAP.** Многомерное представление данных. Любые данные, используемые в программе, можно посмотреть в виде кросс-таблицы и кросс-диаграммы. Пользователю доступен весь набор механизмов манипуляции многомерными данными – группировка, фильтрация, произвольное размещение измерений, детализация, выбор любого способа агрегации, отображение в абсолютных числах и в процентах;
- **Таблица.** Стандартное табличное представление с возможностью фильтрации данных и быстрого расчета статистики (он-лайн статистика);
- **Диаграмма.** График изменения любого показателя;
- **Гистограмма.** График разброса показателей. Гистограмма предназначена для визуальной оценки распределения данных. Распределение данных оказывает значительное влияние на процесс построения модели. Кроме того, по гистограмме можно судить о величине отклонений различной степени (гистограмма распределения ошибок);
- **Статистика.** Статистические показатели выборки;
- **Диаграмма рассеяния.** График отклонения значений, прогнозируемых при помощи модели, от реальных. Может быть построена только для непрерывных величин и только после использования механизмов построения модели, например, нейросети, линейной регрессии или пользовательской модели. Используется для визуальной оценки качества построенной модели;
- **Таблица сопряженности.** Предназначена для оценки результатов классификации вне зависимости от используемой модели. Таблица сопряженности отображает результаты сравнения категориальных значений исходного выходного столбца и категориальных значений рассчитанного выходного столбца. Используется для оценки качества классификации;
- **"Что-если".** Таблица и диаграмма. Позволяют «прогонять» через построенную модель любые интересующие пользователя данные и оценить влияние того или иного фактора на результат. Активно используется для решения задач оптимизации;
- **Обучающая выборка.** Выборка, используемая для построения модели. Выделяются цветом данные, попавшие в обучающее, тестовое и валидационное множество с возможностью фильтрации. Необходима для понимания того, какие записи и каким образом использовались при построении модели;
- **Диаграмма прогноза.** Применяется после использования метода обработки – Прогнозирование. Прогнозные значения выделяются цветом;
- **Граф нейросети.** Визуальное отображение обученной нейросети. Отображается структура нейронной сети и значения весов;
- **Дерево решений.** Отображение дерева решений, полученного при помощи соответствующего алгоритма. Имеется возможность посмотреть детальную информацию по любому узлу и фильтровать попавшие в него данные;
- **Дерево правил.** Отображение в иерархическом виде (в виде дерева) ассоциативных правил. Содержит всегда два уровня. На первом – условие, на втором – следствие правила (или наоборот);
- **Правила.** Отображает в текстовом виде правила, полученные при помощи алгоритма построения деревьев решений или поиска ассоциаций. Такого рода информация легко интерпретируется человеком;
- **Карта Кохонена.** Отображение карт, построенных при помощи соответствующего алгоритма. Широкие возможности настройки – выбор количества кластеров, фильтрация по узлу/кластеру, выбор отображаемых полей. Мощный и гибкий механизм отображения кластеризованных данных;
- **Популярные наборы.** Отображение наиболее часто встречающихся в ассоциативных правилах множеств в виде списка.
- **Описание.** Текстовое описание параметров импорта/обработки/экспорта в дереве сценариев обработки.

Настроенные визуализаторы могут быть вынесены на панель Отчеты. Таким образом, конечный пользователь сможет просто получить и просмотреть необходимый результат, не задумываясь, каким способом он был получен.


Работа с отчетами

Задача тиражирования знаний заключается в предоставлении возможности сотрудникам, не разбирающимся в методиках анализа и способа получения того или иного результата, получать ответ на аналитические запросы на основе моделей, подготовленных экспертом. Для эксперта предназначена панель сценариев, в которой он строит различные модели. Для конечного же пользователя предназначена панель отчетов. Открыть или скрыть эту панель можно, выбрав пункт «Отчеты» меню «Вид» или нажав кнопку  на панели инструментов.

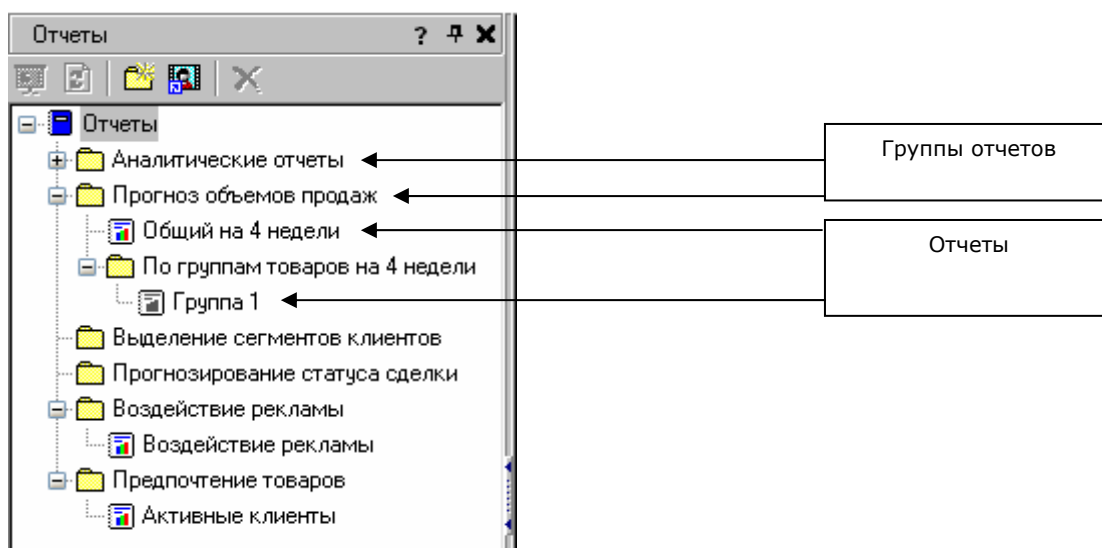
Отчеты представлены в виде древовидного иерархического списка, каждым узлом которого является отдельный отчет или папка, содержащая несколько отчетов. Чтобы добавить новый отчет нужно щелкнуть по кнопке  «Добавить узел» или выбрать соответствующую команду из контекстного меню. В результате откроется окно «Выбор узла», в котором следует выделить узел дерева сценария, где содержится нужная выборка данных и щелкнуть по кнопке «Выбрать».

Чтобы добавить новую папку нужно щелкнуть по кнопке  «Добавить папку» или выбрать соответствующую команду в контекстном меню. В результате в списке отчетов появится новая папка с открытым полем имени, куда следует ввести имя папки. После ввода имени для его сохранения щелкнуть по любому узлу списка. Чтобы поместить отчет в папку, нужно перед вызовом команды «Добавить узел», выделить эту папку.

Для удобства пользователя предусмотрена возможность изменения порядка расположения отчетов и папок в дереве. Для этого выделенный узел перемещается вверх-вниз по дереву в пределах подчинения своему родителю с помощью комбинации клавиш <Ctrl-Вверх> и <Ctrl-Вниз>. Для того чтобы определить, какому узлу сценария соответствует выделенный отчет, следует выбрать пункт всплывающего меню «Найти узел в сценарии».

Дерево отчетов может содержать один и тот же узел дерева сценариев несколько раз. Такая необходимость возникает для отображения одного и того же набора данных разными способами. Например, в одном случае набор будет отображен в виде диаграммы, в другом – в виде кросс-таблицы, а в третьем – в виде кросс-таблицы, но с другими измерениями. Для настройки отображения данных в дереве отчетов предназначена кнопка , после нажатия которой открывается Мастер визуализации, как в дереве сценариев. Надо отметить, что изменение отображения узла в панели отчетов не приводит к изменению отображения узла в панели сценариев. Это позволяет аналитику настраивать отображение в сценарии, так как ему удобно, а конечному пользователю вывести на панель отчетов то отображение, какое ему необходимо.

Отчеты желательно группировать в папки по их смысловому содержанию. Например, папка «аналитические отчеты» может содержать различные кубы данных, папка «прогнозы» может содержать диаграммы прогнозов каких-либо величин. Тогда конечный пользователь открывает панель отчетов, выбирает нужную папку и в этой папке выбирает нужный отчет. После такого выбора программа автоматически выполняет сценарий, соответствующий этому отчету и выдает результат в зависимости от настроенного отображения отчета.



Для формирования отчета достаточно выбрать его в дереве отчетов двойным щелчком мыши. При этом автоматически выполнится сценарий, соответствующий этому отчету. Например, если выбрать отчет «Общий на 4 недели», выполнится сценарий, из предыдущего примера. Результаты будут отображены в соответствии с настройками для панели отчетов, например, в виде диаграммы прогноза. Отчеты, для которых открыт хотя бы один визуализатор, показаны в дереве цветными иконками, в то время как остальные – серыми.

Созданный сценарий и дерево отчетов можно сохранить в файле конфигурации проекта. Для этого необходимо выбрать пункт «Сохранить» меню «Файл». Откроется диалоговое окно сохранения файла. В нем нужно выбрать путь, куда будет сохранен файл проекта, и указать имя файла.

На практике часто встречаются ситуации, когда пользователю требуется получить отчет по некоторому подмножеству всех доступных данных, например, только по одному поставщику или клиенту, по нескольким группам товаров или регионам. В терминах многомерной модели данных такое подмножество называется разрезом. Аналитик может создавать отчеты в предопределенных, наиболее востребованных разрезах, но не в силах предсказать все виды отчетов, которые могут потребоваться пользователю. Поэтому он может дать возможность пользователю самому настраивать срез данных для отчета. Пользовательский срез данных может указываться при импорте данных из хранилища. В этом случае при выполнении отчета будет открыто окно выбора среза (окно *пользовательского фильтра*), в котором пользователю будет предложено выбрать интересующий его разрез данных. Он может указать любой разрез по предусмотренным аналитиком измерениям и получить в результате из хранилища только нужную ему информацию.

Пакетная обработка

Предположим, что в построенном дереве сценариев есть узлы, содержащие экспорт данных во внешний файл. Для такого экспорта можно открыть файл проекта в Deductor Studio, открыть дерево сценариев, найти и открыть все узлы экспорта. Тогда данные, представленные таблицами, будут экспортированы во внешние файлы. Это способ экспорта «вручную».

Но можно воспользоваться пакетной обработкой файла проекта. Для этого необходимо выполнить команду вида:

<путь к программе> <путь к файлу проекта> /параметр_запуска /параметры_лога

Например,

"C:/Program files/BaseGroup/DStudio.exe" C:/project.ded /run

Эту команду можно сохранить и выполнить несколькими равнозначными способами. Первый – это создать ярлык. Для этого в проводнике нужно нажать правой кнопкой мыши, выбрать пункт «Создать» и подпункт «ярлык». Затем, нажав кнопку «Обзор» выбрать путь к Deductor Studio, и в конце строки дописать <путь к файлу проекта> /run. Второй способ – это создать пакетный файл – файл с расширением bat. В этом файле следует написать всего одну строчку: <путь к программе> <путь к файлу проекта> /параметр_запуска.

Параметр_запуска может принимать два значения:

- run. Этот параметр используется для автоматического экспорта данных из программы.
- teach. Этот параметр используется для переобучения моделей, используемых в сценарии. Обычно однажды построенная модель используется в дальнейшем без изменений. Но иногда она перестает давать хорошие результаты на новых данных. Тогда и следует воспользоваться пакетной обработкой с этим параметром.

Допустим, мы создали первым способом ярлык и назвали его Export. После его запуска файл проекта открывается в Deductor Studio, автоматически обнаруживаются все узлы с экспортом данных, и выполняется вся ветвь сценария, содержащая эти узлы. После обработки всех узлов экспорта Deductor Studio закрывается.

В Deductor есть возможность управлять пакетной обработкой при создании сценария. Для этого у каждого узла предусмотрены два свойства: «Выполнить» и «Переобучить» – доступные через всплывающее меню. Если флажок «Выполнить» сброшен, то при запуске в пакетном режиме с параметром Run узел и все его потомки исключаются из процесса выполнения. При установленном флаге узел выполняется в нормальном режиме. Сброшенный флаг «Переобучить» указывает системе, что при запуске с параметром Teach переобучать узел не требуется. При установленном

флаге узел будет переобучен. По умолчанию для вновь создаваемых узлов оба флага установлены, т.е. в пакетном режиме узел может и выполняться, и переобучаться.

Параметры_лога предназначены для управления лог-файлом пакетной обработки. Они могут принимать следующие значения:

- /log - включить подробный режим лог-файла; в логе будет сохраняться информация о времени начала и окончания обработки каждого узла;
- /logfile=<Имя файла> - указывает имя лог-файла;
- /logmode=overwrite - изменяет режим работы с логом на перезапись; по умолчанию используется режим добавления записей в лог.

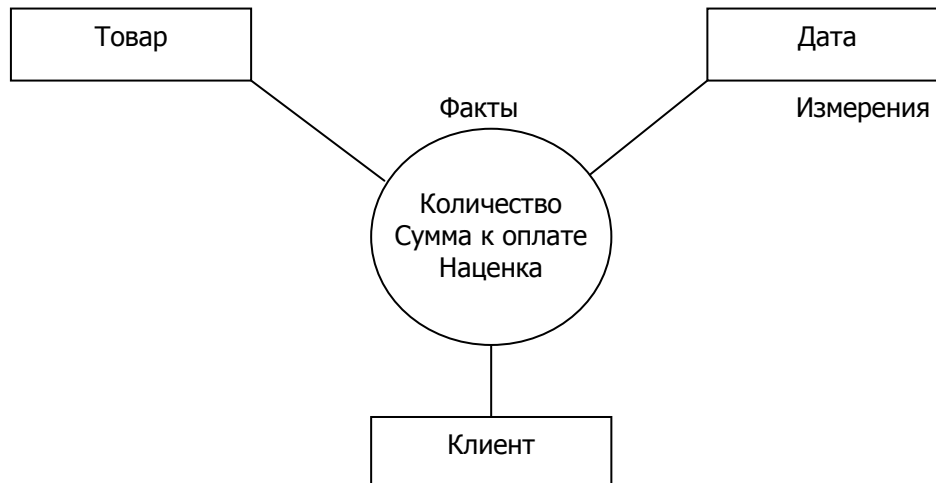
Если в командной строке не указано параметров управления лог-файлом, то используются настройки, сделанные в окне «Настройка».

Архитектура Deductor Warehouse – многомерное хранилище данных

Многомерное представление данных

Deductor Warehouse - многомерное хранилище данных, аккумулирующее всю необходимую для анализа предметной области информацию.

Вся информация в хранилище хранится в структурах типа «звезда», где в центре расположены таблицы фактов, а «лучами» являются измерения.

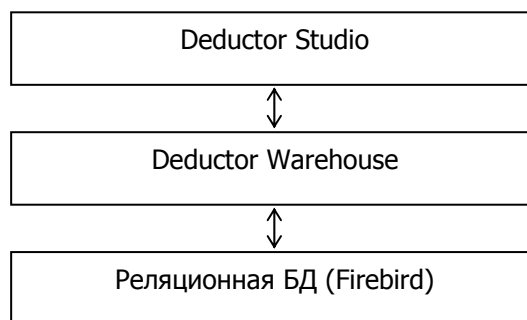


Такая архитектура хранилища наиболее адекватна задачам анализа данных. Каждая «звезда» называется **процессом** и описывает определенное действие, например, продажи товара, отгрузки, поступления денежных средств и прочее. В Deductor Warehouse может одновременно храниться множество процессов, имеющие общие измерения, например, «Товар», фигурирующий в «Поступлении» и в «Отгрузке».



Измерения могут быть как простыми списками, например, дата, так и содержать дополнительные столбцы, называемые **свойствами**. Например, измерение «Товар» может состоять из следующих полей: «Наименование товара» - собственно измерение (первичный ключ), «Вес», «Объем» и прочее - свойства данного измерения.

Что представляет собой хранилище Deductor Warehouse?



Физически – это реляционная база данных, содержащая таблицы для хранения информации и таблицы связей, обеспечивающие целостное хранение сведений. Поверх реляционной базы данных реализован специальный слой, который преобразует реляционное представление к многомерному. Многомерное представление используется потому, что оно намного лучше реляционного соответствует идеологии анализа данных. Благодаря этому слою, пользователь оперирует многомерными понятиями, такими как измерение, факт, а система автоматически производит все необходимые манипуляции, необходимые для работы с реляционной СУБД.

Deductor Warehouse прозрачно для пользователя проводит все необходимые операции по созданию и подключению к реляционной СУБД Firebird. Пользователю остается лишь создать или подключить хранилище данных к Deductor Studio.

Deductor Warehouse реализует универсальное многомерное хранение, т.е. может содержать множество процессов с различным количеством измерений и фактов. Настройка процессов, задание измерений, свойств и фактов может осуществляться с помощью Редактора хранилища Deductor Studio, либо при загрузке в хранилище данных. Вся работа с хранилищем осуществляется средствами Deductor Studio.

Далее кратко описываются процессы создания и подключения локального хранилища данных, создания структуры хранилища, загрузки и импорта данных. Очень подробно концепция источников данных в Deductor, процессы создания и подключения хранилища, импорта и экспорта данных в хранилище описываются в документе «*Источники данных*». Рекомендуется по всем вопросам, возникающим при работе с хранилищем данных, сначала обращаться к нему. Дополнительную информацию о подключении удаленного хранилища, оптимизации работы и текущему администрированию можно найти в «*Руководстве администратора Deductor*».

Создание хранилища данных


Создание хранилища данных производится на панели «Источники данных». Открыть или скрыть эту панель можно, выбрав пункт «Источники данных» в меню «Вид».

Для создания хранилища данных необходимо выполнить следующее. Вызвать контекстное меню щелчком правой кнопки мыши в любом месте панели «Источники данных» и выбрать из списка «Хранилище данных» команду «Создать локальное хранилище данных». В результате будет открыто окно настройки параметров для создания хранилища. В этом окне в поле «Файл базы данных» следует ввести имя файла, в котором должно быть создано новое хранилище. В полях «Имя» и «Метка» можно указать уникальный идентификатор и описание хранилища. Все эти действия можно проделать и в дальнейшем, в редакторе параметров источника данных

После подтверждения настроек с помощью щелчка на кнопке «Ок» на панели источников данных в папке «Хранилища данных» появится новый узел с хранилищем. Хранилище готово к использованию.

Для того чтобы изменить настройки существующего хранилища данных, достаточно двойным щелчком на узле нужного хранилища открыть окно настроек. В нем отображается следующая информация:

- «*Имя*» - текстовое имя, под которым источник данных будет появляться в Мастерах импорта и экспорта данных. Это имя должно быть уникально в пределах одного типа источников, может содержать только латинские буквы, цифры и знак подчеркивания. При этом оно не должно начинаться с цифры, и регистр букв роли не играет.
- «*Описание*» - пользовательское текстовое описание источника данных, содержащее любую дополнительную информацию.

- «*Описание поставщика*» - текстовая строка с именем источника данных. Это поле не может быть изменено.
- «*Версия*» - версия подключаемого хранилища, не может быть изменено, для внутреннего использования.
- «*База данных*» - здесь можно указать имя файла базы данных Interbase\Firebird (gdb-файла), который нужно подключить, и полный путь к нему. При создании нового хранилища это поле будет заполнено после выбора пути, где его следует разместить, и указания имени файла.
- «*Кодовая страница*» - позволяет выбрать кодовую страницу, которая будет использоваться для отображения символов при работе с содержимым хранилища. Для выбора кодовой страницы следует щелкнуть в этой строке и с помощью кнопки  открыть список доступных кодовых страниц, где и выбрать нужную щелчком мыши.
- «*Логин/пароль*» - указывается имя пользователя, по которому будет производиться доступ к хранилищу данных, по умолчанию sysdba, и пароль для указанного имени пользователя, по умолчанию masterkey.
- «*Спрашивать логин/пароль при подключении*» - если данный флажок установлен, то каждый раз при обращении к данному файлу хранилища будет открываться, окно в котором нужно будет указывать имя пользователя и пароль.
- «*Сохранять пароль*» - установка данного флажка позволяет сохранять пароль.
- «*Показывать системные таблицы*» - данный флажок позволяет разрешать и запрещать отображение системных таблиц.


Новое хранилище данных не содержит в себе пока никакой информации. В нем пока еще нет данных и не определены процессы, измерения, факты. Структура хранилища создается с помощью Редактора хранилища или при первой загрузке в него данных.

Подключение к Deductor Warehouse

Перед началом работы с существующим хранилищем данных нужно зарегистрировать его в Deductor Studio, сообщив местонахождение и параметры подключения к базе данных. Эти действия называются подключением к хранилищу данных. Мы только что проделывали подобные шаги при создании нового хранилища. Подключение существующего хранилища проводится аналогично.

Для начала следует вызвать контекстное меню щелчком правой кнопки мыши в любом месте панели «Источники данных» и выбрать из списка «Хранилище данных» команду «Подключить локальное хранилище данных». В открывшемся окне в поле «Файл базы данных» указать существующий файл хранилища. В этом же окне следует указать метку хранилища данных. По этой метке в дальнейшем легко найти нужное хранилище в списках в программе. После принятия всех изменений на панели источников данных в папке «Хранилища данных» появится новый узел с хранилищем.

Для того чтобы изменить настройки существующего хранилища данных, достаточно двойным щелчком на узле нужного хранилища открыть окно настроек, описанное в предыдущем разделе.

После настройки всех необходимых параметров можно проверить доступ к новому хранилищу данных. Для этого следует воспользоваться кнопкой , в результате чего будет предпринята попытка соединения с базой данных хранилища. Если соединение будет успешным, то появится сообщение «Соединение успешно протестировано» и хранилище будет готово к работе. В противном случае будет выдано сообщение об ошибке, вследствие которой соединение невозможно. В этом случае нужно проверить параметры подключения, при необходимости внести в них изменения и протестировать подключение еще раз.

Теперь подключенное хранилище можно использовать для экспорта и импорта данных.

Создание структуры хранилища с помощью Редактора хранилища

Перед тем, как приступить к загрузке данных во вновь созданное хранилище, необходимо задать его структуру, т.е. определить, какие процессы, измерения, факты и свойства будут в нем содержаться. Для этого предназначен Редактор хранилища. Хотя создание процессов и измерений возможно и во время самого процесса загрузки данных, зачастую бывает удобнее заранее спроектировать схему хранилища данных, и производить загрузку в созданные ранее объекты хранилища.

Редактор хранилища предоставляет удобный наглядный интерфейс для работы со своими объектами. В окне Редактора можно создавать новые процессы и измерения, добавлять к процессам факты, а к измерениям свойства, удалять процессы, факты, свойства и неиспользуемые измерения, производить очистку процессов и измерений, т.е. выполнять основные операции с метаданными хранилища.

Редактор хранилища может быть вызван с помощью всплывающего меню или из окна настройки параметров хранилища на панели «Источники данных». В левой части окна Редактора хранилища показано дерево объектов хранилища (процессы, измерения, свойства и факты).

В правой части окна отображаются параметры выделенного объекта:

- Тип объекта - процесс, измерение, факт или свойство;
- ID измерения - внутренний индекс объекта, для служебного использования;
- Имя - уникальное имя объекта, указываемое при создании, для служебного использования;
- Метка - уникальное имя объекта, видимое пользователям Deductor; ее можно изменить для любого объекта, введя новую метку в этом поле.

Для процессов доступно поле с описанием процесса, содержащее произвольные комментарии (оно может быть изменено путем ввода нового описания в это поле) и информация о наличии вспомогательной таблицы. Для всех остальных узлов доступно поле «Тип данных», в котором отображается тип данных объекта, указанный при создании.

С помощью кнопок на панели инструментов Редактора можно выполнить следующие действия:

- Добавить процесс – вызвать окно создания нового процесса;
- Добавить измерение – вызывать окно создания нового измерения;
- Добавить свойство – добавить в измерение новое свойство указанного типа;
- Добавить факт – добавить в процесс новый факт указанного типа;
- Удалить – удаляет выделенный объект из структуры хранилища.

Кроме того, для процесса во всплывающем меню доступна команда «Создать вспомогательную таблицу», с помощью которой для выделенного процесса создается дополнительная таблица, значительно ускоряющая выборку данных из процесса. Создавать вспомогательную таблицу рекомендуется всегда. Но надо отметить, что если в сценарии данные в процесс загружаются по частям в нескольких узлах, то создавать вспомогательную таблицу следует только в последнем узле загрузки. Ранее созданная для процесса вспомогательная таблица удаляется перед новой загрузкой данных в этот процесс, и поэтому ее создание в промежуточных узлах загрузки станет простой тратой времени.

Из любого объекта можно удалить все данные с помощью команды «Очистить» из всплывающего меню. Эта операция бывает полезна при внесении изменений в структуру хранилища или ошибочной загрузке в хранилище неверных данных.


Загрузка данных в хранилище

Для загрузки данных в хранилище нужно воспользоваться Deductor Studio. Для этого необходимо выполнить следующие действия:

1. С помощью Мастера импорта загрузить выборку данных в Deductor Studio.
2. Загруженные данные могут подвергаться любой обработке или преобразованию средствами Deductor Studio. Этот шаг может быть пропущен, если данные не требуют обработки.
3. Запустить Мастер экспорта и в списке форматов выбрать из ветки «Deductor Warehouse» процесс или измерение. В первом случае будет загружена вся выборка в виде процесса («звезды»), а во втором - значения измерения, выбранного пользователем.

Процессы

Для экспорта данных в процесс хранилища данных требуется выполнить следующие действия. В списке доступных хранилищ данных выбрать то, в которое нужно выполнить загрузку. Затем

происходит выбор процесса, в который загружаются данные. Если загружаемая выборка должна быть добавлена к уже существующему процессу, то достаточно выделить этот процесс из списка. Если же выборка загружается в хранилище как новый процесс, нужно активизировать пункт списка «Создать новый процесс...» и открыть окно добавления процесса с помощью кнопки  в правой части строки. В появившемся окне в поле «Название процесса» следует ввести уникальное имя процесса, по которому он будет доступен в Мастерах импорта и экспорта. При желании в поле «Описание процесса» можно ввести краткое описание процесса в произвольной форме. Далее необходимо задать измерения и факты процесса, которые должны быть созданы в хранилище. После нажатия кнопки «Ок» созданный процесс добавится к списку процессов хранилища.

Следующим шагом после выбора процесса становится выбор измерений и фактов, загружаемых в хранилище. В списке «Поле в хранилище» представлены все поля хранилища данных. В столбце «Назначение поля» указан тип поля:

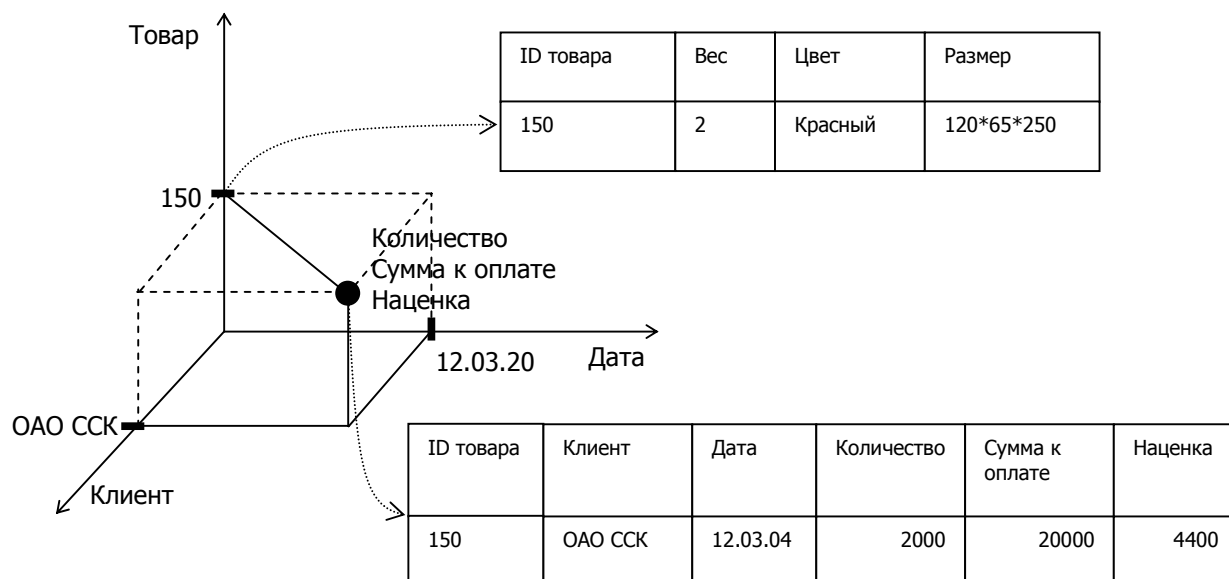
- *факт* - значения будут загружены в это поле как значения фактов (в центр «звезды»);
- *измерение* - значения будут загружены в это поле как значения измерений (в лучи «звезды»).


Для поля хранилища из списка «Поле в источнике данных» выбирается столбец исходной выборки, который должен быть загружен в это поле. Для загрузки процесса необходимо указать соответствия с входными столбцами для всех полей процесса. Установка флажка «Создать вспомогательную таблицу» позволяет создать для загружаемого процесса в хранилище вспомогательную таблицу, которая позволит ускорить доступ к данным в хранилище.

Флаг «Удалить из хранилища, используя измерение» дает возможность выбрать измерение, по которому будет производиться очистка процесса. Его установка позволяет значительно ускорить загрузку данных, но может привести к потерям информации из хранилища. Обычно для каждой загружаемой записи в процессе ищется запись с теми же значениями измерений. Если такая запись есть, то факты в ней обновляются, в противном случае в процесс добавляется новая запись. При установке флага «Удалить из хранилища, используя измерение» для входной выборки данных строится список уникальных значений по указанному измерению, затем из процесса удаляются все строки, значение этого измерения которых содержится в списке. Затем производится быстрая загрузка данных без выполнения каких-либо проверок. Таким образом, если загружаемая выборка по одному значению указанного измерения содержит лишь часть данных, уже имеющихся в процессе, то неизбежны потери информации из хранилища. Таким образом, если данные в процесс загружаются частями, то очистку процесса нужно проводить при загрузке *первой* части данных.

Измерения

Загружать измерение имеет смысл, если у него есть свойства. Например, у измерения «Товар» могут быть свойства «Вес», «Цвет», «Размер». При загрузке процесса свойства измерения *не* загружаются; для того, чтобы добавить их в хранилище, измерение следует загрузить отдельно. В качестве другого примера, измерение «Дата» никаких свойств не имеет, поэтому может загружаться как самостоятельно, так и вместе с процессом.



Для экспорта данных в измерение хранилища данных требуется выполнить следующие действия. В списке доступных хранилищ данных выбрать из списка то, в которое нужно выполнить загрузку. Здесь существует возможность создать новое измерение, активизировав строку «Создать новое измерение» и нажав кнопку  в правой части строки. В появившемся окне следует задать идентификатор измерения и создать его свойства. После подтверждения изменений нажатием кнопки «Ок» новое измерение добавится в список измерений хранилища.

Далее необходимо указать загружаемое измерение и его свойства. В списке «Поле в хранилище» представлены все поля измерения. Для каждого из них в столбце «Назначение поля» указан тип:

- *измерение* – в поле загружаются значения измерения;
- *свойство* – данные в поле загружаются как значения свойства измерения.

Назначение «измерение» может быть указано только для одного поля. Все его значения должны быть уникальными и однозначно идентифицировать измерение.

Для полей измерения из списка «Поле в источнике данных» следует указать столбец входной выборки данных, значения из которого будут загружены в это поле. Для загрузки измерения достаточно указать соответствие хотя бы для поля измерения. Поля свойств могут отсутствовать.

Пример

Пусть во внешнем источнике хранится информация об отгрузках, представленная следующей таблицей.

Таблица. Отгрузки товара.

Код товара	Дата	Клиент	Количество	Сумма к оплате	Наценка
1	01.03.04	ОАО «ССК»	1000	7500	1000
2	10.03.04	ООО «ДСК»	1500	15000	3000
3	12.03.04	ООО «ДСК»	900	9000	2000
3	12.03.04	ОАО «ССК»	2000	20000	4400
4	15.03.04	ООО «ДСК»	500	6000	1000

В другой таблице хранится информация о товаре.













Таблица. Товар.

Код товара	Группа	Цвет	Ширина	Высота	Длина
1	Силикатный	Белый	120	85	250
2	Силикатный	Тонированный	60	88	190
3	Керамический	Красный	120	65	250
4	Керамический	Песочный	120	65	250

Для начала необходимо импортировать эти данные в программу. Для этого вызвать два раза мастер импорта и загрузить данные сначала об отгрузке, затем о товаре (порядок значения не имеет).







Пока наше хранилище пусто и не содержит никаких процессов и измерений. Для создания его структуры в этом примере не будем использовать Редактор хранилища, а выполним все нужные действия прямо в процессе экспорта данных.

Загрузим данные в хранилище. Сначала нужно определиться, что является процессом, фактами и измерениями. Процесс – это отгрузка товаров. Он представлен первой таблицей. Измерения – это товар, дата и клиент. Факты – это количество, сумма к оплате и наценка. Измерение «Товар» содержит свойства, поэтому его нужно загружать в хранилище отдельно. С него и начнем. Выберем узел с таблицей о товаре и вызовем мастер экспорта. В этом мастере из ветви «Deductor Warehouse» нужно выбрать пункт «Измерение». Создадим новое измерение в хранилище, вызвав окно создания измерения и указав назначения и типы полей, как показано на рисунке.













Назначение	Тип поля	Название
 Измерение	12 Целый	 Код товара
 Свойство	ab Строковый	 Группа
 Свойство	ab Строковый	 Цвет
 Свойство	9.0 Вещественный	 Ширина
 Свойство	9.0 Вещественный	 Высота
 Свойство	9.0 Вещественный	 Длина

Номер товара является измерением и однозначно определяет товар. Группа, цвет, ширина, высота и длина – свойства товара. Они могут быть различными у разных товаров. Кроме того, их впоследствии можно будет изменить.

Загрузим наши исходные данные во вновь созданное измерение, указав соответствия полей, как показано на следующем рисунке.







Назначение поля	Поле в хранилище	Поле в источнике данных
 Измерение	Код товара	Код товара
 Свойство	Группа	Группа
 Свойство	Цвет	Цвет
 Свойство	Ширина	Ширина
 Свойство	Высота	Высота
 Свойство	Длина	Длина

В данном примере больше нет измерений со свойствами. Поэтому можно приступить к загрузке процесса. Выберем узел с таблицей отгрузок и вызовем мастер экспорта. В мастере нужно выбрать пункт «Deductor Warehouse – процесс». Создадим новый процесс, выбрав из списка строку «Создать новый процесс...» и вызвав окно создания процесса. Для нашего нового процесса необходимо задать имя. Назовем его «Отгрузки». Далее следует указать, какие измерения и факты есть в процессе, как показано на рисунке.

Назначение	Тип поля	Название
 Измерение	9.0 Вещественный	 Код товара
 Измерение	7 Дата/Время	 Дата
 Измерение	ab Строковый	 Клиент
 Факт	12 Целый	 Количество
 Факт	9.0 Вещественный	 Сумма к оплате
 Факт	9.0 Вещественный	 Наценка

Измерение товар уже есть в хранилище, поэтому его иконка отличается от остальных измерений, а тип поля задан ранее при загрузке измерения и здесь не может быть изменен. В Deductor Warehouse используются только три внутренних типа данных - строковый, вещественный и дата/время - поэтому при загрузке данные целого и логического типов преобразуются к вещественному.

Теперь загрузим данные из таблицы в новый процесс, установив назначения полей в соответствии с рисунком.

Назначение поля	Поле в хранилище	Поле в источнике данных
 Измерение	Код товара	Код товара
 Измерение	Дата	Дата
 Измерение	Клиент	Клиент
 Факт	Количество	Количество
 Факт	Сумма к оплате	Сумма к оплате
 Факт	Наценка	Наценка

☒ Удалить из хранилища, используя измерение Дата

☒ Создать вспомогательную таблицу

Таким образом, будет создана структура хранилища и загружены данные об отгрузках товара.

При загрузке следует указать создание вспомогательной таблицы, так как ее использование значительно повышает быстродействие при импорте данных из хранилища.

Кроме того, укажем удаление данных из хранилища по измерению «Дата». В данном примере при повторной загрузке в процесс из него будут удалены и загружены заново данные на те даты, которые совпадают в источнике и в хранилище. Например, если в хранилище есть данные о том, что на 01.03.04 данному клиенту продано данного товара в количестве 1000, а теперь загружается количество 1200, то реально будет храниться именно 1200. Таким образом, достигается непротиворечивость данных.

Автоматическая загрузка данных в хранилище

Информация в хранилище данных постоянно обновляется, поэтому этот процесс нуждается в автоматизации. Для этих целей в Deductor включены средства автоматического выполнения сценариев.

В сценарии проекта автоматической загрузки данные импортируются из различных источников и экспортируются в измерения и процессы хранилища. Так как загрузка данных в хранилище является экспортом, для такой загрузки можно применять пакетную обработку.

Пример.

Продолжим пример из предыдущего раздела.

Сохраним созданный сценарий выгрузки данных в хранилище под именем «load.ded» в корне диска C:.. Теперь создадим ярлык, в котором укажем команду:

```
«с:\program files\basegroup\deductor\bin\dstudio.exe» c:\load.ded /run
```

При запуске данного ярлыка будет осуществляться автоматическая загрузка данных в хранилище.

Импорт данных из хранилища




Импорт данных из хранилища производится с помощью мастера импорта, в котором необходимо выбрать в качестве типа источника данных процесс или измерение Deductor Warehouse.





Импорт процесса

Для начала нужно выбрать хранилище данных, из которого требуется выполнить импорт. В окне выбора ХД представлены две колонки «Хранилище данных» и «Описание». В колонке «Хранилище данных» указывается имя хранилища, под которым оно зарегистрировано в системе, а в списке «Описание» (это необязательный параметр) – краткая характеристика содержимого хранилища, которая вводилась при его создании.

Затем следует выбрать процесс, из которого нужно импортировать данные. Информация, относящаяся к какому-либо объекту или бизнес-процессу, представлена в хранилище в виде «звезды», где в центре расположены таблицы фактов, а «лучами» являются измерения. Каждая

такая структура называется процессом. В общем случае, таких процессов в хранилище содержится несколько. Поэтому каждый раз при обращении к хранилищу необходимо выбрать процесс, из которого должны импортироваться данные. В списке «Процесс» представлены наименования процессов, содержащихся в данном хранилище, а в списке «Описание» - краткая характеристика процесса (это необязательный параметр). И название, и описание вводятся пользователем при создании процесса.

Далее нужно определить, какие измерения, свойства и факты из выбранного на предыдущем шаге процесса должны быть импортированы. Это необходимо потому, что процесс может содержать много измерений и фактов, а пользователю будут нужны только некоторые из них. Поэтому выбор только тех измерений и фактов, которые нужны в данном случае, позволит сэкономить время на загрузке данных и избежать появления ненужных данных в выборке. Все измерения и факты выбранного процесса представлены в виде дерева, где измерения образуют одну ветвь, а факты - другую. Измерения обозначены значком , а факты - . Слева от каждого измерения и факта расположен флажок. Установка флажка позволяет выбрать соответствующее измерение или факт для импорта, а сброс флажка, наоборот, исключает измерение или факт из числа импортируемых. Кроме этого, с измерением могут быть связаны одно или несколько свойств. В этом случае свойства образуют подветвь ветви измерения, где каждое свойство обозначено значком . Слева от каждого свойства также расположен флажок, который позволяет разрешать или запрещать импорт свойства.

Далее требуется определить срез – то есть выбрать значения измерений, выбранных на предыдущем шаге, которые будут импортированы. Этот шаг желателен потому, что количество значений измерения может быть очень большим, а загрузка всех значений нецелесообразна. Поэтому выбор только тех значений измерения, которые представляют интерес в данном случае, поможет сэкономить время загрузки данных, и не будет загромождать полученную выборку. Чтобы задать параметры среза для данного измерения необходимо выделить его в поле «Измерение», выбрать условие и щелкнуть по кнопке выбора значений . В результате откроется диалоговое окно «Выбор среза». В левой части окна будет представлен список всех значений данного измерения. Чтобы выбрать значение достаточно выделить его щелчком мыши (для перемещения маркера по списку можно также использовать клавиши со стрелками) и щелкнуть по кнопке  или просто нажать «Enter», после чего, выбранное значение появится в правом поле и, таким образом, будет выбрано для построения среза. При необходимости можно отменить выбор значения, для чего выделить его в правом поле и щелкнуть по кнопке . Чтобы очистить весь список выбранных значений следует использовать кнопку . После закрытия окна «Выбор среза», список выбранных значений появится в поле «Выбранные значения измерения». Кнопка «Очистить» позволяет очистить список выбранных значений.

Список условий содержит несколько значений:

- *Нет* – фильтрация по указанному измерению не производится;
- *Входит в список* – указываются значения, которые входят в список импортируемых;
- *Не входит в список* - указываются значения, которые не входят в список импортируемых.

Кроме того, для измерений типа дата/время дополнительно доступны следующие фильтры:

- *В интервале* – выбираются записи, для которых измерение лежит в заданном диапазоне дат;
- *Вне интервала* – выбираются записи, для которых измерение не входит в заданный диапазон дат;
- *За последний период* - выбираются записи, для которых измерение лежит в указанном промежутке времени, предшествующем указанной дате;
- *Кроме последнего периода* - выбираются все записи, кроме тех, для которых измерение лежит в указанном промежутке времени, предшествующем указанной дате.

Если для измерения установлен флаг «Определить при выполнении», то при каждом выполнении узла импорта пользователю будет выводиться окно, аналогичное окну настройки среза, в котором он сможет указать требуемые разрезы по этому измерению. Эта настройка позволяет строить динамические отчеты, в которых пользователю предоставляется только интересующая его информация.

Например, пользователю требуется получать информацию об отгрузках по каждому дилеру. Можно было бы подготовить отдельную ветку сценария и отчеты по каждому дилеру, либо фильтровать

по дилеру в кросс-таблице. Однако первый вариант не подходит, из-за того, что число дилеров может быть очень большим, либо постоянно меняться. Тогда пришлось бы каждый раз перестраивать систему отчетов. Во втором варианте из хранилища данных будут в любом случае выбираться все данные, что скажется на быстродействии системы. Кроме того, таким образом нельзя дополнительно обработать данные или представить в другом виде, нежели куб.

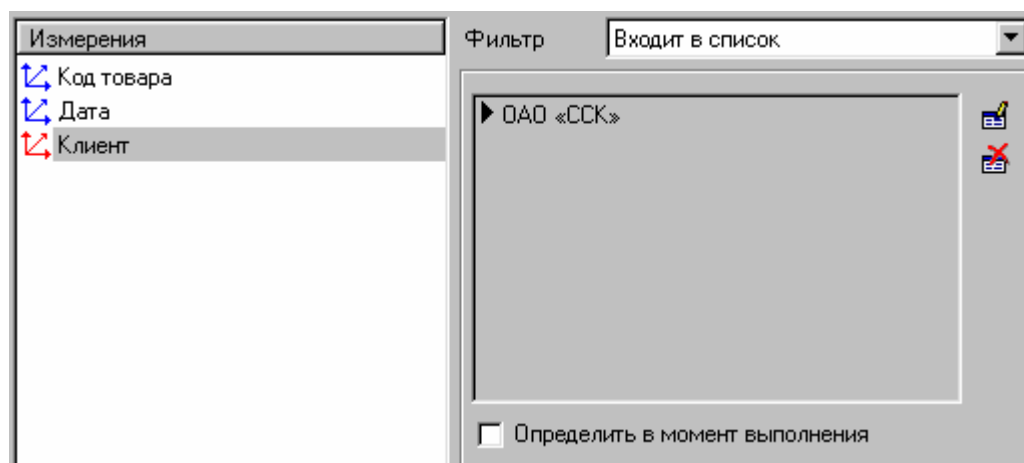
С помощью динамического фильтра можно эффективно решать подобные проблемы. На этапе импорта данных из хранилища пользователь сам сможет указать, в каком именно разрезе ему нужны данные, и работать только с ними. При этом для всех возможных разрезов готовится всего один сценарий обработки и одна система отчетов.

Пример.

Импортируем из хранилища данные о количестве отгруженного товара в разрезе дат и товаров по клиенту ОАО «ССК», оставив свойство товара «Цвет». Вызовем Мастер импорта и выберем источник «Deductor Warehouse». Далее выберем наше хранилище из списка и процесс «Отгрузки». Отметим измерения и факты, импортируемые из хранилища, как показано на рисунке.



Далее укажем фильтр по клиенту.



Таким образом, будет получена таблица следующего содержания.

	Код товара	Группа	Цвет	Дата	Клиент	Количество
▶	1	Силикатный	Белый	01.03.2004	ОАО «ССК»	1000
	3	Керамический	Красный	12.03.2004	ОАО «ССК»	2000

То есть получили количество в разрезе товара, даты и конкретного клиента.

Импорт измерения

Для начала нужно выбрать хранилище данных, из которого требуется выполнить импорт. В окне выбора ХД представлены две колонки «Хранилище данных» и «Описание». В колонке «Хранилище данных» указывается имя хранилища, под которым оно зарегистрировано в системе, а в списке

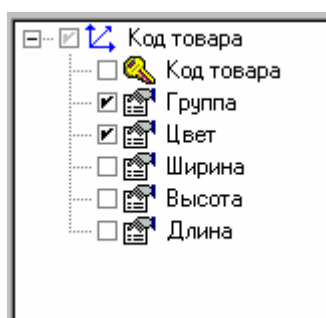
«Описание» (это необязательный параметр) – краткая характеристика содержимого хранилища, вводимая пользователем при его создании.

Затем следует выбрать измерение, из которого нужно импортировать данные. В колонке «Измерение» представлены наименования измерений, содержащихся в хранилище. В колонке «Свойства» находится список свойств каждого измерения или пустая строка в случае их отсутствия.

Теперь нужно выбрать свойства измерения, которые будут включены в выходной набор данных. Для этого нужные свойства помечаются флагом, расположенным слева от имени свойства. Неотмеченные свойства не будут включены в итоговый набор.

Пример

Импортируем из хранилища данные о товарах, которые продает компания. Для этого выберем свойства измерения товаров, как показано на следующем рисунке.



В результате будет получена таблица следующего содержания:

Группа	Цвет
Керамический	Красный
Керамический	Песочный
Силикатный	Белый
Силикатный	Тонированный

То есть получены группа и цвет всех продаваемых товаров.

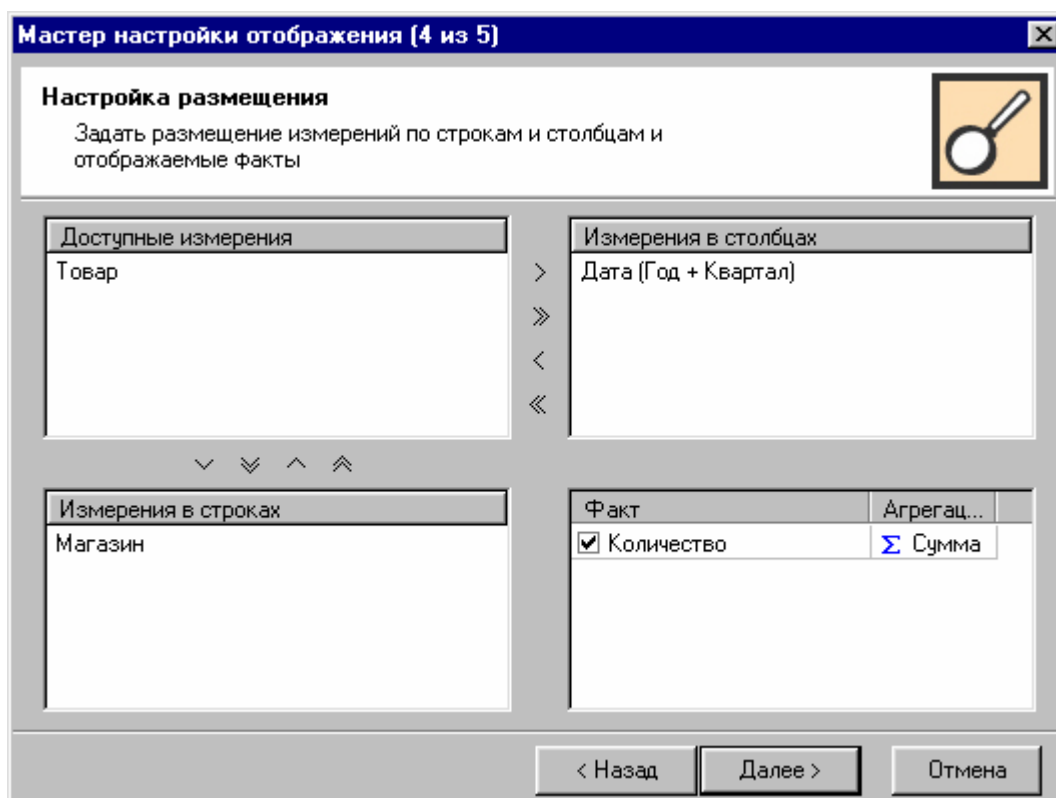
Работа с OLAP-кубом

Кросс-таблица

Кросс-таблица представляет собой размещение многомерных данных на плоскости в виде сводной таблицы. Следовательно, прежде чем строить эту таблицу, необходимо указать измерения и факты. Например, измерения – это Месяц, Наименование товара, Группа товара, а факт – Количество проданного товара.

Размещение измерений

Измерения могут быть размещены в строках и столбцах кросс-таблицы. В мастере настройки отображения изначально весь список выбранных измерений отображается в окне «Доступные измерения». Нажимая кнопки «>» и «>>» справа и снизу от этого окна можно размещать выбранные измерения в строках и столбцах таблицы. Пример приведен на рисунке.



Здесь можно так же выбрать, какие факты отображать в кросс-таблице на пересечении измерений и какую функцию применять при их агрегации (объединении). В данном примере факт «Количество» будет суммироваться.

Построенная кросс-таблица для данного примера будет выглядеть следующим образом.

Измерения в столбцах		Значения фактов				
Скрытые измерения	Товар					
	Дата (Год + Квартал)					
Измерения в строках	Магазин	2004-Q1	2004-Q2	2004-Q3	2004-Q4	Итого
	Магазин 1	234.00	305.00	307.00	229.00	1,075.00
	Магазин 2	276.00	268.00	203.00	171.00	918.00
	Итого	510.00	573.00	510.00	400.00	1,993.00

Измерения в кросс-таблице изображаются специальными полями. Синие поля показывают измерения, участвующие в построении таблицы. Зелеными полями отображаются скрытые

измерения, не участвующие в построении таблицы. Есть возможность перестраивать таблицу с помощью мыши «на лету». Сделать это можно, если перетаскивать поля с заголовками измерений. Приведем различные варианты изменения таблицы таким способом.

1. Сделать измерение, участвующее в построении таблицы скрытым. Для этого нужно перетащить поле с заголовком измерения в строку со скрытыми измерениями. Перетащим, например поле «Дата (Год+Квартал)».

Товар	Дата (Год + Квартал)				
Магазин	2004-Q1	2004-Q2	2004-Q3	2004-Q4	Итого
Магазин 1	234.00	305.00	307.00	229.00	1,075.00
Магазин 2	276.00	268.00	203.00	171.00	918.00
Итого	510.00	573.00	510.00	400.00	1,993.00

Результат:

Дата (Год + Квартал)	Товар
Магазин	Σ Количество
Магазин 1	1,075.00
Магазин 2	918.00
Итого	1,993.00

2. Сделать скрытое измерение участвующим в построении таблицы. При этом его можно добавить к измерениям в строках или столбцах. В исходной таблице перетащим измерение «Товар» и поместим его рядом с измерением «Магазин».

Товар	Дата (Год + Квартал)				
Магазин	2004-Q1	2004-Q2	2004-Q3	2004-Q4	Итого
Магазин 1	234.00	305.00	307.00	229.00	1,075.00
Магазин 2	276.00	268.00	203.00	171.00	918.00
Итого	510.00	573.00	510.00	400.00	1,993.00

Результат:

		Дата (Год + Квартал)				
Товар	Магазин	2004-Q1	2004-Q2	2004-Q3	2004-Q4	Итого
Товар 1	Магазин 1	5.00		8.00		13.00
	Магазин 2	5.00				5.00
	Итого	10.00		8.00		18.00
Товар 10	Магазин 1			3.00	6.00	9.00
	Магазин 2	26.00	11.00	7.00		44.00
	Итого	26.00	11.00	10.00	6.00	53.00
Товар 11	Магазин 1	6.00	3.00	7.00	11.00	27.00
	Магазин 2		2.00	7.00		9.00
	Итого	6.00	5.00	14.00	11.00	36.00
Товар 12	Магазин 1	1.00	8.00	11.00	14.00	34.00

При этом можно было расположить измерение «Товар» как слева, так и справа от измерения «Магазин».

3. Поменять два измерения местами. Например, в предыдущем примере товаров оказалось гораздо больше, чем магазинов. Это привело к многократному повторению значений измерения «Магазин» в таблице. Поменяем местами заголовки этих измерений.

		Дата (Год + Квартал) ▾				
Магазин ▾	Товар ▾	2004-Q1	2004-Q2	2004-Q3	2004-Q4	Итого
Магазин 1	Товар 14	23.00		21.00	4.00	48.00
	Товар 16	18.00	11.00	2.00	10.00	41.00
	Товар 18	9.00		19.00		28.00
	Товар 21	5.00	23.00	1.00		29.00
	Товар 26	9.00	8.00	8.00	7.00	32.00
	Товар 30	17.00	16.00	4.00		37.00
	Товар 41	20.00	10.00	13.00		43.00
	Итого	101.00	68.00	68.00	21.00	258.00
Магазин 2	Товар 14	16.00	13.00	4.00		33.00
	Товар 16	9.00		16.00	8.00	33.00
	Товар 18	30.00		7.00		37.00
	Товар 21		11.00	20.00		31.00

Такое размещение гораздо удобнее. Эта таблица показывает квартальный объем продаж каждого товара в каждом магазине.


Изменять расположение измерений можно, используя операцию транспонирования таблицы. В результате транспонирования данные, ранее отображавшиеся в строках, отображаются в столбцах, а данные в столбцах преобразуются в строки. Транспонирование во многих случаях позволяет оперативно сделать таблицу более удобной для восприятия. Пример транспонирования таблицы представлен на рисунке.

Исходная таблица

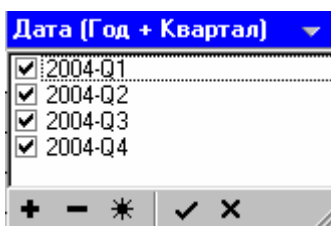
	Товар ▾					
Дата (Год ▾	Товар 1	Товар 10	Товар 2	Товар 3	Товар 4	Товар 5
2004-Q1	11.00	47.00	54.00	44.00	29.00	
2004-Q2	17.00	47.00	58.00	41.00	31.00	
2004-Q3	17.00	29.00	55.00	23.00	28.00	
2004-Q4	10.00	44.00	24.00	17.00	32.00	
Итого	55.00	167.00	191.00	125.00	120.00	1

Такая таблица может не поместиться в пределы экрана и будет неудобна для рассмотрения. В результате транспонирования она примет более удобный вид.

	Дата (Год + Квартал) ▾				
Товар ▾	2004-Q1	2004-Q2	2004-Q3	2004-Q4	Итого
Товар 1	11.00	17.00	17.00	10.00	55.00
Товар 10	47.00	47.00	29.00	44.00	167.00
Товар 2	54.00	58.00	55.00	24.00	191.00
Товар 3	44.00	41.00	23.00	17.00	125.00
Товар 4	29.00	31.00	28.00	32.00	120.00
Товар 5	36.00	22.00	53.00	34.00	145.00
Товар 6	66.00	28.00	43.00	26.00	163.00
Товар 7	51.00	66.00	41.00	41.00	199.00
Товар 8	75.00	24.00	45.00	3.00	147.00
Товар 9	28.00	42.00	73.00	11.00	154.00
Итого	441.00	376.00	407.00	242.00	1,466.00

Чтобы применить операцию транспонирования следует воспользоваться кнопкой «Транспонировать таблицу»  на панели инструментов.

В приведенных выше примерах кросс-таблица строится по всем значениям измерений. Однако иногда возникает необходимость построить таблицу в разрезе лишь некоторых значений, например, за первый и третий кварталы года. Включать или исключать значения измерений в таблицу можно, нажав на треугольник в поле заголовка интересующего измерения. Например, если нажать на треугольник в поле заголовка измерения «Дата (Год+Квартал)» откроется список его значений.



Включать или исключать значения можно, устанавливая или снимая галочку рядом с наименованием. Кнопки внизу списка означают:


- «+» - установить все галочки;
- «-» - убрать все галочки;
- «*» – инверсия, все сброшенные галочки устанавливаются, а все установленные – сбрасываются;
- «V» – применить все изменения для построения таблицы;
- «X» – отменить все сделанные изменения.


Выбирать фильтруемые значения можно и для скрытых измерений.


Способы агрегации и отображения фактов

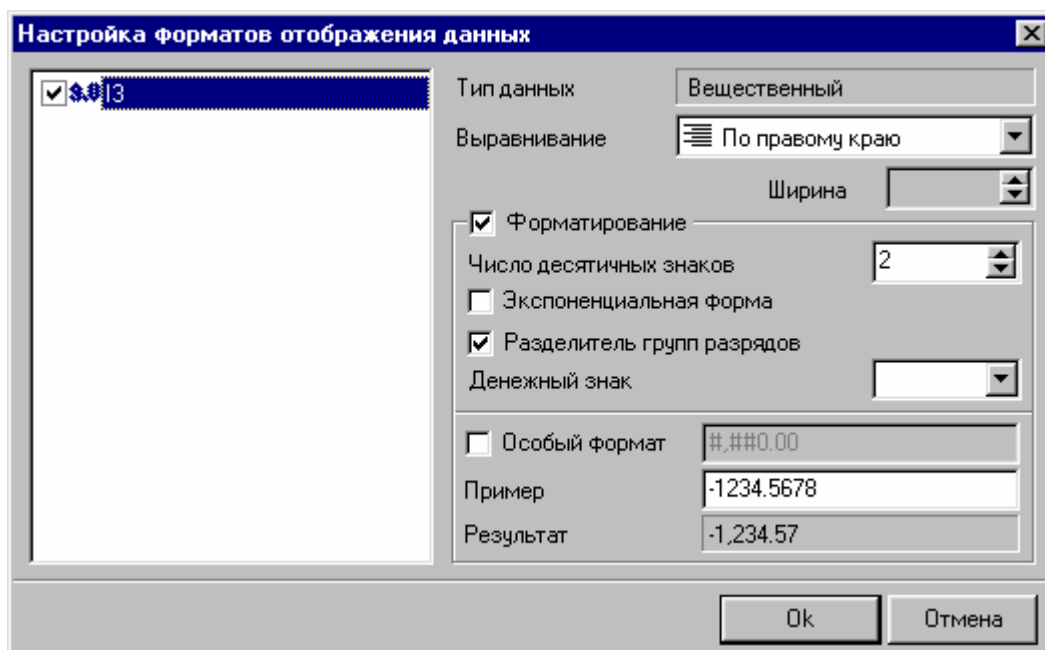
Предусмотрено несколько способов объединения фактов в кросс таблице:

- сумма – вычисляется сумма объединяемых фактов;
- минимум – среди всех объединяемых фактов в таблице отображается только минимальный;
- максимум – среди всех объединяемых фактов в таблице отображается только максимальный;
- среднее – вычисляется среднее значение объединяемых фактов;
- количество – в кросс-таблице будет отображаться количество объединенных фактов.

Для изменения способа агрегации фактов нужно вызвать окно «Настройка размещения», нажав кнопку  «Изменение размещения» на панели инструментов.

Есть возможность отображать факты не только их значениями, но и в процентах по строкам или столбцам кросс-таблицы. Для изменения такого способа отображения нужно нажать кнопку  «Отображать факты как» на панели инструментов.

В кросс-таблице факты отображаются с двумя знаками после запятой и выровнены по правому краю ячейки. Есть возможность изменить форматирование ячеек таблицы, вызвав окно «Настройка форматов отображения данных». Вызвать его можно, нажав кнопку  «Настройка отображения фактов». Пример окна на рисунке.




Назначение полей следующее:

- «Выравнивание» – определяет выравнивание значений в ячейках. Может принимать значения: «По левому краю», «По центру», «По правому краю»;
- «Форматирование» – применить особое форматирование или оставить все как есть;
- «Число десятичных знаков» – определяет число знаков после запятой;
- «Экспоненциальная форма» – если установлен, то число будет отображаться в экспоненциальной форме. Например, число 153.47 будет выглядеть 1.5347E+2;
- «Разделитель групп разрядов» – отображать или не отображать разделитель разрядов. То есть число может выглядеть так 1289 или так 1,289;
- «Денежный знак» – можно в конце значения добавить знак «р.» или «\$»;
- «Особый формат» – позволяет задать формат с помощью строки формата. В поле «Результат» отображается результат применения формата к числу, введенному в поле «Пример».

Селектор – фильтрация данных кросс-таблицы

Селектор является мощным средством фильтрации данных в кросс-таблице. Фильтрация может производиться двумя способами:

- по значениям фактов;
- по значениям измерений, путем непосредственного выбора значений из списка, или отбора их по условию. Фильтрация выполняется отдельно по каждому измерению.

Чтобы приступить к работе с селектором достаточно на панели инструментов нажать на кнопку  «Селектор», после чего будет открыто окно селектора.

Измерения/факты	Выбр...	Всего
Факты		
Дата (Год + Квартал)	4	4
Магазин	2	2
Товар	50	50

Измерение: Товар

Факт: Количество

Агрегация:

- ☒ Сумма
- ☐ Минимум
- ☐ Максимум
- ☐ Среднее
- ☐ Количество

Условие: Первые N

Значение: 10

Ок Отмена

Это пример окна для фильтрации данных по значениям фактов. Слева отображаются все измерения кросс-таблицы и поле «Факты», означающее фильтрацию по фактам. Справа находятся элементы:

- Измерение. Фильтрация подразумевает, что в таблице останется лишь часть значений некоторого измерения. Это поле как раз и задает измерение, значения которого будут отфильтрованы;
- Факт. В кросс-таблице может содержаться один и более фактов. Фильтрация будет происходить по значениям выбранного здесь факта.
- Агрегация – можно выбрать функцию агрегации, в соответствии с которой следует выполнить отбор записей. В результате будут выбраны только те записи, агрегированные значения которых удовлетворяют выбранному условию;
- Условие - условие отбора записей по значениям выбранного факта.

Поле условие может принимать следующие значения:

- Первые N. Значения измерения сортируются в порядке убывания факта и выбираются первые N значений измерений. Таким образом, можно, например, находить лидеров продаж – первые 10 наиболее продаваемых товаров, или первые 5 наиболее удачных дней;
- Последние N. Значения измерения сортируются в порядке убывания факта и выбираются последние N значений измерений. Например, 10 наименее популярных товаров;
- Доля от общего. Значения измерения сортируются в порядке убывания факта. В этой последовательности выбирается столько первых значений измерения, что они в сумме давали заданную долю от общей суммы. Например, можно отобрать клиентов, приносящих 80% прибыли – группа A по ABC классификации;
- Диапазон. Результатом отбора будут записи, для которых значение соответствующего факта лежит в заданном диапазоне;
- Больше. Будут отобраны записи, значение соответствующего факта, для которых будет больше указанного значения;
- Больше или равно. Будут отобраны записи, значение соответствующего факта, для которых будет больше или равно указанного значения;
- Меньше. Будут отобраны записи, значение соответствующего факта, для которых будет меньше указанного значения;
- Меньше или равно. Будут отобраны записи, значение соответствующего факта для которых будет меньше или равно указанного значения;
- Равно. Будут отобраны записи, значение соответствующего факта, для которых будет равно указанному значению;
- Не равно. Будут отобраны записи, значение соответствующего факта, для которых будет не равно указанному значению.

Приведем пример. Пусть нам нужно определить товары, пользующиеся наибольшим спросом. Исходная кросс-таблица содержит 15 товаров.

Товар	Σ Количество
Товар 1	55.00
Товар 10	167.00
Товар 11	110.00
Товар 12	133.00
Товар 13	162.00
Товар 14	145.00
Товар 15	54.00
Товар 2	191.00
Товар 3	125.00
Товар 4	120.00
Товар 5	145.00
Товар 6	163.00
Товар 7	199.00
Товар 8	147.00
Товар 9	154.00
Итого	2,070.00

Применим к ней селектор.

Измерение: **Товар**

Факт: **Количество**

Агрегация:


- ☒ Сумма
- ☐ Минимум
- ☐ Максимум
- ☐ Среднее
- ☐ Количество

Условие: **Первые N**

Значение: **5**

В результате получим 5 наиболее популярных товаров. Такую выборку можно получить по любому факту. В данном примере – это количество. Поэтому мы получили наиболее продаваемые товары. Если отфильтровать по наценке, то получим наиболее прибыльный товар.

Товар	Σ Количество
Товар 10	167.00
Товар 13	162.00
Товар 2	191.00
Товар 6	163.00
Товар 7	199.00
Итого	882.00

При работе с кросс-таблицей доступна операция детализации ячейки. При нажатии <Enter> или кнопки  «Детализация ячейки» для выделенной ячейки в нижней части экрана открывается таблица детализации. В ней отображаются все записи исходного набора данных, которые вносят свой вклад в формирование значения выделенной ячейки. Т.е. при детализации ячейки, выделенной на предыдущем рисунке, в таблице будут показаны все строки набора данных, в которых находится информация о продажах Товара 10.

В таблице детализации будут находиться столбцы, для которых было указано назначение измерение, факт или информационное. Информационные поля не отображаются в самой кросс-таблице, но показываются при детализации. Например, информацию о номере выписанного счета показывать в кросс-таблице не имеет смысла, но если указать для него информационное назначение, то для любой поставки будет легко его узнать, вызвав окно детализации. Для предыдущей таблицы окно детализации будет содержать следующие столбцы: Товар, Дата (Год+Квартал), Магазин и Количество. Если просуммировать все значения по столбцу Количество, то получим суммарные продажи Товара 10 за все время по всем магазинам, т.е. значение из выделенной ячейки – 167.

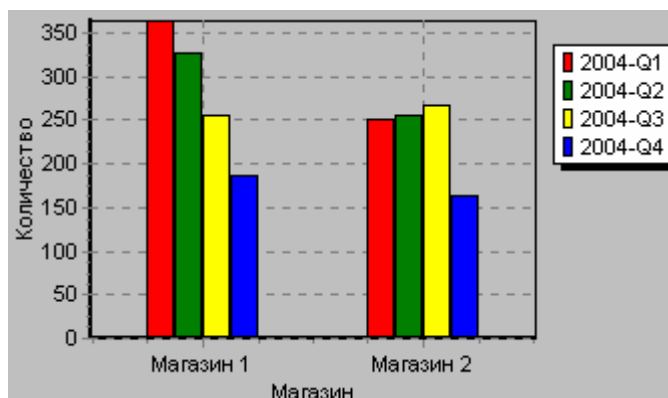
Кросс-диаграмма

Кросс-диаграмма представляет собой диаграмму заданного типа, построенную на основе кросс-таблицы. Основное отличие кросс-диаграммы от обычной диаграммы в том, что она однозначно соответствует текущему состоянию кросс-таблицы и при любых ее изменениях изменяется соответственно.

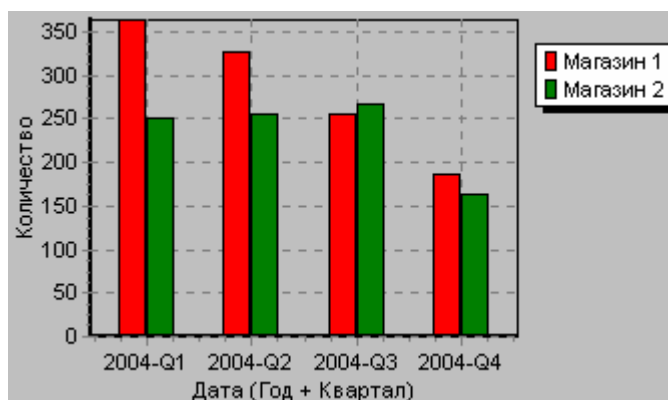
Приведем пример кросс-диаграммы для следующей кросс-таблицы.


	Дата (Год + Квартал) ▼				
Магазин ▼	2004-Q1	2004-Q2	2004-Q3	2004-Q4	Итого
Магазин 1	364.00	326.00	256.00	187.00	1,133.00
Магазин 2	251.00	255.00	267.00	164.00	937.00
Итого	615.00	581.00	523.00	351.00	2,070.00

Кросс-диаграмма для нее имеет следующий вид.





На этой диаграмме можно наблюдать поквартальную тенденцию продаж в различных магазинах. К кросс-диаграмме, так же как и к кросс-таблице, можно применять транспонирование. Результат транспонирования приведенной выше диаграммы будет следующий.




Кросс-диаграмма строится по одному из фактов кросс-таблицы. Выбрать интересующий факт можно, нажав кнопку  на панели инструментов.

При построении диаграммы вводятся ограничения числа серий и числа точек в каждой серии. Данное ограничение вызвано, с одной стороны, большими вычислительными затратами при

построении диаграммы, а с другой - сложностью восприятия больших диаграмм. Кнопка  на панели инструментов будет иметь синий цвет, если ограничения не превышены, и красный  в противном случае. Это предупреждение о том, что на кросс-диаграмме пользователь видит не всю информацию. Щелчок по кнопке выводит окно «Сведения об ограничениях», в котором представлена информация:

- сколько серий и точек, фактически отображается на кросс-диаграмме; под «точками» понимаются группы столбцов, которые соответствуют значениям измерения по строкам. Каждому значению измерения по строкам могут соответствовать несколько значений измерения по столбцам. Для каждого из них строится свой столбец в каждой точке кросс-диаграммы. Количество столбцов в группе называется «серией».
- максимально возможное число серий и точек (по умолчанию 50 и 100 соответственно) для кросс-диаграммы;
- количество серий и точек, минимально необходимое для того, чтобы диаграмма была отображена полностью.

Таким образом, если фактическое количество серий и точек соответствует минимально необходимому для полного отображения кросс-диаграммы, то отображается вся информация.

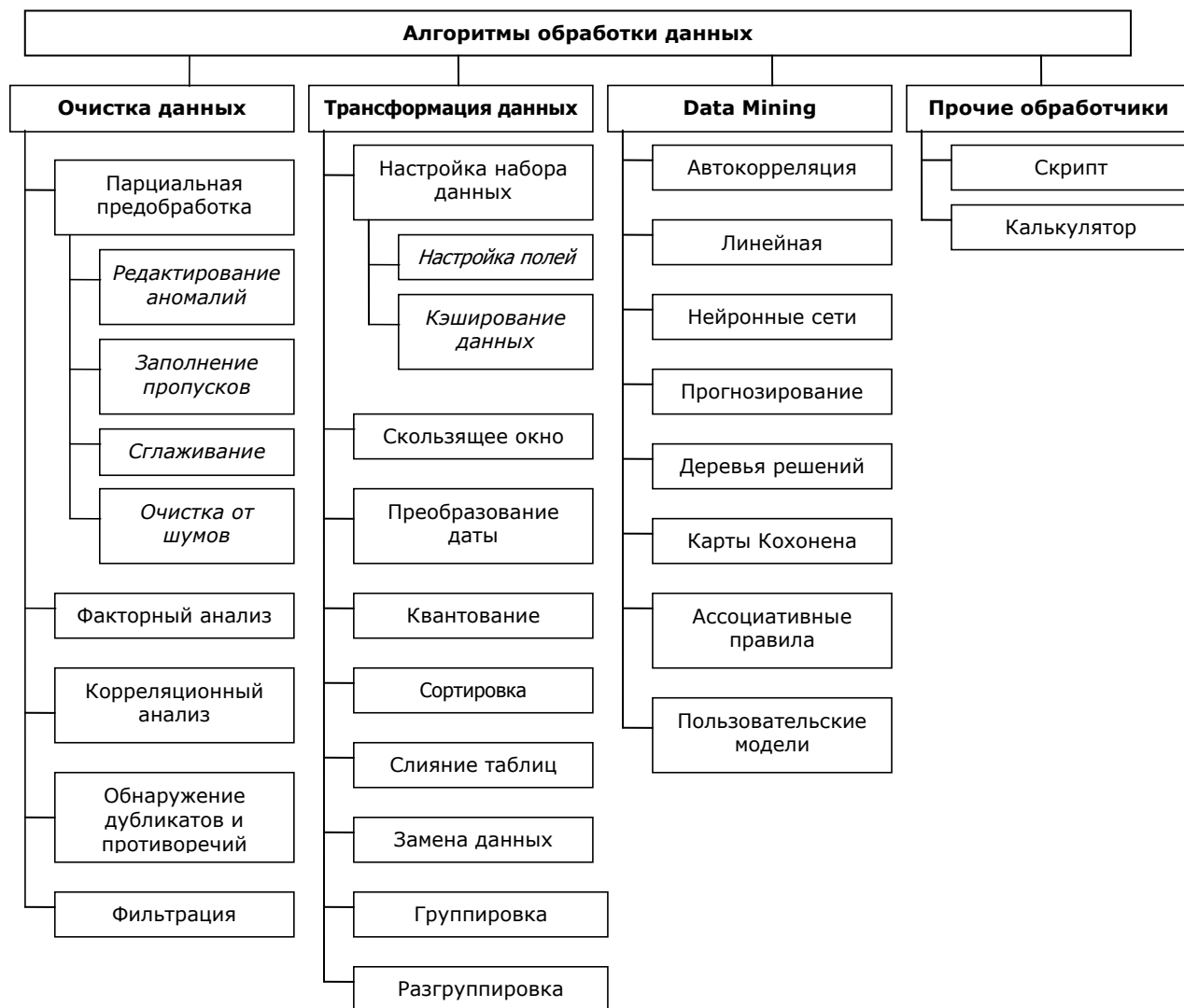
Кросс-диаграмма обладает интересной возможностью – построением тренда. Во многих случаях тренд позволяет увидеть тенденции, которые обычно скрыты из-за большого разброса значений, наличия отклонений, не типичных для отображаемого процесса и т.д. Линия тренда получается путем сглаживания рядов данных, на основе которых построена кросс-диаграмма, посредством выделения и отсекаания больших отклонений, которые в большинстве случаев мешают оценить общий характер процесса. Кнопка  «Тренд» на панели инструментов позволяет включить отображение тренда. Как только режим отображения тренда будет включен, в панели инструментов окна кросс-диаграммы станут доступны настройки «Гладкость линии» и «Степень округления». Гладкость линии определяет степень сглаживания исходного ряда значений, а степень округления определяет «масштаб» отсеиваемых деталей. Чем выше значения этих настроек, тем более гладкой будет линия тренда. Комбинируя эти настройки можно добиться наилучшего результата. Однако следует учесть, что для более быстро изменяющихся процессов эти значения должны быть больше, а для медленных процессов - меньше. Если задать их слишком большими, то исходный процесс будет сглажен и округлен в такой степени, что будет потеряна и полезная информация, а линия тренда вырождается в ступеньку или прямую линию.

Описание аналитических алгоритмов

Кроме консолидации данных работа по созданию законченного аналитического решения содержит несколько этапов.

- Очистка данных. На этом этапе проводится редактирование аномалий, заполнение пропусков, сглаживание, очистка от шумов, обнаружение дубликатов и противоречий.
- Трансформация данных. Производится замена пустых значений, квантование, табличная замена значений, преобразование к скользящему окну, изменение формата набора данных.
- Data Mining. Строятся модели с использованием нейронных сетей, деревьев решений, самоорганизующихся карт, ассоциативных правил и других методов.

На рисунке представлены алгоритмы, которые используются в программе, сгруппированные по назначению.



Очистка данных

Парциальная обработка

Если множество данных, анализ которого требуется выполнить, не соответствуют определенным критериям качества, то их предварительная обработка становится необходимым шагом для обеспечения удовлетворительного результата их анализа. Необходимость в предварительной обработке возникает независимо от того, какие алгоритмы и технологии используются. Более того, эта задача может представлять независимую ценность в областях, не имеющих непосредственное отношение к анализу данных. Очевидно, что исходные («сырые») данные чаще всего нуждаются в очистке. В процессе этого восстанавливаются пропущенные данные, редактируются аномальные значения, вычитается шум. В Deductor Studio при этом используются алгоритмы робастной фильтрации, спектрального и вейвлет-анализа, при этом каждое поле анализируемого набора обрабатывается независимо от остальных полей, то есть данные обрабатываются по частям. По этой причине такая предобработка получила название *парциальной*. В числе процедур предобработки данных, реализованных в Deductor Studio – сглаживание, удаление шумов, редактирование аномальных значений, заполнение пропусков в рядах данных.

Редактирование аномалий

Назначение.

Довольно трудно дать определение понятию аномалия. Аномалии – это отклонения от нормального поведения чего-либо. Это может быть, например, резкое отклонение величины от ее ожидаемого значения.

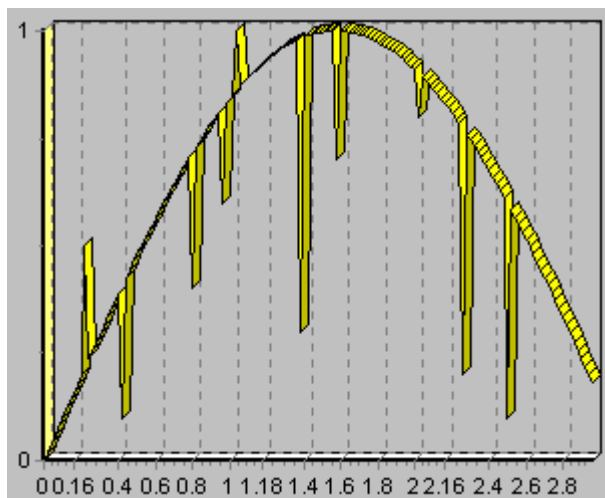
Автоматическое редактирование аномальных значений осуществляется с использованием методов робастной фильтрации, в основе которых лежит использование робастных статистических оценок, таких, например, как медиана. При этом можно задать эмпирически подобранный критерий того, что считать аномалией. Например, задание в качестве степени подавления аномальных данных значения «слабая» означает наиболее терпимое отношение к величине допустимых выбросов.

Настройки.

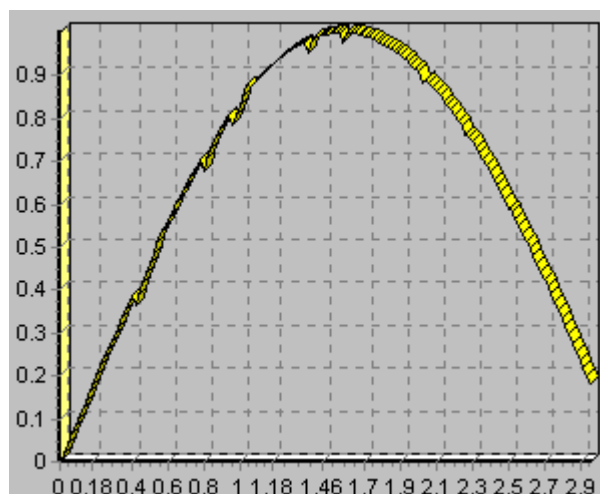
Для применения алгоритма удаления аномалий необходимо указать поле таблицы, к которому его нужно применить (которое содержит аномалии), и указать степень подавления аномальных данных – малую, среднюю или большую.

Пример.

На рисунке пример величины с аномалиями.



После применения алгоритма удаления аномалий та же величина представляется следующим образом.



Здесь была использована большая степень подавления аномалий.

Алгоритм редактирования аномалий выполняется в обработчике «Парциальная обработка».

Заполнение пропусков

Назначение.

Часто бывает так, что в столбце некоторые данные отсутствуют в силу каких-либо причин (данные не известны, либо их забыли внести и т.п.). Обычно, из-за этого пришлось бы убрать из обработки все строки, которые содержат пропущенные данные. Чтобы этого не происходило, в программе предусмотрено два способа заполнения пропущенных данных.

- Аппроксимация - пропущенные данные восстанавливаются методом аппроксимации.
- Максимальное правдоподобие - алгоритм подставляет наиболее вероятные значения вместо пропущенных данных.

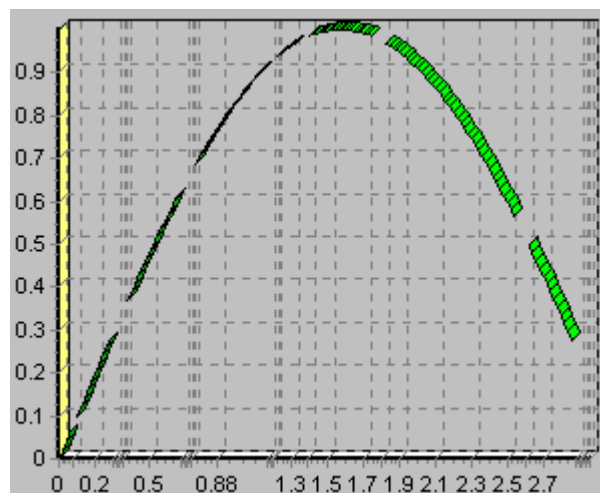
Принцип работы.

Метод аппроксимации используется только для упорядоченных данных, чаще всего это временные ряды. Этот метод использует последовательный рекуррентный фильтр второго порядка (фильтр Калмана). Входные данные последовательно подаются на вход фильтра, и если очередное значение ряда отсутствует, оно заменяется значением, которое экстраполируется фильтром.

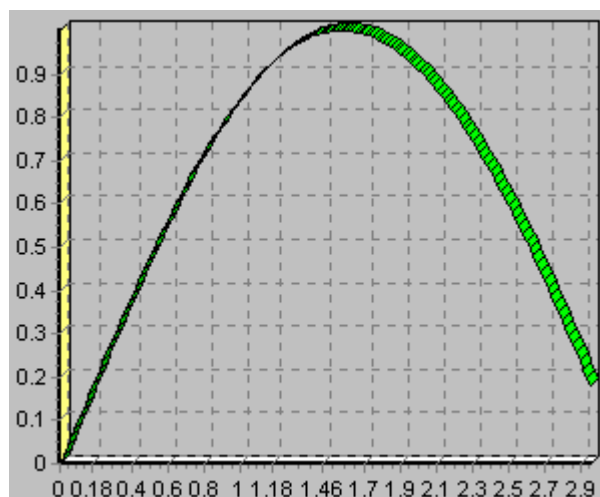
Метод максимального правдоподобия рекомендуется использовать на неупорядоченных данных. При использовании этого метода строится плотность распределения вероятностей, и отсутствующие данные заменяются значением, соответствующим ее максимуму.

Пример.

На рисунке представлены упорядоченные данные с пропусками.



После применения алгоритма аппроксимации эти данные выглядят так.



Алгоритмы заполнения пропусков выполняется в обработчике «Парциальная обработка».

Сглаживание

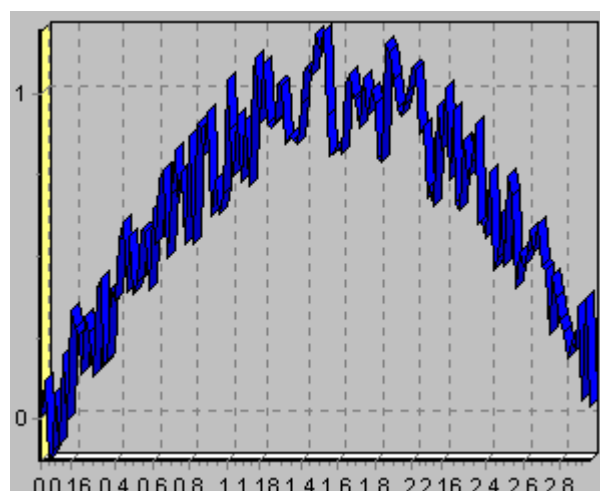
Для сглаживания рядов данных в программе используются два алгоритма.

Первый способ сглаживания – это низкочастотная фильтрация с использованием быстрого преобразования Фурье. При этом задается верхнее значение полосы пропускаемых частот. При подавлении шумов на основе анализа распределения составляющих Фурье спектра на выход фильтра пропускаются спектральные составляющие, превышающие некоторый порог, рассчитанный по эмпирическим формулам в соответствии с заданным критерием степени вычитания шума. Чем больше требуется сгладить данные, тем меньше должно быть значение полосы. Однако слишком узкая полоса может привести к потере полезной информации. Следует заметить, что этот алгоритм наиболее эффективен, если анализируемые данные есть сумма полезного сигнала и белого шума.

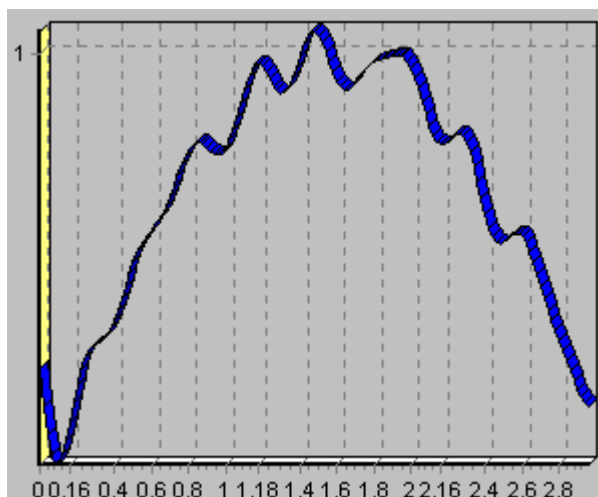
Второй способ сглаживания – это вейвлет-преобразование. Если выбран данный метод, то необходимо задать глубину разложения и порядок вейвлета. Глубина разложения определяет «масштаб» отсеиваемых деталей: чем больше эта величина, тем более «крупные» детали в исходных данных будут отброшены. При достаточно больших значениях параметра (порядка 7-9) выполняется не только очистка данных от шума, но и их сглаживание («обрезаются» резкие выбросы). Использование слишком больших значений глубины разложения может привести к потере полезной информации из-за слишком большой степени «огрубления» данных. Порядок вейвлета определяет гладкость восстановленного ряда данных: чем меньше значение параметра, тем ярче будут выражены «выбросы», и, наоборот, при больших значениях параметра «выбросы» будут сглажены.

Пример.

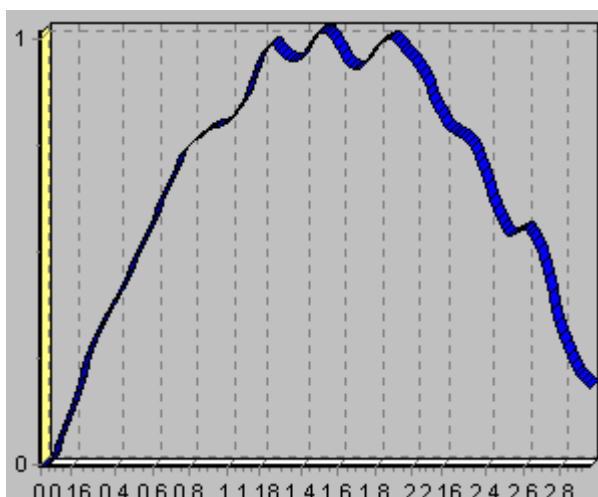
На рисунке показан пример данных с шумом.



На рисунке – диаграмма после применения сглаживания с полосой пропускания равной 15.



На рисунке – диаграмма после применения вейвлет-преобразования с глубиной разложения 3 и порядком 6.



Алгоритмы сглаживания выполняются в обработчике «Парциальная обработка».

Очистка от шумов

Назначение.

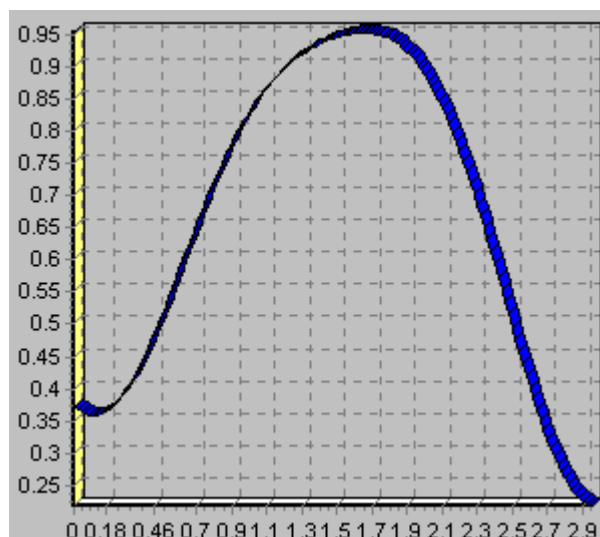
Шумы в данных не только скрывают общую тенденцию, но и проявляют себя при построении модели прогноза. Из-за них модель может получиться с плохими обобщающими качествами.

При выборе режима очистки от шумов необходимо задать степень вычитания шума: малую, среднюю или большую. При использовании вычитания шума следует соблюдать осторожность, т.к. реализованный здесь эвристический алгоритм гарантирует удовлетворительные результаты лишь при выполнении двух условий:

- 1) дисперсия шума значительно меньше энергии полезного сигнала;
- 2) шум имеет нормальное распределение.

Пример удаления шумов.

Применим алгоритм для данных с шумом из предыдущего примера. Результат на рисунке.



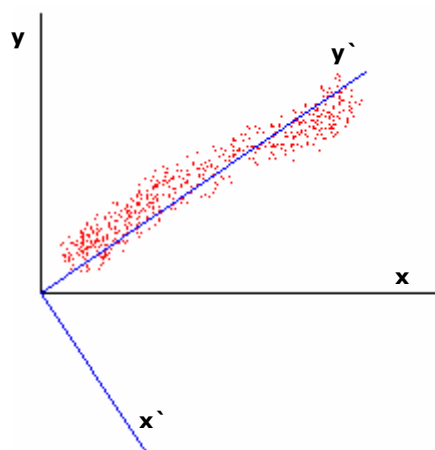
Алгоритм очистки от шумов выполняется в обработчике «Парциальная обработка».

Факторный анализ

При исследовании сложных объектов и систем часто нельзя непосредственно измерить величины, определяющие свойства этих объектов (так называемые *факторы*), а иногда неизвестно даже число и содержательный смысл факторов. Для измерений могут быть доступны иные величины, тем или иным способом зависящие от этих факторов. При этом, когда влияние неизвестного фактора проявляется в нескольких измеряемых признаках, эти признаки могут обнаруживать тесную связь между собой (например, коррелированность). Поэтому общее число факторов может быть гораздо меньше числа измеряемых переменных, которое обычно выбирается исследователем в некоторой степени произвольно. Для обнаружения влияющих на измеряемые переменные факторов используются методы факторного анализа, реализованные в обработчике «Факторный анализ».

В обработчике используется метод главных компонент. Этот метод сводится к выбору новой ортогональной системы координат в пространстве наблюдений. В качестве первой главной компоненты избирают направление, вдоль которого массив данных имеет наибольший разброс. Выбор каждой последующей главной компоненты происходит так, чтобы разброс данных вдоль нее был максимальным, и чтобы эта главная компонента была ортогональна другим главным компонентам, выбранным прежде. В результате получаем несколько главных компонент, каждая следующая из которых несет все меньше информации из исходного набора. Следующим шагом поэтому становится выбор наиболее информативных главных компонент, которые будут использоваться в дальнейшем анализе.

Посмотрим на следующий рисунок. На нем изображено двумерное пространство наблюдений в осях x и y , соответствующих двум измеряемым параметрам.



Как видно, разброс данных велик по обоим направлениям. Теперь повернем систему координат так, чтобы ось y соответствовало направлению наибольшего разброса массива данных, т.е. перейдем в систему координат x' - y' . Теперь по оси x' дисперсия данных невелика и появляется возможность отбросить это направление, перейдя к одномерному пространству.



В этом случае потери некоторой части информации могут компенсироваться удобством работы с данными меньшей размерности. Аналогичные действия выполняются в многомерном случае: система координат последовательно вращается таким образом, чтобы каждый следующий поворот минимизировал остаточный разброс массива данных.

Выбор главных компонент в процессе факторного анализа может осуществляться полуавтоматически: пользователь задает уровень значимости (вклад в результат), который в сумме должны давать главные компоненты. Последней в порядке уменьшения вклада в результат главной компонентой, попадающей в выходной набор, является главная компонента, суммарный уровень значимости (суммарный вклад в результат) которой, не меньше заданного уровня значимости.

Факторный анализ широко используется в следующей ситуации. В очень большом исходном наборе данных есть много полей, некоторые из которых взаимозависимы. На этом наборе данных требуется обучить нейронную сеть. Для того чтобы снизить время, требуемое на обучение сети, с помощью факторного анализа осуществляют переход в новое пространство факторов меньшей размерности. Так как большая часть информативности исходных данных сохраняется в выбранных главных компонентах, то качество модели не ухудшается, зато значительно уменьшается время обучения сети.

Корреляционный анализ

Корреляционный анализ применяется для оценки зависимости выходных полей данных от входных факторов и устранения незначимых факторов. Принцип корреляционного анализа состоит в поиске таких значений, которые в наименьшей степени коррелированы (взаимосвязаны) с выходным результатом. Такие факторы могут быть исключены из результирующего набора данных практически без потери полезной информации. Критерием принятия решения об исключении является порог значимости. Если корреляция (степень взаимозависимости) между входным и выходным факторами меньше порога значимости, то соответствующий фактор отбрасывается как незначимый.

В процессе обработки значащие факторы могут выбираться вручную или автоматически. При ручном выборе около имени каждого входного поля ставится флажок, если это поле нужно включить в выходную выборку, и снимается в противном случае. В автоматическом режиме исключаются все факторы, корреляция которых с выходными полями меньше задаваемого уровня значимости.

Замечание: на практике считается, что корреляция, большая 0.6 означает очень высокую связь между рядами, меньшая 0.3 – отсутствие зависимости, а промежуточные значения констатируют наличие определенной связи. В другом подходе полагается, что зависимость существует, когда корреляция больше 2 поделить на корень из объема выборки.

Пример

В качестве примера рассмотрим, как определить товары-заменители и сопутствующие товары, имея временные ряды объемов продаж. У товаров-заменителей должна быть большая отрицательная корреляция, т.к. увеличение продаж одного товара ведет к спаду продаж второго. А у сопутствующих товаров – большая положительная корреляция.

Пусть есть такие временные ряды продаж товаров:

Товар 1	Товар 2	Товар 3	Товар 4
10	20	15	25
12	22	12	26
14	25	9	26
13	24	10	25
14	25	9	24

14	25	9	23
12	21	12	24
10	18	14	23
16	24	9	22
13	21	9	23
17	25	7	25

Определим корреляцию Товара 1 с остальными товарами.

Входные поля	Корреляция с выходными полями		
	Товар 2	Товар 3	Товар 4
<input checked="" type="checkbox"/> Товар 1	0.832	-0.928	0.301

Как видно из рисунка, ряд продаж Товара 2 имеет очень большую положительную, а Товара 3 – отрицательную корреляцию. Из этого можно сделать вывод, что Товар 2, возможно, является сопутствующим товаром, а Товар 3 – заместителем Товара 1. Корреляция с продажами Товара 4 невысока и находится на уровне практического отсутствия взаимосвязи с Товаром 1.

Обнаружение дубликатов и противоречий

Назначение.

Бывают ситуации, когда проблема неочищенных данных не позволяет построить хорошую модель. В анализируемом наборе данных можно выделить входные и выходные поля. Считается, что значения входных полей полностью определяют значения выходных полей. В этом случае возможно возникновение противоречий, то есть присутствие групп записей, значения в ключевых (входных) полях которых полностью совпадают, а в целевых (выходных) – различаются. Например, если значения в ключевых полях – это коды товаров, а в целевых – цены этих товаров, то присутствие двух записей с одинаковым кодом, но разной ценой как раз и создает противоречие. Обычно бывает так, что только одна запись из группы противоречивых является правильной, а остальные – ошибочными. Очевидно, что присутствие ошибочных данных искажает результаты анализа. Поэтому противоречивые данные лучше вообще исключить из исходной выборки. Однако следует заметить, что искусственное введение противоречий в исходные данные может быть полезным, например, если нужно ввести некоторую неопределенность в данные.

Также в данных могут встречаться записи с одинаковыми входными факторами и одинаковыми выходными, т.е. дубликаты. Таким образом, данные несут избыточность. Присутствие дубликатов в анализируемых данных можно рассматривать как способ повышения «значимости» дублирующейся информации. В некоторых случаях такой прием может быть полезен, например, если при обучении нейросети нужно особо выделить некоторые наборы значений. Однако, в других случаях, дублирование является следствием ошибок при подготовке исходных данных. Дубликаты могут исказить результаты некоторых методов анализа, например, статистического.

Так или иначе, возникает задача выявления дубликатов и противоречий. В Deductor Studio для автоматизации этого процесса есть соответствующий инструмент – обработка «Дубликаты и противоречия».

Дубликаты – записи в таблице, все входные и выходные поля которых одинаковые.

Противоречия – записи в таблице, у которых все входные поля одинаковые, но отличающиеся хотя бы по одному выходному полю.

Суть обработки состоит в том, что определяются входные и выходные поля. Алгоритм ищет во всем наборе записи, для которых одинаковым входным полям соответствуют одинаковые (дубликаты) или разные (противоречия) выходные поля. На основании этой информации создаются два дополнительных логических поля – «Дубликат» и «Противоречие», принимающие значения «правда» или «ложь». В дополнительные числовые поля «Группа дубликатов» и «Группа противоречий» записываются номер группы дубликатов и группы противоречий, в которые попадает данная запись. Если запись не является дубликатом или противоречием, то соответствующее поле будет пустым.

Настройка выявления дубликатов и противоречий заключается в выборе назначений полей исходной выборки данных, то есть в выборе, какие поля входные, какие – выходные.

Пример.

Рассмотрим таблицу. Исходная таблица – это таблица с полями: «Поле 1» – «Поле 4». После применения алгоритма добавлены еще два поля: «Противоречие» и «Дубликат»

Входные поля		Выходные поля		Противоречие	Дубликат	Группа противоречий	Группа дубликатов
Поле 1	Поле 2	Поле 3	Поле 4				
01.01.04	2	1000	1500	Нет	Нет		
21.05.04	3	1000	1500	Да	Нет	1	
21.05.04	3	700	1500	Да	Да	1	1
21.05.04	3	700	1500	Да	Да	1	1
01.09.04	4	1200	1700	Нет	Да		2
01.09.04	4	1200	1700	Нет	Да		2

Вторая строка противоречит третьей. Третья строка противоречит второй и является дубликатом четвертой. Четвертая – противоречит второй и является дубликатом третьей. Пятая и шестая строки – дубликаты.

Обработка дубликатов и противоречий может не проводиться, либо ее следует применять одним из двух способов – в зависимости от смысла анализируемой информации и требований к входным данным.

Обработка дубликатов или противоречий не проводится в тех случаях, когда дубликаты или противоречия были преднамеренно введены в исходные данные. Как правило, этот метод применяется только к одной из описываемых аномалий, то есть либо только дубликаты, либо только противоречия остаются без обработки.

Наличие дубликатов и противоречий может приводить к полному обесцениванию аномальных данных, то есть считается, что присутствие подобных ошибок делает информацию полностью недостоверной. Такая ситуация возникает, например, при обработке социологических данных, когда наличие дубликатов или противоречий свидетельствует о недобросовестности респондента и вызывает недоверие ко всей предоставленной им информации. В этом случае все записи, формирующие группу дубликатов или противоречий, должны быть удалены. Это первый способ обработки.

Существует еще один, наиболее естественный, способ обработки дубликатов. Поскольку все дубликаты представляют собой копии одних и тех же данных, они могут быть сведены к одной записи набора данных, содержащей уникальную копию таких значений. К противоречиям также применим подобный метод обработки, но с некоторыми ограничениями. Напомним, что противоречивые записи содержат одинаковые входные значения, но различные выходные. Приведение таких записей к одной, уникальной, возможно на основе статистической агрегации, то есть вычисления максимума, минимума или среднего из выходных значений и подстановки этой величины в соответствующее поле формируемой уникальной записи. Следует заметить, что такую операцию следует выполнять с осторожностью. Во-первых, семантика, то есть смысл данных, должна допускать возможность вычисления таких статистических значений. Например, статистическая агрегация допустима для цены товара или величины пропускной способности, но бессмысленна для номеров квартир или кодов налогоплательщиков.

Фильтрация

С помощью операции фильтрации можно оставить в таблице только те записи, которые удовлетворяют заданным условиям, а остальные скрыть.

Параметры фильтрации задаются в виде списка условий, который содержит следующие столбцы:

- Операция – позволяет установить функцию отношения «И» или «ИЛИ» между полями, для каждого из которых выполняется фильтрация. Возможна фильтрация по нескольким условиям для нескольких полей одновременно. Практически, в результате фильтрации по каждому из полей или условий будет получено отдельное множество значений. Тогда функция в поле «Операция» устанавливает отношение между этими множествами. Если используется отношение «И», то в результирующий набор будут включены записи, удовлетворяющие условиям фильтрации по обоим полям. Установка отношений возможна, только если настроены два или более условия фильтрации. Для выбора операции следует дважды щелкнуть левой кнопкой мыши в столбце «Операция» для соответствующего условия и из списка, открываемого кнопкой, выбрать нужную функцию отношения. По умолчанию устанавливается отношение «И».
- Имя поля - позволяет выбрать поле, по значениям которого должна быть выполнена фильтрация. Для этого дважды щелкнуть в столбце «Имя поля» и с помощью кнопки открыть список полей текущей выборки, где щелкнуть по нужному полю. Одно и то же поле может быть использовано в нескольких условиях.
- Условие – указывается условие, по которому нужно выполнить фильтрацию для данного поля. Для выбора условия достаточно дважды щелкнуть мышью в соответствующей ячейке и в списке условий, открываемом кнопкой, выделить нужное условие. Доступны следующие условия фильтрации:
 - «=» (равно), «<» (меньше), «<=» (меньше или равно), «>» (больше), «>=» (больше или равно), «<>» (не равно) - отбираются только те записи, значения которых в данном поле соответственно равны содержимому столбца «Значение», меньше, меньше или равны, больше, больше или равны, не равны ему.
 - «пустой» - отбираются только те записи, для которых в данном поле содержится пустое значение. В этом случае поле «Значение» не используется.
 - «не пустой» - отбираются только те записи, для которых в данном поле не содержится пустое значение. В этом случае поле «Значение» не используется.
 - «содержит» - отображаются только записи, которые в данном столбце содержат указанное значение.
 - «не содержит» - отображаются только записи, которые в данном столбце не содержат указанное значение.
 - «в интервале», «вне интервала» - для числовых полей и полей типа «Дата/время» отбираются только те записи, значения которых в данном столбце лежат в выбранном диапазоне (вне выбранного диапазона), то есть между (не между) верхней и нижней границей.
 - «в списке», «вне списка» - отбираются только те записи, которые в данном столбце лежат в выбранном списке (вне выбранного списка).
 - «начинается на», «не начинается на» - для строковых полей отбираются записи, значения которых в данном столбце начинаются (не начинаются) на введенную последовательность символов.
 - «заканчивается на», «не заканчивается на» - для строковых полей отбираются записи, значения которых в данном столбце заканчиваются (не заканчиваются) на введенную последовательность символов.
 - «первый», «не первый» - для полей типа «Дата/время» - по данному полю отбираются первые (не первые) N периодов от выбранной даты. Периодом может быть день, неделя, месяц, квартал, год. Например, если выбрать условие «первые 3 дня от 29.11.2004», то будут отобраны записи, в которых значение данного поля равно «29.11.2004», «30.11.2004», «01.12.2004».
 - «последний», «не последний» - для полей типа «Дата/время» отбираются последние (не последние) N периодов от выбранной даты. Периодом может быть день, неделя, месяц, квартал, год. Например, если выбрать условие «последние 3 дня от 29.11.2004», то будут отобраны записи, в которых значение данного поля равно «29.11.2004», «28.11.2004», «27.11.2004».
- Значение – указывается значение, по которому будет производиться фильтрация записей в соответствии с заданным условием. Способ ввода значения будет различным в зависимости от типа данных и условия. Допустим, в качестве условия выбрана операция отношения «=», «<>», «>» и т.д. Если данные в поле являются непрерывными (т.е. числовыми), то достаточно дважды щелкнуть мышью в соответствующей ячейке, чтобы появился курсор, затем ввести значение (число). Если поле, по которому выполняется фильтрация, имеет тип «строка» (т.е. является дискретным), то в результате двойного щелчка в столбце «Значение» появится

кнопка выбора, которая откроет окно «Список уникальных значений», где будут отображены все уникальные значения поля и их количество. Чтобы выбрать значение для условия отбора достаточно выделить его и щелкнуть «Ok», либо просто выполнить двойной щелчок. Если выбрано условие «между» или «не между», тогда по кнопке выбора (справа от поля) откроется окно, в котором необходимо указать верхнюю и нижнюю границы интервала. Если выбрано условие «в списке» или «вне списка», тогда по кнопке выбора откроется окно, в котором необходимо сформировать список значений, установив галочки рядом с нужными значениями. Если выбрано условие «первый», «не первый», «последний», «не последний», тогда при нажатии кнопки выбора откроется окно, где необходимо указать дату, от которой вести отсчет, тип периода и количество периодов. Дата может быть текущей, от имеющихся в измерении данных, либо дата, указанная вручную. Дата от имеющихся данных означает либо минимальную дату во всем списке значений (если выбрано условие «первый», «не первый»), либо максимальную (если выбрано условие «последний», «не последний»).

Фильтрация может быть полезна для применения различных алгоритмов к группам данных, так как позволяет выделить из выборки только нужную часть. Тем не менее, если требуется провести анализ только части данных, желательно загружать в программу уже отфильтрованный набор. Такая возможность есть, например, при загрузке данных из хранилища. В этом случае значительно экономится память, занимаемая данными.

Например, есть несколько групп товаров и нужно провести определенный анализ для каждой группы отдельно. Тогда можно воспользоваться фильтрацией, оставив в наборе данные только по одной группе, провести анализ. Затем к исходному набору снова применить фильтрацию, оставив другую группу товара, и провести анализ для нее. И так для каждой товарной группы.

Трансформация данных

Анализируемая информация, представленная в виде набора данных, имеет определенный формат. Под форматом данных подразумевается отнесение их к определенному типу (целочисленные, строковые, даты), задание вида (дискретные или непрерывные) и т. п. Для анализа различных аспектов информации может потребоваться преобразование ее формата, или трансформация. Кроме преобразования форматов, трансформация включает в себя изменение представления данных и другие операции, связанные с преобразованиями входного набора данных.

Настройка набора данных

Обработка «Настройка набора данных» предназначена для изменения имени, метки, типа, вида и назначения полей текущей выборки данных и кэширования выходного набора.

Настройка полей

У каждого поля можно изменить метку столбца, которая будет использоваться для дальнейшей работы в программе. Если в текущей выборке данных поле имеет имя «Name», ему можно задать метку «Наименование», что гораздо удобнее при дальнейшем отображении этого поля в таблицах или диаграммах.

Изменение имени поля удобно в тех случаях, когда имена столбцов могут измениться в источнике данных или при перенастройке узлов верхних уровней. В этом случае в узле «Настройка набора данных» имя исходного столбца заменяется другим, на которое и настраиваются все дочерние узлы. После такой операции изменение имен полей на верхних уровнях не требует перенастройки всех дочерних узлов.

Далее каждому полю можно изменить тип:

- логический - данные в поле будут принимать только два значения 0 или 1 (ложь или истина);
- дата/время - поле будет содержать данные типа дата/время;
- вещественный - значениями поля будут числа с плавающей точкой;
- целый - данные в поле будут целыми числами;
- строковый - данные в поле будут представлять собой строки символов (для этого типа поля можно также указать максимальное число символов в строке – Размер поля).

Затем можно указать вид данных:

- непрерывный - значения в столбце могут принимать любое значение в рамках своего типа. Обычно непрерывными являются числовые данные;
- дискретный - данные в столбце могут принимать ограниченное число значений. Обычно дискретный характер носят строковые данные.

В зависимости от содержимого поля «Тип данных», на выбор вида данных накладываются ограничения – например, строковые данные не могут быть непрерывными. К выбору типа и вида данных нужно относиться серьезно, так как это влияет на возможность дальнейшего использования этого поля.

Далее можно изменить назначение полей. В зависимости от дальнейшего использования выборки данных, предлагается изменить текущие назначения полей на следующие:

- непригодное - данные в поле будут непригодны для дальнейшей обработки.
- неиспользуемое - запрещает использование поля в обработке данных и исключает его из выходного набора. В отличие от непригодного поля, такие поля в принципе могут использоваться, просто в этом нет необходимости;
- первичный ключ - поле будет использоваться в качестве первичного ключа;
- входное - поле таблицы будет являться входным полем в последующей обработке (нейронной сети, дерева решений и т.д.).
- выходное - поле таблицы будет являться выходным полем в последующей обработке (например, целевым полем для обучения нейронной сети).
- информационное - поле содержит вспомогательную информацию, которую часто полезно отображать, но не следует использовать при обработке;
- измерение - поле будет использоваться в качестве измерения в многомерной модели данных;
- факт - значения поля будут использованы в качестве фактов в многомерной модели данных.
- свойство - поле содержит описание свойств или параметров некоторого объекта;
- транзакция – поле, содержащее идентификатор событий, происходящих совместно (одновременно). Например, номер чека, по которому приобретены товары. Тогда покупка товара – это событие, а их совместное приобретение по одному чеку - транзакция;
- элемент – поле, содержащее элемент транзакции (событие).

Для установки первоначальных параметров полей необходимо выделить поле или список полей и нажать на кнопку "Сброс параметров".

Кэширование данных

Одно из важных применений обработчика «Настройка полей» состоит в кэшировании данных. Кэширование – это загрузка часто используемой информации в оперативную память для быстрого доступа к ней, минуя многократные считывания с жесткого диска. Кэширование может заметно повысить скорость работы сценария в следующих случаях:

- перед настройкой полей находится узел, в котором проводится большое количество сложных и длительных вычислений. В узле производятся каждый раз по мере обращения к записям, поэтому обращение к данным может занимать много времени. В этом случае установка кэша позволяет однократно вычислить все выражения и сохранить результаты в памяти;
- из узла настройки полей выходит несколько ветвей обработки данных. В обычном режиме при переходе к каждой следующей ветви данные будут читаться с диска, а это достаточно длительная операция. В случае же использования кэша данные однократно загружаются в память и в дальнейшем забираются на обработку уже из нее.

Если объем кэшируемых данных превышает доступные ресурсы оперативной памяти, то значительного прироста в быстродействии может не наблюдаться, т.к. частично они все равно окажутся выгруженными на диск.

Включить кэширование данных в узле «Настройка полей» можно с помощью установки флага «Кэшировать результирующий набор данных».

Скользящее окно

При решении некоторых задач, например, при прогнозировании временных рядов при помощи нейросети, требуется подавать на вход анализатора значения нескольких, смежных, отсчетов из исходного набора данных. Такой метод отбора данных называется скользящим окном (окно – поскольку выделяется только некоторый непрерывный участок данных, скользящее – поскольку это окно «перемещается» по всему набору). При этом эффективность реализации заметно повышается, если не выбирать данные каждый раз из нескольких последовательных записей, а последовательно расположить данные, относящиеся к конкретной позиции окна, в одной записи.

Значения в одном из полей записи будут относиться к текущему отсчету, а в других – смещены от текущего отсчета «в будущее» или «в прошлое». Таким образом, преобразование скользящего окна имеет два параметра: «глубина погружения» - количество «прошлых» отсчетов, попадающих в окно, и «горизонт прогнозирования» – количество «будущих» отсчетов. Следует отметить, что для граничных (относительно начала и конца всей выборки) положений окна будут формироваться неполные записи, т.е. записи, содержащие пустые значения для отсутствующих прошлых или будущих отсчетов. Алгоритм преобразования позволяет исключить такие записи из выборки (тогда для нескольких граничных отсчетов записи формироваться не будут), либо включить их (тогда формируются записи для всех имеющихся отсчетов, но некоторые из них будут неполными). Отметим, что для правильного формирования скользящего окна данные должны быть соответствующим образом упорядочены.

Пример.

Есть история продаж за половину года по месяцам, представленная таблицей:

Первый день месяца	Объем продаж (тыс. руб.)
01.01.2004	1000
01.02.2004	1160
01.03.2004	1210
01.04.2004	1130
01.05.2004	1250
01.06.2004	1300

Если задать глубину погружения 2 и горизонт прогнозирования 1, то получим следующую таблицу с неполными записями.

Первый день месяца	Объем продаж два месяца назад	Объем продаж месяц назад	Объем продаж в текущий месяц	Объем продаж на следующий месяц
				1000
01.01.2004			1000	1160
01.02.2004		1000	1160	1210
01.03.2004	1000	1160	1210	1130
01.04.2004	1160	1210	1130	1250
01.05.2004	1210	1130	1250	1300
01.06.2004	1130	1250	1300	
	1250	1300		
	1300			

Или следующую таблицу с полными записями.

Первый день месяца	Объем продаж два месяца назад	Объем продаж месяц назад	Объем продаж в текущий месяц	Объем продаж на следующий месяц
01.03.2004	1000	1160	1210	1130
01.04.2004	1160	1210	1130	1250
01.05.2004	1210	1130	1250	1300

Такую таблицу можно использовать при построении моделей, например, прогноза. При этом на вход модели для ее обучения будут подаваться поля с текущим и двумя предыдущими месяцами, а на выход – поле с объемом продаж на следующий месяц.

Эту таблицу также можно использовать для вычисления оборотов за определенное количество месяцев. Например, вычисляя разницу между столбцами с объемом продаж за текущий месяц и объемом продаж за предыдущий месяц.

Преобразование даты

Преобразование даты служит для анализа всевозможных показателей за определенный период (день, неделя, месяц, квартал, год). Суть преобразования заключается в том, что на основе столбца с информацией о дате формируются один или несколько столбцов, в которых указывается, к какому заданному интервалу времени принадлежит строка данных. Тип интервала задается аналитиком, исходя из того, что он хочет получить – данные с агрегацией за год, квартал, месяц, неделю, день или сразу по всем интервалам.

Значения нового столбца, полученного после применения преобразования даты, могут быть одного из трех типов: строка, число или дата. Например, нужно из даты «10.04.2004» получить только месяц. Тогда в столбце строкового типа будет содержаться «2004-M04» и его уже нельзя использовать как дату, например, к нему нельзя снова применить преобразование даты. А в столбце типа «дата» будет значение «01.04.2004». К нему снова можно применить преобразование и получить, например, номер квартала. Новый столбец будет содержать значение 2 числового типа.

Пример.

Пример использования преобразования даты приведен в таблице. Первый столбец «Дата» – это исходный столбец. Остальные получены после обработки.

Дата	Год + Квартал	Год + Месяц	Год + Неделя	Квартал	Месяц	Неделя	День года	День недели	День недели
01.01.2004	01.01.2004	01.01.2004	01.01.2004	1	1	1	1	4	4 Четверг
09.01.2004	01.01.2004	01.01.2004	05.01.2004	1	1	2	9	5	5 Пятница
17.01.2004	01.01.2004	01.01.2004	12.01.2004	1	1	3	17	6	6 Суббота
25.01.2004	01.01.2004	01.01.2004	19.01.2004	1	1	4	25	7	7 Воскресенье
02.02.2004	01.01.2004	01.02.2004	02.02.2004	1	2	6	33	1	1 Понедельник
10.02.2004	01.01.2004	01.02.2004	09.02.2004	1	2	7	41	2	2 Вторник
18.02.2004	01.01.2004	01.02.2004	16.02.2004	1	2	8	49	3	3 Среда
26.02.2004	01.01.2004	01.02.2004	23.02.2004	1	2	9	57	4	4 Четверг
05.03.2004	01.01.2004	01.03.2004	01.03.2004	1	3	10	65	5	5 Пятница
13.03.2004	01.01.2004	01.03.2004	08.03.2004	1	3	11	73	6	6 Суббота

Есть таблица с информацией о продажах. Пусть необходимо посмотреть объемы продаж по клиентам и по месяцам. Для этого заменим дни продаж месяцем, в который попадает этот день. Объемы продаж удобно посмотреть с помощью OLAP-куба.

	DATE (Год + Месяц) ▼			
CLIENTNAME ▼	01.01.2004	01.02.2004	01.03.2004	Итого
Покупатель 1	65.00	356.00	100.00	521.00
Покупатель 2	263.00	251.00	260.00	774.00
Покупатель 28	24.00	27.00	120.00	171.00
Покупатель 3	250.00	200.00	205.00	655.00
Покупатель 38	73.00	27.00	90.00	190.00
Покупатель 4	333.00	279.00	232.00	844.00
Покупатель 5	269.00	234.00	241.00	744.00
Покупатель 6	341.00	271.00	302.00	914.00
Покупатель 7	272.00	328.00	243.00	843.00
Покупатель 8	331.00	298.00	260.00	889.00
Итого	2,221.00	2,271.00	2,053.00	6,545.00

Квантование значений

При выполнении этой операции осуществляется разбиение диапазона числовых значений на указанное количество интервалов определенным методом и замена каждого обрабатываемого значения на число, связанное с интервалом, к которому оно относится, либо метку интервала. Интервалы разбиения включают в себя нижнюю границу, но не включают верхнюю, кроме последнего интервала, который включает в себя обе границы. Результатом преобразования может быть: номер интервала (от нуля до значения, на единицу меньшего количества интервалов), значение нижней или верхней границы интервала разбиения, среднее значение интервала разбиения, метка интервала.

Квантование может быть осуществлено интервальным или квантильным методом. Интервальное квантование подразумевает разбиение диапазона значений на указанное количество значений равной длины. Например, если значения в поле попадают в диапазон от 0 до 10, то при интервальном квантовании на 10 интервалов мы получим отрезки от 0 до 1, от 1 до 2 и т. д. При этом 0 будет относиться к первому интервалу, 1 – ко второму, а 9 и 10 – к десятому. Квантильное квантование подразумевает разбиение диапазона значений на равновероятные интервалы, то есть на интервалы, содержащие равное (или, по крайней мере, примерно равное) количество значений. Нарушение равенства возможно только тогда, когда значения, попадающие на границу интервала, встречаются в наборе данных несколько раз. В этом случае все они относятся к одному определенному интервалу и могут вызвать «перевес» в его сторону.

Настройки.

Для настройки квантования требуется для каждого используемого при разбиении поля указать:

1. способ разбиения – по интервалам или по квантилям;
2. количество интервалов;
3. значение, подставляемое вместо значения интервала – номер интервала, нижняя граница, верхняя граница, середина интервала, метка интервала. Если выбрана метка интервала, то нужно еще задать для каждого интервала метку, то есть его наименование.

Кроме того, после окончания автоматического расчета границ интервалов можно вручную изменить вычисленные границы. При этом нижняя граница любого интервала не может быть больше верхней, хотя совпадать они могут. Ручное изменение границ может потребоваться в тех случаях, когда исходная выборка данных не отражает всего диапазона значений, которые может принимать исследуемая величина на практике.

Принцип работы.

В таблице рассматриваемой величине соответствует столбец. Значения из этого столбца попадают в определенный интервал и заменяются значением, соответствующим этому интервалу (номер, нижняя граница, верхняя граница и т.д.). Весь диапазон рассматриваемой величины разбивается на заданное число интервалов. Пусть задан способ разбиения по интервалам. Каждый интервал при этом имеет одинаковую длину. Если задан способ разбиения по квантилям, то длины

интервалов вычисляются так, чтобы в каждый интервал попало одинаковое количество величин из столбца таблицы.

Пример.

Допустим, у нас есть таблица с информацией о кредиторах и суммой взятых кредитов. Нужно узнать активность разных возрастных групп кредиторов.

N п/п	Возраст	Сумма
1	37	7000
2	38	7500
3	60	14500
4	28	15000
5	59	32000
6	25	11500
7	57	5000
8	45	61500
...

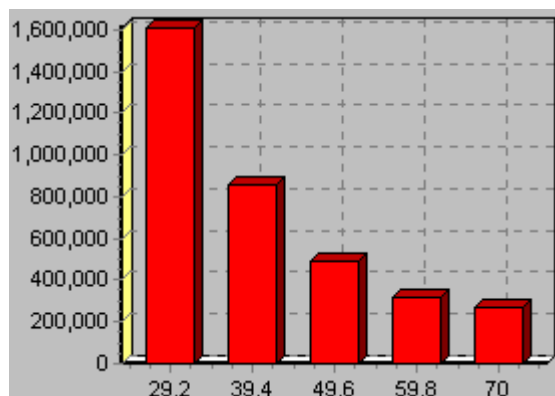
Статистика показывает, что возраст кредиторов лежит в диапазоне от 19 до 70 лет. Разобьем возраст на 5 равных интервалов, заменив возраст номером интервала.

Номер интервала	Нижняя граница	Верхняя граница
1	19	29.2
2	29.2	39.4
3	39.4	49.6
4	49.6	59.8
5	59.8	70

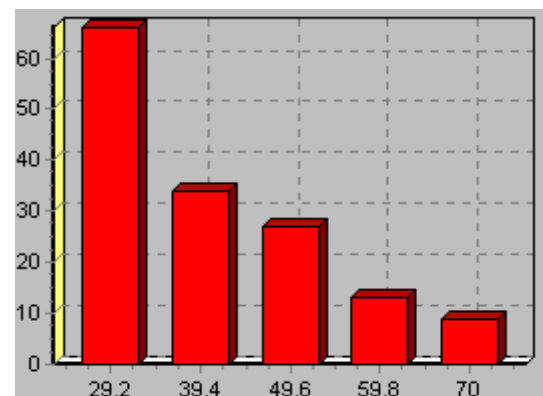
Получим таблицу.

N п/п	Номер интервала	Сумма
1	2	7000
2	2	7500
3	5	14500
4	1	15000
5	4	32000
6	1	11500
7	4	5000
8	3	61500
...

Теперь можно посмотреть количество кредиторов в каждой возрастной группе и сумму взятых кредитов по этим группам.



Сумма кредитов.



Количество кредиторов.

По таким данным можно делать выводы о необходимости стимулирования малоактивных возрастных групп.

Сортировка

С помощью сортировки можно изменять порядок следования записей в исходной выборке данных в соответствии с заданным пользователем алгоритмом сортировки. Результатом выполнения сортировки будет новая выборка данных, записи в которой будут следовать в соответствии с заданными параметрами сортировки.

Если сортировка производится по одному полю, то все записи исходной выборки располагаются в порядке возрастания или убывания его значений. Если сортировка производится по двум или более полям, то действует следующий алгоритм:

1. Сначала записи сортируются в заданном порядке для первого поля.
2. В каждом наборе одинаковых значений первого поля, записи располагаются в заданном порядке для второго поля.

И так далее для всех полей, подлежащих сортировке.

В окне настройки параметров сортировки представлен список условий сортировки, в котором содержатся две графы:

- Имя поля – содержит имена полей, по которым следует выполнить сортировку.
- Порядок сортировки – содержит порядок сортировки данных в соответствующем поле – по возрастанию или по убыванию.

Слияние

Обработка «Слияние» предназначена для объединения двух таблиц по нескольким одинаковым полям. Для этого необходимо задать общие поля двух таблиц. Они называются измерениями. Объединить таблицы можно только путем «присоединения» одной (исходной) таблицы к другой. Предполагается, что в присоединяемой таблице есть поля, отсутствующие в исходной. Они называются фактами. После слияния к исходной таблице добавятся поля-факты. Их значения будут браться из присоединяемой таблицы из полей, в которых значения измерений совпадают со значениями измерений исходной таблицы. Так как в присоединяемой таблице значения измерений могут повторяться в разных строках, а значение обязательно должно быть только одно (уникально), факты перед слиянием будут агрегированы по измерениям. Функцию агрегации можно задавать: сумма, количество, минимум, максимум, среднее.

Настройки.

Для настройки следует указать, какие поля присоединяемой таблицы являются измерениями, и сопоставить их полям исходной таблицы. Затем следует указать, какие поля присоединяемой таблицы являются фактами.

Пример.

Пусть дана исходная таблица.

Поставщик	Товар
ЖБИ	Бетон
ЖБИ	Плита
КРЗ	Рубероид
КРЗ	Картон

Нужно присоединить столбец с объемами поставок каждого товара по каждому поставщику. Эту информацию можно взять из таблицы с историей поставок.

Дата	Поставщик	Товар	Количество
10.02.2004	ЖБИ	Бетон	100
10.03.2004	КРЗ	Рубероид	10
10.03.2004	КРЗ	Рубероид	20
10.03.2004	ЖБИ	Плита	5
11.03.2004	ЖБИ	Бетон	130
11.03.2004	КРЗ	Картон	20

Чтобы произвести слияние нужно настроить назначение полей следующим образом.

Импортируемое поле	Назначение	Поле/Агрегировать
Дата	Неиспользуемое	
Поставщик	Измерение	Поставщик
Товар	Измерение	Товар
Количество	Факт	Сумма

После слияния будет получена таблица.

Поставщик	Товар	Сумма (Количество)
ЖБИ	Бетон	230
ЖБИ	Плита	5
КРЗ	Рубероид	30
КРЗ	Картон	20

Например, поставщик «ЖБИ» и товар «Бетон» одновременно встречались в присоединяемой таблице в двух строках – в одной количество равно 100, в другой – 130. При слиянии использовалась функция агрегации сумма. Поэтому в соответствующей строке результирующей таблицы поле «Количество» содержит 230.

Замена данных

В результате выполнения этой операции производится замена значений по таблице подстановки, которая содержит пары, состоящие из исходного значения и выходного значения. Например, 0 – «красный», 1 – «зеленый», 2 – «синий». Или «зима» – «январь», «весна» – «апрель», «лето» – «июль», «осень» – «октябрь». Для каждого значения исходного набора данных ищется соответствие среди исходных значений таблицы подстановки. Если соответствие найдено, то значение меняется на соответствующее выходное значение из таблицы подстановки. Если значение не найдено в таблице, оно может быть либо заменено значением, указанным для замены «по умолчанию», либо оставлено без изменений (если такое значение не указано).

Пример.

Пусть, например, есть список клиентов и каким-либо образом каждый клиент был отнесен в одну из трех групп. Группа задана номером. Таблица может быть такой.

Наименование клиента	Группа
Клиент 1	1
Клиент 2	3
Клиент 3	2
Клиент 4	1
Клиент 5	2
...	...

Понять, что представляет каждый клиент по такой таблице невозможно. Но известно, что соответствует каждой группе.

Группа	Наименование
1	Постоянные
2	Случайные
3	Потерянные

Это таблица подстановки. Воспользуемся ей для замены значений группы в таблице клиентов.

Наименование клиента	Группа
Клиент 1	Постоянные
Клиент 2	Потерянные
Клиент 3	Случайные
Клиент 4	Постоянные
Клиент 5	Случайные
...	...

Группировка

Назначение.

Трудно судить по данным каждой записи в отдельности. Аналитику для принятия решения часто необходима сводная информация. Совокупные данные намного более информативны, тем более, если их можно получить в разных разрезах. В Deductor Studio предусмотрен инструмент, реализующий сбор сводной информации – «Группировка». Группировка позволяет объединять записи по полям - измерениям, агрегируя данные в полях-фактах для дальнейшего анализа.

Настройки.

Для настройки группировки требуется указать, какие поля являются измерениями, а какие – фактами. Для каждого факта требуется указать функцию агрегации. Это может быть сумма, среднее, максимум, минимум, количество. Количество – это число агрегируемых значений фактов для каждой комбинации измерений. Для строковых полей в качестве функции агрегации можно указать только максимум, минимум и количество. При этом максимум (минимум) двух строк рассчитывается посимвольным сравнением. Сначала сравниваются два первых символа строки. Если коды этих символов одинаковы, сравниваются вторые символы и т.д. Как только в строках появится первый несовпадающий символ, функция агрегации принимает значение строки, код символа в которой оказался больше (меньше).

Принцип работы.

В таблице данных ищутся записи с одинаковыми полями-измерениями. К полям-фактам таких записей применяются функции агрегации.

Пример.

Сгруппируем объемы продаж по клиентам и месяцам. Для этого нужно назначить измерениями поля: клиент и месяц. Поле с объемом продаж назначить фактом и указать для него функцию агрегации – сумма.

К результату можно снова применить операцию группировки. Например, можно задать измерением поле месяц, а фактом – объемы продаж, указав в качестве функции агрегации среднее. Результатом будут объемы продаж в месяц в среднем по каждому клиенту.

Зачем проводить группировку данных, если это может сделать OLAP-куб? При использовании инструмента «Группировка» формируется таблица со сгруппированными значениями, которую, в отличие от OLAP-куба, можно использовать для обработки другими алгоритмами программы.

Например, необходимо построить прогноз объемов продаж. Обычно данные о продажах собираются в определенный промежуток времени, например, раз в неделю. В таком случае желательно группировать объемы продаж по неделям. Использование выборки по дневным продажам даст плохие результаты, так как продажи по каждому дню в отдельности могут очень сильно отличаться. Однако объемы продаж за неделю или за месяц в среднем не так сильно зашумлены. Поэтому, перед построением прогноза желательно применить две обработки: преобразование даты – для приведения даты к неделе или месяцу, в который она попадает, и группировка для вычисления объемов продаж за неделю или месяц.

Разгруппировка

Назначение.

Группировка используется для объединения фактов по каким-либо измерениям. При этом под объединением понимается применение некоторой функции агрегации. Если в исходном наборе данных присутствовали какие-либо другие измерения, то теряется информация о значениях фактов в разрезе этих измерений. Алгоритм разгруппировки позволяет восстановить эти факты, но их значения восстанавливаются не точно, а пропорционально известному вкладу в сгруппированные значения.

Пример.

Пусть есть таблица с объемами продаж некоторого товара за два месяца.

Месяц	Номер товара	Количество
1	1	100
1	2	10
2	1	110
2	2	20

Для вычисления объемов продаж всех товаров за месяц используется группировка с измерением «Месяц» и фактом «Количество».

Месяц	Количество
1	110
2	130

Для восстановления объемов продаж каждого товара за месяц необходимо сделать разгруппировку.

Месяц	Номер товара	Количество
1	1	96,25
1	2	13,75
2	1	113,75
2	2	16,25

Поясним, как выполнена такая разгруппировка. Общее количество товара за 2 месяца равно 240. При этом первый товар внес в это количество 210 или 87.5%. После группировки выяснилось, что общее количество товара за первый месяц равно 110. Из них 87.5% приходится на товар номер 1. Это составляет 96.25. Количество товара за второй месяц равно 130. Из них 87.5% приходится на первый товар. Это составляет 113.75. Точно также восстанавливаются значения для второго товара.

В этом примере мы посчитали вклад первого товара в сумму по всем месяцам. Однако такой расчет может оказаться неактуальным, так как пропорциональное соотношение продаваемого товара может изменяться с течением времени. Поэтому можно посчитать вклад первого товара в общее количество по последнему месяцу (в общем случае, разгруппировывать можно по любому числу последних месяцев, недель, дней и вообще по произвольному числу значений любого измерения). Тогда получим такую таблицу.

Месяц	Номер товара	Количество
1	1	93,078
1	2	16,923
2	1	110,0
2	2	20,0

Общий объем за второй месяц равен 130. Из них 110 приходится на первый товар. Это 84.6%. На второй приходится 15.4%.

Допустим, мы сделали прогноз объемов продаж на третий месяц. Часто прогноз строится не по каждому товару отдельно, а по всем товарам или группам товаров. Тогда, используя разгруппировку, можно восстановить прогнозные значения для каждого товара. Например, прогноз на третий месяц составил 100. Используя разгруппировку по двум предыдущим месяцам, получим прогноз для первого товара – 87.5 и для второго товара – 12.5.

Для настройки разгруппировки нужно выбрать поле, значения которого нужно восстановить, и указать ему назначение «Факт» (в примере это количество). Затем следует выбрать восстанавливаемое измерение (в примере это номер товара). Для восстановления значений факта необходимо выбрать столбец, по которому проводится разгруппировка. В нашем примере прогнозируемое количество продаж товаров восстанавливается по полю «Количество», т.е. разгруппировка рассчитывается на основании значений столбца «Количество» за прошлые периоды времени. Далее нужно выбрать способ восстановления: по всей выборке (в примере - по всем месяцам) или по последним N значениям какого-либо измерения (в примере – по одному последнему месяцу). В последнем случае предлагается выбрать измерение и количество его последних значений.

Data Mining

Автокорреляция

Целью автокорреляционного анализа является выяснение степени статистической зависимости между различными значениями (отсчетами) случайной последовательности, которую образует поле выборки данных. В процессе автокорреляционного анализа рассчитываются коэффициенты корреляции (мера взаимной зависимости) для двух значений выборки, отстоящих друг от друга на определенное количество отсчетов, называемые также лагом. Совокупность коэффициентов корреляции по всем лагам представляет собой автокорреляционную функцию ряда (АКФ):

$$R(k) = \text{corr}(X(t), X(t+k)), \text{ где } k > 0 - \text{целое число (лаг)}.$$

По поведению АКФ можно судить о характере анализируемой последовательности, наличии периодичности (например, сезонной).

Очевидно, что при $k=0$, автокорреляционная функция будет максимальной и равной 1 - т.е. значение последовательности полностью коррелировано само с собой - степень статистической взаимозависимости максимальна. Действительно, если факт появления данного значения имел место, то и соответствующая вероятность равна 1. По мере увеличения числа лагов, т.е. увеличения расстояния между двумя значениями для которых вычисляется коэффициент корреляции, значения АКФ будут убывать из-за уменьшения статистической взаимозависимости между этими значениями (вероятность появления одного из них все меньше влияет на вероятность появления другого). При этом чем быстрее убывает АКФ, тем быстрее изменяется анализируемая последовательность. И наоборот, если АКФ убывает медленно, то и соответствующий процесс является относительно гладким. Если в исходной выборке имеет место тренд (плавное увеличение или уменьшение значений ряда), то плавное изменение АКФ также будет иметь место. При наличии сезонных колебаний в исходном наборе данных, АКФ также будет иметь периодические всплески.

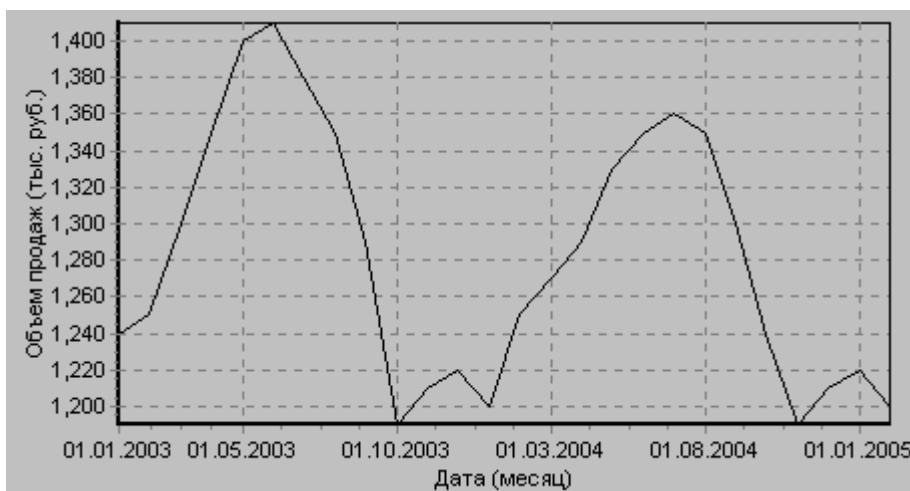
Для применения алгоритма автокорреляции в Deductor Studio необходимо выбрать поле, для которого вычисляется АКФ. В поле «Количество отсчетов» требуется указать количество отсчетов, для которых будут рассчитаны значения АКФ.

Пример.

Есть таблица продаж некоторого товара за два с небольшим года.

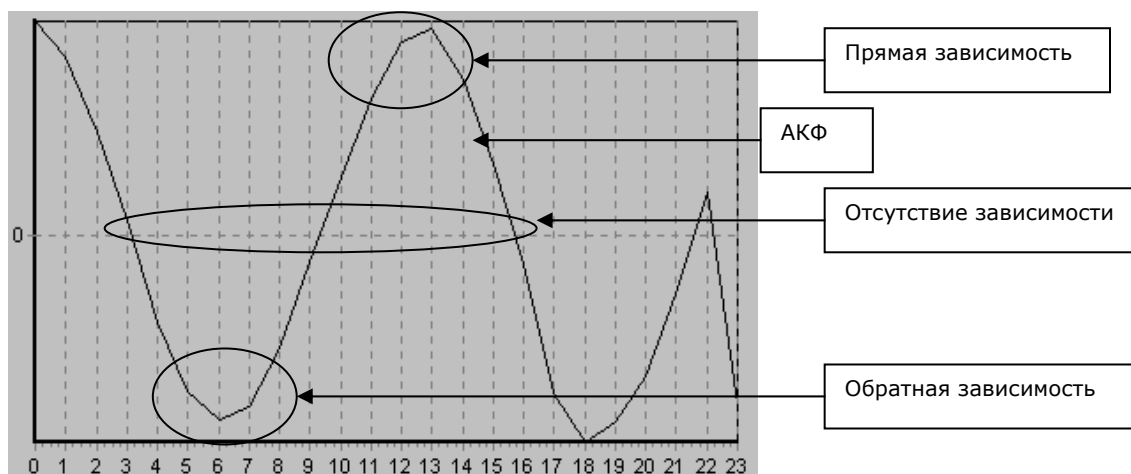
Дата (месяц)	Объем продаж (тыс. руб.)
01.01.2003	1240.0
01.02.2003	1250.0
01.03.2003	1300.0
01.04.2003	1350.0
01.05.2003	1400.0
...	...

График зависимости поля «Объем продаж (тыс.руб.)» от поля «Дата (месяц)» показан на графике.



Попробуем определить наличие сезонных зависимостей продаж этого товара.

Для оценки сезонности выбирают количество отсчетов, обычно, большее 12, если один отсчет соответствует месяцу, то есть сезонность ищется за период больше одного года. Вычислим автокорреляционную функцию для поля «Объем продаж», установив количество отсчетов равным 24 (два года). На диаграмме АКФ выглядит следующим образом.



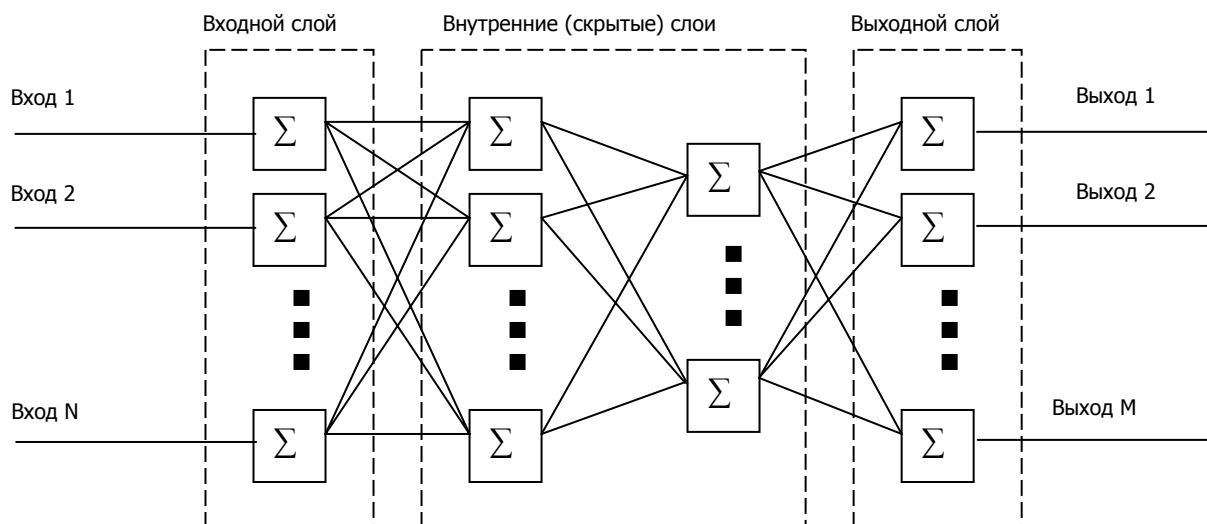
Первое максимальное значение АКФ находится на 12 и 13-м отсчетах (лагах). Это соответствует периоду сезонности, оказавшемуся равным одному году. Таким образом, номер отсчета, соответствующий максимуму АКФ показывает количество месяцев, по прошествии которого наблюдается та же тенденция продаж.

На 6 отсчете АКФ имеет первое минимальное значение и это значение близко к -1. Это так называемая обратная автокорреляционная зависимость. Она присутствует не всегда. В нашем случае это соответствует половине периода сезонности.

Нейронные сети

Нейронные сети (НС) представляют собой вычислительные структуры, моделирующие простые биологические процессы, аналогичные процессам, происходящим в человеческом мозге. Нейросети – это распределенные и параллельные системы, способные к адаптивному обучению путем реакции на положительные и отрицательные воздействия. В основе построения сети лежит элементарный преобразователь, называемый *искусственным нейроном* или просто *нейроном* по аналогии с его биологическим прототипом.

Структуру нейросети можно описать следующим образом. Нейросеть состоит из нескольких слоев: входной, внутренние (скрытые) и выходной слою. Входной слой реализует связь с входными данными, выходной – с выходными. Внутренних слоев может быть от одного и больше. В каждом слое содержится несколько нейронов.



Между нейронами есть связи, называемые *весами*.

Назначение и подготовка обучающей выборки.

Нейросеть способна имитировать какой-либо процесс. Любое изменение входов нейросети ведет к изменению ее выходов. Причем выходы нейросети однозначно зависят от ее входов.

Но перед тем как использовать нейросеть ее необходимо обучить. Задача обучения здесь равносильна задаче аппроксимации функции, то есть восстановление функции по отдельно взятым ее точкам – таблично заданной функции. Таким образом, для обучения нужно подготовить таблицу с входными значениями и соответствующими им выходными значениями, то есть подготовить обучающую выборку. По такой таблице нейросеть сама находит зависимости выходных полей от входных. Далее эти зависимости можно использовать, подавая на вход нейросети некоторые значения. На выходе будут восстановлены зависимые от них значения. Причем на вход можно подавать значения, на которых нейросеть не обучалась.

Важно следующее. Обучающая выборка не должна содержать противоречий, так как нейросеть однозначно сопоставляет выходные значения входным. После обучения на вход нейросети необходимо подавать значения из диапазона, на котором она обучалась. Например, если при обучении нейросети на один из ее входов подавались значения от 0 до 100, то в дальнейшем следует на этот вход подавать значения из диапазона от 0 до 100. Допускается подавать значения, лежащие рядом с диапазоном.

Настройка назначения полей.

В самом начале работы с нейросетью нужно определиться, что является ее входами, а что – выходами. Предполагается, что у нас уже есть таблица с обучающей выборкой. Обычно для ее подготовки пользуются методами очистки и трансформации данных – редактируются аномалии, заполняются или удаляются пропуски, устраняются дубликаты и противоречия, производится квантование и табличная замена, данные приводятся к скользящему окну, преобразуется формат данных.

Нормализация значений полей.

После того, как указаны входные и выходные поля, следует нормализация данных в обучающей выборке. Целью нормализации значений полей является преобразование данных к виду, наиболее подходящему для обработки алгоритмом. Для таких обработчиков как нейронная сеть, дерево решений, линейная модель прогнозирования данные, поступающие на вход, должны иметь числовой тип, а их значения должны быть распределены в определенном диапазоне. Нормализатор может преобразовать дискретные данные к набору уникальных индексов или значения, лежащие в произвольном диапазоне к диапазону $[0..1]$. Для нейросети доступны следующие виды нормализации полей.

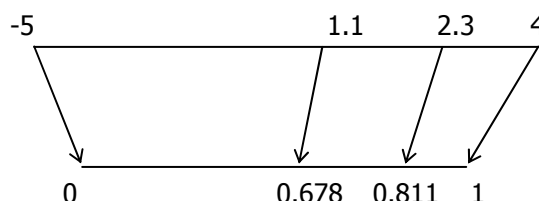
1. **Линейная нормализация.** Используется только для непрерывных числовых полей. Позволяет привести числа к диапазону $[\min..max]$, то есть минимальному числу из исходного диапазона будет соответствовать \min , а максимальному – max . Остальные значения распределятся между \min и max .
2. **Уникальные значения.** Используется для дискретных значений. Такими являются строки, числа или даты, заданные дискретно. Чтобы привести непрерывные числа в дискретные можно, например, воспользоваться обработкой «квантование». Так следует поступать для величин, для которых можно задать отношение порядка, то есть, если для двух любых дискретных значений можно указать, какое больше, а какое меньше. Тогда все значения необходимо расположить в порядке возрастания. Далее они нумеруются по порядку и значения заменяются их порядковым номером.
3. **Битовая маска.** Используется для дискретных значений. Этот вид нормализации следует использовать для величин, которые можно только сравнивать на равенство или неравенство, но нельзя сказать, какое больше, а какое меньше. Все значения заменяются порядковыми номерами, а номер рассматривается в двоичном виде или в виде маски из нулей и единиц. Тогда каждая позиция маски рассматривается как отдельное поле, содержащее ноль или единицу. К такому полю можно применить линейную нормализацию, то есть заменить ноль некоторым минимальным значением, а единицу – максимальным. После такой нормализации на вход нейросети будет подаваться не одно это поле, а столько полей, сколько разрядов в маске.

Пример нормализации полей.

1. **Линейная нормализация.** Приведем значения к диапазону $[0..1]$.

Поле до нормализации	Поле после нормализации
-5	0
2.3	0,81111
1.1	0,67778
4	1
3.5	0,94444

Графически такое преобразование можно представить так.



2. Уникальные значения.

Поле до нормализации	Поле после нормализации
Маленький	1
Средний	2
Большой	3
Огромный	4

3. Битовая маска. Заменяем значения битовой маской и приведем к диапазону [-1..1].

Поле до нормализации	Маска	Поля после нормализации	
Москва	00	-1	-1
Воронеж	01	-1	1
Рязань	10	1	-1
Тула	11	1	1

Кроме того, существует настройка нормализации полей по умолчанию. То есть, запустив обработчик, в данном случае нейронную сеть, этот этап можно пропустить. В этом случае нормализация будет произведена автоматически в зависимости от типов и видов полей:

- 1) вид дискретный – нормализация списком уникальных значений;
- 2) вид непрерывный
 - тип целый или дата – линейная нормализация в диапазоне [-1..1];
 - тип вещественный - линейная нормализация в диапазоне [0..1].

Настройка обучающей выборки.

После нормализации полей следует настроить обучающую выборку. Обучающую выборку разбивают на два множества – обучающее и тестовое.

Обучающее множество - включает записи (примеры), которые будут использоваться непосредственно для обучения сети, т.е. будут содержать входные и желаемые выходные (целевые) значения.

Тестовое множество - также включает записи (примеры), содержащие входные и желаемые выходные (целевые) значения, но используемое не для обучения сети, а для проверки результатов обучения.

Разбивать исходную выборку на эти множества можно двумя способами: либо по порядку, либо случайно. Если разбиение происходит по порядку, то тестовое множество выбирается либо из начала, либо из конца исходной выборки.

Настройка структуры нейросети.

Далее задаются параметры, определяющие структуру нейронной сети - количество скрытых слоев и нейронов в них, а также активационная функция нейронов.

Замечание. К выбору количества скрытых слоев и количества нейронов для каждого скрытого слоя нужно подходить осторожно. Хотя до сих пор не выработаны четкие критерии выбора, дать

некоторые общие рекомендации все же возможно. Считается, что задачу любой сложности можно решить при помощи двухслойной нейросети, поэтому конфигурация с количеством скрытых слоев, превышающих 2, вряд ли оправдана. Для решения многих задач вполне подойдет однослойная нейронная сеть. При выборе количества нейронов следует руководствоваться следующим правилом: «количество связей между нейронами должно быть значительно меньше количества примеров в обучающем множестве». Количество связей рассчитывается как связь каждого нейрона со всеми нейронами соседних слоев, включая связи на входном и выходном слоях. Слишком большое количество нейронов может привести к так называемому «переобучению» сети, когда она выдает хорошие результаты на примерах, входящих в обучающую выборку, но практически не работает на других примерах.

Обучение нейросети.

После настройки конфигурации сети следует выбрать алгоритм ее обучения.

Метод *обратного распространения ошибки* – итеративный градиентный алгоритм обучения, который используется с целью минимизации среднеквадратичного отклонения текущих значений выходов сети от требуемых. Одним из важнейших свойств алгоритма обратного распространения ошибки является высокая устойчивость, а, следовательно, надежность. Хотя нейронные сети, использующие алгоритм обратного распространения, являясь мощным инструментом поиска закономерностей, прогнозирования и качественного анализа, получили широкое распространение, им свойственны некоторые недостатки. К ним относится невысокая скорость сходимости (большое число требуемых итераций), что делает процесс обучения слишком долгим, и поэтому данный алгоритм оказывается неприменимым для широкого круга задач, требующих быстрого решения. Другие алгоритмы обучения нейросетей, хотя и работают быстрее, в большинстве случаев, обладают меньшей устойчивостью.

Для алгоритма обратного распространения ошибки нужно указать два параметра:

- «Скорость обучения» - определяет величину шага при итерационной коррекции весов в нейронной сети (рекомендуется в интервале 0...1). При большой величине шага, сходимость будет более быстрой, но имеется опасность «перепрыгнуть» через решение. С другой стороны, при малой величине шага, обучение потребует слишком многих итераций. На практике величина шага берется пропорциональной крутизне склона, так, что алгоритм замедляется вблизи минимума. Правильный выбор скорости обучения зависит от конкретной задачи и обычно делается опытным путем;
- «Момент» - задается в интервале 0...1. Рекомендуемое значение 0.9 ± 0.1 .

Метод *Resilient Propagation (Rprop)* - эластичное распространение. Алгоритм использует так называемое «обучение по эпохам», когда коррекция весов происходит после предъявления сети всех примеров из обучающей выборки. Преимущество данного метода заключается в том, что он обеспечивает сходимость, а, следовательно, и обучение сети в 4-5 раз быстрее, чем алгоритм обратного распространения.

Для алгоритма Resilient Propagation указываются параметры:

- «Шаг спуска» - коэффициент увеличения скорости обучения, который определяет шаг увеличения скорости обучения при недостижении алгоритмом оптимального результата;
- «Шаг подъема» - коэффициент уменьшения скорости обучения. Задается шаг уменьшения скорости обучения в случае пропуска алгоритмом оптимального результата.

Далее необходимо задать условия, при выполнении которого обучение будет прекращено.

- «Считать пример распознанным, если ошибка меньше» - критерием останова в данном случае является условие, что рассогласование между эталонным и реальным выходом сети становится меньше заданного значения.
- «По достижении эпохи» - установка данного режима позволяет задать число эпох (циклов обучения) по достижении которого обучение останавливается независимо от величины ошибки. Если флажок сброшен, то обучение будет продолжаться, пока ошибка не станет меньше заданного значения, но при этом есть вероятность заикливания, когда ошибка никогда не будет достигнута. Поэтому, все-таки, желательно установить флажок по «Достижению эпохи», чтобы алгоритм был остановлен в любом случае.

Теперь все готово к процессу обучения сети. В начале все веса нейросети инициализируются случайными значениями. После обучения эти веса принимают определенные значения. Обучение

может с большой долей вероятности считаться успешным, если процент распознанных примеров на обучающем и тестовом множествах достаточно велик (близок к 100%).

Пример.

Рассмотрим пример построения системы оценки кредитоспособности физического лица. Предположим, что эксперты определили основные факторы, определяющие кредитоспособность. Ими оказались: возраст, образование, площадь квартиры, наличие автомобиля, длительность проживания в данном регионе. В организации была накоплена статистика возвратов или невозвратов взятых кредитов. Эта статистика представлена таблицей.

Сумма кредита	Возраст	Образование	Площадь квартиры	Автомобиль	Срок проживания	Давать кредит
7000	37	Специальное	37	отечественная	22	Да
7500	38	Среднее	29	импортная	12	Да
14500	60	Высшее	34	Нет автомобиля	30	Нет
15000	28	Специальное	14	отечественная	21	Да
32000	59	Специальное	53	отечественная	29	Да
11500	25	Специальное	28	отечественная	9	Да
5000	57	Специальное	18	отечественная	34	Да
61500	29	Высшее	26	Нет автомобиля	18	Нет
13500	37	Специальное	46	отечественная	28	Нет
25000	36	Специальное	20	Нет автомобиля	21	Нет
25500	68	Высшее	45	отечественная	30	Нет
...

Это обучающая выборка.

Теперь необходимо нормализовать поля. Поля «Сумма кредита», «Возраст», «Площадь квартиры» и «Длительность проживания» – непрерывные значения, которые преобразуем к интервалу $[-1..1]$. Образование представлено тремя уникальными значениями, которые можно сравнивать на большее или меньшее, а точнее лучшее или худшее. То есть образование можно упорядочить так: среднее, специальное, высшее. Значения поля с наличием автомобиля упорядочить нельзя. Его нужно преобразовать к битовой маске. Для кодирования трех значений требуется два бита. Следовательно, это поле будет разбито на два.

Наличие автомобиля	Первый бит маски	Второй бит маски
Импортная	0	0
Отечественная	0	1
нет автомобиля	1	0

На этом нормализация закончена.

Обучающую выборку разобьем на обучающее и тестовое множества так, как программа предлагает это сделать по умолчанию. Т.е. в обучающее множество попадут случайные 95 процентов записей, а остальные 5 процентов – в тестовое.

Конфигурация сети будет такой. Во входном слое – 7 нейронов, то есть по одному нейрону на один вход (в обучающей выборке 6 столбцов, но столбец «Автомобиль» представлен битовой маской из двух бит, для каждого из которых создан новый вход). Сделаем один скрытый слой с двумя нейронами. В выходном слое будет один нейрон, на выходе которого будет решение о выдаче кредита.

Выберем алгоритм обучения сети – Resilient Propagation с настройками по умолчанию. Условие окончания обучения оставим без изменения.

Обученную таким образом нейросеть можно использовать для принятия решения о выдаче кредита физическому лицу. Это можно сделать, применяя анализ «что-если». Для его включения нужно выбрать визуализатор «Что-если». Тогда откроется форма, представленная на рисунке.

Поле	Значение
Входные	
9.0 Сумма кредита	7000
9.0 Возраст	37
ab Образование	специальное
9.0 Площадь квартиры	37
ab Автомобиль	отечественная
9.0 Срок проживания	22
Выходные	
ab Давать кредит	Да

После изменения в этой таблице входных полей система сама принимает решение о выдаче кредита и в поле «Давать кредит» проставляет либо «Да», либо «Нет». Столбцы «Минимум» и «Максимум» определяют диапазон значений, на которых обучалась нейросеть. Следует придерживаться этих ограничений, хотя и возможно взять значения немного выходящие за границы диапазона.

Кроме такой таблицы анализ «что-если» содержит диаграмму, на которой отображается зависимость выходного поля от одного из входных полей при фиксированных значениях остальных полей. Например, требуется узнать, на какую сумму кредита может рассчитывать человек, обладающий определенными характеристиками. Это можно определить по диаграмме.



То есть, человек в возрасте 37 лет со специальным образованием, имеющий квартиру площадью 37 кв. м, отечественный автомобиль и проживающий в данной местности 22 года может рассчитывать на сумму кредита не больше 20000.

Качество построенной модели можно определить по таблице сопряженности, которая является одним из визуализаторов.

Давать кредит	Классифицировано		
	Да	Нет	Итого
Фактически			
Да	50	9	59
Нет	27	63	90
Итого	77	72	149

Чем больше правильно классифицированных записей, тем лучше построенная модель.

Линейная регрессия

В результате работы данного компонента строится линейная модель данных. Применяется следующий алгоритм построения модели.

Пусть имеется набор входных значений X_i , где $i=1...n$, т.е. $\{X_1, X_2, ..., X_n\}$. Тогда можно указать такой набор выходных значений Y_j ($j=1...m$), который будет соответствовать линейной комбинации входных значений с коэффициентами a_i ($i=1...n$):

$$[1, X_1, X_2, ..., X_n] [a_0, a_1, a_2, ..., a_n] = [Y_1, Y_2, ..., Y_m]$$

Если для простоты предположить, что выходное значение одно, то можно записать $a_0 + X_1 a_1 + X_2 a_2 + ... + X_n a_n = Y$. Таким образом, задача сводится к подбору коэффициентов a_i . Подбор производится путем итеративного обучения модели, когда на входные поля подается обучающий набор данных, а на выход - целевое значение Y . Целью обучения является получение такого набора коэффициентов линейного преобразования a_i , при котором ошибка рассогласования реального Y и рассчитанного Y' значений выходов будет минимальной.

Линейная регрессия может быть получена с помощью нейросетей. Для этого нужно создать нейросеть с одним внутренним слоем, состоящим из одного нейрона и с линейной функцией активации. С помощью линейной регрессии модель может быть построена в несколько раз быстрее. Поэтому ее эффективней применять на больших наборах данных. Однако нужно помнить, что она предназначена для поиска линейных зависимостей в данных. Если же зависимости нелинейные, то модель с использованием линейной регрессии может быть не построена вообще. Это будет сразу видно на диаграмме рассеяния – прогнозные значения величины будут сильно разбросаны относительно действительных значений. В этом случае нужно использовать нейронные сети.

Подготовка и настройка обучающей выборки, настройка назначений полей и их нормализация производятся так же, как и для нейронных сетей.

Пример.

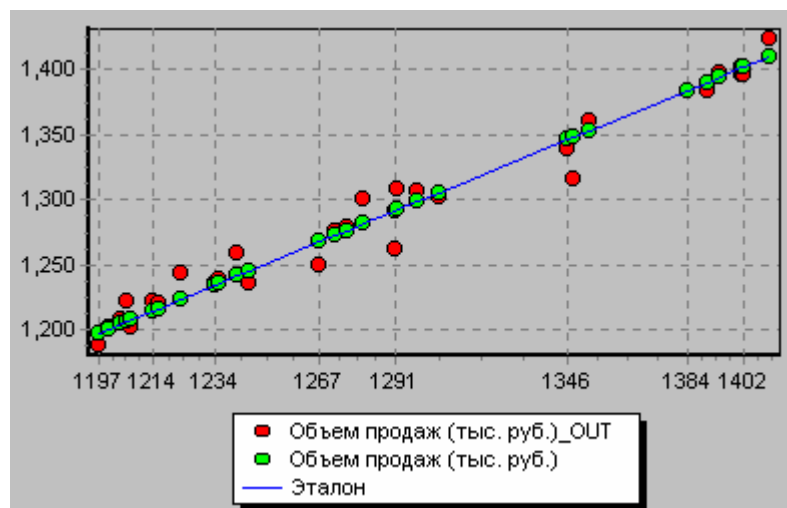
Есть данные о продажах, представленные таблицей.

Первый день месяца	Объем продаж (тыс. руб.)
...	...
01.01.2004	1000
01.02.2004	1160
01.03.2004	1210
01.04.2004	1130
01.05.2004	1250
01.06.2004	1300

Пусть, например, необходимо составить прогноз объемов продаж на следующий месяц, основываясь на продажах за текущий и два предыдущих месяца. Для этого преобразуем таблицу к скользящему окну.

Первый день месяца	Объем продаж три месяца назад (X-3)	Объем продаж два месяца назад (X-2)	Объем продаж месяц назад (X-1)	Объем продаж в текущем месяце (X)
...
01.04.2004	1000	1160	1210	1130
01.05.2004	1160	1210	1130	1250
01.06.2004	1210	1130	1250	1300

Входными полями у модели являются X-3, X-2, X-1, а выходным – X. Способ разбиения множеств оставим по умолчанию и перейдем сразу к построению модели. Для оценки качества построенной модели воспользуемся диаграммой рассеяния.



Судя по диаграмме, разброс между эталонными значениями выходного поля и значениями, рассчитанными моделью, достаточно невелик. Из этого можно сделать следующий вывод. Временной ряд хорошо укладывается в линейную модель и, следовательно, на основании этой модели можно строить прогноз на будущие периоды времени.

Прогнозирование

Прогнозирование позволяет получать предсказание значений временного ряда на число отсчетов, соответствующее заданному горизонту прогнозирования. Алгоритм прогнозирования работает следующим образом. Пусть в результате преобразования методом скользящего окна была получена последовательность временных отсчетов:

$$X(-n), \dots, X(-2), X(-1), X$$

где X – текущее значение. Прогноз на $X(+1)$ строится на основании построенной модели. Чтобы построить прогноз для значения $X(+2)$ нужно сдвинуть всю последовательность на один отсчет влево, чтобы ранее сделанный прогноз $X(+1)$ тоже вошел в число исходных значений. Затем снова будет запущен алгоритм расчета прогнозируемого значения - $X(+2)$ будет рассчитан с учетом $X(+1)$ и так далее в соответствии с заданным горизонтом прогноза.

Для настройки алгоритма прогнозирования необходимо задать горизонт прогноза, а также поля таблицы, которые необходимо подавать на вход модели для построения прогноза (для вычисления выходного поля модели).

Пример.

Продолжим предыдущий пример. На основании построенной модели временного ряда спрогнозируем продажи на следующий месяц.

Для этого на входы модели необходимо подать значения о продажах за 3 месяца – 01.04.2004, 01.05.2005, 01.06.2004. А это есть поля $X-2$, $X-1$ и X . Следовательно, необходимо сопоставить поля, как показано в таблице.

Столбец	При очередном шаге брать значения из
X-3	X-2
X-2	X-1
X-1	X
X	

После применения алгоритма прогноза будет получена таблица.

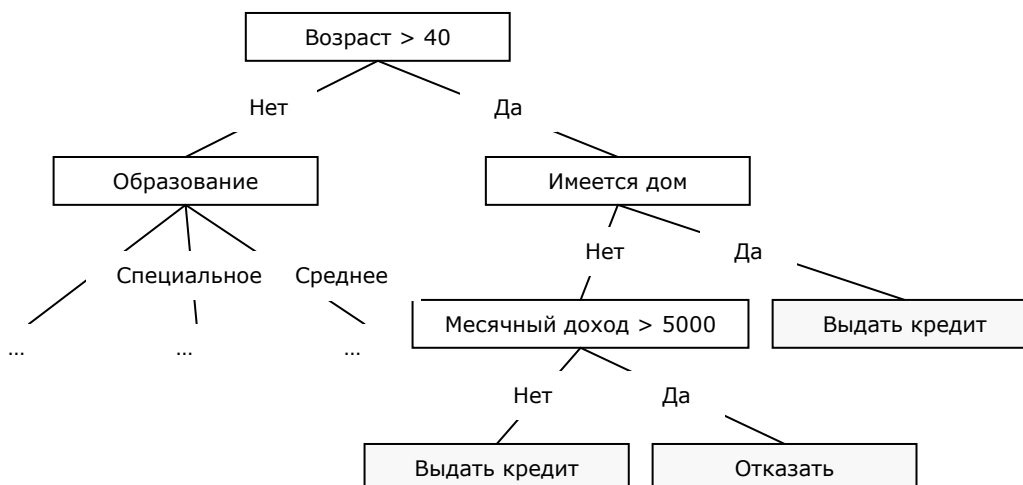
Первый день месяца	X-3	X-2	X-1	X	Шаг прогноза
...	
01.03.2004	1000	1160	1210	1130	
01.04.2004	1160	1210	1130	1250	
01.05.2004	1210	1130	1250	1300	
01.05.2004	1130	1250	1300	1400	1

Столбец X показывает историю продаж, включая спрогнозированное значение на следующий месяц. Для получения прогнозного значения 1400 на вход модели были поданы значения: 1130, 1250, 1300. Если задать глубину прогноза 2, то для получения значения на второй шаг прогноза будут использоваться значения: 1250, 1300, 1400, то есть уже будет участвовать прогнозное значение, полученное на предыдущем шаге. И так далее на любое число шагов прогноза. В связи с тем, что для получения прогноза на большое число шагов используются не реальные данные, а вычисленные моделью, ошибка такого прогноза может быть очень велика. Поэтому при построении прогноза не следует заглядывать слишком далеко вперед, так как с увеличением погрешности ценность полученного прогноза очень быстро падает.

Деревья решений

Деревья решений (decision trees) являются одним из наиболее популярных подходов к решению задач добычи данных. Они создают иерархическую структуру классифицирующих правил типа «ЕСЛИ...ТО...» (if-then), имеющую вид дерева. Чтобы принять решение, к какому классу следует отнести некоторый объект или ситуацию, требуется ответить на вопросы, стоящие в узлах этого дерева, начиная с его корня. Вопросы имеют вид «значение параметра A больше B?». Если ответ положительный, осуществляется переход к правому узлу следующего уровня. Затем снова следует вопрос, связанный с соответствующим узлом и т. д. Приведенный пример иллюстрирует работу так называемых бинарных деревьев решений, в каждом узле которых ветвление производится по двум направлениям (т. е. на вопрос, заданный в узле, имеется только два варианта ответов, например, Да или Нет). Однако, в общем случае, ответов а, следовательно, ветвей, выходящих из узла, может быть больше.

Дерево решений состоит из узлов – где производится проверка условия, и листьев – конечных узлов дерева, указывающих на класс (узлов решения).



Область применения деревьев решений в настоящее время весьма широка, но все задачи, решаемые этим аппаратом, могут быть объединены в три класса.

1. Описание данных. Деревья решений позволяют хранить информацию о данных в компактной форме. Вместо громоздких массивов данных можно хранить дерево решений, которое содержит точное описание объектов.
2. Классификация. Деревья решений отлично справляются с задачами классификации, т.е. отнесения объектов к одному из заранее известных классов.
3. Регрессия. Если целевая переменная является непрерывной, деревья решений позволяют установить зависимость целевой переменной от независимых (входных) переменных.

Например, к этому классу относятся задачи численного прогнозирования (предсказания значений целевой переменной).

Подготовка обучающей выборки.

Для построения дерева решений готовится обучающая выборка так же, как это описано для нейросети. Разница заключается в том, что выходное поле для дерева решений может быть только одно и только дискретно.

Нормализация значений полей.

Для полей, подаваемых на входы и выход дерева решений, также задается нормализация. Можно задать либо линейную нормализацию, либо нормализацию уникальными значениями (описание в разделе по нейросети).

Настройка обучающей выборки.

Настройка обучающей выборки такая же, как для нейросети.

Обучение дерева решений.

Параметры обучения дерева решений следующие:

- *Минимальное количество примеров, при котором будет создан новый узел.* Задается минимальное количество примеров, которое возможно в узле. Если примеров, которые попадают в данный узел, будет меньше заданного - узел считается листом (т.е. дальнейшее ветвление прекращается). Чем больше этот параметр, тем менее ветвистым получается дерево.
- *Строить дерево с более достоверными правилами в ущерб сложности.* Включает специальный алгоритм, который, усложняя структуру дерева, увеличивает достоверность результатов классификации. При этом дерево получается, как правило, более ветвистым.
- *Уровень доверия, используемый при отсечении узлов дерева.* Значение этого параметра задается в процентах и должно лежать в пределах от 0 до 100. Чем больше уровень доверия, тем более ветвистым получается дерево, и, соответственно, чем меньше уровень доверия, тем больше узлов будет отсечено при его построении.

Качество построенного дерева после обучения можно оценить по нескольким параметрам. Во-первых, это число распознанных примеров в обучающем и тестовом наборах данных. Чем выше это число, тем качественнее построенное дерево. Во-вторых, это количество узлов в дереве. При очень большом их числе дерево становится трудным для восприятия. Это также означает очень слабую зависимость выходного поля от входных полей. Каждое правило характеризуется поддержкой и достоверностью.

Поддержка – общее количество примеров классифицированных данным узлом дерева.

Достоверность – количество правильно классифицированных данным узлом примеров.

Пример.

Продолжим пример, рассмотренный в пункте, посвященном нейронным сетям. Всех заемщиков можно разделить на два класса – кредитоспособных и некредитоспособных. Очевидно, существуют некоторые правила отнесения заемщиков к тому или иному классу. Но при достаточно большом числе выбранных характеристик вручную практически невозможно определить эти правила. Решить эту задачу позволяют деревья решений.

Обучающая выборка, а так же правила получения обучающего и тестового множества будут теми же, что и в примере с нейросетями. Нормализацию полей оставим заданной по умолчанию в программе. При построении правил зададим минимальное количество примеров, при котором будет создан новый узел равным 10. Будем строить дерево с более достоверными правилами в ущерб сложности.

Полученное дерево решений содержит 20 узлов и 11 правил.



Такое дерево содержит в себе правила, следуя которым можно отнести заемщика в одну из групп риска и сделать вывод о выдаче кредита. Правила читаются с узлов, расположенных правее. Например, если сумма кредита меньше 24000 и срок проживания меньше 11, тогда выдать кредит. Следует заметить, что характеристики, лежащие ближе к вершине дерева, то есть левее, являются более значимыми.

Построенные правила можно также просмотреть в виде списка правил.

N	Условие	Решение (Давать кредит)	Поддержка		Достоверность	
			%	Кол-во	%	Кол-во
1	Сумма кредита < 24000 И Срок проживания < 11	Да	10.56	15	86.67	13
2	Сумма кредита < 24000 И Срок проживания >= 11 И Сумма кредита < 13750 И Сумма кредита < 6250	Нет	9.86	14	57.14	8
3	Сумма кредита < 24000 И Срок проживания >= 11 И Сумма кредита < 13750 И Сумма кредита >= 6250 И Сумма кредита < 9250	Да	7.75	11	100	11

Как и для нейросети, общее количество правильно классифицированных примеров можно посмотреть в таблице сопряженности.

Карты Кохонена

Самоорганизующиеся карты (карты Кохонена) могут использоваться для решения таких задач как моделирование, прогнозирование, поиск закономерностей в больших массивах данных, выявление наборов независимых признаков и сжатие информации.

Алгоритм функционирования самоорганизующихся карт (Self Organizing Maps - SOM) представляет собой один из вариантов кластеризации многомерных векторов - алгоритм проецирования с сохранением топологического подобию.

Примером таких алгоритмов может служить алгоритм k-ближайших средних (k-means). Важным отличием алгоритма SOM является то, что в нем все нейроны (узлы, центры классов) упорядочены в некоторую структуру (обычно двумерную сетку). При этом в ходе обучения модифицируется не только нейрон-победитель (нейрон карты, который в наибольшей степени соответствует вектору входов и определяет, к какому классу относится пример), но и его соседи, хотя и в меньшей степени. За счет этого SOM можно считать одним из методов проецирования многомерного

пространства в пространство с более низкой размерностью. При использовании этого алгоритма, вектора, близкие в исходном пространстве, оказываются рядом и на полученной карте.

SOM подразумевает использование упорядоченной структуры нейронов. Обычно используются одно- и двумерные сетки. При этом каждый нейрон представляет собой n -мерный вектор-столбец, где n определяется размерностью исходного пространства (размерностью входных векторов). Применение одно- и двумерных сеток связано с тем, что возникают проблемы при отображении пространственных структур большей размерности (при этом опять возникают проблемы с понижением размерности до двумерной, представимой на мониторе).

Обычно, нейроны располагаются в узлах двумерной сетки с прямоугольными или шестиугольными ячейками. При этом, как было сказано выше, нейроны также взаимодействуют друг с другом. Величина этого взаимодействия определяется расстоянием между нейронами на карте.

При реализации алгоритма SOM заранее задается конфигурация сетки (прямоугольная или шестиугольная), а также количество нейронов в сети. Некоторые источники рекомендуют использовать максимально возможное количество нейронов в карте. При этом начальный радиус обучения (neighbourhood в англоязычной литературе) в значительной степени влияет на способность обобщения при помощи полученной карты. В случае, когда количество узлов карты превышает количество примеров в обучающей выборке, успех использования алгоритма в большой степени зависит от подходящего выбора начального радиуса обучения. Однако, в случае, когда размер карты составляет десятки тысяч нейронов, время, требуемое на обучение карты, обычно бывает слишком велико для решения практических задач. Таким образом, необходимо достигать допустимый компромисс при выборе количества узлов.

Подготовка обучающей выборки.

Отличием в подготовке обучающей выборки в этом алгоритме заключается в том, что выходные поля в такой выборке могут отсутствовать совсем. Даже если в обучающей выборке будут присутствовать выходные поля, они не будут участвовать при обучении нейросети. Однако они будут участвовать при отображении карт.

Нормализация значений полей.

Нормализация полей здесь такая же, как для деревьев решений.

Настройка обучающей выборки.

Обучающая выборка настраивается так же, как и для нейросети и дерева решений.

Обучение.

Перед началом обучения карты необходимо проинициализировать весовые коэффициенты нейронов. Удачно выбранный способ инициализации может существенно ускорить обучение и привести к получению более качественных результатов.

Существуют три способа инициирования начальных весов.

- инициализация случайными значениями, когда всем весам даются малые случайные величины;
- инициализация примерами, когда в качестве начальных значений задаются значения случайно выбранных примеров из обучающей выборки;
- линейная инициализация. В этом случае веса инициализируются значениями векторов, линейно упорядоченных вдоль линейного подпространства, проходящего между двумя главными собственными векторами исходного набора данных.

Визуализация.

Полученную в результате обучения карту можно представить в виде слоеного пирога, каждый слой которого представляет собой раскраску, порожденную одной из компонент исходных данных. Полученный набор раскрасок может использоваться для анализа закономерностей, имеющихся между компонентами набора данных. После формирования карты, получается набор узлов, который можно отобразить в виде двумерной картинки. При этом каждому узлу карты можно поставить в соответствие участок на рисунке (четырёх или шестиугольный), координаты которого определяются координатами соответствующего узла в решетке. Теперь для визуализации остается только определить цвет ячеек этой картинки. Для этого и используются значения компонент. Самый простой вариант - использование градаций серого. В этом случае ячейки, соответствующие

узлам карты, в которые попали элементы с минимальными значениями компонента или не попало вообще ни одной записи, будут изображены черным цветом, а ячейки, в которые попали записи с максимальными значениями такого компонента, будут соответствовать ячейке белого цвета. В принципе можно использовать любую градиентную палитру для раскраски.

Полученные раскраски в совокупности образуют атлас, отображающий расположение компонент, связи между ними, а также относительное расположение различных значений компонент.

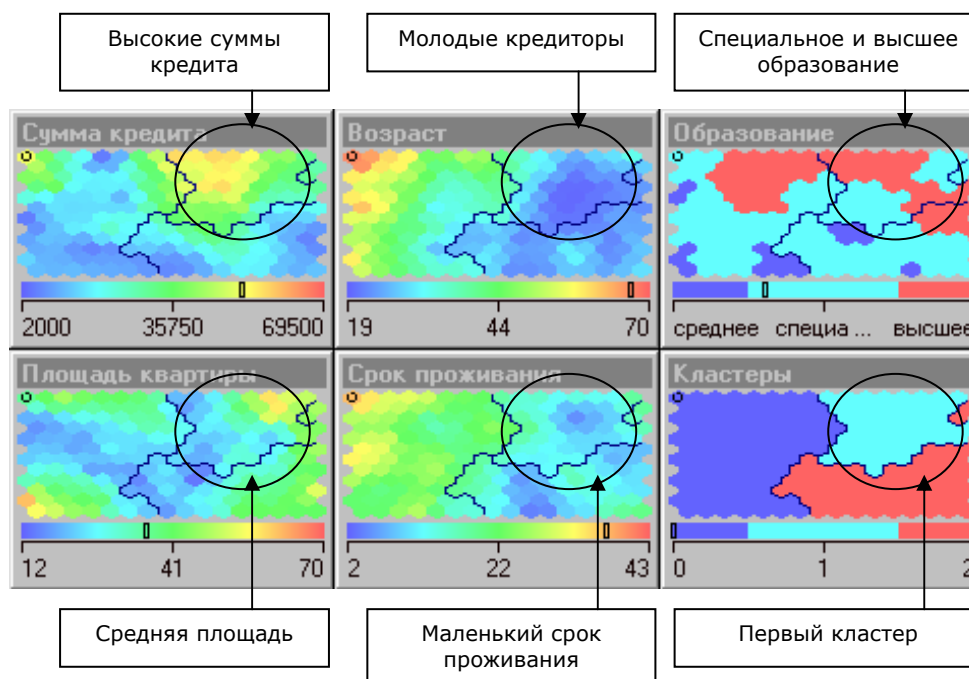
Пример.

Продолжим пример с кредитованием физических лиц. С помощью самоорганизующихся карт Кохонена можно посмотреть зависимости между различными характеристиками заемщиков и выделить сегменты заемщиков, объединив их по схожим признакам.

Обучающей выборкой будет та же, что и для нейросетей и деревьев решений. Но поле «Автомобиль» использовать не будем. Это нецелесообразно, так как данные, подаваемые на вход карт Кохонена должны быть такими, чтобы между ними можно было вычислить расстояние или, по крайней мере, была возможность расположить их в порядке возрастания или убывания. Значения же «отечественная», «импортная» и «нет автомобиля» сравнивать между собой нельзя.

Нормализация полей будет такая же, как для деревьев решений. Разбиение обучающей выборки на два множества оставим по умолчанию, так же, как и остальные настройки.

Результаты работы алгоритма отображаются на картах. Каждому входному полю соответствует своя карта.



К примеру, в один и тот же кластер были сгруппированы молодые заемщики с маленьким сроком проживания в данной местности, специальным или средним образованием, средней жилплощадью, берущие высокие суммы кредита. В результате кластеризации заемщики со схожими характеристиками попадут в один кластер, и поэтому для них можно применять одинаковые правила выдачи кредита, т.е. для каждого кластера определить, стоит ли выдавать кредит его представителям.

Ассоциативные правила

Ассоциативные правила позволяют находить закономерности между связанными событиями. Примером такого правила, служит утверждение, что покупатель, приобретающий 'Хлеб', приобретет и 'Молоко' с вероятностью 75%. Впервые эта задача была предложена для поиска ассоциативных правил для нахождения типичных шаблонов покупок, совершаемых в супермаркетах, поэтому иногда ее еще называют анализом рыночной корзины (market basket analysis).

Транзакция – это множество событий, произошедших одновременно. Пусть имеется база данных, состоящая из покупательских транзакций. Каждая транзакция – это набор товаров, купленных покупателем за один визит. Такую транзакцию еще называют рыночной корзиной.

После определения понятия транзакция, можно перейти к определению ассоциативного правила. Пусть имеется список транзакций. Необходимо найти закономерности между этими событиями. Как в условии, так и вследствие правила должны находиться элементы транзакций.

Пусть $I = \{i_1, i_2, \dots, i_n\}$ – множество элементов, входящих в транзакции. D – множество транзакций.

Ассоциативным правилом называется импликация $X \Rightarrow Y$ (читается « X дает Y » или «из X следует Y »), где $X \subset I$, $Y \subset I$ и $X \cap Y = \emptyset$.

Правило $X \Rightarrow Y$ имеет поддержку s (support), если $s\%$ транзакций из D , содержат $X \cup Y$, $\text{supp}(X \Rightarrow Y) = \text{supp}(X \cup Y)$.

Достоверность правила показывает, какова вероятность того, что из X следует Y . Правило $X \Rightarrow Y$ справедливо с достоверностью (confidence) c , если $c\%$ транзакций из D , содержащих X , также содержат Y , $\text{conf}(X \Rightarrow Y) = \text{supp}(X \cup Y) / \text{supp}(X)$.

Покажем на конкретном примере: '75% транзакций, содержащих хлеб, также содержат молоко. 3% от общего числа всех транзакций содержат оба товара'. 75% – это достоверность (confidence) правила, 3% это поддержка (support), или «Если 'Хлеб', то 'Молоко'» с вероятностью 75%.

Другими словами, целью анализа является установление следующих зависимостей: если в транзакции встретился некоторый набор элементов X , то на основании этого можно сделать вывод о том, что другой набор элементов Y также же должен появиться в этой транзакции. Установление таких зависимостей дает нам возможность находить очень простые и интуитивно понятные правила.

Алгоритмы поиска ассоциативных правил предназначены для нахождения всех правил $X \Rightarrow Y$, причем поддержка и достоверность этих правил должны быть выше некоторых заранее определенных порогов, называемых соответственно минимальной поддержкой (minsupport) и минимальной достоверностью (minconfidence). Аналогично, поддержка и достоверность ограничиваются сверху порогами максимальной поддержки (maxsupport) и максимальной достоверности (maxconfidence). В результате получаются два окна, в которые должны попасть поддержка и достоверность правила, чтобы оно было предъявлено аналитику.

Значения для параметров минимальная (максимальная) поддержка и минимальная (максимальная) достоверность выбираются таким образом, чтобы ограничить количество найденных правил. Если поддержка имеет большое значение, то алгоритмы будут находить правила, хорошо известные аналитикам или настолько очевидные, что нет никакого смысла проводить такой анализ. С другой стороны, низкое значение поддержки ведет к генерации огромного количества правил, что, конечно, требует существенных вычислительных ресурсов. Большинство интересных правил находится именно при низком значении порога поддержки, хотя слишком низкое значение поддержки ведет к генерации статистически необоснованных правил. Ассоциативные правила с высокой поддержкой могут применяться для формализации хорошо известных правил, например, в автоматизированных системах для управления процессами или персоналом. Надо отметить, что понятия «высокая» и «низкая» поддержка или достоверность очень сильно зависят от предметной области. Например, в торговле 1% вероятности совместного приобретения хлеба и молока не значит ничего, в то время как вероятность в 1% отказа двигателя самолета совершенно неприемлема, и такое правило становится чрезвычайно важным.

Поиск ассоциативных правил совсем не тривиальная задача, как может показаться на первый взгляд. Одна из проблем – алгоритмическая сложность при нахождении часто встречающихся наборов элементов, т.к. с ростом числа элементов экспоненциально растет число потенциальных наборов элементов.

Обычные ассоциативные правила – это правила, в которых как в условии, так и в следствии присутствуют только элементы транзакций и при вычислении которых используется только информация о том, присутствует ли элемент в транзакции или нет. Фактически все приведенные выше примеры относятся к обычным ассоциативным правилам.

Для поиска обычных ассоциативных правил в программе служит обработка «Ассоциативные правила».

Настройки.

Для начала необходимо указать, что является идентификатором (ID) транзакции, а что элементом транзакции. Например, идентификатор транзакции – это номер чека или код накладной. А элемент – это наименование товара в чеке.

Затем следует настройка параметров поиска правил. Всего четыре параметра:

- минимальная и максимальная поддержка. Ассоциативные правила ищутся только в некотором множестве всех транзакций. Для того чтобы транзакция вошла в это множество, она должна встретиться в исходной выборке количество раз, большее минимальной поддержки и меньше максимальной. Например, минимальная поддержка равна 1%, а максимальная – 20%. Количество элементов «Хлеб» и «Молоко» столбца «Товар» с одинаковым значением столбца «Номер чека» встречаются в 5% всех транзакций (номеров чека). Тогда эти две строки войдут в искомое множество.
- минимальная и максимальная достоверность. Это процентное отношение количества транзакций, содержащих все элементы, которые входят в правило, к количеству транзакций, содержащих элементы, которые входят в условие. Если транзакция – это заказ, а элемент – товар, то достоверность характеризует, насколько часто покупаются товары, входящие в следствие, если заказ содержит товары, вошедшие во всё правило.

Пример.

Дана таблица транзакций.

Транзакция (номер чека)	Элемент (товар)
1	Булочка
4	Спички
2	Сигареты
2	Зажигалка
3	Молоко
3	Кефир
3	Булочка
1	Кефир
4	Сигареты

Список транзакций будет выглядеть так:

Номер транзакции	Элементы, вошедшие в транзакцию
1	Булочка, Кефир
2	Сигареты, Зажигалка
3	Молоко, Кефир, Булочка
4	Спички, Сигареты

Всего исходные данные состоят из 4 транзакций. Полный список множеств для поиска правил выглядит так:

Множество элементов	Встречается раз в списке транзакций (количество)	Встречается раз в списке транзакций (%)
Булочка	2	50
Кефир	2	50
Сигареты	2	50
Зажигалка	1	25
Молоко	1	25
Спички	1	25
Булочка и кефир	2	50
Сигареты и зажигалка	1	25
Молоко и кефир	1	25
Молоко и булочка	1	25
Спички и сигареты	1	25

Если установить минимальную поддержку 30%, а максимальную – 60%, То останется только часть списка множеств (часто встречающиеся множества):

Множество элементов	Встречается раз в списке транзакций (количество)	Встречается раз в списке транзакций (%)
Булочка	2	50
Кефир	2	50
Сигареты	2	50
Булочка и кефир	2	50

Правила будут искажаться именно в этом последнем списке часто встречающихся множеств. Первые три множества в таблице одноэлементные, а последнее – двухэлементное. Ассоциативное правило можно построить только на основе 2-х и более элементного множества. Соответственно, если будут найдены только одноэлементные множества, то количество ассоциативных правил будет равно нулю. В этом случае следует уменьшить минимальную поддержку и/или увеличить максимальную. Тогда список множеств будет увеличен и, возможно, в него попадут двух и более элементные множества.

Выявление действительно интересных правил – это одна из главных подзадач, при вычислении ассоциативных зависимостей. Для того чтобы получить действительно интересные зависимости, нужно разобраться с несколькими эмпирическими правилами:

1. Уменьшение минимальной поддержки приводит к тому, что увеличивается количество потенциально интересных правил, однако это требует существенных вычислительных ресурсов. Одним из ограничений уменьшения порога минимальной поддержки является то, что слишком маленькая поддержка правила делает его статистически необоснованным.
2. Уменьшение порога достоверности также приводит к увеличению количества правил. Значение минимальной достоверности также не должно быть слишком маленьким, так как ценность правила с достоверностью 5% настолько мала, что это правило и правилом считать нельзя.
3. Правило со слишком большой поддержкой с точки зрения статистики представляет собой большую ценность, но, с практической точки зрения, это, скорее всего, означает то, что, либо правило всем известно, либо товары, присутствующие в нем, являются лидерами продаж, откуда следует их низкая практическая ценность.
4. Правило со слишком большой достоверностью практической ценности в контексте решаемой задачи не имеет, т.к. товары, входящие в следствие, покупатель, скорее всего, уже купил.

Если значение верхнего предела поддержки имеет слишком большое значение, то в правилах основную часть будут составлять товары – лидеры продаж. При таком раскладе не представляется возможным уменьшить минимальный порог поддержки до того значения, при котором могут появляться интересные правила. Причиной тому является просто огромное число правил, и, как следствие, нехватка системных ресурсов. Причем получаемые правила процентов на 95 содержат товары – лидеры продаж.

Варьируя верхним и нижним пределами поддержки, можно избавиться от очевидных и неинтересных закономерностей. Как следствие, правила, генерируемые алгоритмом, принимают приближенный к реальности вид.

При большом ассортименте товара важно отобразить построенные правила в удобном виде. Для этого в Deductor Studio служат два визуализатора: «Дерево правил» и «Что-если».

Дерево правил – это всегда двухуровневое дерево. Оно может быть построено либо по условию, либо по следствию. При построении дерева правил по условию, на первом (верхнем) уровне находятся узлы с условиями, а на втором уровне – узлы со следствием.

Пример.

<div> <div>Шпаклевки</div> <div>Эмали</div> <div>Герметики</div> <div>Клей - ж.гвозд</div> <div>Пена монтажн</div> <div>Герметики И К</div> <div>Герметики И Г</div> <div>Клей - ж.гвозд</div> <div>Грунтовки</div> </div>	Количество правил: 6; Условие: Эмали			
	Следствие	Поддержка		Достоверность, %
		N	%	
Герметики	25	2.48	54.30	
Клей - ж.гвозди	24	2.38	52.20	
Пена монтажная	25	2.48	54.30	
Герметики И Клей - ж.гвозди	21	2.08	45.70	
Герметики И Пена монтажная	21	2.08	45.70	
Клей - ж.гвозди И Пена монтажная	21	2.08	45.70	

В этом примере узлы «Шпаклевки», «Эмали», «Грунтовки» находятся на верхнем уровне дерева и представляют собой условия. А «Герметики», «Клей – ж.гвозди» и т.д. – следствия. Это означает, что человек, купивший эмаль купит еще и герметик с достоверностью 54.3%, клей с достоверностью 52.2% и т.д. В окне слева расположен список со следствиями для конкретного узла с условием. Для каждого следствия указана поддержка и достоверность. Например, в исходной выборке данных и эмаль, и герметик встретились одновременно в 25 транзакциях (чеках).

Второй вариант дерева правил – дерево, построенное по следствию. Здесь на первом уровне располагаются узлы со следствием.

<div> <div>Панели пластиков</div> <div>Герметики</div> <div>Затирки</div> <div>Клей - ж.гвозд</div> <div>Краски водоз</div> <div>Очиститель пе</div> <div>Пена монтажн</div> <div>Пена професс</div> <div>Пигменты</div> </div>	Количество правил: 77; Следствие: Герметики			
	Условие	Поддержка		Достоверность, %
		N	%	
Пигменты	28	2.78	42.40	
Пистолеты для герметиков	10	0.99	66.70	
Решетки радиаторные	16	1.59	44.40	
Уплотнители	27	2.68	42.90	
Шпаклевки	11	1.09	42.30	
Эмали	25	2.48	54.30	




Например, для того, чтобы человек приобрел герметик, он должен купить хотя бы один предмет из следующего списка: пигменты, пистолеты для герметиков, решетки радиаторные, уплотнители, шпаклевки, эмали и т.д. И для каждого из этих правил отображены поддержка и достоверность.

Чтобы перестраивать дерево по условию или по следствию служат две кнопки на панели инструментов: «Группировать по условию» и «Группировать по следствию».

<div> <div>Элемент</div> <div>Поддержка, %</div> </div>	Количество правил: 4	
	Следствие	Достоверность, %
Герметики	17.56	
Затирки	3.87	
Клей - ж.гвозди	10	0.99
Краски водоземлюсионные	11	1.09
Пена монтажная	11	1.09
Пигменты	10	0.99

Наиболее удобным и оперативным инструментом использования ассоциативных правил является анализ «что-если». Внешний вид формы для проведения такого анализа представлен на рисунке.

Слева находится список всех элементов транзакций, то есть весь ассортимент товара. Для каждого элемента указана поддержка – количество транзакций (чеков), в которых встречался данный элемент. Предположим, что человек купил герметики и затирки. Двойным щелчком мыши по

элементу он переносится в список условий. Используя ассоциативные правила можно предложить этому человеку сопутствующий товар, который приобретался совместно с тем, что он заказал. Этот товар отображается в списке следствий в правом нижнем окне. То есть этому человеку можно предложить купить еще и клей, либо краски вододисперсионные, либо пену монтажную, либо пигменты. Список следствий можно отсортировать по следствию, поддержке или достоверности, либо отфильтровать, оставив в нем часть следствий. Для вычисления следствий по условиям служит кнопка  «Обновить правила» на панели инструментов списка следствий. В связи с тем, что список элементов может быть очень большим, для быстрого поиска нужного элемента можно отсортировать список или воспользоваться поиском. Для этого нужно воспользоваться кнопками  «Сортировка элементов» и  «Найти...» на панели инструментов.

Пользовательские модели

В процессе анализа данных встречаются такие ситуации, когда сложные аналитические модели, основанные на линейной регрессии и нейронных сетях, не могут описать предметную область. Например, в тех случаях, когда объем исходной выборки мал, либо ее качество недостаточно для того, чтобы обучить нейронную сеть. Кроме того, иногда заранее известны модели некоторых процессов, т.е. описывающие их формулы и оценки коэффициентов. Например, в *оценках рисков* существуют несколько хорошо известных моделей, в которые эксперту требуется лишь подставить коэффициенты, описывающие конкретную ситуацию. Для того чтобы дать аналитику возможность самому строить модели, основанные на экспертных оценках, в Deductor включен обработчик «Пользовательские модели».


Пользовательская модель позволяет на основании исходных данных и формул, указанных аналитиком, построить произвольную модель и работать с ней так же, как с моделями на основе, скажем, нейросетей. Примерами такой модели может служить модель прогноза на основе авторегрессии с коэффициентами, задаваемыми экспертом.

При создании пользовательской модели нужно пройти, в основном, те же шаги, что и при настройке других моделей Deductor.

Нормализация значений полей.

Для полей, подаваемых на входы и выход дерева решений, настройки нормализации недоступны. Можно лишь посмотреть параметры нормализации, используемые по умолчанию, и гистограмму распределения выборки (описание в разделе по нейросети).

Задание модели

Задание формулы, по которой рассчитывается модель, производится в окне Калькулятора. В Deductor предусмотрены несколько стандартных моделей, включая скользящее среднее, авторегрессию, ARMA, ARIMA и др., которые можно использовать на этом этапе. Для этих моделей достаточно указать необходимые коэффициенты. Для того чтобы ввести в Калькуляторе формулу одной из стандартных моделей, следует с помощью кнопки « Функция» вызвать окно выбора функции. В разделе «Модели» будут представлены стандартные модели с описанием применения, формул и требуемых коэффициентов. Здесь нужно выделить название нужной модели и добавить ее в окно Калькулятора, нажав кнопку «Ok». Теперь в формулу модели следует добавить нужные коэффициенты, и на этом создание модели закончено.

В окне Калькулятор можно задавать формулы не только для полей, назначение у которых является выходным, но и добавлять новые поля. Разница состоит в том, что для выходных полей будут доступны визуализаторы группы Data Mining, предназначенные для оценки качества модели, – диаграмма рассеяния и таблица сопряженности. Создание модели с использованием выходных полей аналогично обучению нейросети с учителем – есть эталонные значения выходов и есть выходы, рассчитанные моделью. Чем они ближе, тем лучше модель описывает исходную выборку данных. Диаграмма рассеяния и таблица сопряженности позволяют оценить степень близости эталонных и рассчитанных выходов, а следовательно, качество модели. При добавлении же новых полей эталонных выходов нет, есть только рассчитанные моделью, и оценка качества модели с помощью подобных инструментов невозможна.

Визуализация

Для пользовательских моделей доступны три вида визуализаторов из группы Data Mining: диаграмма рассеяния, «что-если» и таблица сопряженности. Первый из них позволяет оценить

качество построенной модели по тому, насколько точно она описывает имеющиеся данные. Второй дает возможность анализировать модель по принципу «что будет, если» и позволяет исследовать ее поведение при подаче на вход тех или иных данных. С помощью таблицы сопряженности можно сравнить значения выходных полей, рассчитанные моделью, с выходными значениями полей исходной выборки, и определить, насколько точно выходы модели соответствуют эталонным значениям.

Пример

В качестве примера рассмотрим построение прогноза объемов продаж на основании короткого временного ряда. Пусть задан следующий временной ряд объемов продаж товара.

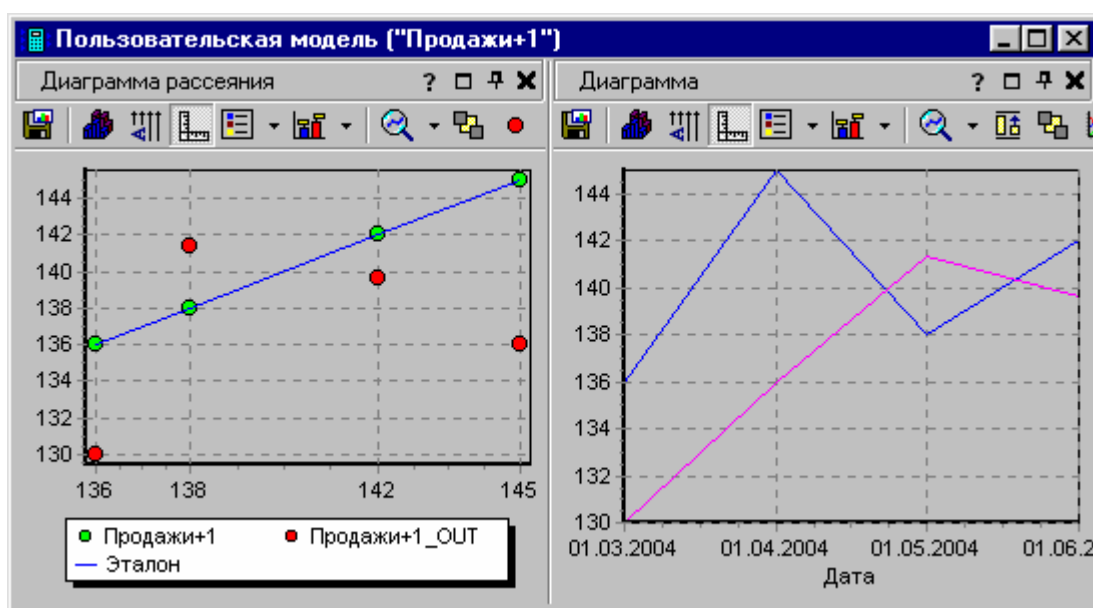
Дата	Продажи
01.01.04	118
01.02.04	129
01.03.04	143
01.04.04	136
01.05.04	145
01.06.04	138
01.07.04	142

Необходимо построить прогноз продаж на следующий месяц. Объем выборки очень мал, и построить качественный прогноз по ней нельзя. Тем не менее, даже не самый лучший прогноз зачастую бывает лучше чем ничего. Воспользуемся для прогнозирования моделью скользящего среднего. Для начала преобразуем данные скользящим окном, указав глубину погружения 2 и горизонт прогнозирования 1. Таким образом, мы будем строить прогноз на основании данных за последние три месяца. На этом шаге получим следующие результаты:

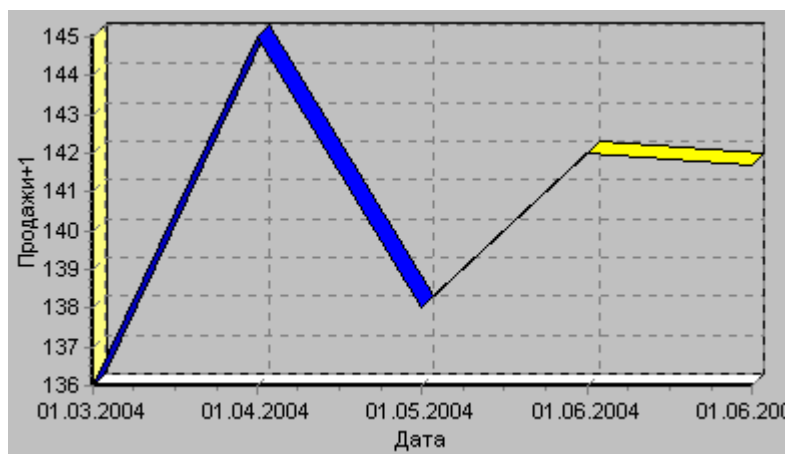
Дата	Продажи-2	Продажи-1	Продажи	Продажи+1
01.03.2004 0:00	118	129	143	136
01.04.2004 0:00	129	143	136	145
01.05.2004 0:00	143	136	145	138
01.06.2004 0:00	136	145	138	142

Теперь построим пользовательскую модель. В качестве формулы, по которой будет рассчитываться модель, укажем следующую: $Продажи+1 = MOVINGAVERAGE(COL2;COL2B1;COL2B2)$.

Диаграмма рассеяния и диаграмма модели показаны на следующем рисунке.



Как видим, качество построенной модели значительно ниже по сравнению с моделями на основе нейросетей или линейной регрессии. Тем не менее, учитывая столь маленький объем выборки, даже такая модель прогноза представляет определенный интерес. Диаграмма прогноза, построенного на один месяц, показана на следующем рисунке.



Прогнозируемое значение объема продаж составило 141.67, т.е. средний объем продаж за последние три месяца.

Вспомогательные методы обработки

Помимо перечисленных выше алгоритмов в программе есть обработки, которые трудно отнести к очистке, трансформации или методам Data Mining. Но эти механизмы обработки могут быть важной частью при создании последовательности действий по обработке данных.

Скрипт

Скрипты предназначены для автоматизации процесса добавления в сценарий однотипных ветвей обработки. По сути скрипт представляет собой динамическую копию выбранного участка сценария. Скрипт является готовой моделью, и поэтому входящие в него узлы не могут быть изменены отдельно от исходной ветки сценария. Тем не менее, на скрипте отражаются все изменения, вносимые в ветку, на которую он ссылается. То есть, при переобучении или перенастройке узлов этой ветки все сделанные изменения будут внесены в работу скрипта.

Аналогом скрипта является функция или процедура в языках программирования. Ветвь обработки строится один раз, а затем скриптами выполняются заложенные в ней универсальные обработки.

Для настройки скрипта достаточно указать начальный и конечный узлы, находящиеся на одной ветви обработки. Тогда при выполнении узла скрипта будут последовательно выполнены начальный, все промежуточные узлы между начальным и конечным и конечный узел. Основное назначение скриптов – применение готовой модели к различным наборам данных в рамках одного проекта.

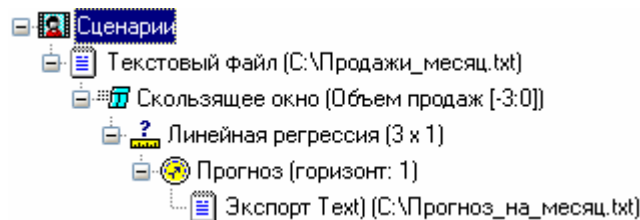
Пример

Пусть даны две таблицы с объемами продаж товара. В первой таблице продажи заданы в разрезе по дням, во второй – по месяцам. Требуется построить прогноз объемов продаж товара на один месяц по обеим таблицам, так как не известно, какая из них будет подаваться на вход в будущем.

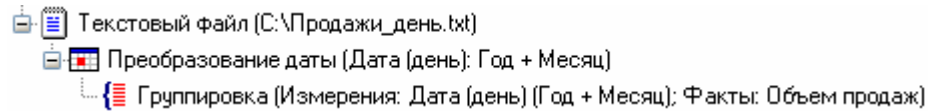
Дата (месяц)	Объем продаж
01.01.2005	312
01.02.2005	210
01.03.2005	246
01.04.2005	258
01.05.2005	234
01.06.2005	276
01.07.2005	174

Дата (день)	Объем продаж
01.01.2005	11
02.01.2005	08
03.01.2005	13
04.01.2005	12
05.01.2005	07
06.01.2005	10
...	...

Построим ветвь прогноза объема продаж по таблице с данными в месячном разрезе (см. рисунок).



Теперь импортируем данные из второй таблицы и преобразуем дату к представлению по месяцам:

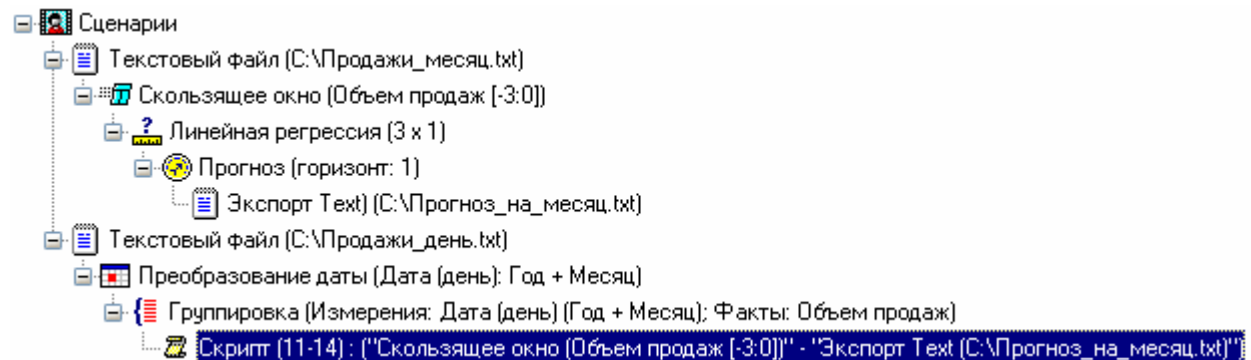


После узла группировки добавим обработчик Скрипт со следующими настройками начального узла:

Начальный этап обработки	
Скользящее окно (Объем продаж [-3:0])	
Соответствия исходных столбцов результирующим	
Исходный столбец	Результирующий столбец
7 Дата (день) (Год + Месяц)	7 Дата (месяц)
9.0 Объем продаж	9.0 Объем продаж

Конечным узлом укажем экспорт в текстовый файл с результатами прогноза.

Нам не пришлось строить модель прогноза для второй таблицы, вместо этого мы воспользовались уже готовой моделью, применив ее в другой ветви сценария. В конечном варианте сценария мы можем получить прогноз по таблице с месячными продажами, выполнив узел «Экспорт Text...», или по таблице с ежедневными продажами, выполнив узел скрипта.



Калькулятор

С помощью компонента «Калькулятор» в исходную выборку могут быть добавлены поля, значения которых вычисляются по формуле из значений других полей.

Для создания выражения нужно выполнить следующие действия:

В поле «Название выражения» ввести метку, под которой это выражение будет видно пользователю. По умолчанию, для нового выражения назначается метка «Выражение_N», где N – номер, обеспечивающий уникальность. При необходимости, можно назначить для выражения (и соответственно, для вычисляемого с его помощью поля) более информативное имя, например, «Сумма», «Доля» и так далее. Справа от имени выражения необходимо указать тип получаемого результата.

1. В поле «Выражение» ввести собственно выражение. Правила составления выражений соответствуют общим правилам составления выражений в математике, в частности число открывающих скобок должно равняться числу закрывающих.
2. Выражение может содержать:
 - числа в явном виде;
 - переменные в виде имен столбцов;
 - скобки, определяющие порядок выполнения операций;

- знаки математических операций и отношений;
- имена функций;
- даты в формате «ДД.ММ.ГГ», указываемые в двойных кавычках; такой способ указания даты может оказаться непереносимым между разными компьютерами, поэтому лучше использовать для этой цели функцию StrToDate.
- строки, обязательно указываемые в кавычках.

Выражение можно ввести вручную с клавиатуры, однако, удобнее выбирать функции, переменные и знаки операций с помощью мыши. В поле «Выражение» всегда отображается то выражение, которое в данный момент выделено в списке. Для его редактирования достаточно щелкнуть в поле мышью, вызвав курсор, а затем редактировать как обычное текстовое поле.

Интерпретация результатов

Одним из важных этапов анализа данных является интерпретация его результатов. Под интерпретацией результатов понимается их анализ экспертом предметной области. Результаты должны быть описаны на языке предметной области. Так как невозможно построить идеальную модель необходимо оценить, удовлетворяет ли построенная модель запросам, проверить ее качество.

Например, после кластеризации (сегментации) клиентов по частоте и периодичности приобретения услуг, а так же количеству приобретаемых услуг, каждому кластеру (сегменту) может быть сопоставлена степень важности или ценности входящих в него клиентов. Это и есть интерпретация полученных сегментов клиентов. Для решения задачи сегментации обычно используется история предоставления услуг за некоторый промежуток времени. Поэтому точность отнесения клиентов к тому или иному сегменту будет зависеть от объема выборки.

В программе есть средства для оценки качества построенной модели. К ним относятся таблица сопряженности, диаграмма рассеяния, в деревьях решений и ассоциативных правилах можно посмотреть поддержку и достоверность по каждому узлу и по правилу целиком.


Для того чтобы оценить качество классификации данных, обычно используют таблицу сопряженности. Для решения задачи классификации используется таблица, в которой уже есть выходной столбец, содержащий класс объекта. После применения алгоритма добавляется еще один столбец с выходным полем, но его значения уже вычисляются, используя построенную модель. При этом значения в столбцах могут отличаться. Чем больше таких отличий, тем хуже построенная модель классификации. Пример таблицы после классификации.

Входные поля					Выходное поле	Выходное поле после классификации
Сумма кредита	Возраст	Образование	...	Срок проживания	Давать кредит	Давать кредит
7000	37	Специальное	...	22	Да	Да
7500	38	Среднее	...	12	Да	Нет
14500	60	Высшее	...	30	Нет	Нет
15000	28	Специальное	...	21	Да	Да
32000	59	Специальное	...	29	Да	Да
5000	57	Специальное	...	34	Да	Нет
61500	29	Высшее	...	18	Нет	Нет
13500	37	Специальное	...	28	Нет	Да
25500	68	Высшее	...	30	Нет	Нет
...

Ошибки классификации выделены жирным шрифтом.

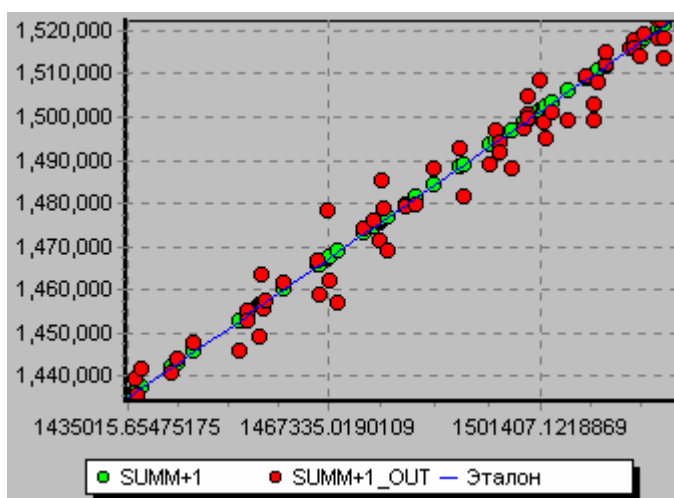
Таблица сопряженности выглядит так.

Давать кредит	Классифицировано		
	Да	Нет	Итого
Фактически			
Да	50	9	59
Нет	27	63	90
Итого	77	72	149


На главной диагонали показано количество правильно классифицированных примеров. В данном примере всего два класса, поэтому таблица сопряженности имеет размер 2 на 2 ячейки. По нажатию кнопки  на панели инструментов откроется таблица, содержащая примеры, отнесенные к этой ячейке. Например, если выбрать первую ячейку, то откроется таблица с правильно классифицированными примерами, попавшими в класс «Да».

Если количество неправильно классифицированных примеров довольно велико, это говорит о плохо построенной модели, и нужно изменить параметры построения модели, увеличить обучающую выборку, либо изменить набор входных полей. Если же количество неправильно классифицированных примеров мало, это может говорить о том, что данные примеры являются аномалиями. В этом случае можно посмотреть, чем же характеризуются такие примеры и, возможно, добавить новый класс для их классификации.

Для оценки качества моделей прогноза какой-либо непрерывной величины, т.е. при решении задачи *регрессии*, можно воспользоваться диаграммой рассеяния. На этой диаграмме отображается отклонение прогнозного значения величины от ее истинного значения.



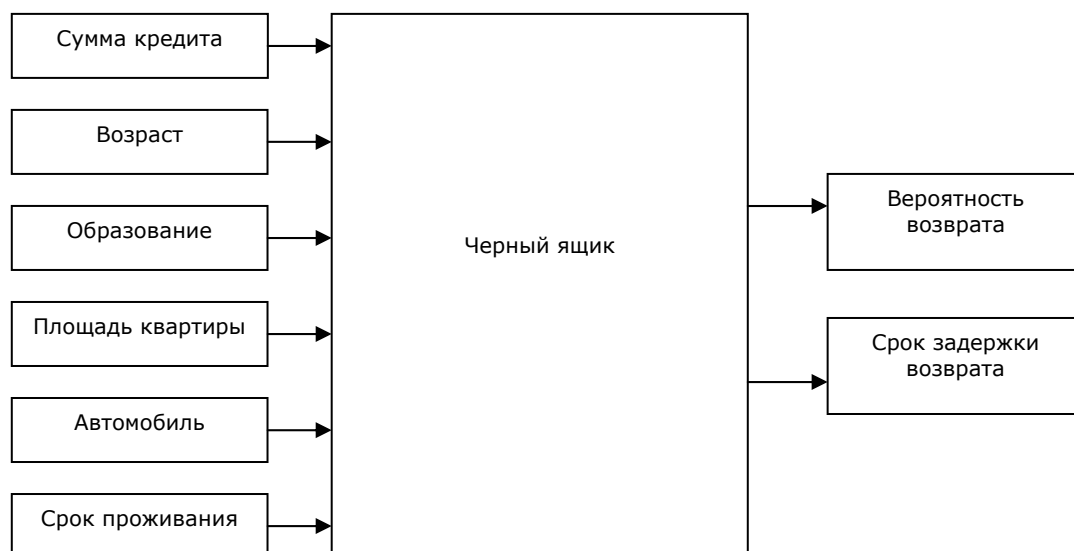
Слишком большое отклонение величины от ее истинного значения говорит о плохо построенной модели и необходимости увеличения обучающей выборки либо предобработки данных. Например, удалить аномалии, убрать шумы, изменить набор входных параметров. Слишком точное совпадение прогнозных значений с истинными может говорить о переобучении модели. То есть модель «запомнила» все примеры, используемые при ее обучении. В таком случае модель выдает отличные результаты именно на этих данных, но совершенно бесполезна на каких-либо других, например, на данных за другой промежуток времени. Это может произойти, если, например, для построения модели использовалась нейросеть с большим числом слоев.

Для точки на диаграмме рассеяния можно посмотреть детализацию. Для этого на панели инструментов следует нажать кнопку , и под диаграммой появится таблица детализации. Если теперь левой кнопкой мыши щелкнуть на интересующей точке диаграммы, в таблице появится пример выборки, которому она соответствует. С помощью детализации можно понять, что за строка набора данных скрывается за точкой с большим отклонением и определить свои дальнейшие действия – проигнорировать этот пример, исключить его из выборки или изменить параметры модели.

Анализ «что-если»

Анализ «что-если» означает ответ на вопрос: «**Что** получим, **если** задать такие значения факторов?» Здесь подразумевается, что есть некоторая величина (а в общем случае может быть несколько величин), которая зависит от нескольких факторов. Изменяя значения этих факторов, будет изменяться и значение зависимой величины. Решение обратной задачи, то есть поиск значений факторов для получения желаемого значения величины, является задачей оптимизации.

Рассмотрим пример. Есть зависимость между вероятностью возврата кредита и характеристиками кредитора. Эту зависимость можно представить в виде «черного ящика».



Прежде, чем приступить к анализу, нужно построить модель этой зависимости. Для этого в программе есть такие инструменты как нейронные сети, деревья решений, линейная регрессия. Для анализа совершенно безразлично, каким методом была построена модель. Главное, что она имитирует работу нашего «черного ящика», то есть, может по входным параметрам вычислить значение выходного. Сделать это можно с помощью таблицы «что-если».

Таблица «что-если»

В этой таблице перечислены все входные и выходные поля. Указан диапазон значений для числовых полей и количество значений для строковых. В колонке «Значение» для каждого входного поля можно указать значения, для которых требуется вычислить выходное поле.

Поле	Значение
Входные	
9.0 Сумма кредита	10000
9.0 Возраст	20
ab Образование	среднее
9.0 Площадь квартиры	37
ab Автомобиль	отечественная
9.0 Срок проживания	22
Выходные	
9.0 Вероятность возврата	0.63465

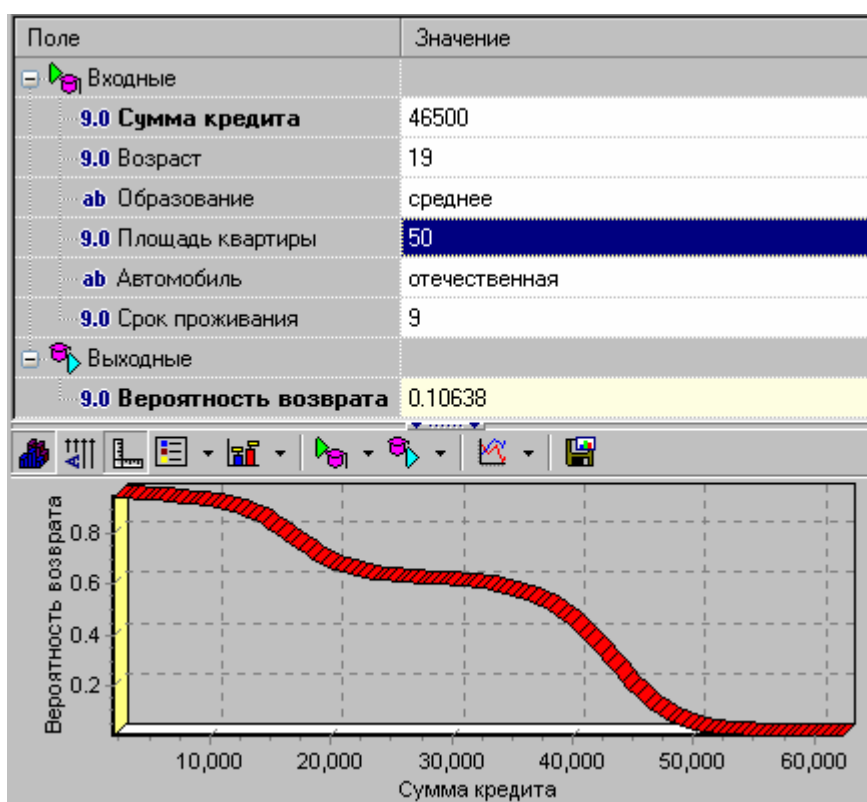
Для вычисления значения служит кнопка «Рассчитать выходы» на панели инструментов. При вводе в таблицу строковых значений предлагается выбрать значение из списка. Например, для поля «Образование» может быть список: «Среднее», «Специальное», «Высшее». При вводе числовых значений можно ввести любое число. Однако желательно вводить числа из диапазона

«Минимум» и «Максимум», так как модель была построена на значениях именно из этого диапазона.

Таким образом, таблица «что-если» имитирует работу «черного ящика». Зная значения входных полей, можно вычислить значения выходных. Это имеет практическую ценность, когда известны значения всех входных полей. Например, при решении задачи оценки кредитоспособности человека.



Диаграмма «что-если»

Часто возникает ситуация, когда необходимо подобрать значение одного из входных полей для получения желаемого значения выходного поля. Например, человеку необходимо взять кредит на определенную сумму. Вероятность возврата, вычисленная в таблице «что-если» получилась низкой. Это может говорить о слишком высокой сумме кредита, которую он запросил. Возникает вопрос, на какую сумму может рассчитывать этот человек. Ответ на него дает диаграмма «что-если». Эта диаграмма показывает зависимость выходного поля от одного из входных при фиксированных значениях остальных полей. В данном случае нас интересует зависимость вероятности возврата от суммы запрашиваемого кредита.

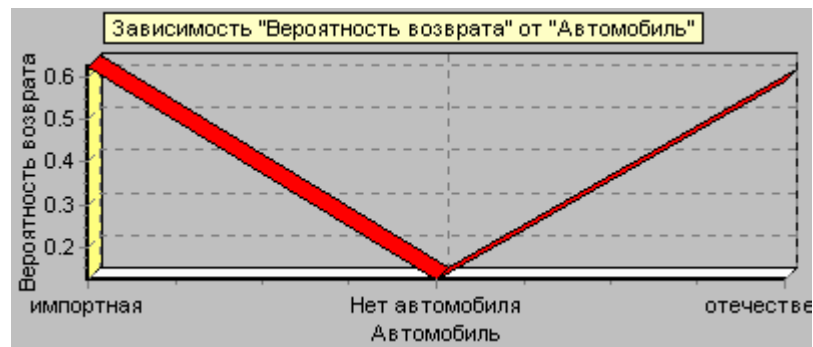


Рассмотрим, например, такую ситуацию.

Человек возрастом 19 лет со средним образованием, квартирой 50 кв. метров, отечественным автомобилем, проживающий 9 лет в данной местности запросил кредит на сумму 46500. Анализ по таблице «что-если» показал очень низкую вероятность возврата кредита – примерно 1%. На какую же сумму он может рассчитывать? Диаграмма «что-если» показывает, что на самом деле для людей с такими характеристиками кредит более 20000 слишком большой и вероятность возврата низкая (менее 60 %). А вот сумму меньше 20000 можно выдать.

Чтобы выбрать поле, по которому строить зависимость служит кнопка  «Вход» на панели инструментов диаграммы. Нажимая на нее, перебираются все входные поля. А, нажав треугольник  рядом с кнопкой, можно выбрать необходимое поле из списка.

Например, зависимость вероятности возврата кредита от наличия и типа автомобиля может быть такой.



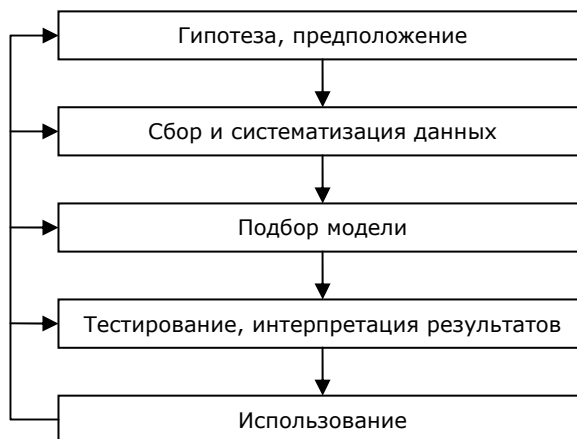
То есть, если придут два человека с абсолютно одинаковыми характеристиками и запросят одну и ту же сумму кредита, то его выдача будет зависеть от наличия у них автомобиля.

Подготовка данных для анализа

Методика анализа с использованием механизмов Data Mining базируется на различных алгоритмах извлечения закономерностей из исходных данных, результатом работы которых являются модели. Таких алгоритмов довольно много, но, несмотря на их обилие, использование машинного обучения и т.п., они не способны гарантировать качественное решение. Никакой самый изощренный метод сам по себе не даст хороший результат, т.к. критически важным становится вопрос качества исходных данных. Чаще всего именно качество данных является причиной неудачи.

Ниже описана методика, следуя которой, можно подготовить качественные данные в нужном объеме для анализа. В этой последовательности действий все достаточно просто и логично, но, несмотря на это, пользователи почти всегда допускают одни и те же тривиальные ошибки.

Общая схема использования методов Data Mining состоит из следующих шагов.



Эта последовательность действий не зависит от предметной области, поэтому ее можно использовать для любой сферы деятельности.

Выдвижение гипотез

Гипотезой в данном случае будем считать предположение о влиянии определенных факторов на исследуемую нами задачу. Форма этой зависимости в данном случае значения не имеет. То есть мы можем сказать, что на продажи влияет отклонение нашей цены на товар от среднерыночной, но при этом не указывать, как, собственно, этот фактор влияет на продажи. Для решения этой задачи и используется Data Mining. Автоматизировать процесс выдвижения гипотез не представляется возможным, по крайней мере, на сегодняшнем уровне развития технологий. Эту задачу должны решать эксперты – специалисты в предметной области. Полагаться можно и нужно на их опыт и здравый смысл. Нужно постараться максимально использовать их знание о предмете и собрать как можно больше гипотез/предположений. Обычно для этих целей хорошо работает тактика мозгового штурма. На первом шаге нужно собрать и систематизировать все идеи, их оценку будем производить позже. Результатом данного шага должен быть список с описанием всех факторов.

Например, для задачи прогнозирования спроса это может быть список следующего вида: сезон, день недели, объемы продаж за предыдущие недели, объем продаж за аналогичный период прошлого года, рекламная компания, маркетинговые мероприятия, качество продукции, бренд, отклонение цены от среднерыночной, наличие данного товара у конкурентов и т.д.

После подготовки таблицы с описанием факторов нужно экспертно оценить значимость каждого из факторов. Эта оценка не является окончательной, она будет отправной точкой. В процессе анализа вполне может оказаться, что фактор, который эксперты посчитали очень важным, таковым не является и, наоборот, незначимый с их точки зрения фактор может оказывать значительное влияние.

Формализация и сбор данных

Далее необходимо опередить способ представления данных, выбрав один из 4-х видов – число, строка, дата, логическая переменная (да/нет). Определить способ представления, т.е. формализовать, некоторые данные просто – например, объем продаж в рублях, это определенное

число. Но довольно часто возникают ситуации, когда непонятно как представить фактор. Чаще всего такие проблемы возникают с качественными характеристиками. Например, на объемы продаж влияет качество товара. Качество – это довольно сложное понятие, но если этот показатель действительно важен, то нужно придумать способ его формализации. Например, определять качество по количеству брака на тысячу единиц продукции, либо экспертно оценивать, разбив на несколько категорий – отлично/хорошо/удовлетворительно/плохо.

Необходимо оценить стоимость сбора нужных для анализа данных. Дело в том, что некоторые данные легко доступны, например, их можно извлечь из существующих информационных систем. Но есть информация, которую не просто собрать, например, сведения о конкурентах. Поэтому необходимо оценить, во что обойдется сбор данных.

Сбор данных не является самоцелью. Если информацию получить легко, то, естественно, нужно ее собрать. Если данные получить сложно, то необходимо соизмерить затраты на ее сбор и систематизацию с ожидаемыми результатами.

Есть несколько методов сбора, необходимых для анализа данных:

1. Получение из учетных систем. Обычно, в учетных системах есть различные механизмы построения отчетов и экспорта данных, поэтому извлечение нужной информации из них, чаще всего, относительно несложная операция.
2. Получение сведений из косвенных данных. О многих показателях можно судить по косвенным признакам и этим нужно воспользоваться. Например, можно оценить реальное финансовое положение жителей определенного региона следующим образом. В большинстве случаев имеется несколько товаров, предназначенных для выполнения одной и той же функции, но отличающихся по цене: товары для бедных, средних и богатых. Если получить отчет о продажах товара в интересующем регионе и проанализировать пропорции, в которых продаются товары для бедных, средних и богатых, то можно предположить, что чем больше доля дорогих изделий из одной товарной группы, тем более состоятельны в среднем жители данного региона.
3. Использование открытых источников. Большое количество данных присутствует в открытых источниках, таких как статистические сборники, отчеты корпораций, опубликованные результаты маркетинговых исследований и прочее.
4. Проведение собственных маркетинговых исследований и аналогичных мероприятий по сбору данных. Это может быть достаточно дорогостоящим мероприятием, но, в любом случае, такой вариант сбора данных возможен.
5. Ввод данных «вручную», когда данные вводятся по различного рода экспертным оценкам сотрудниками организации. Этот метод наиболее трудоемкий.

Стоимость сбора информации различными методами существенно отличается по цене и необходимому для этого времени, поэтому нужно соизмерять затраты с результатами. Возможно, от сбора некоторых данных придется отказаться, но факторы, которые эксперты оценили, как наиболее значимые нужно собрать обязательно, несмотря на стоимость этих работ, либо вообще отказаться от анализа. Очевидно, что если эксперт указал на некоторый фактор как важный, то не учитывать его просто нельзя, т.к. мы рискуем провести анализ, ориентируясь на второстепенные малозначимые факторы. И, следовательно, получить модель, которая будет давать плохие и нестабильные результаты. А такая модель не представляет практической ценности.

Представление и минимальные объемы необходимых данных

Для анализируемых процессов различной природы данные должны быть подготовлены специальным образом.

Упорядоченные данные

Такие данные нужны для решения задач прогнозирования, когда необходимо определить каким образом поведет себя тот или иной процесс в будущем на основе имеющихся исторических данных. Чаще всего в качестве одного из фактов выступает дата или время, хотя это и не обязательно. Речь может идти и о неких отсчетах, например, данных, с определенной периодичностью собираемых с датчиков.

Для упорядоченных данных (обычно это временные ряды), каждому столбцу соответствует один фактор, а в каждую строку заносятся упорядоченные по времени события с единым интервалом между строками. Не допускается наличие группировок, итогов и прочее – нужна обычная таблица.

№ п/п	Дата	Частота закупок	Объем продаж (руб.)
1	01.05.2004	256	459874.00
2	02.05.2004	278	515687.00

Если для процесса характерна сезонность/цикличность, необходимо иметь данные хотя бы за один полный сезон/цикл с возможностью варьирования интервалов (понедельное, ежемесячное...). Т.к. цикличность может быть сложной, например, внутри годового цикла квартальные, а внутри кварталов недельные, то необходимо иметь полные данные как минимум за один самый длительный цикл.

Максимальный горизонт прогнозирования зависит от объема данных:

- Данные на 1,5 года – прогноз максимум на 1 месяц;
- Данные за 2-3 года – прогноз максимум на 2 месяца.

В общем случае максимальный горизонт прогнозирования (время, на которое можно строить достаточно достоверные прогнозы) ограничивается не только объемом данных. Мы исходим из предположения, что факторы, определяющие развитие процесса будут оказывать влияние и в будущем примерно такое же, что и на текущий момент. Данное предположение справедливо не всегда. Например, в случае слишком быстрого изменения ситуации, появления новых значимых факторов и т.п. это правило не работает. Поэтому в зависимости от задачи требования к объему могут сильно изменяться. Использование слишком большого объема данных для анализа так же нецелесообразно, т.к. в этом случае мы будем строить модель по старой истории, и, следовательно, возможно, будем учитывать факторы, уже утратившие свою значимость.

Неупорядоченные данные

Такого рода данные нужны для задач, где временной фактор не имеет значения, например, оценка кредитоспособности, диагностика, сегментация потребителей. В таких случаях мы считаем ситуацию статичной и поэтому информация о том, что одно событие произошло раньше другого, значения не имеет.

Для неупорядоченных данных каждому столбцу соответствует фактор, а в каждую строку заносится пример (ситуация, прецедент). Упорядоченность строк не требуется. Не допускается наличие группировок, итогов и прочее – нужна обычная таблица.

Номер прецедента	Стаж работы	Наличие автомобиля	Объем кредита (руб.)
1	Больше 5 лет	Да	150000.00
2	Меньше 5 лет	Нет	125000.00

Количество примеров (прецедентов) должно быть значительно больше количества факторов. В противном случае высока вероятность, что случайный фактор окажет серьезное влияние на результат. Если нет возможности увеличить количество данных, то придется уменьшить количество анализируемых факторов, оставив наиболее значимые.

Желательно, чтобы данные покрывали как можно больше ситуаций реального процесса и пропорции различных примеров (прецедентов) должны примерно соответствовать реальному процессу. Мы пытаемся построить модели на основе предложенных данных, поэтому, чем ближе данные к действительности, тем лучше. Необходимо понимать, что система не может знать о чем-либо, что находится за пределами собранных для анализа данных. Например, если при создании системы диагностики больных подавать только сведения о больных, то система не будет знать о существовании в природе здоровых людей. И соответственно, любой человек с ее точки зрения будет обязательно чем-то болен.

Транзакционные данные

Транзакционные данные используются в алгоритмах поиска ассоциативных правил, этот метод часто называют «анализом потребительской корзины». Под транзакцией подразумевается несколько объектов или действий, сгруппированных в логически связанную единицу. Очень часто данный механизм используется для анализа покупок (чеков) в супермаркетах. Но, в общем случае, речь может идти о любых связанных объектах или действиях, например, продажа туристических туров с набором сопутствующих услуг (оформление виз, доставка в аэропорт, услуги гида и прочее). Используя данный метод анализа, находятся зависимости вида, «если произошло событие А, то с определенной вероятностью произойдет событие Б».

Транзакционные данные для анализа необходимо подготовить в следующем виде:

Код транзакции	Товар
10200	Йогурт «Чудо» 0.4
10200	Батон «Рязанский»
10201	Вода «Боржоми» 0,5
10201	Сахарный песок, пачка 1 кг.
10201	Хлеб «Бородинский»

Код транзакции соответствует коду чека, счета, накладной. Товары с одинаковым кодом входят в разовую покупку.

Описанного представления данных достаточно для работы обычных ассоциативных правил, где находятся связи между каждым объектом в отдельности. Пример, «Если купили Йогурт Чудо 0,4, то приобретут и Батон Рязанский».

Анализ транзакций целесообразно производить на большом объеме данных, иначе могут быть выявлены статистически необоснованные правила. Алгоритмы поиска ассоциативных связей способны быстро перерабатывать огромные массивы информации, т.к. основное их достоинство заключается в масштабируемости, т.е. способности обрабатывать большие объемы данных.

Примерное соотношение между количеством объектов и объемом данных:

- 300-500 объектов – более 10 тыс. транзакций;
- 500-1000 объектов – более 300 тысяч транзакций.

При недостаточном количестве транзакций целесообразно уменьшить количество анализируемых объектов, например, сгруппировав их.

Построение моделей – анализ

В целом, можно дать следующие рекомендации, не зависящие от конкретного алгоритма обработки:

- Уделить большое внимание очистке данных. Собрав данные в нужном объеме, нельзя быть уверенным, что они будут хорошего качества. Чаще всего, качество данных оставляет желать лучшего, поэтому необходимо их предобработать. Для этого есть множество методов: удаление шумов, сглаживание, редактирование аномалий и прочее;
- Комбинировать методики анализа. Это позволяет шире смотреть на проблему. Более того, использование различных методов для решения одной и той же задачи может привести к ценным идеям;
- Не гнаться за абсолютной точностью и начинать использование при получении первых приемлемых результатов. Все равно идеальный результат получить невозможно. Если мы получили результат, пусть не идеальный, но лучше, чем был ранее, то есть резон начать его использование. Во-первых, это позволяет быстрее получить практическую отдачу. Во-вторых, только на практике можно действительно оценить полученный результат. В-третьих, можно и нужно параллельно работать над совершенствованием модели с учетом полученных на практике результатов;
- При невозможности получения приемлемых результатов – вернуться на предыдущие шаги схемы. К сожалению, ошибки могут быть допущены на любом шаге: может быть некорректно сформулирована первоначальная гипотеза, могут возникнуть проблемы со сбором необходимых данных и прочее. К этому нужно быть готовым. При возникновении такого рода проблем возвращаться на предыдущие пункты и рассмотреть альтернативные варианты решения.

Для оценки адекватности полученных результатов необходимо привлекать экспертов в предметной области. Интерпретация модели, так же как и выдвижение гипотез может и должна делаться экспертом, т.к. для этого нужно более глубокое понимание процесса, выходящее за пределы анализируемых данных. Кроме того, нужно воспользоваться и формальными способами оценки качества модели – тестировать построенные модели на различных выборках для оценки их обобщающих способностей, т.е. способности давать приемлемые результаты на данных, которые не предоставлялись системе при построении модели. Некоторые механизмы анализа могут «запоминать» предъявленные ей данные и на них демонстрировать прекрасные результаты, но при этом полностью терять способность к обобщению и на тестовых (из неизвестных системе ранее) данных выдавать очень плохие результаты. При формальной оценке можно отталкиваться от идеи,

что если на тестовых данных модель дает приемлемые результаты, значит, она имеет право на существование.

Оптимизация работы и создания сценариев

Анализ реальной бизнес-информации зачастую связан с обработкой больших объемов данных. В связи с этим появляются проблемы оптимизации работы готового сценария. На быстродействие создаваемой аналитической системы могут значительно повлиять особенности разработанных сценариев анализа, поэтому рассмотрим подробнее пути повышения производительности при обработке сценариев. Кроме того, рассмотрим методы создания сценариев, которые позволяют значительно упростить и ускорить разработку.

Какие источники использовать

Deductor может одинаково успешно работать с большим количеством разнообразных источников данных. Однако скорость извлечения данных во многом определяется типом используемого источника. При создании законченного решения этот фактор становится очень важным, так как оперативность получения аналитической информации имеет большое значение на практике.

Самыми быстрыми источниками данных являются хранилище Deductor Warehouse, базы данных, подключаемые напрямую (MS SQL, Oracle) или через драйверы ODBC, и прямой доступ к текстовым файлам и файлам dbf. Остальные источники работают значительно медленнее.

Использовать для получения данных источники, подключаемые через интерфейс ADO, следует только в крайнем случае, при отсутствии других возможностей доступа, как, например, к таблицам MS Excel и MS Access. На больших объемах данных доступ к текстовому файлу через ADO гораздо медленнее прямого доступа. Кроме того, при прямом доступе можно указать большое число настроек разбора текстового файла. Подключение к базам данных через ADO приводит к тому, что загрузка данных в процесс хранилища из таблицы DBF занимает в 10-12 раз больше времени по сравнению с прямым доступом к этой таблице.

Очень медленным источником данных является конфигурация 1С:Предприятия. Тем не менее, удобство работы с ней напрямую часто превосходит недостаточное быстродействие. Для того чтобы ускорить загрузку в Deductor данных из 1С:Предприятия следует пользоваться механизмом внешних отчетов 1С. В этом случае подготавливается отчет, в котором нужные данные выгружаются в таблицы dbf (гораздо быстрее использования текстового файла), и уже из них производится загрузка данных в Deductor. Если для загрузки данных в хранилище применять пакетную загрузку, настроив ее запуск на ночное время, то создание промежуточных отчетов может и не потребоваться. Объем данных, хранимых в конфигурациях 1С:Предприятия обычно не очень велик (сотни тысяч строк в документах, тысячи в регистрах и справочниках). Вдобавок к этому документы будут загружаться только за последний период, и это не займет много времени. Справочники можно загружать изредка, только при появлении в них изменений. Поэтому наибольшее время займет загрузка регистров. По опыту работы, 1С:Предприятие с базами на основе MS SQL работает при прямой выгрузке данных быстрее, чем при использовании баз dbf.

В случае неизбежности получения данных из медленных источников (ADO, 1С:Предприятие) обычно следует их загружать в хранилище Deductor Warehouse. При активной работе с данными это позволит гораздо меньше задумываться о вопросах производительности при импорте данных в программы Deductor.

Кэширование

Значительно увеличить скорость выполнения сценариев может установка в «узких» местах кэша данных в оперативной памяти. Кэширование данных включается при установке в узле «Настройка набора данных» флажка «Кэшировать выходной набор данных». В результате данные, получаемые на выходе этого узла, будут полностью загружены в память. Обращение к ним будет происходить гораздо быстрее, но за счет дополнительных затрат памяти.

По умолчанию Deductor работает с данными таким образом, чтобы минимизировать объем используемой памяти. Каждый раз, когда какому-нибудь узлу требуется новая порция данных, он обращается к родительскому узлу с запросом на ее предоставление. Если в родительском узле данные не закэшированы, он в свою очередь отправляет запрос вверх по иерархии, и так далее, пока данные не будут получены. В предельном случае, узлом, предоставляющим данные, может стать узел импорта данных. Таким образом, в Deductor применен подход уменьшения требуемого объема памяти за счет роста количества вычислений, необходимых для получения каждой порции данных.

Обычно внутренняя организация работы программы не имеет особого значения. Тем не менее, могут возникать такие ситуации, когда хранение полной копии данных в некотором узле может радикально повысить скорость выполнения сценария. Для примера рассмотрим следующие ситуации.

В узле «Калькулятор» (или любом другом) производятся сложные и объемные расчеты, причем данные из этого узла и его потомков активно используются при визуализации в интерактивном режиме. В этом случае каждый раз при отображении данных могут возникать задержки, связанные с тем, что каждый раз в «Калькуляторе» производятся расчеты. Установка узла кэширования после него позволит работать при каждом обращении к данным с единожды рассчитанными результатами.

Узел является родительским для нескольких ветвей обработки. В этом случае каждая ветка будет независимо от остальных запрашивать данные у этого узла. Ему, в свою очередь, потребуется обращение на верхние уровни. Таким образом, одни и те же данные будут многократно запрашиваться и обрабатываться вышестоящими узлами. В этом случае удобней было бы после этого узла установить узел кэширования, и уже из него наследовать ветви последующей обработки.

Существуют некоторые ситуации, в которых кэширование данных приведет лишь к избыточным ненужным затратам памяти:

Кэширование данных после узла импорта не имеет смысла, так как в узле импорта уже присутствуют все данные внешнего источника.

Не имеет смысла кэшировать данные после каждого узла обработки. Это не даст большого прироста скорости, но на больших объемах данных очень быстро исчерпает имеющиеся ресурсы памяти. Кэширование следует применять только в «узких» местах сценария, где это действительно дает заметный прирост производительности.

Не имеет смысла кэширование данных перед узлами, выполняющими нормализацию данных. Это узлы, построенные на нейронных сетях (Нейросеть, карты Кохонена), Ассоциативные правила, Дерево решений, Линейная регрессия и Пользовательская модель. При нормализации в любом случае осуществляется преобразование данных, после которого они кэшируются в память специальным образом.

Следует отметить, что кэшировать имеет смысл относительно небольшие объемы данных, размер которых не превышает свободные ресурсы оперативной памяти. Если в результате установки кэша данные не поместятся полностью в оперативную память, операционная система начнет выгружать их на жесткий диск в файл подкачки. В результате обращений к жесткому диску работа сценария может замедлиться даже по сравнению с работой в отсутствие кэша.

Динамические фильтры

Ключевым моментом, сказывающимся на скорости получения результатов, служит объем анализируемых данных. В некоторых случаях существует возможность значительно уменьшить объем получаемых из хранилища по запросу пользователя данных, установив в узле импорта данных пользовательский фильтр.

При построении отчета часто возникает ситуация, когда пользователю требуется просмотреть его в некотором узком разрезе. Например, его интересует информация по конкретному поставщику и товару, или продажи за последний месяц, или отчет по работе одного дилера. Так как заранее предсказать конкретный запрос пользователя и создать на каждый запрос отдельный отчет физически невозможно, появляется необходимость в динамической генерации отчетов. Отчасти такую проблему решает использование OLAP-кубов. При этом можно посмотреть доступные данные в любом разрезе, отфильтровав ненужное. Минусом такого подхода является то, что из хранилища данных в этом случае выбирается большой объем избыточной информации. Если пользователю нужен отчет *по одному* поставщику, из хранилища все равно будут выбраны данные *по всем* поставщикам. При активной работе с хранилищем нескольких пользователей это может значительно замедлить получение ими затребованной информации. Кроме того, избыток измерений в кубе усложняет работу с ним и увеличивает усилия, затрачиваемые на получение каждого отчета.

Для решения этой проблемы в Deductor введен механизм *динамических* (или *пользовательских*) фильтров. Эти фильтры становятся доступными при импорте данных из хранилища. При создании

узла импорта на странице настройки среза есть возможность для каждого фильтра установить флаг «*Определить при выполнении*». Его установка включает пользовательский фильтр. Когда узел импорта будет выполняться в следующий раз, на экран будет выведено окно настройки среза с предложением указать нужный разрез извлекаемых данных. Пользователь имеет выбор, оставить разрез, предлагаемый аналитиком по умолчанию, или задать свои настройки выбираемых данных. После задания разреза и закрытия этого окна программа начнет импорт данных из хранилища. При этом будут извлекаться только данные, которые удовлетворяют условиям фильтрации, заданным пользователем. Если пользователь выбрал одного поставщика, то будут получены данные только по этому поставщику, благодаря чему нагрузки на сервер хранилища данных и сеть заметно уменьшаются. Сценарий затем обрабатывает полученные данные в обычном режиме и выдает на выходе отчет в ранее определенном виде, но уже только в том разрезе, в котором он затребован пользователем.

Другой механизм создания динамических отчетов заключается в использовании фильтров за период. Например, часто возникает задача сравнить два соседних отчетных периода (месяца, квартала или года). В этом случае при импорте из хранилища или в обработчике «*Фильтрация*» по измерению «*Дата*» устанавливается фильтр, в котором указывается условие – отобрать два последних месяца от имеющихся данных. На выходе фильтра получим выборку, содержащую только эти два периода. Фильтр «*от имеющихся данных*» позволяет работать только с фактически имеющимися в фильтруемом измерении периодами времени. Если в измерении «*Дата*» присутствуют данные за период с 10 января 2005 по 27 мая 2005, а сегодняшнее число 14 июля 2005, то в предыдущем примере получим данные за апрель и май 2005 года. Указав тот же период, но *от текущей даты*, получим данные уже за июнь и июль. В приведенном примере, так как информации за эти месяцы еще нет в хранилище, то на выходе получим пустой набор данных. Фильтр «*от даты*» позволяет использовать в качестве точки отсчета конкретную дату.

При построении прогноза часто бывает так, что данные за последний и первый периоды не полные. Например, при месячном прогнозе есть данные только за период с 14 числа первого месяца по 10 число последнего. Если учитывать их при построении модели прогноза, то получим резкое падение продаж за последний месяц, а это может сильно исказить результаты прогноза. Влияние первого месяца значительно слабее, но также может сказаться, например, при выявлении сезонности. Поэтому перед тем, как строить прогноз, следует удалить эти данные из выборки. Для этого можно воспользоваться фильтром при импорте данных из хранилища или обработчиком «*Фильтрация*» с динамическими фильтрами «*кроме первого периода от имеющихся данных*» и «*кроме последнего периода от имеющихся данных*».

Быстрая подготовка сценариев (скрипты)

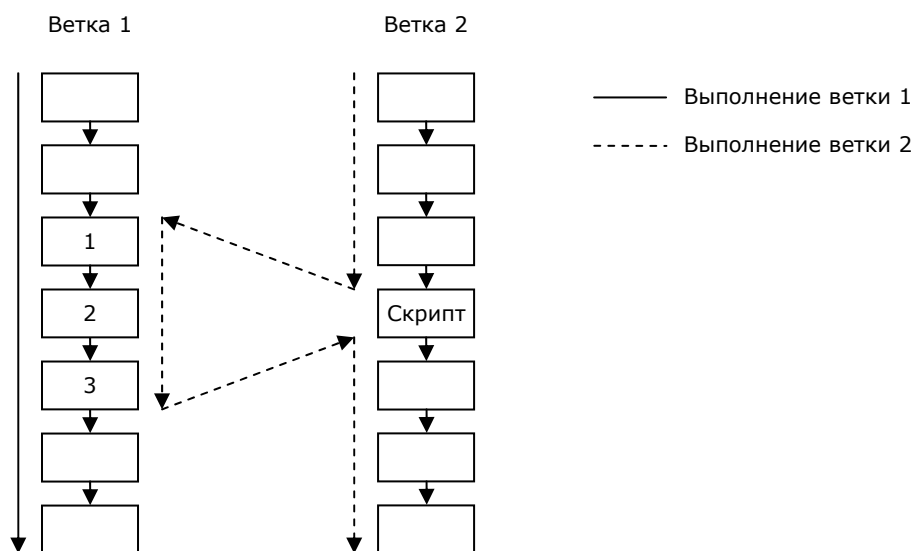
Оптимизировать требуется не только источники данных или сценарии обработки, но и работу аналитика по подготовке проектов анализа данных. Deductor включает в себя инструменты, использование которых поможет в разы ускорить создание сценариев и избавить аналитика от рутинной работы.

Ускорить разработку сценариев и предоставить возможность повторного использования однажды созданной модели призван обработчик «Скрипт». Он является аналогом функции или процедуры в языках программирования. Алгоритм работы скрипта определяется последовательностью настроенных обработчиков, входящих в его состав, а входным параметром служит поступающий набор данных.

При добавлении в проект узла «Скрипт» требуется указать начальный и конечный узлы, находящиеся на одной ветви обработки. В скрипт войдут все узлы от начального до конечного включительно. В скрипте нельзя выполнить настройку отдельного узла. Скрипт является законченным блоком обработки. Изменить работу скрипта можно, только поменяв настройки узлов в его ветви-оригинале. Внесенные в нее настройки сказываются на работе скрипта. Исходя из сказанного, переобучать нейронные сети, входящие в состав скрипта, также можно только в ветке-оригинале. В состав скрипта может войти любой узел, кроме узлов импорта и экспорта данных, то есть он может включать и другие скрипты. Импорт данных не может входить в состав скрипта по тем соображениям, что основное назначение скрипта состоит в обработке нового набора данных на уже существующей модели. Поэтому не имеет смысла пропускать через него те же данные, что и в оригинальной модели.

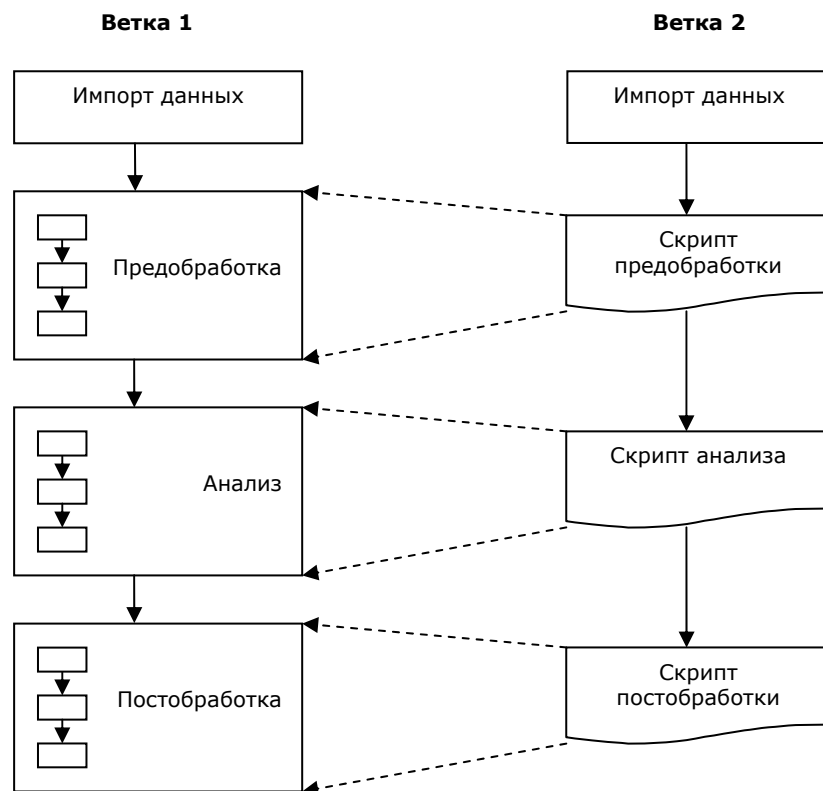
На рисунке показана схема выполнения ветви со скриптом, включающим три узла из другой ветки сценария. Сначала, до узла со скриптом, последовательно выполняются узлы второй ветки. Затем

осуществляется переход на начальный узел скрипта, находящийся в Ветви 1. Далее последовательно выполняются уже узлы первой ветки, пока не будет достигнут конечный узел скрипта. После этого осуществляется возврат к Ветке 2 на следующий после скрипта этап обработки, и выполнение продолжается. На ход выполнения первой ветки скрипт при этом не оказывает никакого влияния.



На основе скриптов могут быть реализованы три этапа обработки данных: предобработка, анализ и постобработка.

После импорта данных в программу осуществляется их предобработка (1 этап). Предобработка может быть одинаковой для различных наборов данных (например, сглаживание, очистка, фильтрация, сортировка и т.д.), поэтому в других ветках можно выполнить ее с помощью скрипта. Затем данные проходят через построенную аналитическую модель, например, строится прогноз на базе линейной регрессии (2 этап). Если данные подчиняются одним и тем же закономерностям (в этом примере временной ряд является линейным), то аналитическую модель можно построить только для одного набора данных, а для остальных использовать обработку на основе скриптов. После окончания этапа анализа данные подвергаются постобработке (3 этап), для того чтобы перевести их на язык предметной области и представить пользователю в удобном виде. Этот этап также может быть реализован в виде скриптов. Схема возможного включения скриптов в ветви сценария показана на рисунке.



Любой из этих скриптов может отсутствовать, а между двумя скриптами могут находиться произвольные узлы обработки.

Дополнительное преимущество, даваемое применением скриптов в проектах, состоит в возможности избежать ошибок и легкости модернизации сразу всех моделей обработки данных. При обнаружении ошибки достаточно исправить ее в одном месте, в ветке-оригинале, и она автоматически исправляется во всех остальных ветвях, где используется скрипт. Аналогично изменение исходной модели синхронно скажется на всех ветвях обработки, построенных на скриптах. Наряду с уменьшением размера файла проекта и ускорением его загрузки, простота внесения изменений является основным преимуществом использования скриптов перед *копированием ветви сценария* (см. «Руководство аналитика», глава «Подготовка сценариев»).

Перенастройка узла

Перенастройка узла является механизмом, с помощью которого можно изменить параметры однажды созданного узла. При перенастройке вызывается Мастер обработки (импорта или экспорта, в зависимости от типа узла), в котором можно пройти заново все этапы настройки узла. Эта операция оказывается очень полезной на этапе исследования данных и разработки модели, когда еще неизвестны оптимальные параметры каждого узла сценария. В результате можно экспериментальным путем подобрать настройки узлов.

Побочным эффектом перенастройки узла является то, что после подтверждения внесенных изменений на последней странице Мастера обработки текущий узел и все его потомки закрываются, а затем текущий узел выполняется с новыми параметрами. Таким образом, потребуются новое выполнение всех узлов-потомков. На больших наборах данных эта особенность может ограничивать применение перенастройки для поиска оптимальных решений, но в любом случае проще изменить настройки одного узла, чем заново создавать всю ветку обработки.

Перенастройка узла может привести к тому, что дочерние узлы окажутся неработоспособными. Такая ситуация может возникнуть из-за удаления какого-то столбца из набора данных, его переименования, изменения типа, появления пустых значений и т.д. В этом случае исполнение дочернего узла может вызвать сообщение об ошибке. В таком случае следует вернуть внесенные изменения, либо перенастроить узел с ошибкой на работу с новым набором данных.

Пример создания законченного аналитического решения

Рассмотрим пример создания аналитического решения для организации, занимающейся розничной торговлей. Оно будет решать следующие задачи: консолидацию данных, то есть сбор данных о продажах из распределенных баз (например, находящихся в разных магазинах), построение аналитической отчетности, анализ потребительской корзины, прогнозирование объемов продаж, поиск оптимальной наценки и сегментацию клиентов.

Создание хранилища данных

Для анализа нам потребуется информация о продажах товаров. Будем собирать информацию, представленную следующими измерениями и фактами.

Измерения:

- Дата – дата покупки;
- Клиент – покупатель;
- Товар – приобретаемый товар;
- Номер чека – для анализа потребительской корзины.

Факты:

- Цена товара – закупочная цена товара – для расчета скидки и наценки в процентах;
- Сумма наценки – общая сумма наценки;
- Сумма скидки – общая сумма скидки;
- Количество товара – количество приобретенного за один раз товара.

Эта информация должна быть представлена тремя таблицами.

Клиент.

N клиента	Наименование
Ключевое поле	Свойство

Товар.

N товара	Наименование	Группа
Ключевое поле	Свойство	Свойство

Продажа.

Дата	N клиента	N товара	Номер чека	Цена товара	Сумма наценки	Сумма скидки	Количество
Измерения				Факты			

Таблицы «Клиент» и «Товар» содержат информацию о соответствующих измерениях. Необходимость в них возникла из-за наличия свойств у этих измерений. Таблица «Продажа» содержит информацию о процессе, если говорить в терминах хранилища Deductor Warehouse.

Откроем панель «Источники данных» и создадим новое хранилище, указав путь к файлу хранилища и метку (его название, которое будет отображать при экспорте и импорте данных).

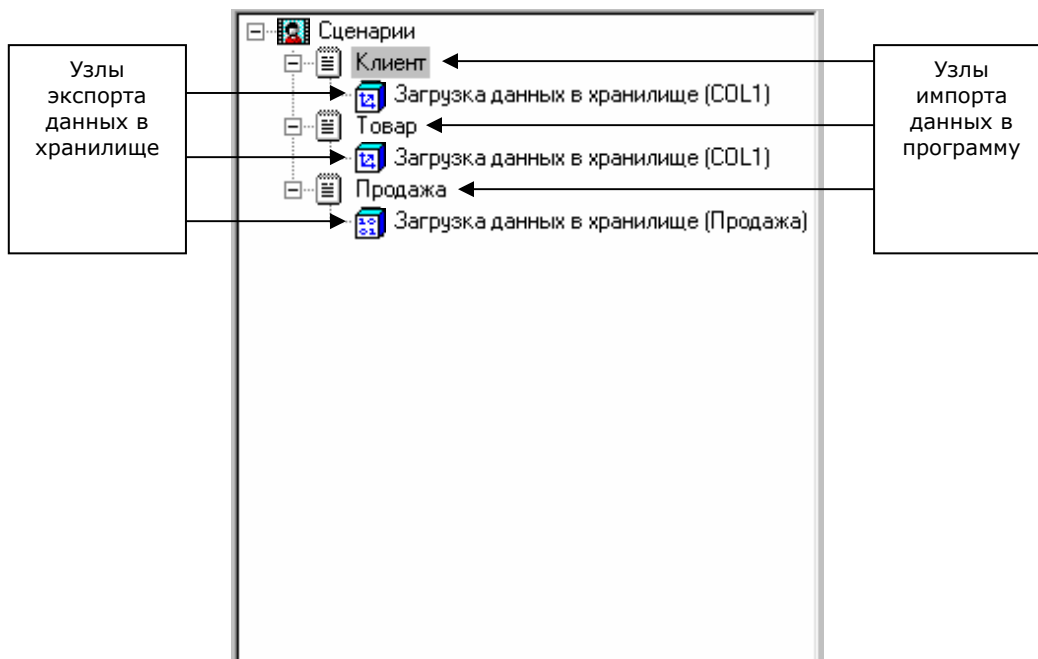
Вторым шагом при создании хранилища является импорт данных в программу. Для этого откроем Deductor Studio и вызовем мастер импорта. Предположим, что исходные таблицы содержатся в текстовых файлах с разделителем. В мастере импорта нужно будет указать, что первая строка таблицы является заголовком, указать разделитель столбцов, а также разделитель дробной части в числах. Последнее зависит от настроек операционной системы и может быть либо точкой, либо запятой. Импортируем все три таблицы в программу. Узлы импорта назовем соответственно «Клиент», «Товар» и «Продажа».

Третьим шагом будет выгрузка измерений «Товар» и «Клиент» в хранилище. Для этого для узлов «Клиент» и «Товар» поочередно вызовем мастер экспорта, выбрав в нем источник «Deductor Warehouse – измерение». Далее укажем, что является измерением, а что свойствами, как показано в таблицах выше.

Четвертым шагом будет выгрузка данных о продажах в процесс хранилища. Для этого для узла «Продажа» вызовем мастер экспорта, выбрав в нем источник «Deductor Warehouse – процесс». Так

как мы создаем новый процесс, укажем для него имя «Продажа». В поле «Описание» можно ввести любой текст. Далее укажем, что является измерениями, что фактами, как показано в таблице выше. Зададим создание вспомогательной таблицы для ускорения извлечения данных. Чтобы ускорить будущие загрузки данных в хранилище, установим флажок «Удалять из хранилища, используя измерение» и укажем измерение «Дата».

Таким образом, мы создали сценарий загрузки данных в хранилище.



Созданный сценарий можно сохранить и использовать при загрузке новых данных в хранилище.

Для анализа данных, находящихся в хранилище, создадим новый сценарий.

Прогнозирование объемов продаж

Прогнозирование является неотъемлемой частью для решения задач оптимизации. Торговые организации стремятся свести к минимуму время, которое товар лежит на складе, а также место, которое он там занимает. С другой стороны, необходимо чтобы на складе всегда лежал требуемый в настоящее время товар. Прогнозирование объемов продаж является важным шагом на пути принятия решения по оптимизации работы предприятия.

Важными данными для построения прогноза объемов продаж – это статистика продаж за предыдущие периоды. Такая информация есть в нашем хранилище.

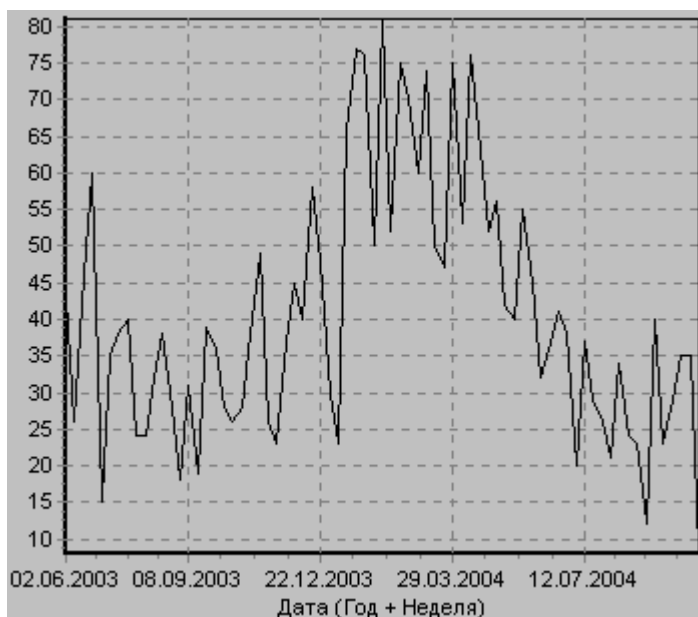
Для начала нужно импортировать данные из хранилища в программу. Нас интересуют не все данные, а только количество продаваемого товара в разрезе даты и товара. Вызовем мастер импорта и выберем в нем источник «Deductor Warehouse». Отметим загружаемые измерения и факты. После импорта озаглавим узел «Дата, товар, количество».

Рассматривать продажи по дням не имеет смысла, так как каждый день может очень сильно отличаться от другого по объему продаж. Но продажи по неделям или по месяцам разбросаны не так сильно. Поэтому вторым шагом будет преобразование даты к неделе. В таблице появится новый столбец «Дата (Год+Неделя)».

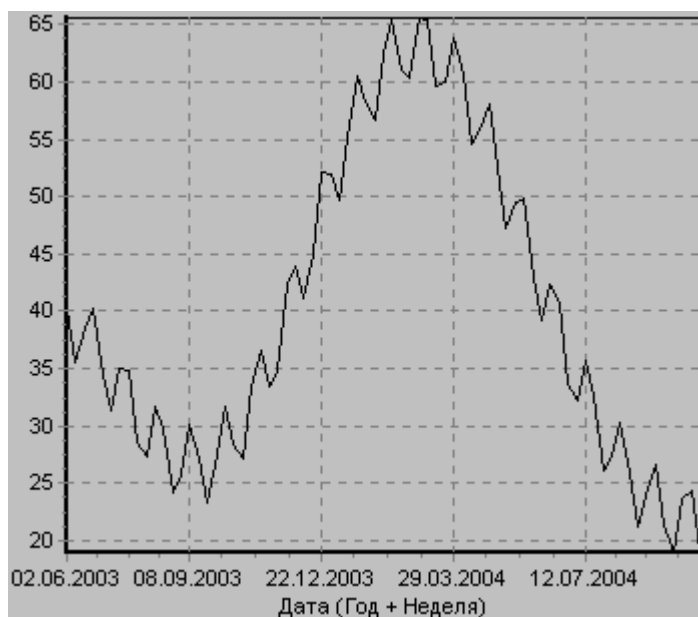
Прогнозировать объемы по разным группам товаров не имеет смысла, так как тенденция продаж разных групп товаров может очень сильно отличаться. Поэтому сделаем фильтрацию по каждой группе товара. Для этого к узлу с преобразованием даты по неделям нужно применить обработку «Фильтрация», указав условие, например, «Группа = Группа 1».

Затем следует сгруппировать количество объемов продаж по неделям. Для этого применим обработку «Группировка» к узлу с фильтрацией по группе товара. В качестве измерения укажем поле «Дата (Год+Неделя)», в качестве факта – поле «Количество». Таким образом, будет

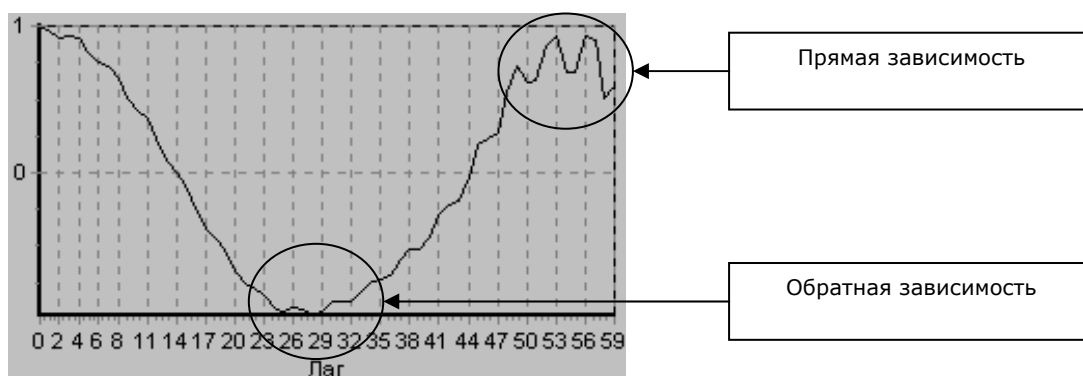
получена таблица с продажами товара группы 1 сгруппированные по неделям. Кривую продаж можно посмотреть на диаграмме.



Кривая продаж может содержать шумы и выбросы, которые необходимо удалить для получения более качественного прогноза. Для этого нужно воспользоваться парциальной обработкой. Вызовем мастер парциальной обработки для узла с группировкой. В настройках укажем для поля «Количество» вычитание шума с большой степенью вычитания. Результат такого преобразования можно посмотреть на диаграмме.



Теперь есть достаточно хорошо подготовленные для построения модели данные. Но нам необходимо определить сезонность продаж данной группы товара. Для этого служит обработка «Автокорреляция». Применим ее к узлу с парциальной предобработкой. Будем искать зависимости в поле «Количество». Такую зависимость следует искать, например, в течение 60 недель, то есть немного больше одного года. Для этого укажем количество отсчетов равное 60. Полученную автокорреляционную функцию можно посмотреть на диаграмме.



На ней видна обратная зависимость в течение примерно 28 недель и прямая зависимость в течение 53 недель, что соответствует половине года и году. Для построения модели будем использовать годовую зависимость объемов продаж.

При построении модели будем основываться на продажах за две предыдущие недели для того, чтобы учесть общее развитие рынка, и на продажах за два аналогичных месяца год назад, чтобы учесть сезонность продаж. Таким образом, необходимо подготовить такую выборку, чтобы в ней содержались поля.

Дата (Год+Неделя)	Количество-53	Количество-52	Количество-2	Количество-1	Количество
Неделя	Объем продаж год назад		Объем продаж предыдущие две недели		Текущий объем продаж

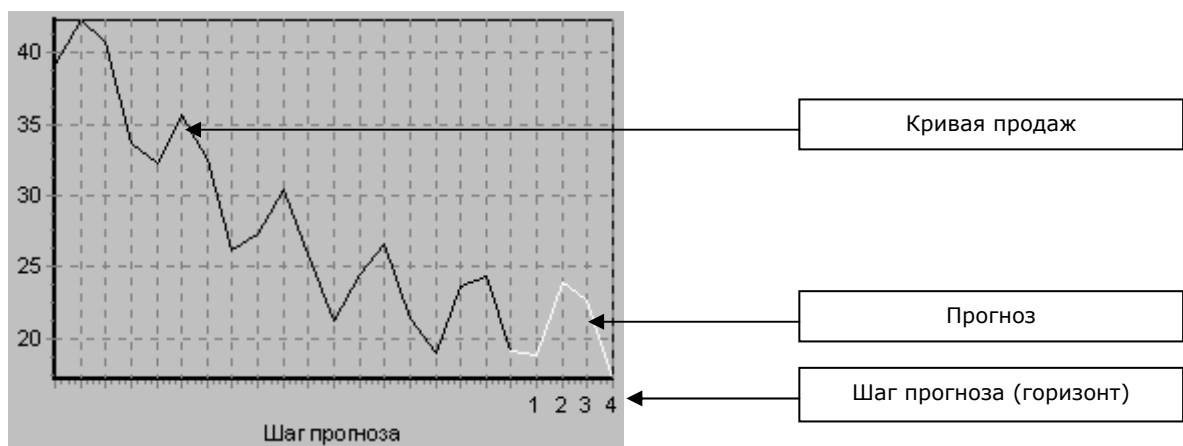
Это ни что иное, как преобразование столбца «Количество» к скользящему окну. Вызовем обработку «Скользящее окно» к узлу с парциальной обработкой. Укажем столбец «Количество» используемым и установим для него глубину равной 53. Полученная таблица будет содержать столбцы не только указанные выше, но и столбцы «Количество-3» – «Количество-51».

Построим модель, используя линейную регрессию. Вызовем после узла с преобразованием к скользящему окну обработку «Линейная регрессия». Поле «Дата (Год+Неделя)» укажем информационным, чтобы оно не участвовало в построении модели, но присутствовало в результирующей таблице. Поля «Количество-53», «Количество-52», «Количество-2» и «Количество-1» укажем входными. Поле «Количество» – выходным. А поля «Количество-3» – «Количество-51» укажем неиспользуемыми или информационными. На этом настройка линейной регрессии завершена. После построения модели можно посмотреть на диаграмме рассеяния качество построенной модели.

Теперь у нас есть модель и можно с ее помощью узнать прогнозные значения объемов продаж на небольшой промежуток времени. Для этого вызовем обработку «Прогнозирование» для узла линейной регрессии и укажем в ней горизонт прогноза, равный четырем. То есть построим прогноз на 4 недели вперед. После построения прогноза результат можно посмотреть в таблице либо на диаграмме прогноза. В таблице будет содержаться примерно следующее.

Дата (Год + Неделя)	Количество -53	Количество -52	Количество -2	Количество -1	Количество	Шаг прогноза
20.09.2004	23.4086	27.6052	21.4422	18.9691	23.6201	
27.09.2004	27.6052	31.6763	18.9691	23.6201	24.4068	
04.10.2004	31.6763	28.3353	23.6201	24.4068	19.0879	
04.10.2004	28.3353	27.1191	24.4068	19.0879	18.8843	1
04.10.2004	27.1191	33.4255	19.0879	18.8843	23.8759	2
04.10.2004	33.4255	36.6289	18.8843	23.8759	22.5807	3
04.10.2004	36.6289	33.3729	23.8759	22.5807	17.1315	4

В поле «Количество» в последних четырех строках содержатся прогнозные значения количества продаж на четыре недели в сумме по группе товара номер 1. На диаграмме прогноза можно посмотреть будущую тенденцию продаж.



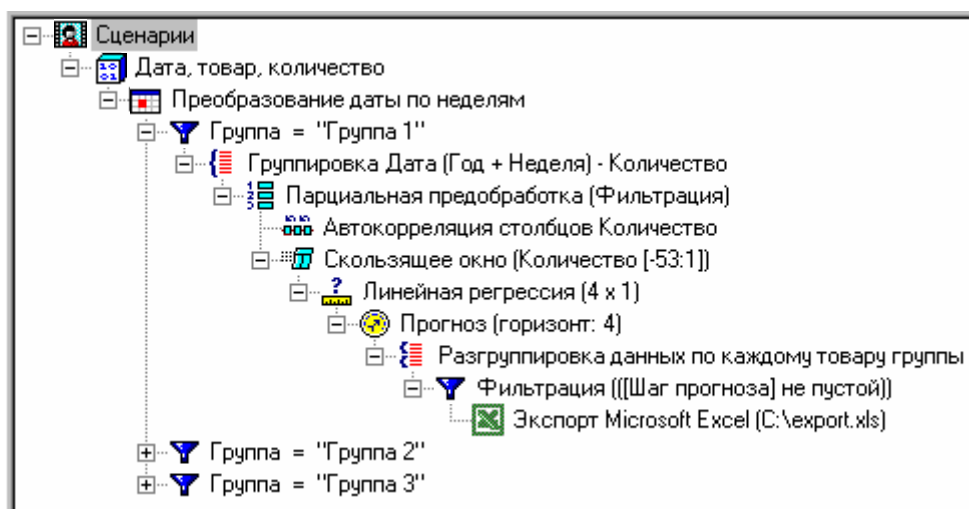
Было бы интересно узнать прогноз объемов не по группе товара, а по каждому товару в группе. Это можно сделать в предположении, что пропорциональный вклад каждого товара в объемы продаж в прошлом сохранится и в будущем. Применим к узлу прогнозирования обработчик Разгруппировка. Укажем все поля измерениями, а поле «Количество» – фактом. В поле «Исходный столбец» выберем «Количество», в поле «Восстановить» выберем «Наименование». Установим переключатель «По последним значениям». В поле «Управляющий столбец» выберем «Дата (Год+Неделя)», а в поле «Количество значений» укажем 5. То есть будем считать пропорциональный вклад каждого товара в общее количество за последние 5 недель и использовать эту пропорцию для восстановления количества каждого товара группы. Наглядно результаты разгруппировки можно посмотреть с помощью куба. Выберем визуализатор «Куб» и в его настройках укажем измерения «Наименование», «Шаг прогноза» и факт «Количество».

В результате получим таблицу.

	Шаг прогноза ▼				
Наименование ▼	1	2	3	4	Итого
Товар 1	0.71	0.90	0.85	0.65	3.11
Товар 10	1.51	1.91	1.81	1.37	6.60
Товар 11	1.79	2.26	2.13	1.62	7.80
Товар 12	1.50	1.90	1.79	1.36	6.55
Товар 2	1.45	1.83	1.73	1.31	6.33
Товар 3	1.52	1.93	1.82	1.38	6.66
Товар 4	1.44	1.82	1.72	1.30	6.27
Товар 5	1.71	2.16	2.04	1.55	7.46
Товар 6	2.10	2.65	2.51	1.90	9.16
Товар 7	1.46	1.85	1.75	1.33	6.38
Товар 8	2.00	2.53	2.39	1.82	8.74
Товар 9	1.70	2.14	2.03	1.54	7.41
Итого	18.88	23.88	22.58	17.13	82.47

Как видно, значения «Итого» для каждого шага прогноза совпадают со значениями из таблицы прогноза. Теперь можно сделать вывод, какой товар необходим на складе в течение следующих четырех недель. Это важный шаг на пути решения задачи оптимизации складов. Однако сам Deductor Studio не решает задачи оптимизации. Это необходимо делать в других программах, которые используют прогнозные значения для решения задачи оптимизации. Таким образом, возникает необходимость экспортировать результаты прогноза во внешний файл. Но при прогнозировании мы добавили в результирующую выборку исходные данные. Это те строки таблицы, где значение поля «Шаг прогноза» равен пустому значению. Вызовем для узла разгруппировки обработку фильтрации и укажем в ней условие: «Шаг прогноза не пустой». В результате будет получена таблица, готовая для экспорта. Вызовем для этого последнего узла фильтрации мастер экспорта и выберем источник, например, «Microsoft Excel». Далее следует указать экспортируемые поля таблицы. Укажем поля: «Количество», «Шаг прогноза» и «Наименование». В итоге будет сформирован файл Excel с прогнозными значениями количества продаваемого товара на следующие 4 недели.

Такую процедуру можно повторить на остальных группах товара. В результате всех этих действий будет получен сценарий прогнозирования объемов продаж.



Поиск оптимальной наценки

Предоставление скидки покупателям является стимулом для увеличения объемов закупок. Чем больше продается некоторого товара, тем больше прибыль. С другой стороны, чем больше предоставляется скидка, тем меньше наценка на товар, и тем меньше прибыли приносят продажи этого товара. Для нахождения оптимальной скидки необходимо построить зависимость прибыли от процентной ставки скидки.

В нашем хранилище отсутствуют данные о скидке в процентах и прибыли. Зато есть информация, с помощью которой их можно вычислить.

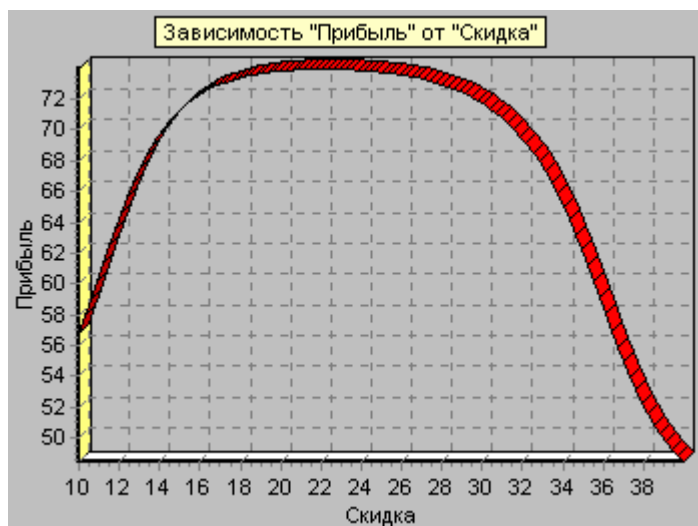
Импортируем в программу из хранилища все факты в разрезе измерений: «Дата», «Клиент», «Товар», «Номер чека».

Применим к узлу обработку «Вычисляемые данные». Назовем выражение «Скидка» и в окно формулы введем выражение: $\text{Сумма скидки} / (\text{Сумма скидки} + \text{Цена})$. Вводить выражение нужно с помощью мыши, выбирая соответствующие поля в окне со списком столбцов. Затем добавим еще одно выражение, назвав его «Прибыль». В окне формулы необходимо ввести: $\text{Сумма наценки} - \text{Сумма скидки}$.

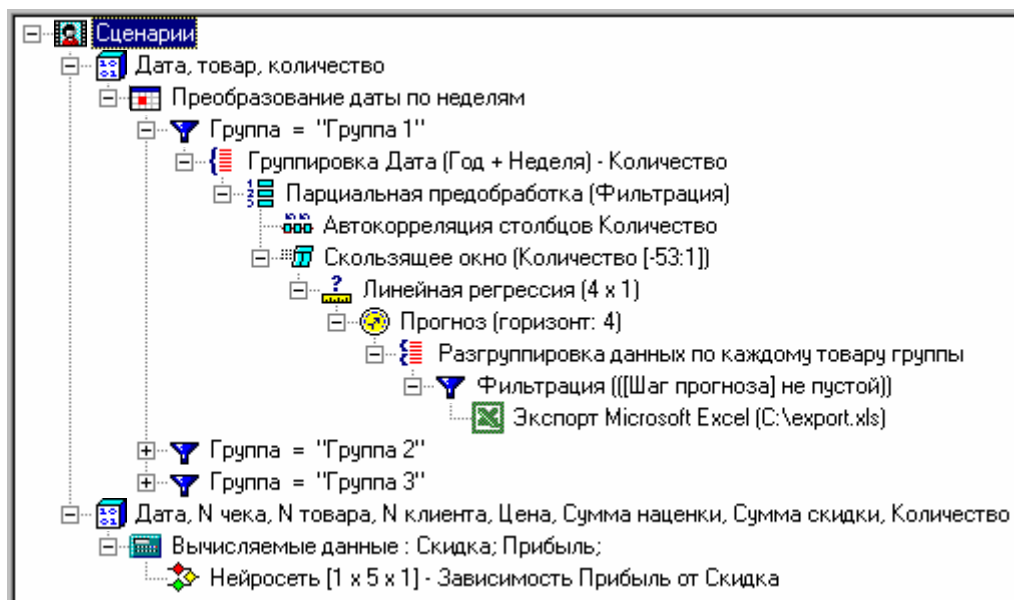
В результате работы обработки в таблицу будет добавлено два поля «Скидка» и «Прибыль».

Так как зависимость прибыли от скидки может быть нелинейной, воспользуемся нейронными сетями. Применим эту обработку к узлу с вычисляемыми данными. Укажем поле «Скидка» входным, а поле «Прибыль» – выходным. Остальные поля сделаем неиспользуемыми. Запустим алгоритм, оставив все остальные настройки без изменения. Оценить качество построенной зависимости можно с помощью диаграммы рассеяния. Если прогнозируемое алгоритмом значение прибыли разбросано относительно истинных значений, то необходимо изменить настройки нейросети, например, увеличив число слоев или нейронов в слое, или изменив функцию активации.

Нас же интересует результат, представленный диаграммой «Что-если».



Эта диаграмма наглядно показывает, при каком значении скидки достигается максимум прибыли. В результате, сценарий будет выглядеть следующим образом.



Анализ потребительской корзины

Обычно клиенты покупают не один товар, а несколько. Причем эти товары могут быть каким-то образом взаимосвязаны. Например, человек, приобретающий дверь, скорее всего, купит еще и дверные ручки. А если он приобретает дверь конкретного вида, то и ручки он купит соответствующие этой двери. Зачем нужна такая информация? Например, для размещения товара на прилавках в виде, более удобном для покупателя. Зачем идти в другой конец магазина, чтобы посмотреть, какие ручки подойдут к понравившейся двери? Или другой вариант. Покупатель приобрел дверь, но «забыл» про ручки, тогда продавец, зная такие правила приобретения товара, может сам их предложить.

Информация о товарах, приобретаемых совместно, содержится в нашем хранилище в поле «Номер чека». Найдем правила совместного приобретения товаров с помощью инструмента «Ассоциативные правила».

Импортируем в программу из хранилища количество товара в разрезе номера чека и товара. Применим к этому узлу ассоциативные правила. Полю «Номер чека» выберем назначение «Транзакция», а полю наименование – назначение «Элемент».

Выполним сначала алгоритм, не изменяя других настроек. После его выполнения возможны следующие варианты:

- найдены только одноэлементные множества. Это факт видно на диаграмме на странице обучения в мастере. На диаграмме будет отображен только один столбец. В этом случае нужно вернуться на шаг назад и уменьшить минимальную поддержку либо увеличить максимальную поддержку. И так до тех пор, пока не будут получены двух и более элементные множества;
- количество правил равно нулю, что отображается в соответствующем поле на странице обучения мастера. Тогда нужно вернуться на шаг назад и уменьшить минимальную достоверность либо увеличить максимальную.

В результате будут получены ассоциативные правила, которые можно посмотреть с помощью визуализаторов: «Правила», «Дерево правил» и «Что-Если». Например, дерево правил может выглядеть так.

+

Товар 7 (3.18%; 47)

-

Товар 16 И Товар 3 (0.27%; 4)

+

Товар 15 (0.14%; 2)

-

Товар 9 (0.14%; 2)

+

Товар 16 И Товар 9 (0.27%; 4)

+

Товар 3 И Товар 9 (0.34%; 5)

+

Товар 21 И Товар 28 (0.20%; 3)

Количество правил: 2; Условие: Товар 16 И ТОВА...

Следствие	Поддержка		Достоверность, %
	N	%	
Товар 15	2	0.14	50.00 <div></div>
Товар 9	2	0.14	50.00 <div></div>

Например, если человек приобрел одновременно «Товар 16» и «Товар 3», то он, скорее всего, купит еще и «Товар 15» с достоверностью 50% и/или «Товар 9» с достоверностью 50%.

Аналитическая отчетность

С помощью OLAP-куба можно быстро получать аналитические отчеты на основе данных, содержащихся в хранилище.

Проведем, к примеру, ABC анализ клиентов. Для этого нам понадобится информация о продажах в разрезе клиентов. Вызовем мастер визуализации для узла «Вычисляемые данные: Скидка; Прибыль», созданный в разделе «Поиск оптимальной наценки». Выберем визуализатор «Куб». В настройках куба укажем, что поле наименование клиента является измерением, а поле прибыль является фактом. Разместим измерение в строках. Полученная кросс таблица будет выглядеть так:

Наименование	Прибыль
Клиент 11	11500
Клиент 14	10700
Клиент 2	12357
...	...

Эта таблица содержит всех клиентов. Чтобы оставить только клиентов группы А, нужно вызвать «Селектор», указать в нем фильтрацию факта «Прибыль» по измерению «Наименование» и выбрать «Сумма» в списке функций. В поле «Условие» выбрать «Доля от общего», а в поле значение указать 50. После этого в таблице останутся только клиенты, приносящие в сумме 50% прибыли. Это и есть клиенты группы А. Если сделать то же самое, указав значение 80%, то будет получена таблица с клиентами групп А и В.

Аналогичный ABC анализ можно провести и для товаров, то есть выделить наиболее выгодные товары.

Иногда перед построением куба требуется некоторая предобработка данных. Например, нужно отследить постоянных, новых и утерянных клиентов. Это можно сделать, основываясь на информации о количестве обращений клиентов в месяц. Например, если клиент на протяжении нескольких месяцев не обращался в организацию, то его будем считать утерянным. Если клиент обращается примерно одинаковое число раз каждый месяц, то его можно считать постоянным.

Применим к узлу «Вычисляемые данные: Скидка; Прибыль» преобразование даты и заменим дату покупки месяцем этой даты. Для нового узла выберем визуализатор «Куб». Укажем в нем измерения «Дата (Год+Месяц)», «Наименование», а факт – «Номер чека». Для факта сразу выберем функцию агрегации «количество». Разместим «Наименование» в строках, а «Дата (Год+Месяц)» в столбцах. Полученная таблица будет иметь вид.

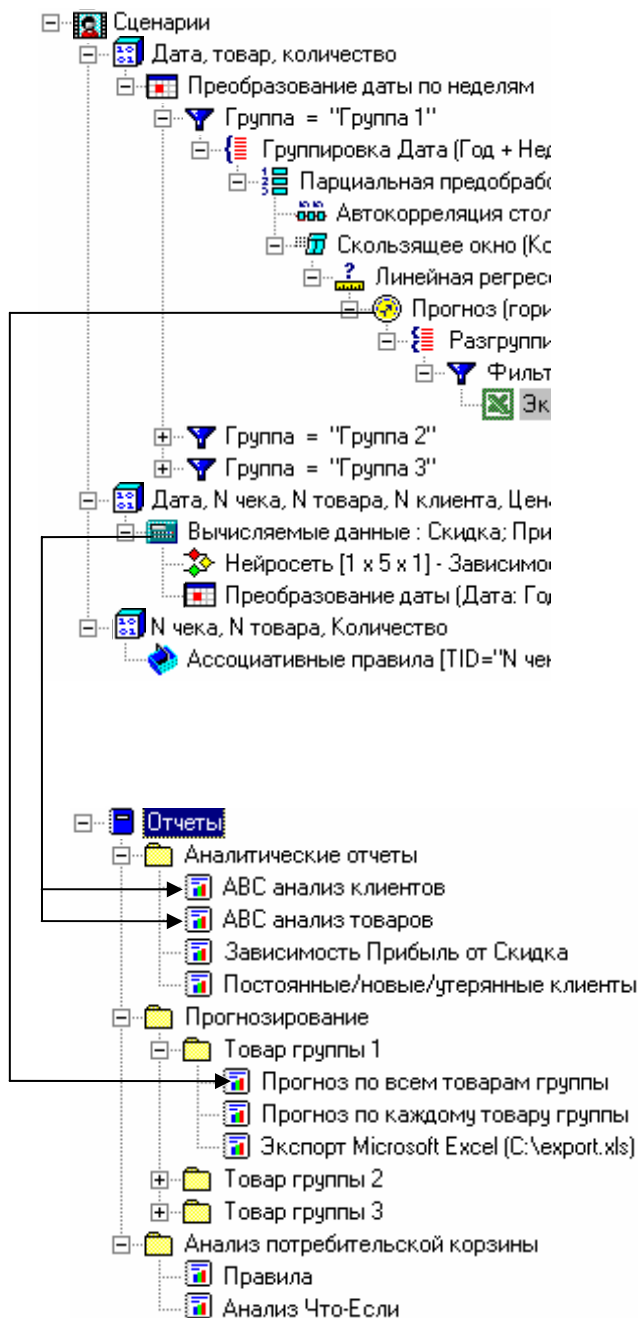
	Дата (Год+Месяц)			
Наименование	01.06.2003	01.07.2003	01.08.2003	01.09.2003
Клиент 1				2

Клиент 2	2	3	1	2
Клиент 3	3			

Клиент 1 – новый клиент на сентябрь 2003 года. Клиент 2 – постоянный клиент. Клиент 3 – утерянный.

Создание отчетности

Весь сценарий, построенный выше, обычно готовит аналитик организации. Конечным же пользователям необходимо быстро получить доступ к результатам анализа. Для этого в программе предназначена панель отчетов. Откроем эту панель и создадим на ней дерево отчетов, как показано на рисунке. Снизу изображено дерево отчетов, сверху – дерево сценариев, стрелками показано, какой узел сценария соответствует каждому отчету.



Для каждого отчета настраивается свой способ отображения. Это удобно, так как одному узлу дерева сценариев может соответствовать несколько узлов дерева отчетов.

На этом создание решения закончено, и оно готово к использованию.

Выше был приведен всего лишь один из вариантов решения поставленной задачи. В анализе данных всегда существует много путей достижения одной цели. Тем не менее, базовые вещи остаются неизменными, и при создании большинства проекта будут пройдены те же самые этапы, что и в этом примере.

Что делать при возникновении ошибок

В ходе обработки данных с использованием Deductor могут возникать различные ошибки. Иногда определить причину возникновения той или иной ошибки нелегко, но существуют некоторые рекомендации, которые помогут определить место и причину появления ошибки и исправить ее. Локализация и устранение ошибок в равной мере ложатся на плечи аналитика и администратора системы.

В процессе выполнения сценария обработки возможно появление различных ошибок, о которых Deductor будет сообщать пользователю. Рассмотрим способы решения некоторых из них.

Первое, что следует сделать при появлении окна с сообщением об ошибке – это установить источник сообщения. Если сообщение об ошибке исходит от операционной системы, значит, возникли серьезные неполадки в ее работе или работе программы. К сообщениям операционной системы, в частности, относятся все, связанные с ошибками доступа к памяти (access violation). В этом случае следует перезапустить программу, возможно, с перезагрузкой компьютера. Если ошибка не исчезает, отправьте описание своих действий, текст сообщения об ошибке, конфигурацию используемых аппаратных и программных средств и, по возможности, набор входных данных разработчикам программы по адресу deductor@basegroup.ru. Описывать действия, приведшие к появлению ошибки, следует как можно более подробно, так чтобы разработчики смогли ее повторить и установить причину возникновения. Информация о способах устранения ошибки будет выслана в ответном письме.

Эти же действия следует выполнить, если Deductor некорректно самостоятельно завершает работу («падает») или зависает.

Если сообщение об ошибке исходит от Deductor, следует определить место и причину ее возникновения. Ошибка, как правило, возникает при выполнении какого-либо определенного узла. В интерактивном режиме этот узел можно определить вручную, последовательно выполняя все узлы сценария. В пакетном режиме – с помощью подробного лога, в который записывается информация о выполнении каждого узла. После определения места возникновения ошибки следует внимательно ознакомиться с текстом сообщения. В нем обычно есть краткая информация о причинах ошибки. Например, сообщение *«Столбец XXX должен существовать в исходном источнике данных»* говорит о том, что столбец, обрабатываемый в узле, был удален из набора данных, либо у него просто поменялось имя. Такая ошибка возникает при перенастройке вышестоящих узлов или появлении изменений в источнике данных. Для ее устранения нужно либо откатить внесенные перенастройкой изменения, либо перенастроить узел, вызывающий ошибку, на работу с новым набором данных.

Часто ошибки вызываются тем, что в наборе данных появляются пустые значения (NULL-значения). Многие обработчики не способны работать с полями, содержащими пустые значения. Это касается, прежде всего, обработчиков группы Data Mining. Пустые значения могут появиться из-за изменений в источнике данных, перенастройки узлов, особенностей обработки некоторых граничных значений в узлах и т.д. От пустых значений следует избавляться с помощью фильтрации или табличной замены. Фильтрация полностью убирает из набора данных строки, содержащие пустые значения в указанных полях. С помощью же табличной замены можно поменять пустые значения на другие, нейтральные для дальнейшей обработки, например, на ноль.

При импорте данных из текстового файла и экспорте данных в хранилище по окончании операции возможно появление лог-файла, открываемого в текстовом редакторе. При импорте такой лог создается в случае появления ошибок преобразования типов. Например, в качестве разделителя целой и дробной части числа была указана точка, в то время как в действительности им является запятая. В результате Deductor не сможет выполнить преобразование прочитанного из файла числа к вещественному типу и добавит сообщение в лог. При экспорте в хранилище лог создается при наличии в наборе данных NULL-значений или в случае, когда тип загружаемых данных не соответствует типу данных объекта хранилища.

При работе Deductor под ОС Windows 9x может наступить исчерпание ресурсов системы. Так как Deductor активно работает с графической информацией, то в системе может наступить дефицит графических ресурсов. Это вызовет появление бесконечного числа окон с сообщениями об ошибках. В такой ситуации следует по возможности закрыть окна визуализаторов и другие приложения, выполняющиеся в системе. После этого сообщения об ошибках пропадут. В

критических случаях поможет принудительная перезагрузка системы. Такая проблема не касается операционных систем с архитектурой Windows NT.

Часто возникают ситуации, связанные с ошибкой доступа к источнику данных. Например, когда недоступен нужный сетевой ресурс, локальный источник данных был перемещен в другой каталог, переименован настроенный источник данных. В сообщении об ошибке в этом случае обычно находится достаточно информации для локализации и устранения ошибки. При получении сообщения об ошибке вида «*Хранилище данных с именем XXX не найдено*», «*Файл XXX не найден*» и т.п. следует проверить указанный источник данных на существование и доступность и внести соответствующие изменения в настройки источника.

При импорте данных из конфигурации 1С:Предприятия сообщение об ошибке сервера 1С может быть вызвано рядом причин.

- Недоступность сервера в сетевой версии.
- Сообщение об ошибке блокировки данных вызывается тем, что в настройке источников данных было указано использовать монопольный доступ. При этом к базе данных конфигурации уже подсоединен клиент. В этом случае следует отсоединить всех клиентов от базы или использовать сетевой доступ.
- Сообщение об ошибке инициализации сервера 1С:Предприятия может быть связано с тем, что после некорректного завершения работы с базой на основе dbf-файлов требуется переиндексирование базы. В этом случае следует самостоятельно запустить 1С:Предприятие в монопольном режиме и выполнить индексирование или отказаться от него.

При работе сценария с хранилищем Deductor Warehouse для импорта данных используются идентификаторы объектов хранилища. Идентификаторы создаются автоматически при создании новых объектов. По этой причине в тех случаях, когда структура хранилища создается заново, у объектов с теми же параметрами и именами, что и в предыдущем хранилище, могут оказаться другие идентификаторы. В результате созданные ранее сценарии не смогут работать с новым хранилищем, выводя сообщения об ошибках при импорте. Таким образом, сценарии по возможности следует настраивать на ту структуру хранилища, которая будет использоваться в будущем. Удалять и создавать новые объекты следует очень аккуратно. Для внесения изменений в существующие объекты хранилища следует использовать возможности Редактора хранилища – добавление фактов и свойств, очистка процессов и измерений (подробное описание см. в «Руководстве аналитика» глава «Создание структуры хранилища с помощью Редактора хранилища»). Только в крайнем случае (добавление в процесс нового измерения или удаление существующего) следует создавать объект заново.

Заключение

Выше рассмотрены вопросы, связанные с построением аналитических систем.

На рынке программного обеспечения существует множество разрозненных приложений, предназначенных для консолидации и анализа данных. Построение аналитической системы в этом случае происходит следующим образом. Для очистки данных используется отдельная программа. Данные для нее готовятся в специальном формате. После очистки, обработанные данные сохраняются в какой-либо файл или базу данных. Далее, для построения моделей используются другие программы, для которых данные также должны быть представлены в определенном формате и сохранены в отдельный файл. Обычно эти программы содержат собственные способы визуализации результатов. Но чтобы получать различные сводки данных необходимо опять использовать отдельную программу для построения OLAP куба. Причем для каждой программы необходимо где-то сохранять настройки. Как видно такой подход к построению аналитической системы достаточно неудобен и требует много времени. Нередко не обходится без написания дополнительных программ, обеспечивающих взаимодействие различных аналитических модулей.

Deductor 4 позволяет пройти все шаги построения аналитических систем в рамках единой платформы. Имеется возможность интеграции с различными источниками данных: файлы различных форматов, базы данных, собственное хранилище данных. Результаты обработки представляются в виде таблиц и к ним снова можно применять различные методы обработки. Нет необходимости использовать промежуточные источники данных. Архитектура построения сценариев позволяет произвести практически любой анализ. Deductor 4 является универсальной платформой и никак не привязан к какой-либо предметной области. Он позволяет пройти все этапы общего для любой предметной области алгоритма извлечения полезных знаний из данных.

Наличие самообучающихся алгоритмов позволяет легко адаптировать построенную систему к специфике работы предприятия. Возможность переобучить модель на новых данных обеспечивает переносимость сценария в филиалы организации.

Широкий спектр визуализаторов позволяет получать результаты анализа в удобном для восприятия виде. OLAP-куб, как один из методов визуализации данных, позволяет получать разнообразные срезы данных и сам предоставляет аналитические возможности. Он строится на основе таблиц данных, поэтому его можно получить на любом шаге сценария. Анализ «что-если» позволяет протестировать построенную модель на новых данных.

Экспорт данных дает возможность интегрировать программу с существующими корпоративными учетными системами.

Наличие в Deductor всех необходимых инструментов анализа, минимальные сроки разработки готовых решений, возможность наращивать и адаптировать созданное решение гарантируют быстрое получение качественного результата и превосходную отдачу в будущем.

Список литературы

1. *Quinlan, J. R. (John Ross)* C4.5: programs for machine learning.
2. *Айвазян С.А., Бухштабер В.М., Енюков И.С., Мешалкин Л.Д.* Прикладная статистика: Классификация и снижение размерности. Справочное издание под ред. Айвазяна С.А. – М.: Финансы и статистика, 1989. – 607 с.
3. *Айвазян С.А., Енюков И.С., Мешалкин Л.Д.* Прикладная статистика: Основы моделирования и первичная обработка данных. Справочное издание под ред. Айвазяна С.А. – М.: Финансы и статистика, 1983. – 471 с.
4. *Айвазян С.А., Енюков И.С., Мешалкин Л.Д.* Прикладная статистика: Исследование зависимостей. Справочное издание под ред. Айвазяна С.А. – М.: Финансы и статистика, 1985. – 471 с.
5. *Барсегян А.А., Куприянов М.С., Степаненко В.В., Холод И.И.* Методы и модели анализа данных: OLAP и Data Mining. – СПб.: БХВ-Петербург, 2004. – 336 с.: ил.
6. *Бэстенс Д.-Э., ван ден Берг В.-М., Вуд Д.* Нейронные сети и финансовые рынки: принятие решений в торговых операциях. – Москва: ТВП, 1997. – xx, 236 с.
7. *Галушкин А.И.* Теория нейронных сетей. Кн. 1: Учеб. Пособие для вузов / Общая ред. А. И. Галушкина. – М.: ИПРЖР, 2000. – 416 с.: ил. (Нейрокомпьютеры и их применение).
8. *Головкин В.А.* Нейронные сети: обучение, организация и применение. Кн. 4: Учеб. Пособие для вузов / Общая ред. А. И. Галушкина. – М.: ИПРЖР, 2001. – 256 с.: ил. (Нейрокомпьютеры и их применение).
9. *Дебок Г., Кохонен Т.* Анализ финансовых данных с помощью самоорганизующихся карт / Пер. с англ. – М.: Издательский Дом «АЛЬПИНА», 2001. – 317 с.
10. *Джук В., Самойленко А.* Data mining: учебный курс. – СПб: Питер, 2001. – 368 с.: ил.
11. *Загоруйко Н.Г.* Прикладные методы анализа данных и знаний. – Новосибирск: Изд-во Ин-та математики, 1999. – 270 с.
12. *Осовский С.* Нейронные сети для обработки информации / Пер. с польского И. Д. Рудинского. – М.: Финансы и статистика, 2002. – 344 с.: ил.
13. *Рутковская Д., Пилиньский М., Рутковский Л.* Нейронные сети, генетические алгоритмы и нечеткие системы: Пер. с польск. И. Д. Рудинского. – М.: Горячая линия – Телеком, 2004. – 452 с.:ил.
14. *Тюрин Ю.Н., Макаров А.А.* Статистический анализ данных на компьютере / Под ред. В. Э. Фигурнова – М.: ИНФРА-М, 1998. – 528 с., ил.
15. *Уосерман Ф.* Нейрокомпьютерная техника: теория и практика. – М.: Мир, 1985. – 294 с.
16. *Ханк Д.Э., Уичерн Д.У., Райтс А.Дж.* Бизнес-прогнозирование, 7-е издание.: Пер. с англ. – М.: Издательский дом «Вильямс», 2003. – 656 с.: ил. – Парал. тит. англ.

Ссылки в Internet

1. <http://www.basegroup.ru/> - большое количество статей по вопросам анализа данных и применяемым при этом алгоритмам, примеры эффективного использования методов анализа данных в бизнесе, доступные для скачивания библиотеки компонентов для анализа данных.
2. <http://forum.basegroup.ru/> - форум, посвященный проблемам прогнозирования и анализа данных при помощи современных технологий.
3. <http://glossary.basegroup.ru/> - глоссарий по нейросетевой терминологии, в котором можно уточнить значение непонятного термина.
4. <http://www.megaputer.ru/doc.php?classroom/books/books.html&print> - ссылка на книгу Арсеньева С. «Обнаружение Знаний в медицинских базах данных».
5. <http://www.statsoft.ru/home/textbook/default.htm> - электронный учебник по статистике.

6. http://lii.newmail.ru/lect_p1.htm - конспект лекций по курсу «Основы проектирования систем искусственного интеллекта».
7. <http://soft.neurok.ru/pub/lectures.shtml> - лекции и книги по нейроинформатике и практическому применению нейронных сетей.
8. <http://www.kdnuggets.com/> - англоязычный портал по всем вопросам Data Mining, Knowledge Discovery, Genomic Mining и Web Mining.