МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования

«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ
И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)

Кафедра автоматизации обработки информации (АОИ)

КОМПЛЕКС ЛАБОРАТОРНЫХ РАБОТ

Методические указания по выполнению лабораторных работ
по дисциплине: «Модели и алгоритмы распознавания и обработки данных».

Уровень основной образовательной программы: магистратура

подготовки магистра: Направление подготовки магистра 09.04.04 –
«Программная инженерия»

Магистерская программа: "Методы и технологии индустриального
проектирования программного обеспечения»

Проф. каф. АОИ
Н.В. Замятин

Томск 2016

Содержание

Введение ……..……3
ЛР № 1. Знакомство с Аналитической Платформой “Deductor” (АП DD) 4
ЛР № 2. Хранилища данных (Организация структуры) ……11
ЛР № 3. Поиск ассоциативных правил…… 18
ЛР № 4. Распознавание образов данных (НС Хемминга).....30
ЛР № 5. Кластерная обработка данных (НС Кохонена) ….....36
ЛР № 6. Классификация данных (НС с ВР). …….....43
ЛР № 7. Алгоритмы распознавания прецедентов.....46
ЛР № 8. Фильтрация данных (Фильтр Калмана) …….....52
ЛР № 9. Парциальная обработка данных…….....63
Список рекомендуемой литературы……..……..66

ВВЕДЕНИЕ

Методические указания к лабораторным работам (ЛР) по дисциплине "Модели и алгоритмы распознавания и обработки данных" представляют описание девяти четырехчасовых лабораторных работ, которые расположены последовательно, начиная от знакомства с аналитической платформой Deductor, выполнения исследований по распознаванию образов данных и их обработке с применением современных параллельных систем в виде нейронных сетей. В комплекс включены работы по предобработке данных, индуктивному обучению и поиску прецентов в данных фильтраци на основе алгоритма Кальмана.

Лабораторные работы ориентированы на использование IBM PC, совместимых ПЭВМ, реализованных на микропроцессорах семейства 8086.

Целью проведения лабораторных работ является закрепление практических навыков распознавания и обработки данных путем программирования и визуального моделирования на аналитической платформе Deductor.

Организация и проведение лабораторных работ

Студенты обьединяются в группы из 2–3 человек, работающих на закрепленном компьютере. Каждый студент получает индивидуальное задание в соответствии с номером в журнале и оформляет отчет по лабораторной работе.

Выполнение лабораторной работы предполагает предварительное изучение соответствующего раздела дисциплины и методических указаний к очередной работе.

Для допуска к выполнению лабораторной работы студент должен ознакомиться с темами для проработки и предварительно подготовить план работ и текст программы в соответствии с индивидуальным заданием.

Текст программы составляется на одном из языков программирования по указанию преподавателя или желанию студента с учетом уровня знаний конкретного языка.

В ходе выполнения лабораторной работы студент должен ответить на контрольные вопросы по предыдущей лабораторной работе. К лабораторной работе не допускаются студенты, не сдавшие более двух лабораторных работ.

Пропущенные лабораторные работы выполняются в конце семестра.

В процессе выполнения лабораторных работ следует ограничить перемещения студентов в лаборатории.

Лабораторная работа № 1. Знакомство с Аналитической Платформой «Deductor (АП DD)

Целью выполнения данной лабораторной работы является:

- получение первоначальных сведений о возможностях аналитической платформы;
- изучение основных модулей; работа с мастерами импорта, экспорта, обработки и визуализации данных.

Теоретическая часть

АП «Deductor» применима для решения задач распознавания и обработки данных, таких как парциальная обработка данных (подготовка к анализу) прогнозирование, поиск закономерностей и пр. Платформа применима в задачах, где требуется консолидация и отображение данных различными способами, построение моделей и последующее применение полученных моделей к новым данным.

Задачи, решаемые АП:

- Системы корпоративной отчетности. Готовое хранилище данных и гибкие механизмы предобработки, очистки, загрузки, визуализации позволяют быстро создавать законченные системы отчетности в сжатые сроки.
- Обработка нерегламентированных запросов**.** Конечный пользователь может получить ответ на вопросы типа "Сколько было продаж товара по группам за прошлый год с разбивкой по месяцам?" и просмотреть результаты наиболее удобным для него способом.
- Анализ тенденций и закономерностей, планирование, ранжирование. Простота использования и интуитивно понятная модель данных позволяет вам проводить анализ по принципу «Что, если...?», соотносить ваши гипотезы со сведениями, хранящимися в базе данных, находить аномальные значения, оценивать последствия принятия бизнес-решений.
- Прогнозирование. Построив модель на исторических примерах, можно использовать ее для прогнозирования ситуации в будущем. По мере

изменения ситуации нет необходимости перестраивать все, необходимо всего лишь дообучить модель.

- Управление рисками. Реализованные в системе алгоритмы дают возможность достаточно точно определиться с тем, какие характеристики объектов и как влияют на риски, благодаря чему можно прогнозировать наступление рискового события и заблаговременно принимать необходимые меры к снижению размера возможных неблагоприятных последствий.

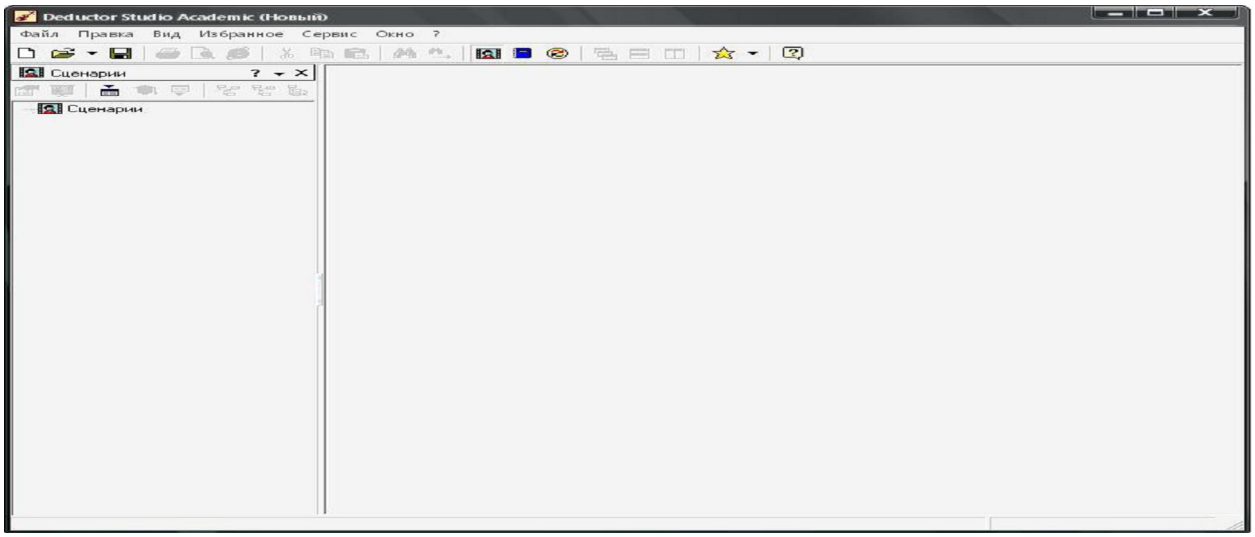
- Анализ данных маркетинговых и социологических исследований. Анализируя сведения о потребителях, можно определить, кто является вашим клиентом и почему. Как изменяются их пристрастия в зависимости от возраста, образования, социального положения, материального состояния и множества других показателей.

- Диагностика. Механизмы анализа, имеющиеся в системе Deductor, с успехом применяются в медицинской диагностике и диагностике сложного оборудования. Например, можно построить модель на основе сведений об отказах. При ее помощи быстро локализовать проблемы и находить причины сбоев.

- Обнаружение объектов на основе нечетких критериев**.** Часто встречается ситуация, когда необходимо обнаружить объект, основываясь не на таких четких критериях, как стоимость, технические характеристики продукта, а на размытых формулировках, например, найти продукты, похожие на ваши с точки зрения потребителя.

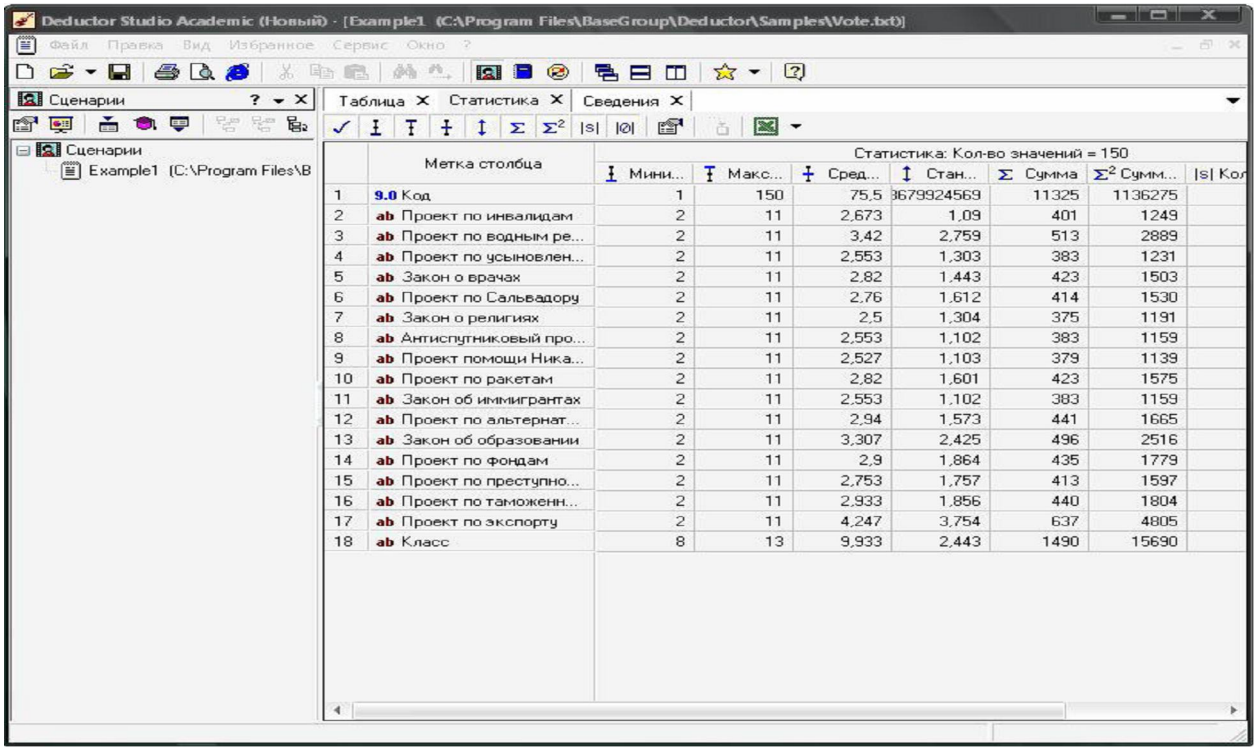
Ход работы

После запуска «Deductor Studio Academic» появится главное окно программы. Главное окно после запуска программы Deductor Studio__



Главное окно после запуска программы Deductor Studio

Выполнив вышеуказанные действия по импорту данных, на панели «Сценарии» формируется новый узел, с заданными именем, меткой и описанием.



	Метка столбца	Мини...	Макс...	Сред...	Стан...	Сумма	Σ² Сумм...	|s| Кол
1	9.0 Код	1	150	75,5	3679924569	11325	1136275	
2	ab Проект по инвалидам	2	11	2,673	1,09	401	1249	
3	ab Проект по водным ре...	2	11	3,42	2,759	513	2889	
4	ab Проект по усыновлен...	2	11	2,553	1,303	383	1231	
5	ab Закон о врачах	2	11	2,82	1,443	423	1503	
6	ab Проект по Сальвадору	2	11	2,76	1,612	414	1530	
7	ab Закон о религиях	2	11	2,5	1,304	375	1191	
8	ab Антиспутниковый про...	2	11	2,553	1,102	383	1159	
9	ab Проект помощи Ника...	2	11	2,527	1,103	379	1139	
10	ab Проект по ракетам	2	11	2,82	1,601	423	1575	
11	ab Закон об иммигрантах	2	11	2,553	1,102	383	1159	
12	ab Проект по альтернат...	2	11	2,94	1,573	441	1665	
13	ab Закон об образовании	2	11	3,307	2,425	496	2516	
14	ab Проект по фондам	2	11	2,9	1,864	435	1779	
15	ab Проект по преступно...	2	11	2,753	1,757	413	1597	
16	ab Проект по таможенн...	2	11	2,933	1,856	440	1804	
17	ab Проект по экспорту	2	11	4,247	3,754	637	4805	
18	ab Класс	8	13	9,933	2,443	1490	15690	

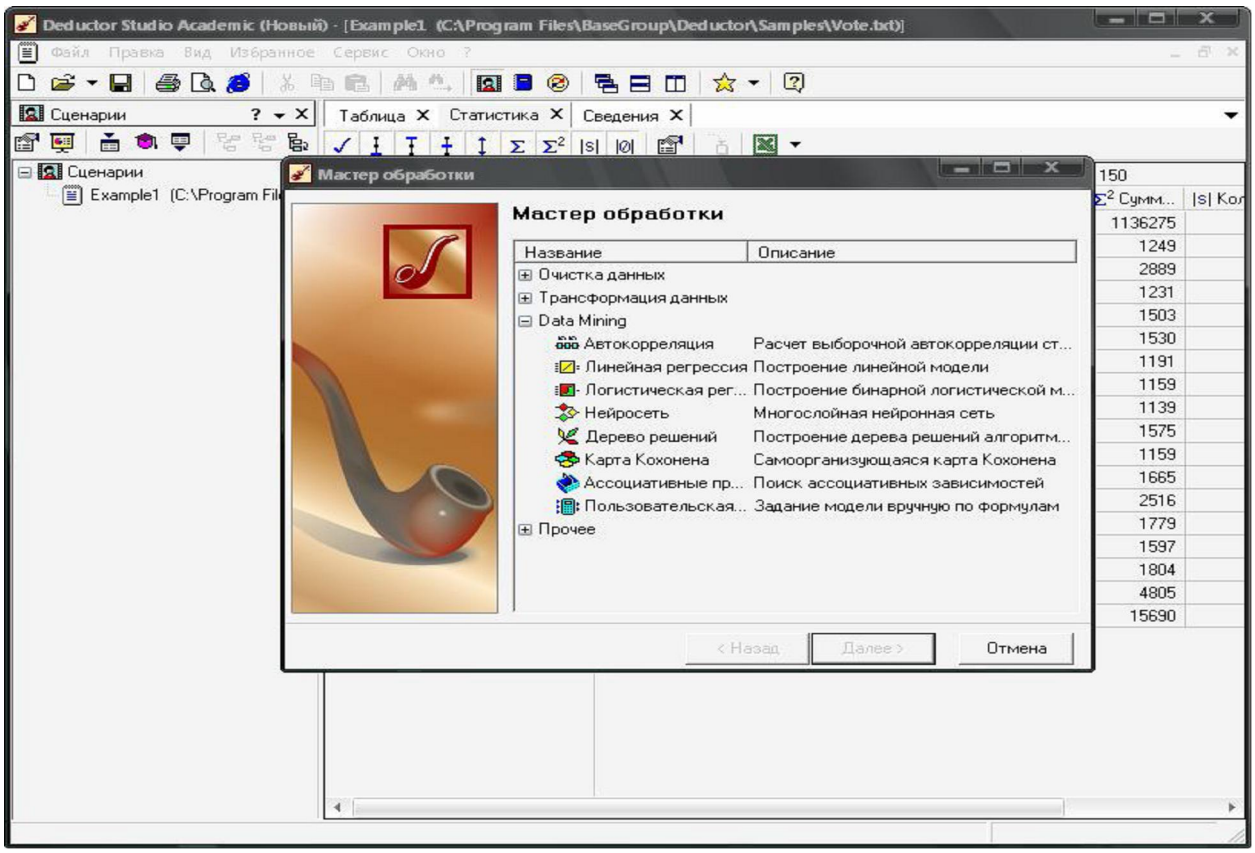
Пример создания сценария, вкладка «Статистика»

Для изучения возможности мастера обработки (кнопка в левой части главного окна либо клавиша F7). После запуска мастера обработки появится список возможных способов обработки данных.

Все способы разделены на четыре основные группы: очистка данных, трансформация данных, Data Mining, пр. Каждый способ обработки имеет название и краткое описание. Выбор способа зависит от целей обработки

данных (например, сортировка и фильтрация данных, построение дерева решений и пр.).

Мастер визуализации позволяет определить способ отображения данных, указать метки и добавить описание к проекту. Запустить его можно с помощью кнопки либо клавишей F5.



Список доступных способов обработки данных

Готовый проект можно экспортировать, воспользовавшись мастером экспорта (кнопка основного окна либо клавиша F8).

Указав параметры, проект можно перенести в один из доступных форматов.

Задание

1. Опишите назначение и возможности АП «Deductor».

2. Запустите программу «Deductor Studio Academic», ознакомьтесь с назначением кнопок и контекстным меню главного окна программы.

3. Воспользуйтесь мастером импорта данных (импортируйте файл с данными Вашей предметной области или из C:\Program Files\ BaseGroup\ Deductor\Samples\ *.txt), или из репозитория данных.

4. Ознакомьтесь с доступными способами обработки данных.

5. Изучите возможности мастера визуализации и экспорта.

Содержание отчета

1. Цель работы.

2. Краткое описание хода работы с описанием возможности Deductor дл распознавания и обработки данных и приведением скриншотов.

3. Ответы на вопросы.

Вопросы:

1. Какие существуют другие платформы для распознавания и обработки данных?
2. Какие возможности имеет АП Deductor для распознавания данных?
3. Какие возможности имеет АП Deductor для обработки данных?
   4. Какие параметры доступны для мастера экспорта данных?
4. В чем заключается процедура визуализации данных?

ЛАБОРАТОРНАЯ РАБОТА №2.

ХРАНИЛИЩЕ ДАННЫХ В АНАЛИТИЧЕСКОЙ ПЛАТФОРМЕ DEDUCTOR.

Цель работы: изучить программную среду хранилища данных в DeductorWarehouse, ознакомиться с архитектурой научиться создавать, и наполнять информацию из хранилища данных.

Ход работы:

1. Хранилище данных (ХД) **DeductorWarehouse** - это специально организованная база данных, ориентированная на решение задач анализа данных и поддержки принятия решений, обеспечивающая максимально быстрый и удобный доступ к информации. **DeductorWarehouse 6** соответствует модели ROLAP (схема «снежинка»).

Хранилище данных **DeductorWarehouse** включает в себя потоки данных, поступающие из различных источников, и специальный семантический слой, содержащий так называемые метаданные (данные о данных). Семантический слой и сами данные хранятся в одной базе данных. Все данные в хранилище

DeductorWarehouse хранятся в структурах типа «снежинка», где в центре расположены таблицы фактов, а «лучами» являются измерения, причем каждое измерение может ссылаться на другое измерение. Именно эта схема чаще всего встречается в хранилищах данных (рис.4.9.).

Объекты хранилища данных DeductorWarehouse следующие.

Измерение - это последовательность значений одного из анализируемых параметров. Например, для параметра «время» это последовательность календарных дней, для параметра «регион» - список городов. Каждое значение измерения может быть представлено координатой в многомерном пространстве процесса, например, Товар, Клиент, Дата.

Атрибут - это свойство измерения (т.е. точки в пространстве). Атрибут как бы скрыт внутри другого измерения и помогает пользователю полнее описать исследуемое измерение. Например, для измерения Товар атрибутами могут выступать Цвет, Вес, Габариты.

Факт - значение, соответствующее измерению. Факты - это данные, отражающие сущность события. Как правило, фактами являются численные значения, например, сумма и количество отгруженного товара, скидка.

Ссылка на измерение - это установленная связь между двумя и более измерениями. Бизнес-понятия (соответствующие измерениям в хранилище данных) могут образовывать иерархии,

например, Товары могут включать Продукты питания и Лекарственные препараты, которые, в свою очередь, подразделяются на группы продуктов и лекарств и т. д. В этом случае первое измерение содержит ссылку на второе, второе - на третье и т.д.

Процесс - совокупность измерений, фактов и атрибутов. По сути, процесс и есть «снежинка». Процесс описывает определенное действие, например, продажи товара, отгрузки, поступления денежных средств и прочее.

Атрибут процесса - свойство процесса. Атрибут процесса в отличие от измерения не определяет координату в многомерном пространстве. Это справочное значение, относящееся к процессу, например, № накладной,

Валюта документа и так далее. Значение атрибута процесса в отличие от измерения может быть не всегда определено.

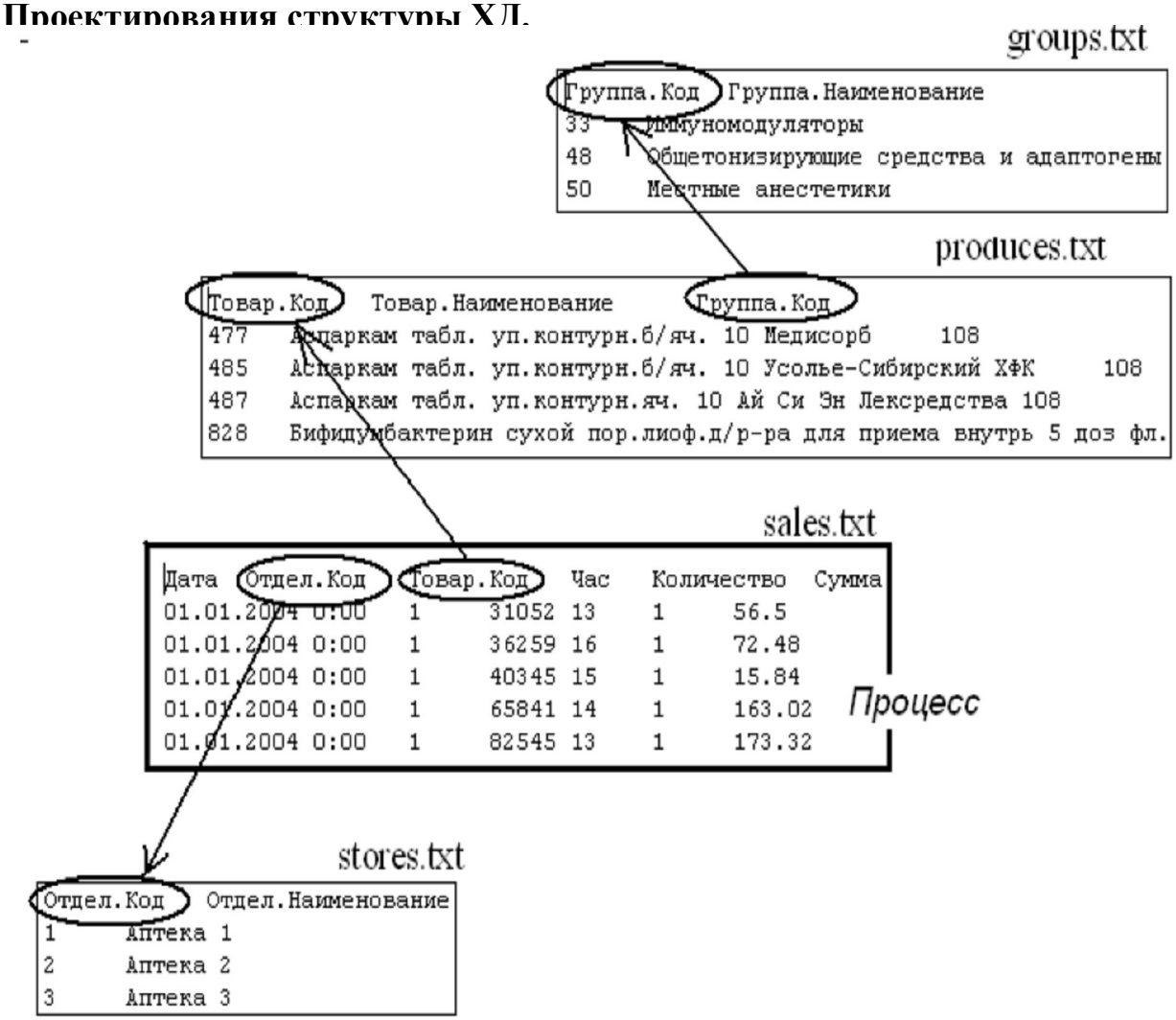
В DeductorWarehouseможет одновременно храниться множество процессов, имеющих общие измерения, например, измерение *Товар*, фигурирующее в процессах *Поступления*и *Отгрузка*.

Все загружаемые в ХД данные обязательно должны быть определены как измерение, атрибут либо факт. Принадлежность данных к типу (измерение, ссылка на измерение, атрибут или факт) содержится в семантическом слое хранилища. Обратим внимание на то, что:

■ таблицы *измерений* содержат только справочную информацию (коды, наименования и т.п.) и ссылки на другие измерения при необходимости;

■ таблица *процесса* содержит только факты и коды измерений (без их атрибутов).

Проектирования структуры ХД.



В таблице groups.txt *Код группы* является измерением, а *Наименование группы* - его атрибутом.

В таблице produces.txt *Код товара* является измерением, а *Наименование товара* - его атрибутом, а *Код группы* - ссылкой на одноименное измерение.

В таблице stores.txt *Код отдела* является измерением, а *Наименование отдела* - его атрибутом.

В таблице sales.txt *Дата* является измерением, *Отдел, Код товара* и *Код группы* как было сказано выше – измерения. *Час покупки* - измерение, *Количество* и *Сумма*- факты, т.е. таблица sales.txt является описанием процесса продаж в трех аптеках.

2. Создание хранилища данных в DeductorWarehouse.

Откройте программу DeductorStudio, используя ярлык на рабочем столе или через кнопку Пуск.

Для создания нового хранилища данных или подключения к существующему в DeductorStudio необходимо перейти на закладку **Подключения** и запустить **Мастер подключений**.

На экране появится первый шаг **Мастера**, в котором следует выбрать тип источника (приемника), к которому нужно подключиться. Выберите **DeductorWarehouse**и нажмите кнопку**Далее.**

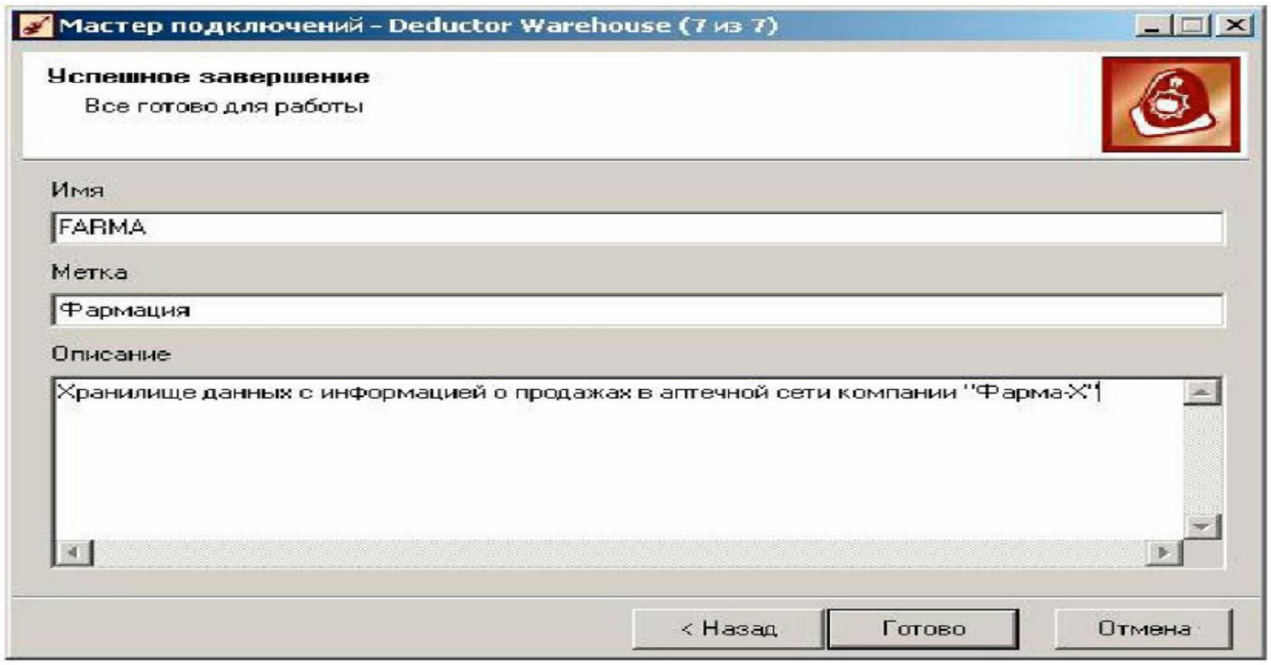
На следующем шаге из единственно доступного в списке типа базы данных выберем **Firebird**и перейдем на третий шаг мастера. В нем зададим параметры базы данных, в которой будет создана физическая и логическая структура хранилища данных (рис. 2), Нажмите **Далее**.

На следующей вкладке выберем последнюю версию для работы с ХД **DeductorWarehouse6** (предыдущие версии необходимы для совместимости с ранними версиями хранилищ).

На следующем шаге при нажатии на кнопку

По указанному ранее пути будет создан файл **farma.gdb** (появится сообщение об успешном создании). Это и есть пустое хранилище данных, готовое к работе.

На последних двух шагах осталось выбрать визуализатор для подключения (здесь это **Сведения** и **Метаданные**) и задать имя, метку и описание для нового хранилища.



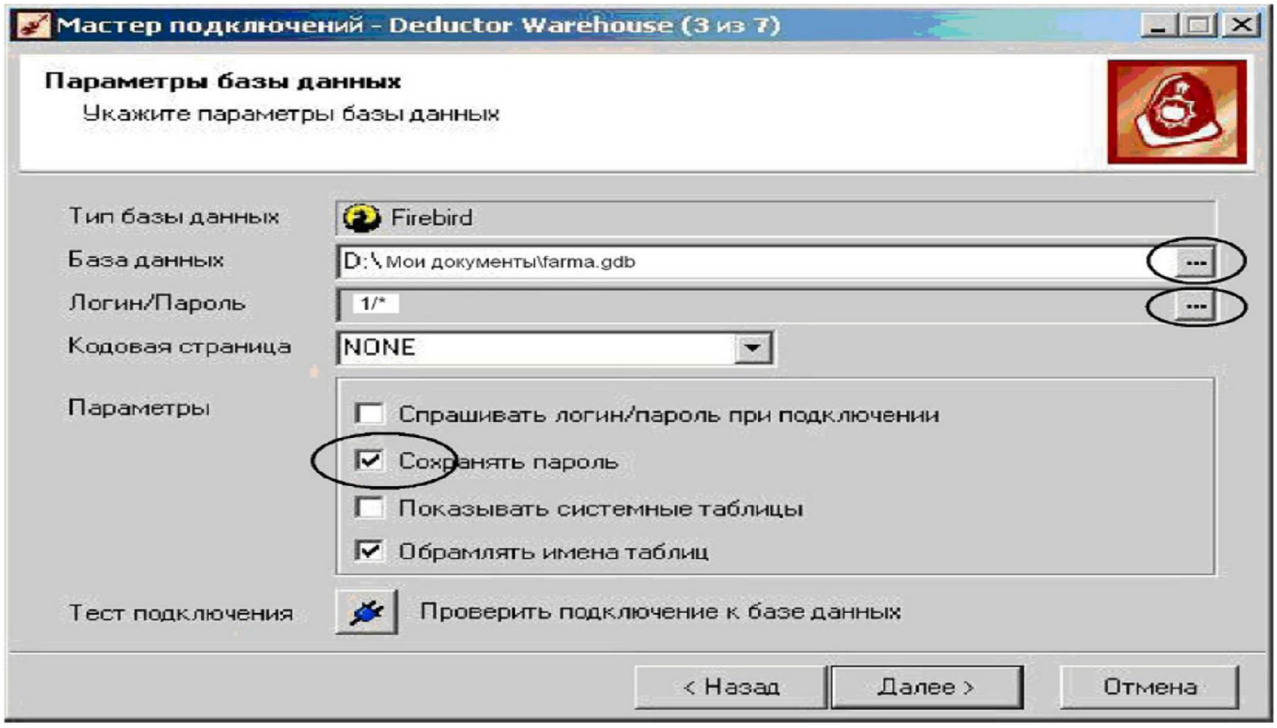
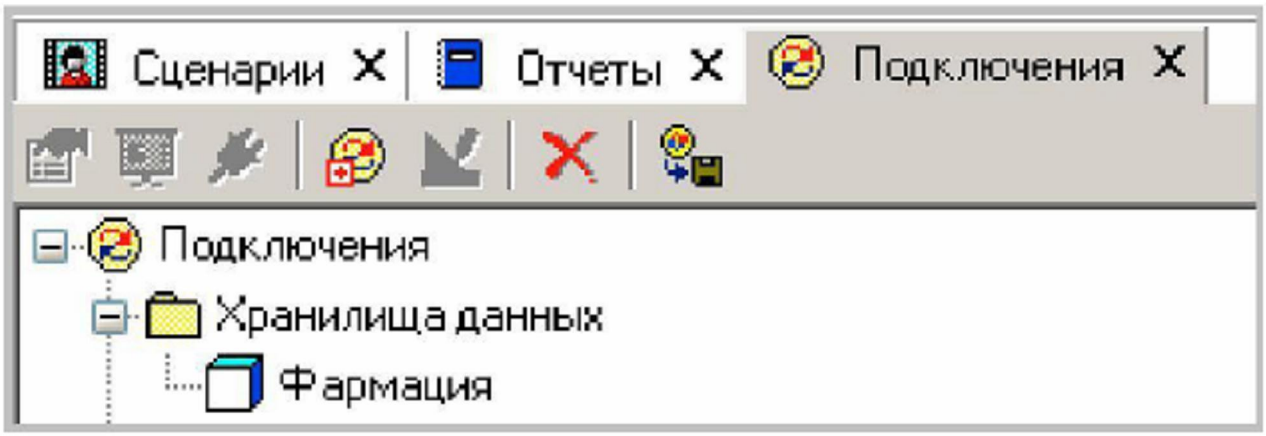


Рис.2. Установка параметров базы данных

После нажатия на кнопку **Готово** на дереве узлов подключений появится метка хранилища.



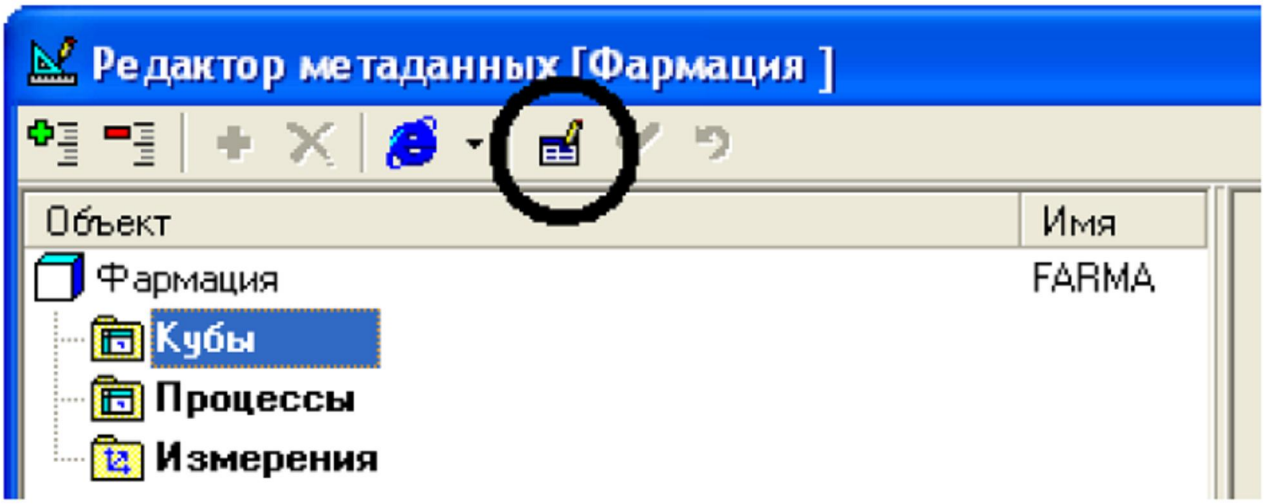
Если соединение по какой-либо причине установить не удалось, то будет выдано сообщение о ответствующей ошибке. В этом случае нужно проверить параметры подключения хранилища данных и при необходимости внести в них изменения (используйте для этого кнопку **Настроить подключение**.

Для проверки доступа к новому хранилищу данных воспользуйтесь кнопкой Если спустя некоторое время появится сообщение «Тестирование соединения прошло успешно», то хранилище готово к работе.

Сохраните настройки подключений, нажав на кнопку сохранения .

После создания хранилища необходимо спроектировать его структуру, т.к. в пустом хранилище нет ни одного объекта (процессов, измерений, фактов). Для этого предназначен «Редактор метаданных», который вызывается кнопкой на вкладке **Подключения**. Нажмите ее.

Для перехода в режим внесения изменений в структуру хранилища нажмем кнопку **Разрешить редактировать**.



Появится диалоговое окно с предупреждением. Нажмем **Да** и в открывшемся окне редактора метаданных, встав на узле **Измерения**, при помощью кнопки **Добавить** добавим в метаданные первое измерение Код группы со следующими параметрами:

имя – GR_ID;

метка - Группа.Код;

тип данных-целый.

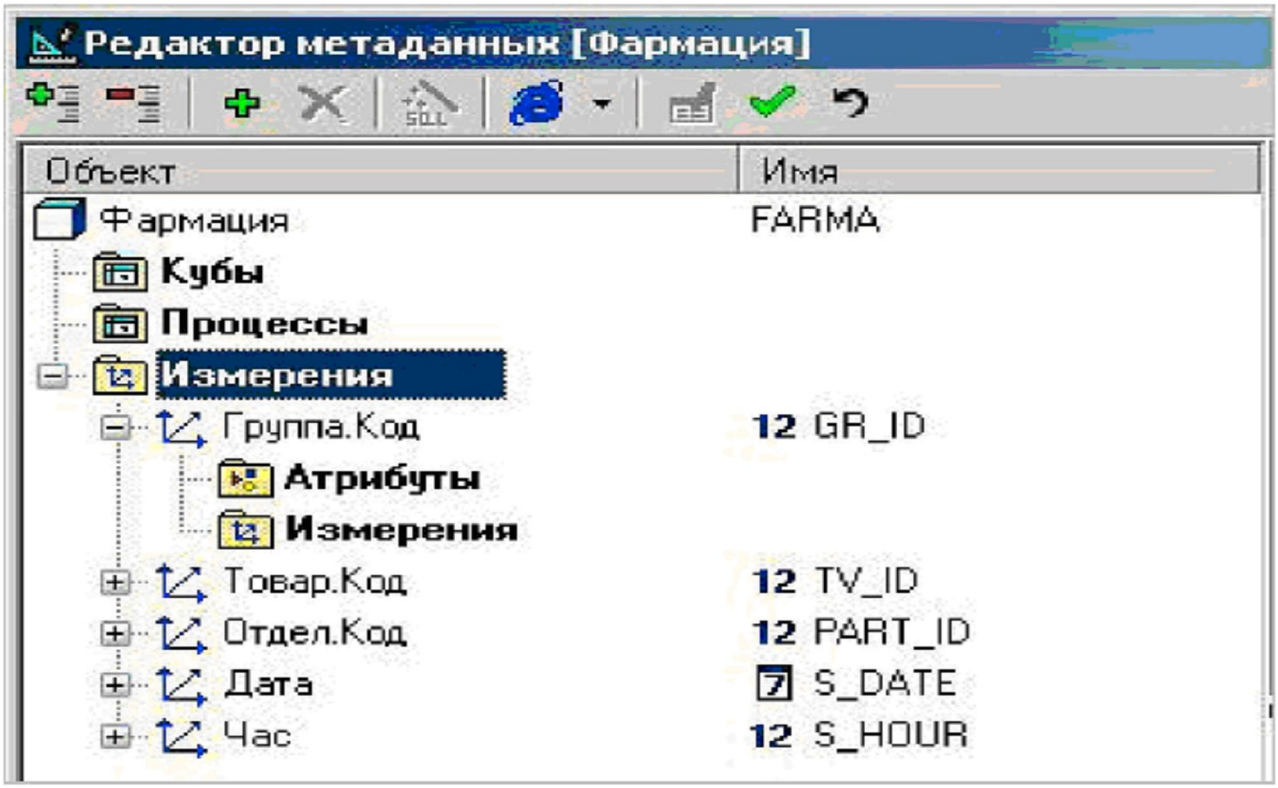
Имя - это семантическое название объекта хранилища данных, которое увидит пользователь, работающий с ХД. (Эти параметры для таблицы «Товарные группы»).

Выполните аналогичные действия для создания всех остальных измерений, взяв параметры из таблицы 1.

Таблица 1. - Параметры измерений

Имя	**Метка**	**Тип данных**
Измерение		
Код группы	GR-ID	Группа.Код	целый
Код товара	TV_ID	Товар.Код	целый
Код отдела	PART_ID	Отдел.Код	целый
Дата	S_DATE	Дата	дата/время
Час покупки	S_HOUR	Час	целый

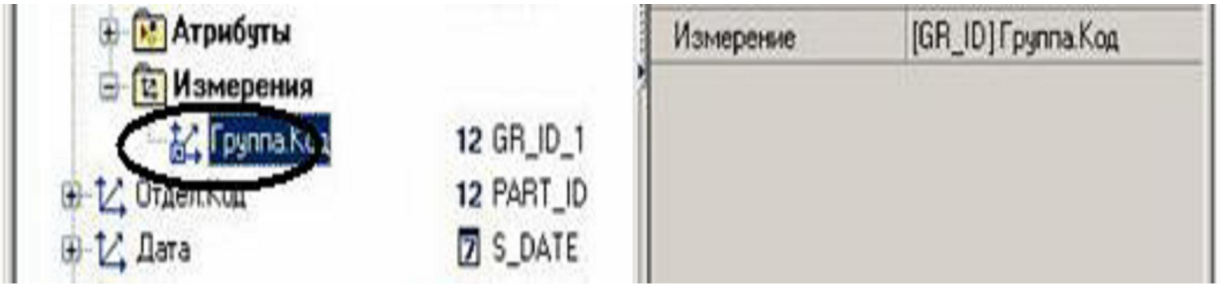
В результате структура метаданных нашего хранилища будет содержать 5 измерений.



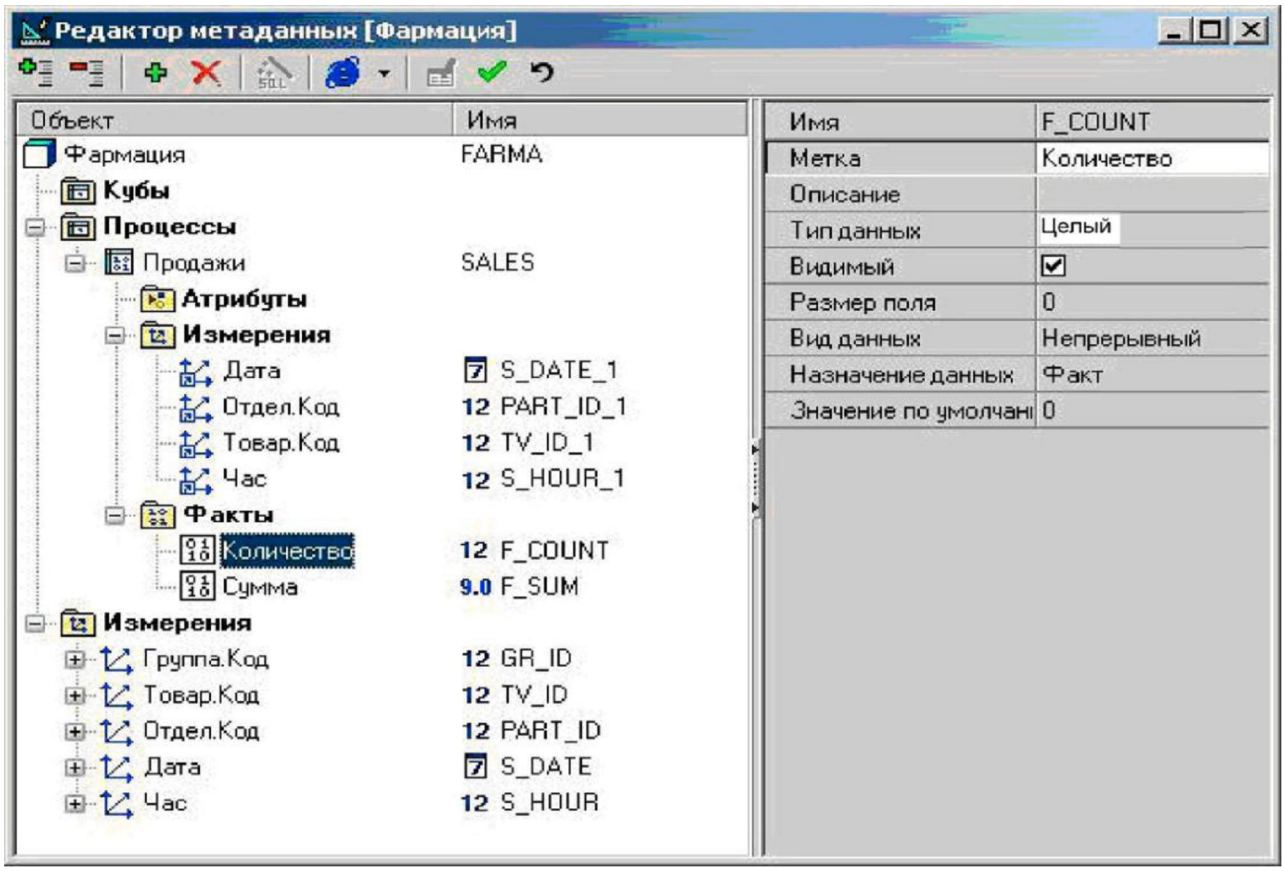
К каждому измерению, кроме Дата и Час, теперь добавим по текстовому атрибуту. Для этого в измерении «Группа.Код» правой кнопкой мыши откроем **Атрибуты** и справа в поле «Метка» введем название атрибута - Группа.Наименование. Тип данных оставим строковым. Размер поля в строковых атрибутах предлагается равным 100, оставим это без изменений. Аналогично введите названия атрибутов :для измерения Товар.Код - Товар.Наименование, для измерения

Отдел.Код - Отдел.Наименование..

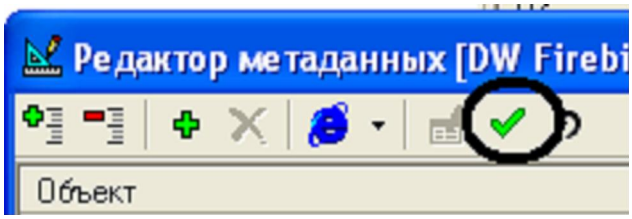
Каждое измерение может ссылаться на другое измерение, реализуя тем самым иерархию измерений (схема «снежинка»). В нашем случае измерение Товар.Код ссылается на Группа.Код (см. табл. 1 и табл. 2). Эту ссылку и установим путем добавления объекта к измерению, для этого в измерении «Товар.Код» правой кнопкой мыши откроем Измерение и выберем пункт Добавить. Имя ссылки зададим GR_ID_1, а метку - Группа.Код. Ссылка на измерение отображается иконкой .



После того как все измерения и ссылки на измерения созданы, приступают к формированию процесса. Назовем его *Продажи* и «соберем» его из 4 существующих измерений: Дата, Отдел.Код, Товар.Код, Час (кнопка). Кроме них в нашем процессе присутствуют два факта: Количество и Сумма, причем первый - целочисленный, второй – вещественный. Результат представлен на рисунке 3.



На этом проектирование структуры и метаданных ХД закончено. Для того чтобы принять все изменения, нужно нажать кнопку **Принять изменения**



После этого закройте окно редактора. Структура хранилища данных готова.

Задание. Для выбранной предметной области сформировать хранилище данных, заполнить его. Привести примеры вывода данных.

Содержание отчета

1. Цель работы.
2. Ход работы.
3. Ответы на вопросы.
4. Листинг программы.
5. Заключение.

Вопросы

1. Что такое «Редактор метаданных» в DeductorStudio?

2. Как создать новое пустое хранилище данных?

3. Как сделать иерархию измерений?

4. Какие типы данных могут быть у объектов хранилища?

5. Чем факт отличается от измерения?

ЛАБОРАТОРНАЯ РАБОТА №3. Поиск ассициативных правил

Цель работы. Изучить возможность поиска ассоциативных правил используя аналитическую платформу **DEDUCTOR**

Теоретическая часть. В последнее время растет интерес к методам «обнаружения знаний в базах данных». Большие объемы современных баз данных выззвают спрос на новые алгоритмы распознавания и обработки данных. Одним из распространенных аналитических методов обработки данных является аффинитивный анализ (англ: affinityanalysis), название произошедшее от английского слова affinity – близость, сходство. Метод определяет взаимные связи между событиями, происходящие совместно. Одним из применения аффинитивного анализа является анализ рыночной корзины (англ: marketbasketanalysis), цель которого – обнаружить ассоциации между различными данными, т.е. найти правила для количественного описания взаимной связи между двумя или более данными. Такие правила называются **ассоциативными правилами** (англ.: associationrules) и применяются в data mining.

Примерами приложения ассоциативных правил могут быть следующие задачи:

1. Обнаружение наборов товаров, которые в супермаркетах часто покупаются вместе или никогда не покупаются вместе.

2. Определение доли клиентов, положительно относящихся к нововведениям в их обслуживании.

3. Определение профиля посетителя веб-ресурса.

4. Определение доли случаев, в которых новое лекарство показывает опасный побочный эффект.

Базовым понятием в теории ассоциативных правил является транзакция. Транзакция – некоторое множество событий, происходящих совместно. Типичной транзакцией является приобретение клиентом некоторого товара в супермаркете. В табл. 1 представлен простой пример набора транзакций. В каждой строке содержится комбинация продуктов, приобретенных за одну покупку.

Таблица 1. Пример набора транзакций.

№ транзакции	**Товары**
1	сливы, салат, помидоры
2	сельдерей, конфеты
3	конфеты
4	яблоки, морковь, помидоры, картофель, конфеты
5	яблоки, апельсины, салат.конфеты, помидоры
6	персики, апельсины, сельдерей, помидоры
7	фасоль, салат, помидоры
8	апельсины, салат.помидоры
9	яблоки, сливы, морковь, помидоры, лук, конфеты
10	яблоки, картофель

На практике обрабатываются миллионы транзакций, в которых участвуют десятки и сотни различных продуктов. Данный пример ограничен 10 транзакциями, содержащими 13 видов продуктов, что достаточно для иллюстрации методики обнаружения ассоциативных правил. В большинстве случаев клиент приобретает не один товар, а некоторый набор товаров, называемых рыночной корзиной. Существует связь между спросом на товары, которую может обнаружить ассоциативное правило, утверждающее, что покупатель, купивший молоко, с вероятностью 75% купит и хлеб. Такие связи сущетвуют и в других областях, например в медицинской или технической диагностике, выборе профессий и т.д.

Анализ рыночной корзины – это анализ наборов данных для определения комбинаций товаров, связанных между собой. Иными словами, производится поиск товаров, присутствие которых в транзакции влияет на вероятность наличия других товаров или комбинаций товаров [4].

Современные кассовые аппараты в супермаркетах позволяют собирать информацию о покупках, которая может храниться в базе данных. Накопленные данные затем могут использоваться для построения систем поиска ассоциативных правил.

Визуальный анализ примера (табл.1) показывает, что все четыре транзакции, в которых фигурирует салат, также включают и помидоры, и что четыре из семи транзакций, содержащих помидоры, также содержат и салат. Салат и помидоры в большинстве случаев покупаются вместе. Ассоциативные правила позволяют обнаруживать и количественно описывать такие совпадения.

Ассоциативное правило состоит из двух наборов предметов, называемых условие (англ: antecedent) и следствие (англ: consequent), записываемых в виде $X \rightarrow Y$, что читается «из X следует Y». Таким образом, ассоциативное правило формулируется в виде «Если условие, то следствие».

Условие часто ограничивают содержанием только одного предмета. Правила обычно отображаются с помощью стрелок, направленных от условия к следствию, например, (помидоры) → (салат). Условие и следствие часто называются соответственно левосторонним (LHS – left-handside) и правосторонним (RHS – right-handside) компонентом ассоциативного правила.

Ассоциативные правила описывают связь между наборами предметов, соответствующим условию и следствию. Эта связь характеризуется двумя показателями – поддержкой и достоверностью.

Обозначим Dкак базу данных транзакций, а Nкак число транзакций в этой базе. Каждая транзакция Diпредставляет собой некоторый набор предметов.

Зададим, что S(англ.: support) – поддержка, C (англ.: confidence) – достоверность.

Поддержка ассоциативного правила – это число транзакций, содержащих как условие, так и следствие.

Например, для ассоциации A→Bможно записать:

$$S(A \to B) = P(A \cap B) = \frac{\text{количество транзакций, содержащих A и B}}{\text{общее количество транзакций}}$$

Достоверность ассоциативного правила- это мера точности правила, которая определяется как отношение количества транзакций, содержащих как условие, так и следствие, к количеству транзакций, содержащих только условие.

Например, для ассоциации A→B можно записать:

$$C(A \to B) = P(A \mid B) = P(A \cap B)/P(A) = \frac{\text{количество транзакций, содержащих A и B}}{\text{количество транзакций, содержащих только A}}$$

Если поддержка и достоверность достаточно высоки, то это позволяет с большой вероятностью утверждать, что любая будущая транзакция, которая включает условие, будет также содержать и следствие.

Рассмотрим пример для вычисления поддержки и достоверности для ассоциаций из табл.1. Возьмем ассоциацию (салат) →(помидоры). Поскольку количество транзакций, содержащее как (салат), так и (помидоры), равно 4, а общее число транзакций 10, то поддержка данной ассоциации будет:

S((салат)→ (помидоры)) = 4/10 = 0,4 .

Поскольку Поскольку количество транзакций, содержащее только (салат) как условие, равно 4, то достоверность данной ассоциации будет:

C((салат) →(помидоры)) = 4/4 = 1.

Все наблюдения, содержащие салат, также содержат и помидоры, что позволяет сделать вывод о том, что данная ассоциация может рассматриваться как правило. С точки зрения интуитивного поведения такое правило вполне объяснимо, поскольку оба продукта широко используются для приготовления растительных блюд и часто покупаются вместе.

Рассмотрим ассоциацию (конфеты) →(помидоры), в которой содержатся, в общем-то, слабо совместимые в гастрономическом плане продукты (тот, кто планирует сделать растительное блюдо, вряд ли станет покупать конфеты, а покупатель, желающий приобрести что-нибудь к чаю, скорее всего, не станет покупать помидоры). Поддержка данной ассоциации S = 3/10 = 0,3 , а достоверность**C**= 3/7 = 0,43. Таким образом, сравнительно невысокая достоверность данной ассоциации дает повод усомниться в том, что она является правилом.

Аналитики могут отдавать предпочтение правилам, которые имеют только высокую поддержку или только высокую достоверность, либо, что является наиболее частым, оба эти показателя. Правила, для которых значения поддержки или достоверности превышают некоторый, заданный пользователем порог, называются сильными правилами (strongrules). Например, аналитика может интересовать, какие товары в супермаркете, покупаемые вместе, образуют ассоциации с минимальной поддержкой 20% и минимальной достоверностью 70 %. С другой стороны, при анализе с целью обнаружения мошенничеств, аналитику может потребоваться уменьшение поддержки до 1%, поскольку сравнительно небольшое число транзакций являются связанными с мошенничеством.

Значимость ассоциативных правил

Методики поиска ассоциативных правил обнаруживают все ассоциации, которые удовлетворяют ограничениям на поддержку и достоверность, наложенные пользователем. Это часто приводит к необходимости рассмотреть десятки и сотни тысяч ассоциаций, что делает невозможным «ручную» обработку такого большого количества данных. Очевидно, что желательно уменьшить число правил таким образом, чтобы проанализировать только наиболее значимые правила. Часто значимость связана с разностью между поддержкой правила в целом и произведением поддержки только условия и поддержки только следствия.

Выделяют объективные и субъективные меры значимости правил. Объективными являются такие меры, как поддержка и достоверность, которые могут применяться независимо от конкретного приложения. Субъективные меры связаны со специальной информацией, определяемой пользователем в контексте решаемой задачи. Такими субъективными мерами являются **лифт** (англ: lift) и **левередж**(от англ. leverage- плечо, рычаг).

Лифт – это отношение частоты появления условия в транзакциях, которые также содержат и следствие, к частоте появления следствия в целом. Лифт (оригинальное название - интерес) определяется следующим образом: $L(A \rightarrow B) = C(A \rightarrow B)/S(B)$. Значения лифта большие, чем единица показывают, что условие более часто появляется в транзакциях, содержащих и следствие, чем в остальных. Можно сказать, что лифт является обобщенной мерой связи двух предметных наборов: при значениях лифта >1 связь положительная, при 1 она отсутствует, а при значениях <1 -отрицательная.

Другой мерой значимости правила является **левередж**(англ: leverage; предложена Г. Пятецким-Шапиро):

Левередж – это разность между наблюдаемой частотой, с которой условие и следствие появляются совместно (т.е., поддержкой ассоциации), и произведением частот появления (поддержек) условия и следствия по отдельности.

$$\mathbf{L}(\mathbf{A} \rightarrow B) = S(A \rightarrow B) - S(A)S(B).$$

Такие меры, как лифт и левередж могут использоваться для последующего ограничения набора рассматриваемых ассоциаций путем установки порога значимости, ниже которого ассоциации отбрасываются.

Генерация ассоциативных правил

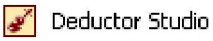
В DeductorStudio для решения задач ассоциации используется обработчик **Ассоциативные правила**. В нем реализован алгоритм apriori. Обработчик требует на входе два поля: идентификатор транзакции и элемент транзакции.

Например, идентификатор транзакции – это номер чека или код клиента. А элемент - это наименование товара в чеке или услуга, заказанная клиентом.

Оба поля (идентификатор и элемент транзакции) должны быть дискретного вида.

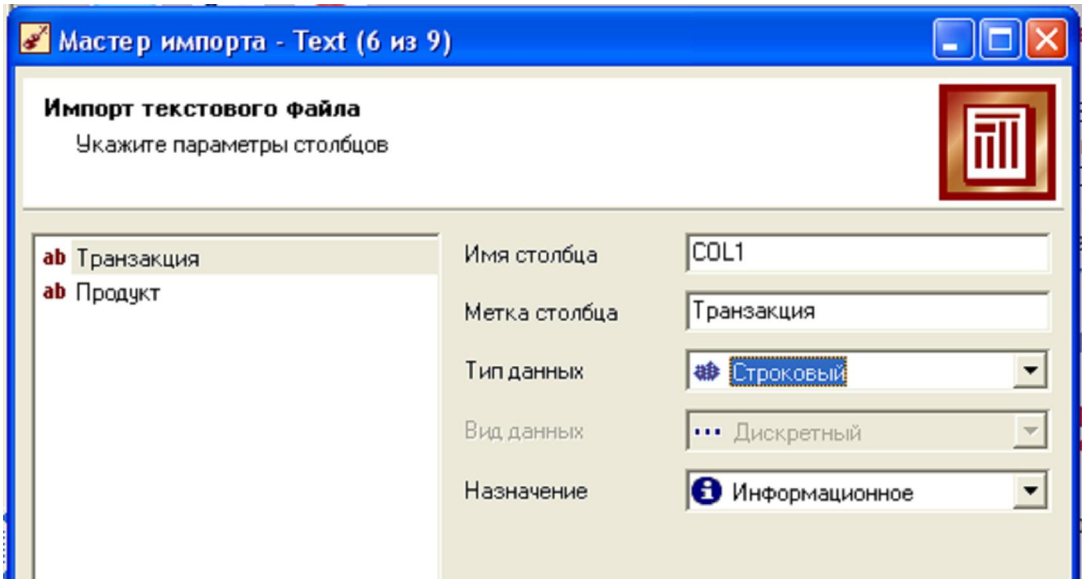
Пример решения конкретной задачи ассоциации из области розничной торговли:

- предсказать то, какие товары покупатели могут выбрать в зависимости от того, что уже есть в их корзинах;
- предложить рекламные акции типа «Каждому купившему товары А и В, товар С в подарок».

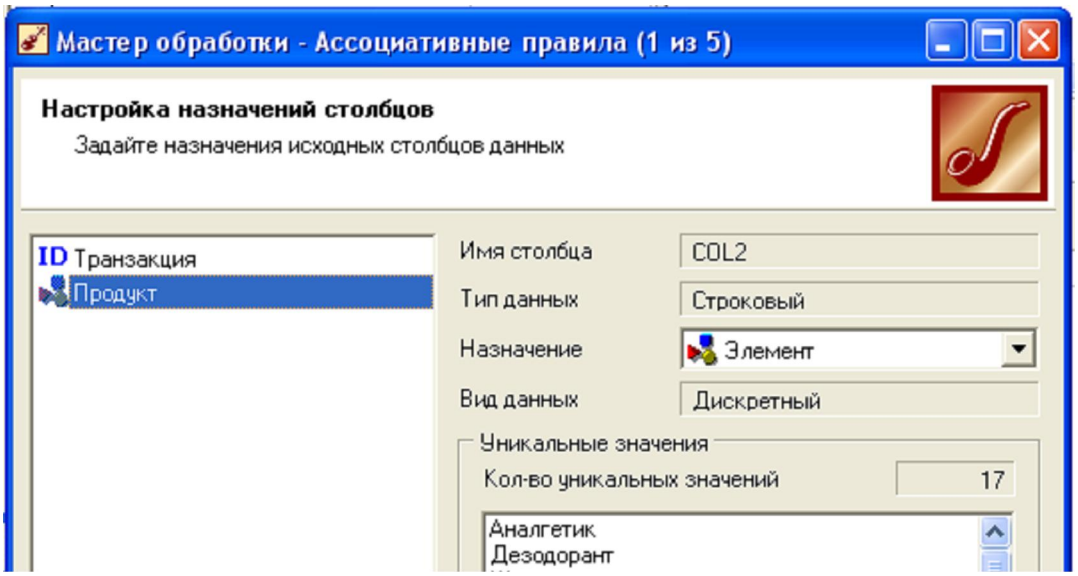
Откройте программу DeductorStudio 

, используя ярлык на рабочем столе или через кнопку Пуск.

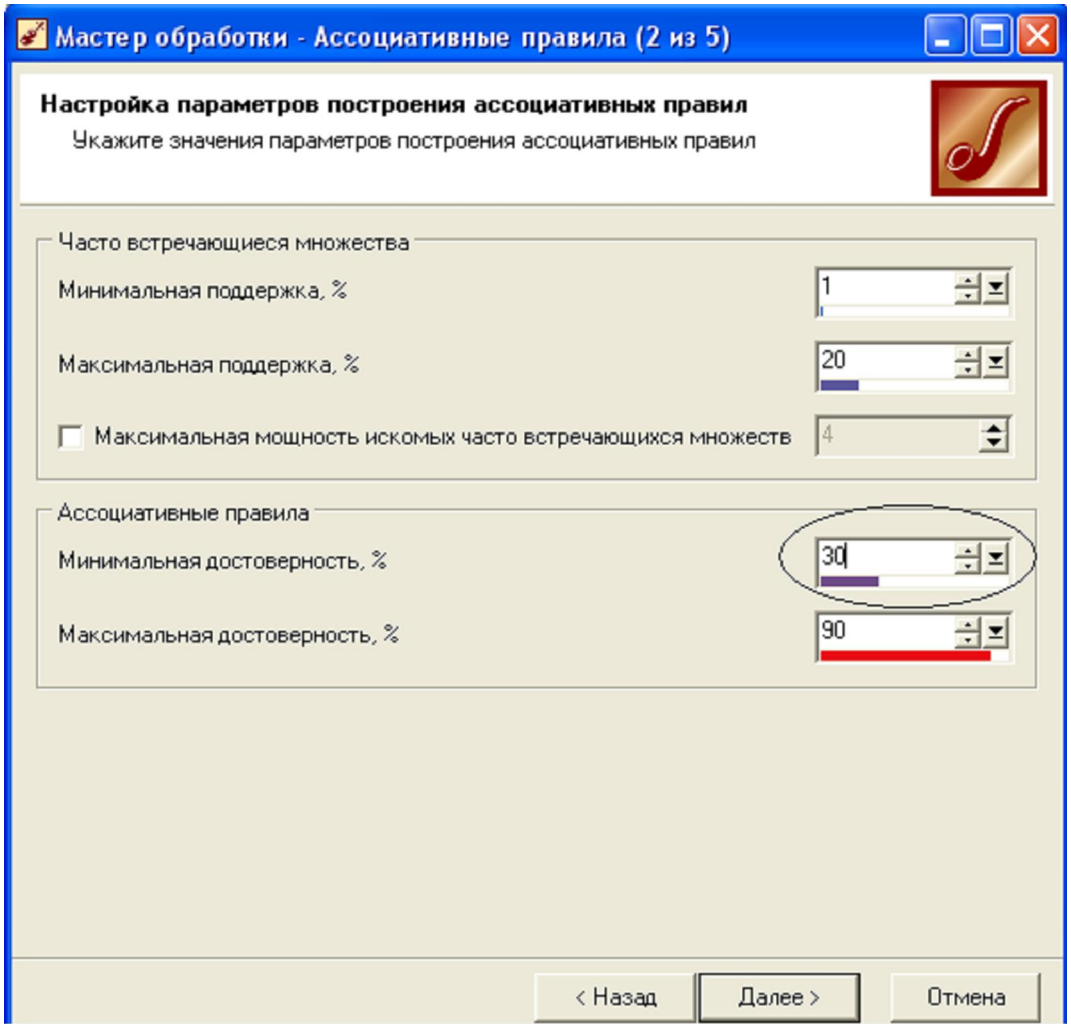
Импортируйте данные из текстового файла transactions.txt в DeductorStudio. В файле данных имеются два столбцаТранзакция и Продукт, для которых нужно Тип поля нужно установить строковый.



После импорта к данному загруженному файлу применим обработчик **Ассоциативные правила**. Столбец Транзакция сделаем идентификатором транзакции, а столбецПродукт – ее элементом:



На следующем шаге мастера настроим параметры построения ассоциативных правил, что, по сути, есть



параметры алгоритма apriori:

Здесь для изменения доступны следующие параметры.

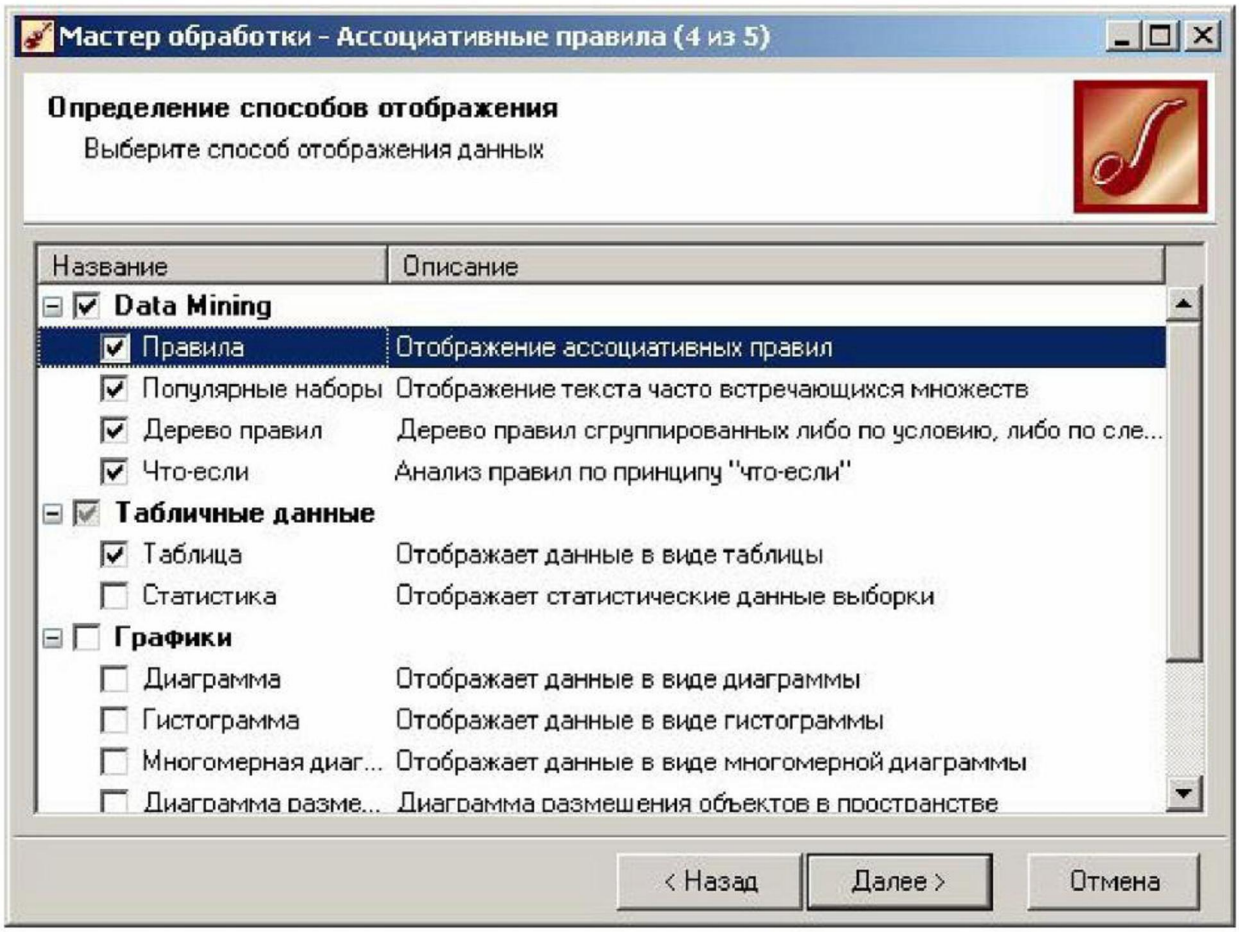
Минимальная и максимальная поддержка в % – ограничивают пространство поиска часто встречающихся предметных наборов. Эти границы определяют множество популярных наборов, из которых и будут создаваться ассоциативные правила.

Минимальная и максимальная достоверность в % – в результирующий набор попадут только те ассоциативные правила, которые удовлетворяют условиям минимальной и максимальной достоверности.

Максимальная мощность искомых часто встречающихся множеств – параметр ограничивает длину k-предметного набора. Например, при установке значения 4 шаг генерации популярных наборов будет остановлен после получения множества 4-предметных наборов. В конечном итоге это позволяет избежать появления длинных ассоциативных правил, которые трудно интерпретируются.

Нажмите на кнопку **Пуск,** что приведет к работе алгоритма поиска ассоциативных правил. По окончании его работы справа в полях появится следующая информация:

Далее выбираем все доступные специализированные визуализаторы DataMining и визуализатор Таблица:



Все эти визуализаторы, кроме **Что-если**, отображают результаты работы алгоритма в различных формах.

На вкладке **Правила** помимо самих ассоциативных правил приводятся их основные расчетные характеристики:

Правила | Популярные наборы | Дерево правил | Что-если | Таблица

Правил: 6 из 6 | Фильтр: Без фильтрации

№	Номер правила	Условие	Следствие	Поддержка		Достоверность	Лифт
				Кол-во	%		
1	1	Зубная щетка	Парфюм	446	2,23	35,20	3,952
2	2	Зубная паста Карандаши	Конфеты	218	1,09	45,04	2,618
3	3	Карандаши Конфеты	Зубная паста	218	1,09	33,90	2,125
4	4	Зубная паста Конфеты	Поздравительная открыт	272	1,36	34,61	2,288
5	5	Зубная паста Поздравительная открыт	Конфеты	272	1,36	40,60	2,360
6	6	Конфеты Поздравительная открыт	Зубная паста	272	1,36	30,63	1,920

поддержка, достоверность и лифт.

На вкладке **Популярные наборы** отображается множество найденных популярных предметных наборов в виде списка. Кнопка предлагает на выбор несколько вариантов сортировки списка, а кнопка * вызывает окно настройки фильтра множеств. Например, задав в фильтре минимальное значение поддержки 3% и отсортировав их по убыванию поддержки, получим 17 популярных наборов (на картинке изображено только 12):

Правила | Популярные наборы | Дерево правил | Что-если | Таблица

Множеств: 17 из 26 | Фильтр: Минимальная поддержка = 3,00

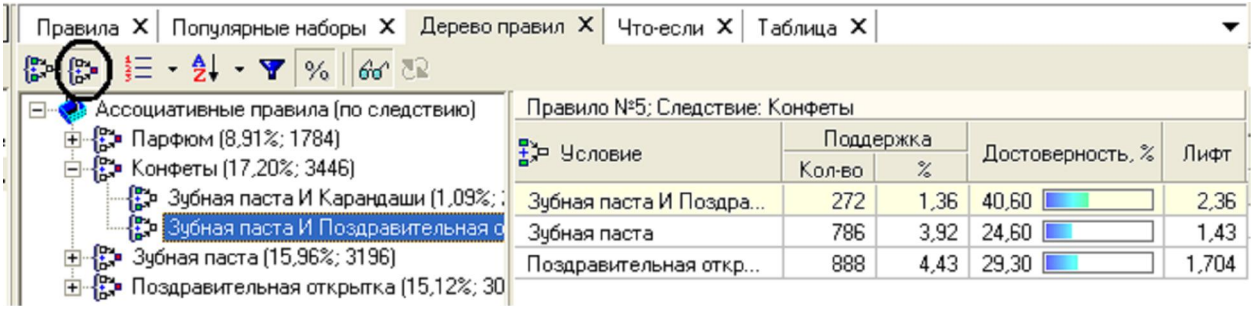
№	Номер множества	Элементы	Поддержка		Мощность
			Кол-во	%	
1	6	Конфеты	3446	17,20	1
2	2	Зубная паста	3196	15,96	1
3	13	Поздравительная открытка	3029	15,12	1
4	10	Набор ручек	2922	14,59	1
5	4	Карандаши	2714	13,55	1
6	12	Парфюм	1784	8,91	1
7	3	Зубная щетка	1267	6,33	1
8	5	Карта флеш-памяти	1198	5,98	1
9	7	Лекало	1093	5,46	1
10	11	Оберточная бумага	1025	5,12	1
11	24	Конфеты Поздравительная открытка	888	4,43	2
12	9	Мыло	832	4,15	1

выявить наиболее популярные товарные наборы, состоящие из более, чем 1 предмета;

На вкладке **Дерево правил** предлагается еще один удобный способ отображения множества ассоциативных правил, которое строится 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. В дереве, построенном по следствию, наоборот, на первом уровне располагаются узлы со следствием.

Справа от дерева расположен список правил, построенный по выбранному узлу дерева, например по правилу №5:

предложить рекламные акции типа «Каждому купившему товары A и B, товар C в подарок».



Для каждого правила отображаются поддержка и достоверность и лифт. Если дерево построено по условию, то вверху списка отображается условие правила, а список состоит из его следствий. Тогда правила отвечают на вопрос, что будет при таком условии. Если же дерево построено по следствию, то вверху списка отображается следствие правила, а список состоит из его условий. Эти правила отвечают на вопросы, что нужно, чтобы было заданное следствие или какие товары нужно продать для того, чтобы продать товар из следствия.

Сохраните сценарий под именем **assosiation.ded**.

СОДЕРЖАНИЕ ОТЧЕТА

1. Цель работы.
2. Краткое описание хода работы.
3. Исходые данные
4. Выявленные ассоциативные правила
5. Ответы на вопросы.
6. Заключение.

Вопросы:

1. Какой алгоритм генерации ассоциативных правил имеется в Deductor?

2. Какие входные поля набора данных необходимы для запуска обработчика Ассоциативные правила в Deductor?

3. Какие специализированные визуализаторы предлагаются к узлу-обработчику Ассоциативные правила?

Лабораторная работа №4. Распознавание образов данных (Сеть Хемминга)

Цель работы. Изучение функционирования нейроподобных элементов в виде сети Хемминга. Разработка программы для разпознавания образов при преобразовании информации.

Общие сведения

В настоящее время дальнейшее повышение производительности компонентов связывает с системами, обладающими свойствами массового параллелизма.

Одна из таких систем – это нейрокомпьютер, использующий искусственную нейросеть. Искусственная нейросеть (ИНС) – это параллельная структура, которая естественным образом реализует принцип потока данных. Обычно под ИНС понимается набор элементарных нейроподобных преобразователей информации – нейронов, соединенных друг с другом каналами обмена информацией для их совместной работы.

Сформировались две ветви исследований. Первая, нейробиологическая, основывается на моделировании работы живого мозга, имея цель объяснить, каким образом в нем отображаются сложные объекты и связи между ними, как устанавливается соответствие между хранящейся и поступающей извне информацией, и другие вопросы, касающиеся функционирования мозга. Второе направление исследований направлено на решение с помощью ИНС задач переработки информации в различных областях знаний, особенно плохо формализованных, где существующие модели субъективны и неадекватны. Впечатляющие результаты использования ИНС достигнуты при распознавании образов, при построении ассоциативной памяти, при создании самообучающихся Экспертных систем, при решении оптимизационных задач большой размерности.

Предложено и изучено большое количество моделей нейросетей. основными являются три типа сетей, которые соответствуют трем известным методам обучения: самоорганизации, последовательному подкреплению знаний, обучению с учителем.

Теоретическая часть. Сеть Хемминга (СХ) представляет сеть с двухслойной топологией, прямой связью между слоями и с обучением с супервизором. Число нейронов N на входном слое равно размерности векторов памяти, а число нейронов в выходном слое равно числу M векторов памяти.

Сеть состоит из двух слоев. Первый и второй слои имеют по m нейронов, где m – число образцов. Нейроны первого слоя имеют по n синапсов, соединенных со входами сети (образующими фиктивный нулевой слой). Нейроны второго слоя связаны между собой ингибиторными

(отрицательными обратными) синаптическими связями. Единственный синапс с положительной обратной связью для каждого нейрона соединен с его же аксоном.

Работы сети заключается в нахождении расстояния Хэмминга от тестируемого образа до всех образцов. Расстоянием Хэмминга называется число отличающихся битов в двух бинарных векторах. Сеть должна выбрать образец с минимальным расстоянием Хэмминга до неизвестного входного сигнала, в результате чего будет активизирован только один выход сети, соответствующий этому образцу.

На стадии инициализации весовым коэффициентам первого слоя и порогу активационной функции присваиваются следующие значения:

$w_{ik} = \frac{x_i^k}{2}$, *i=0...n-1, k=0...m-1* (5)

$T_k = n / 2$, *k = 0...m-1* (6)

Здесь x_i^k – i-ый элемент k-ого образца.

Весовые коэффициенты тормозящих синапсов во втором слое берут равными некоторой величине 0 < □ < 1/m. Синапс нейрона, связанный с его же аксоном имеет вес +1.

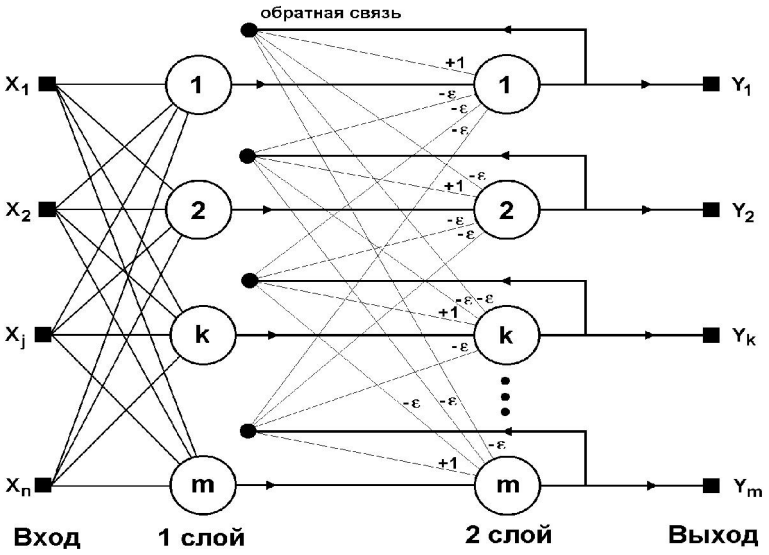


Рис.1 . Схема сети Хемминга

Ход работы. Алгоритм функционирования сети Хэмминга следующий:

1. На входы сети подается неизвестный вектор X = {x_i:i=0...n-1}, исходя из которого рассчитываются состояния нейронов первого слоя (верхний индекс в скобках указывает номер слоя):

$y_j^{(1)} = s_j^{(1)} = \sum_{i=0}^{n-1} w_{ij} x_i + T_j$, *j=0...m-1* (7)

После этого полученными значениями инициализируются значения аксонов второго слоя: $y_j^{(2)} = y_j^{(1)}$, *j = 0...m-1*

2. Вычислить новые состояния нейронов второго слоя:

$$s_j^{(2)}(p+1) = y_j(p) - \varepsilon \sum_{k=0}^{m-1} y_k^{(2)}(p), k \neq j, j = 0...m-1$$

и значения их аксонов:

$$y_j^{(2)}(p+1) = f\left[s_j^{(2)}(p+1)\right], j = 0...m-1$$

Активационная функция f имеет вид порога (рис. 2б), причем величина F должна быть достаточно большой, чтобы любые возможные значения аргумента не приводили к насыщению.

3. Проверить, изменились ли выходы нейронов второго слоя за последнюю итерацию. Если да – перейди к шагу 2. Иначе – конец.

Из оценки алгоритма видно, что роль первого слоя весьма условна: воспользовавшись один раз на шаге 1 значениями его весовых коэффициентов, сеть больше не обращается к нему, поэтому первый слой может быть вообще исключен из сети (заменен на матрицу весовых коэффициентов), поэтому так можно сделать в ее конкретной реализации,

Сеть классифицирует произвольные бинарные или аналоговые образы $x = (x_1 ... x_n)$ в один из M классов. При этом начальное значение $y^j(0)$ нейронов в выходном слое определяется двумя способами в зависимости от характера векторов памяти. Но в обоих случаях вектор стимула x с начало нормируется.

Если векторы памяти являются бинарными, то $y^j(0)$ соответствует перекрытиям нормированного вектора стимула с нормированными векторами памяти. Если векторы памяти являются аналоговыми, то $y^j(0)$ выбирается в соответствии с величиной расстояния Хемминга между нормированными векторами памяти и стимула, с помощью пороговой функции F.

После сформирования начальных значений нейронов выходного слоя (в этом слое все нейроны связаны между собой) выполняются итерации, про которых самодействие каждого нейрона является положительным, а вклад остальных нейронов этого слоя отрицателен. С помощью итераций выделяется тот нейрон, у которого значение $y^j(0)$ было максимальным, т. е. итерации прекращаются, когда только один из нейронов имеет ненулевое значение, а номер этого нейрона и определяет результат классификации.

4. Алгоритм программы

1. Инициализация весов (можно взять из файла “obraz.txt”).
2. Ввод распознаваемого образа.
3. Определение кодовое расстояние $d[j]$.
4. Определение максимального сходство искомого образа с одним из исходных.

$$y[j] = porog - d[j].$$

5. Вывод результата.

Пример результата работы сети Хемминга при разпознавании образоа цифры.

1. Введите образ:

1111
1111
0011
0011
0011
0011

Результат: 7

Введите образ:

1111
0011
1111
1111
1100
1111

2. Результат: 2

Содержание отчета

1. Цель работы.
2. Краткое описание хода работы.
3. Ответы на вопросы.
4. Листинг программы
5. Заключение.

ВОПРОСЫ

1. Приведите примеры использовния сети Хемминга.
2. Сколько слоев имеет сеть Хемминга?
3. Какую роль играют обратные связи?
4. Каким образом определяется распонаваемый образ?
5. Какой вид имеет авктивационная функция для сети Хеммнга?
6. Совпадает ли количество входов и выходов в сети Хемминга?

Приложение:

1. Варианты интерфейса с результатами работы программы

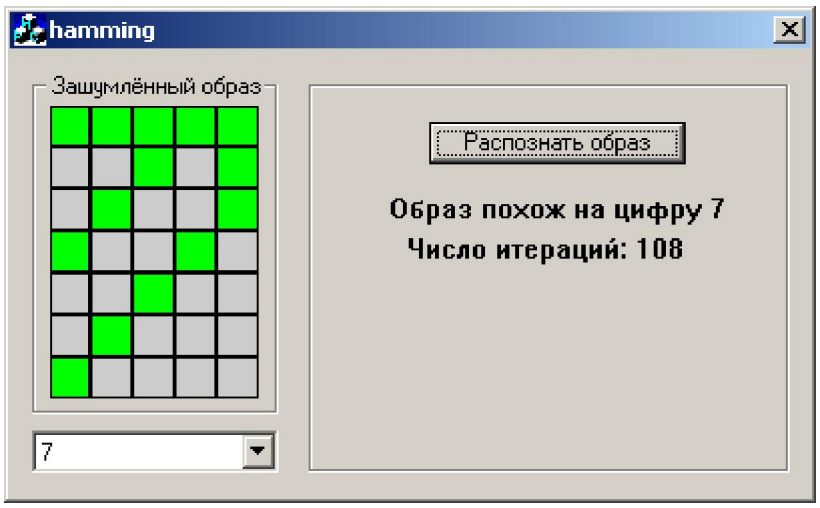


Рис. 2. Зашумлённая цифра 7.

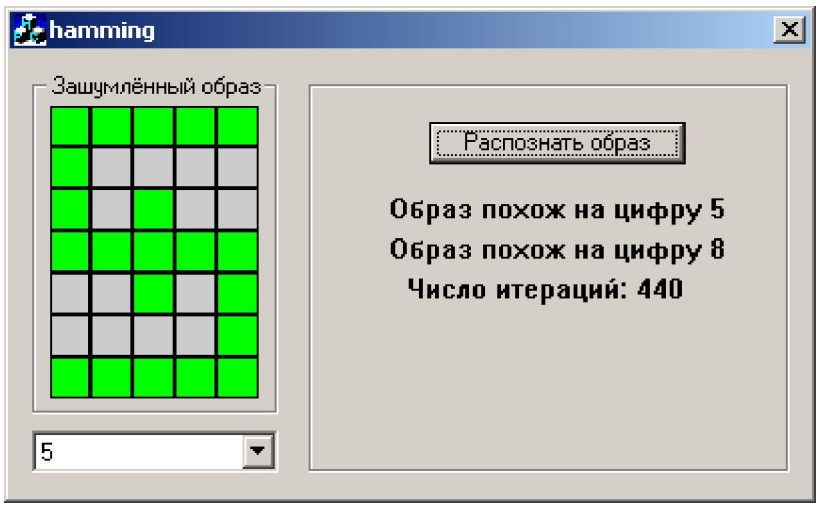
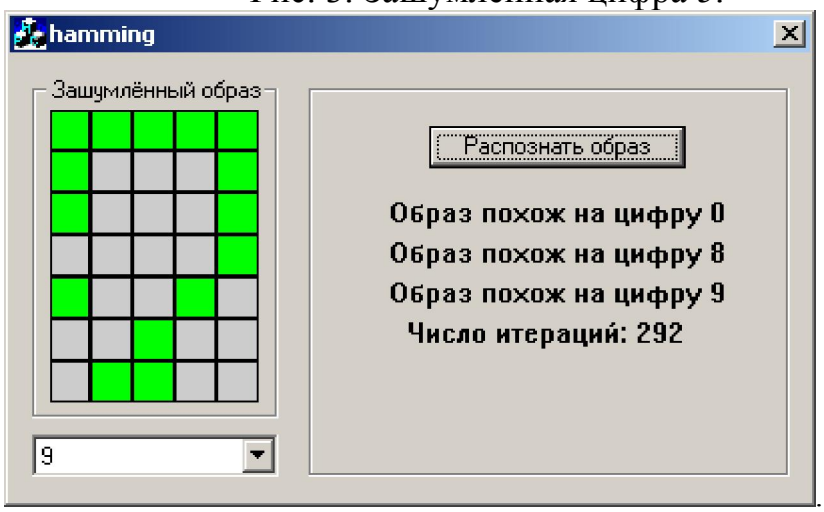


Рис. 3. Зашумлённая цифра 5.



1. Листинг процедур

Процедура распознавания зашумлённого образа

```
void CHammingDlg::OnRecognition()
```

```
{
	// TODO: Add your control notification handler code here
	Invalidate();
	UpdateWindow();

	int k, i, j, iter=0;
	for(k=0; k<m_N; k++) {y1[k]=y2[k]=y2new[k]=tmp[k]=0.0;}

	//Инициализация сети
	T = m_m*m_n/2.0; srand((unsigned)time(NULL));
	 double e = rand()%((m_m*m_n))/(m_m*m_n*100.0);

	//Вычисление состояния нейронов первого слоя
	for(k=0; k<m_N; k++) {
		for(i=0; i<m_m; i++)
			for(j=0; j<m_n; j++)
				y1[k] += m_image[i][j] * m_w[k][i*m_n+j];
		y1[k]+=T;
	}

	//Инициализация значений аксонов второго слоя полученными
значениями
	for(k=0; k<m_N; k++) y2new[k] = y1[k];

	do
	{
		for(k=0; k<m_N; k++) y2[k] = y2new[k];

		//Вычислить новые состояния нейронов второго слоя
		for(k=0; k<m_N; k++) {
			double sum=0;
			for(j=0; j<m_N; j++) if(j!=k) sum += y2[j];
			y2new[k] = y2[k] - e * sum;
		}

		//Вычислить значения аксонов второго слоя
		for(k=0; k<m_N; k++) {
			if(y2new[k]<0) y2new[k]=0;
			else if(y2new[k]>=m_m*m_n) y2new[k]=m_m*m_n;
		}
		iter++;
	}
	while(Change());

	//Определение схожих образов
	double etalon=-100; int index[10], l=0;
	for(k=0; k<m_N; k++) index[k] = -1;
	for(k=0; k<m_N; k++)
		if(y2new[k]>etalon)
			{etalon = y2new[k]; index[0] = k;}

	for(k=0; k<m_N; k++)
		if(fabs(y2new[k]-etalon)<0.001 && k!=index[0])
			index[++l] = k;

	CDC *pdc2 = m_results.GetDC();
	pdc2->SetBkMode(TRANSPARENT);
	char str[50]={0}; int s=0;
	for(k=0; k<m_N; k++) {
		if(index[k]>-1) {
			sprintf(str,"Образ похож на цифру %d", index[k]);
			pdc2->TextOut(40, 60+20*s, str); s++;
		}
```

```
      }
      sprintf(str,"Число итераций: %d", iter);
      pdc2->TextOut(50, 60+20*s, str); s++;
      m_results.ReleaseDC(pdc2);

}
```

Процедура проверки изменения выходов нейронов второго слоя

```
int CHammingDlg::Change()
{
      for(int k=0; k<m_N; k++)
            if(fabs(y2new[k]-y2[k])>0.001) return 1;
      return 0;
}
```

Считывание эталонных образов из файла

```
FILE *f;
if((f=fopen("cifir.txt","r"))==NULL){printf("Невозможно открыть
файл!\n"); return FALSE;}
      for(int k=0; k<m_N; k++)
            for(i=0; i<m_m; i++)
                  for(int j=0; j<m_n; j++)
                        fscanf(f, "%d", &m_etalon[k][i*m_n+j]);
fclose(f);
```

Лабораторная работа № 5. Кластерная обработка данных (карты Кохонена)

Цельработы. Научиться использовать метод кластерной обработки данных в виде самоорганизующихся карт Кохонена».

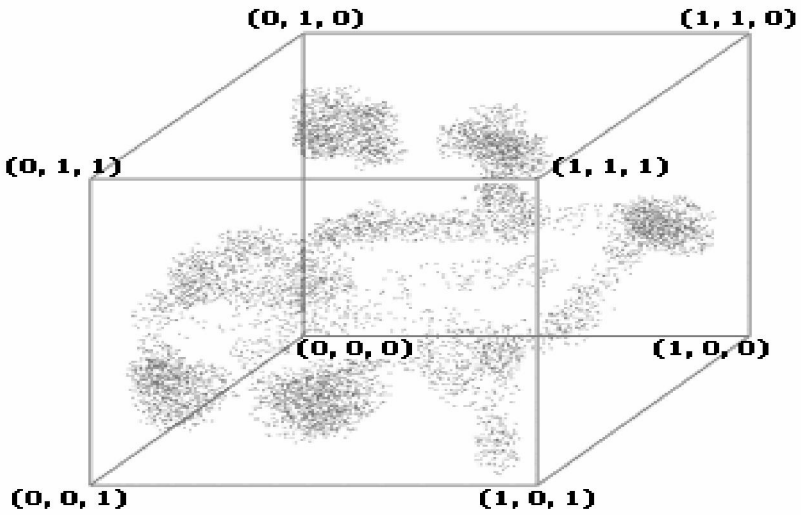
Теоретическая часть. Существуют задачи анализа данных, которые затруднительно представить в числовой форме. При этом нужно извлечь данные, принципы отбора которых заданы нечетко: выделить надежных партнеров, определить перспективный товар и т.п. Также необходимо на основании имеющихся априорных данных получить прогноз на дальнейший период. Существует метод, позволяющий автоматизировать все действия по поиску закономерностей – метод анализа с использованием самоорганизующихся *карт Кохонена*.

Самоорганизующаяся карта Кохонена (англ. Self-organizing map — SOM) — нейронная сеть с обучением без учителя, выполняющая задачу визуализации и кластеризации. Является методом проецирования многомерного пространства в пространство с более низкой размерностью (чаще всего

двумерное), применяется также для решения задач моделирования, прогнозирования и др.

Каждый объект характеризуется набором различных *параметров*, описывающих его состояние. Например, параметрами будут данные из финансовых отчетов. Эти параметры часто имеют числовую форму или могут быть приведены к ней. Таким образом, нам надо на основании анализа параметров объектов выделить схожие объекты и представить результат в форме, удобной для восприятия. Эти задачи решаются самоорганизующимися картами Кохонена.

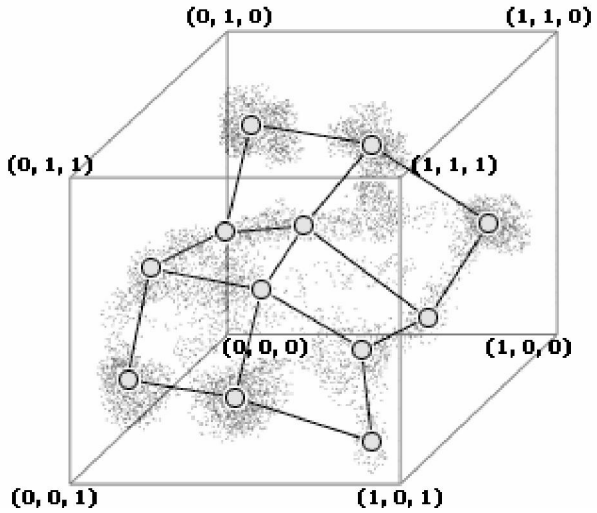
Пусть объект расположен в трехмерном пространстве. Тогда каждый объект с признаками можно представить в виде точки в данном пространстве, и пронормировать эти признаки в интервал [0,1]), в результате чего все точки попадут в куб единичного размера. Отобразим эти точки.



Расположение объектов в пространстве

На рисунке видно, как расположены объекты в пространстве, причем легко заметить участки, где объекты группируются, т.е. у них схожи параметры, значит, и сами эти объекты, скорее всего, принадлежат одной группе. Но так можно поступить только в случае, когда признаков немного. Значит, надо найти способ, преобразующий данную систему в простую для восприятия, желательно двумерную систему (потому что уже трехмерную картинку

невозможно корректно отобразить на плоскости) так, чтобы соседние в искомом пространстве объекты оказались рядом и на полученной картинке. Для этого используем самоорганизующуюся карту Кохонена. В первом приближении ее можно представить в виде «гибкой» сети. Предварительно «скомкав», бросаем сеть в пространство признаков, где уже имеются объекты, и далее поступаем следующим образом: берем один объект (точку в этом пространстве) и находим ближайший к нему узел сети. После этого узел подтягивается к объекту (т.к. сетка «гибкая», то вместе с этим узлом так же, но с меньшей силой подтягиваются и соседние узлы). Затем выбирается другой объект (точка), и процедура повторяется. В результате получется карта, расположение узлов которой совпадает с расположением основных скоплений объектов в исходном пространстве. Полученная карта обладает следующим замечательным свойством – узлы ее расположились таким образом, что объектам, похожим между собой, соответствуют соседние узлы карты. Теперь находим, какие объекты попали в какие узлы карты. Это также определяется ближайшим узлом – объект попадает в тот узел, который находится ближе к нему.



Вид пространства после наложения карты

В результате данных операций объекты со схожими параметрами попадут в один узел или в соседние узлы. Хотя задача поиска похожих объектов и их группировки решена, но на этом возможности карт Кохонена не заканчиваются. Они позволяют также представить полученную информацию

в простой и наглядной форме путем нанесения раскраски полученной карты (точнее ее узлы) цветами, соответствующими интересующим нас признакам объектов.

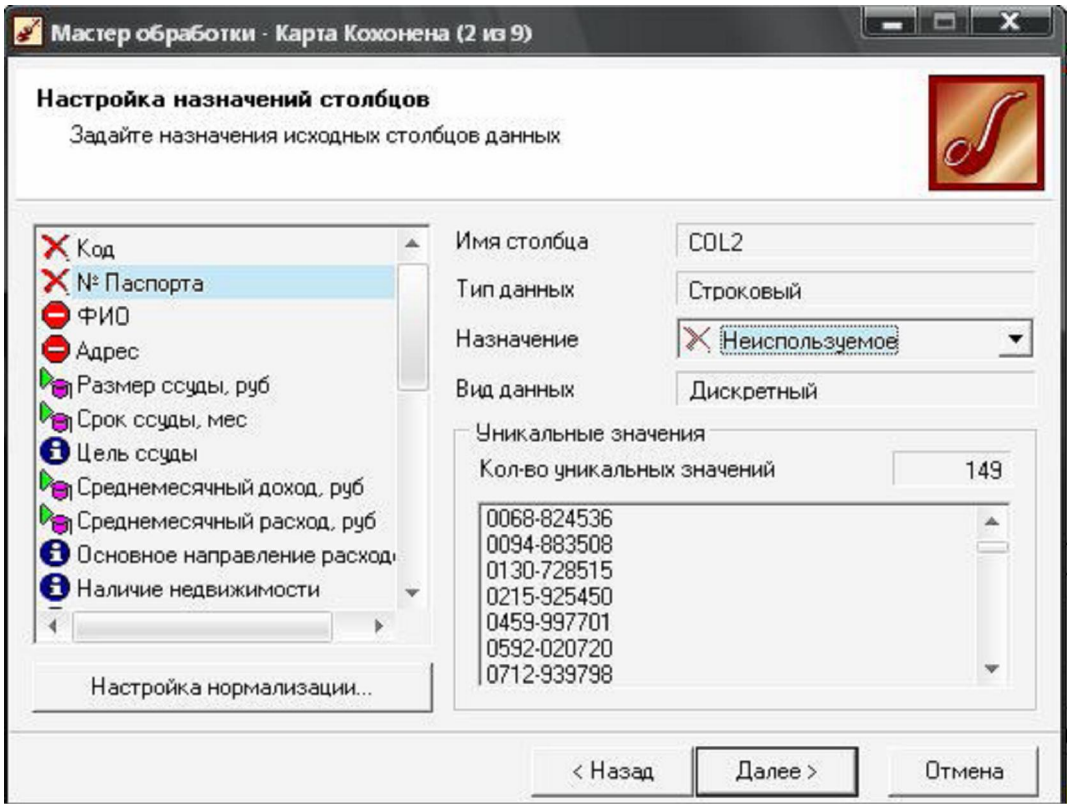
Также можно получить информацию о зависимостях между параметрами. Нанеся на карту раскраску, соответствующую различным статьям отчетов, можно получить так называемый атлас, хранящий в себе информацию о состоянии рынка. Можно анализировать, сравнивать расположение цветов на раскрасках, порожденных различными параметрами, тем самым получая все новую информацию.

При всем этом описанная технология является универсальным методом анализа. С ее помощью можно анализировать различные стратегии деятельности, производить анализ результатов маркетинговых исследований, проверять кредитоспособность клиентов и т.д.

Ход работы

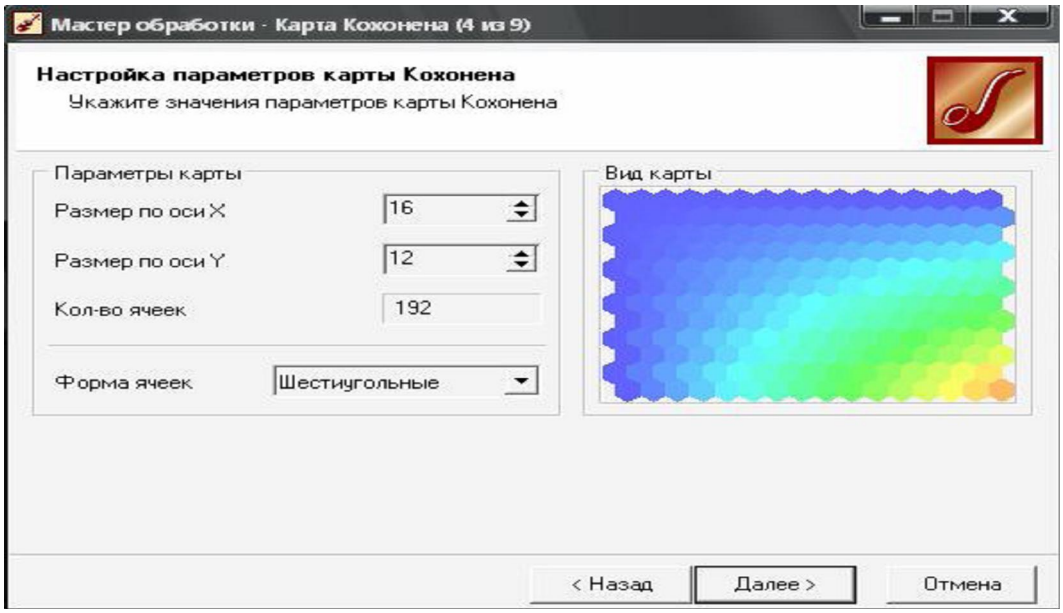
Импортируйте в АП «Deductor» исходные данные из файла C:\Program\Files\BaseGroup\Deductor\Samples\CreditSample.txt. Процесс построения карты Кохонена состоит из 10 этапов.

Запустите *мастер обработки*, в котором в разделе «Data Mining» выберете способ обработки данных «Карта Кохонена», нажмите «Далее». В окне настройки назначения столбцов необходимо обозначить столбцы «Код» и «№ паспорта» как «Неиспользуемые» (так как значения этих столбцов уникальны, а это не позволит их классифицировать по общим признакам). Определите поле «Давать кредит» как «Выходное».



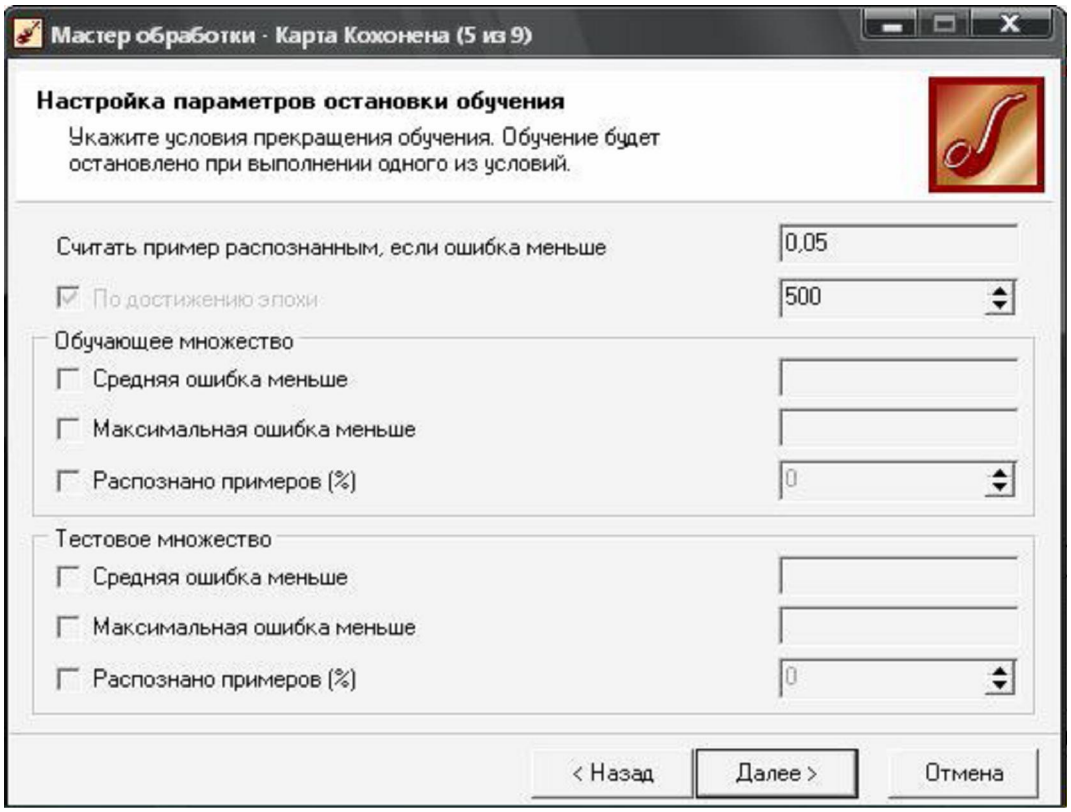
Пример настройки назначений столбцов

Настройку обучающей выборки и параметров карты Кохонена можно оставить без изменений.



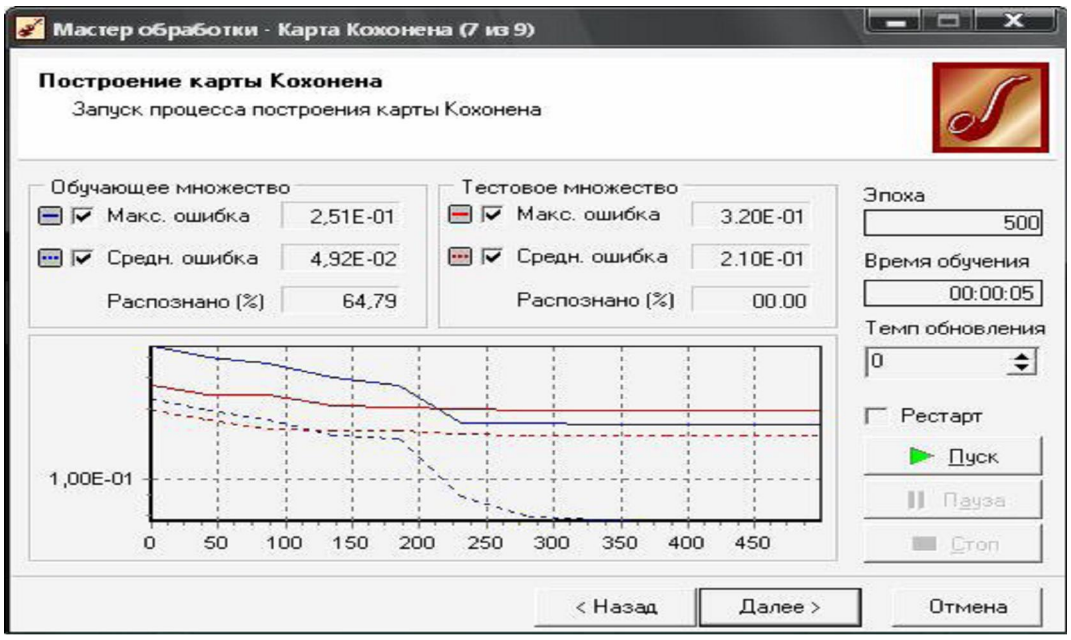
Настройка параметров карты Кохонена

Настройте параметры остановки обучения, указав уровень допустимой погрешности, если он будет превышен, анализ данного множества будет прекращен. Можно оставить значения «по умолчанию».



Настройка параметров остановки обучения.

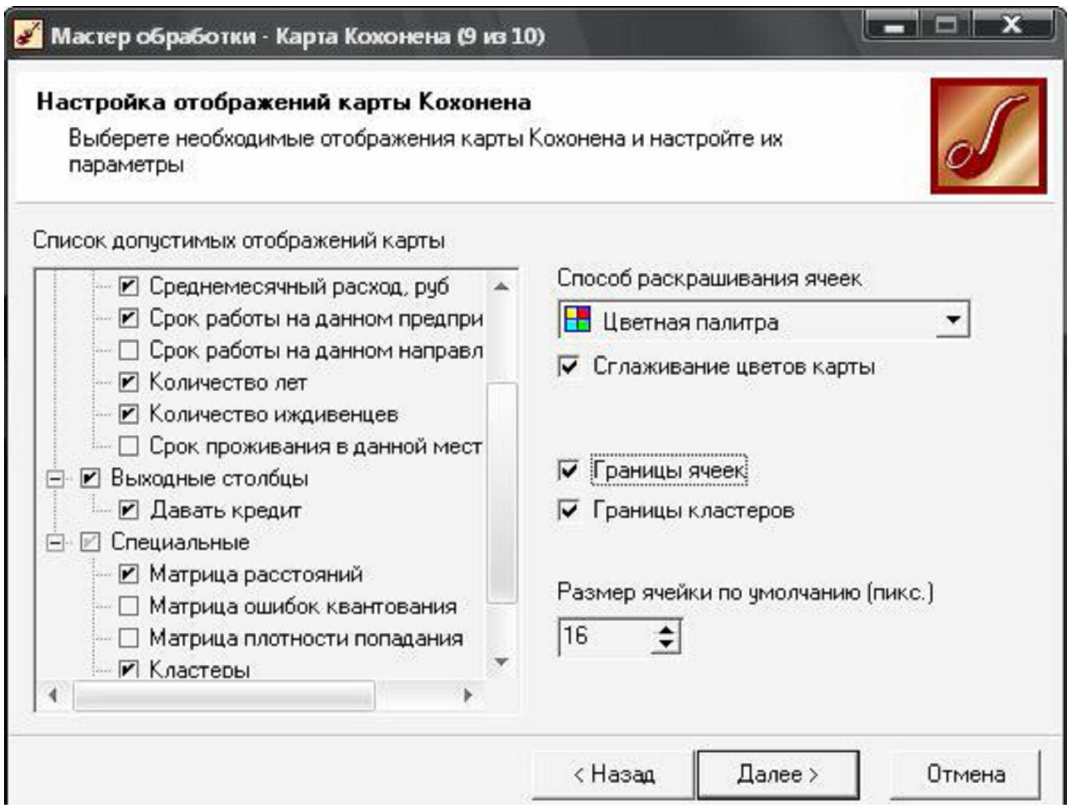
Настройку параметров обучения также оставьте без изменений. Далее запустите процесс построения карты Кохонена, нажав кнопку «Пуск».



Итог построения карты Кохонена

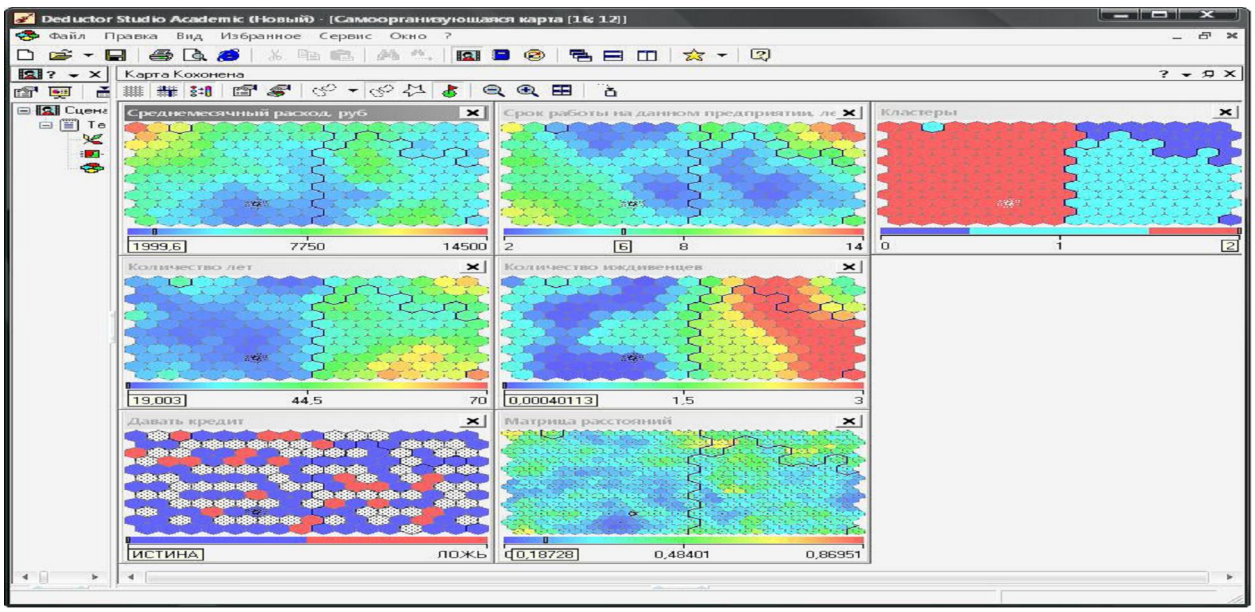
На вкладке «Выбор способа отображения данных» поставьте галочку напротив пункта «Самоорганизующаяся карта Кохонена». Теперь необходимо провести настройку отображения карты:

отметьте разделы «Давать кредит» и «Кластеры» и другие разделы – по желанию.



Настройка отображений карты Кохонена

Далее задайте имя, метку и описание карты (по желанию). В результате получатся карты Кохонена, подобные изображенным на рисунке.



Примеры карт Кохонена

Щелкнув левой клавишей мыши по любому шестиугольнику на любой карте, выделятся соответствующие ему ячейки на остальных картах, в том

числе на картах «Давать кредит» и «Кластеры». При этом на шкалах в нижней части карт отобразятся значения соответствующих параметров.

Задание

1. Выполните описанные выше действия по построению карт Кохонена. Проанализируйте результаты, что можно сказать о вероятности возврата кредита для групп 2, 3 и 4?

2. Используя различные отображения карты Кохонена, постройте 3-4 правила выдачи кредитов.

Содержание отчета

1. Цель работы.
2. Краткое описание хода работы
3. Вид карт Кохонена
4. Ответы на вопросы.
5. Листинг программы
6. Заключение.

Вопросы:

1. Для чего используются карты Кохонена?
2. По какому принципу происходит перенос многомерного пространства на пространство меньшей размерности?
3. Что обзначает фраза “Победитель берет все”
4. Какие метрики испоьзуются пи разбиении на кластеры
5. Целессобразность применения карт Кохонена при клстеризации данных

Лабораторная работа №6. Классификация данных

Цель работы. Целью данной лабораторной работы является исследование способности нейронной сети решать задачи классификации. Сеть необходимо обучить классификации по пяти классам по 10-20 числовым признакам. Используемая модель: одномерная сеть Кохонена.

Теория. Задача классификации заключается в идентификации объекта и отнесении его к одному из нескольких множеств. При этом задача классификации предполагает, что множества попарно не пересекаются. Применительно к нейронным сетям задачу классификации можно поставить

следующим образом: пусть имеется N множеств D1,D2…Dn признаков объектов. Сеть обучается на парах векторов X и Y, где: X = (x1,x2…xm) – входной вектор признаков; Y = (y1,y2…yn) = C(X) – выходной вектор, классифицирующий вектор X. При этом возможно несколько случаев:

1. Y = k, классификатор имеет скалярный характер. k – порядковый номер множества, к которому относится X.

2. Y = (y1,y2 …yk … yn). При этом только yk=1, остальные компоненты вектора равны 0. Таким образом работает звено Кохонена

3. Y = (y1,y2 … yn). При этом каждая компонента yk характеризует степень принадлежности к множеству Dk. В режиме нормального функционирования сеть по входному вектору X выдает вектор Z по правилам, аналогичным описанным для векторов Y. Точность решения определяется статистикой: сколько раз вектор Z правильно классифицировал объект с признаками X, соотнося его с той или иной группой Dk. Для пункта 3 возможно вычисление погрешности, при наличии функции-скаляризатора степени принадлежности вектора X к множествам Dk. Задача может быть дополнена введением «шума», однако смысл от этого не изменится. Шум лишь изменит границы областей D1…Dn.

Для решения задачи классификации с линейной и нелинейной разделимостью классов используются классические модели нейронных сетей: многослойный персептрон и рекуррентные сети на его основе, радиально-базисные сети, сети Кохонена, гибридные сети, рекуррентные самоорганизующиеся сети. При наложении множества объектов друг на друга, то задача классификации становится более общей и предполагает, что объект характеризуется степенью принадлежности к тому или иному множеству, то есть имеет место задача классификации с вероятностной разделимостью классов.

Ход работы:

1. Необходимо выбрать предметную область, отобрать не менее 10 числовых характеристик объектов и задать их диапазоны.

2. Сгенерировать обучающую выборку размерностью от 10 до 20 примеров для каждого класса. Предусмотреть нормализацию входных векторов.

3. Написать программу, имитирующую работу нейронной сети Кохонена

4. Провести обучение сети Кохонена по алгоритму Кохонена с прямоугольным соседством.

5. Исследовать эффективность алгоритмов обучения от значения коэффициента обучения.

6. Исследовать зависимость погрешности классификации от алгоритма обучения.

7. Исследовать зависимость погрешности классификации от объёма обучающей выборки.

8. Исследовать зависимость погрешности классификации от числа итераций обучения.

Содержание отчета

1. Цель лабораторной работы.
2. Краткое описание хода работы.
3. Файл обучающей выборки.
4. Результатов работы нейросетевого клссификатора Выводы по результатам работы.
5. Ответы на вопросы.
6. Ответы на вопросы.

Вопросы

1. В чем суть классификации данных?

2. Отличие классификации от кластеризации.

3. Какие существуют методы классификации кроме нейросетевого?

4. Целесообразность применения нейросетевого метода для лссификации данных.

5. Как определяется погрешность классификации?

Лабораторная работа №8 . Алгоритмы распознавания прецендентов

Цель работы: изучить существующие алгоритмы распознавания образов в виде прецендентов.

Теоретическая часть. Обзор алгоритмов распознавания образов

1. Алгоритм ближайшего соседа Процедуравзятая в качестве решающего правила: оставить в памяти машины все реализации обучающей выборки и классифицируемую точку (образ) отнести к тому классифицирующему образу, чья реализация оказалась ближайшей. Это – правило ближайшего соседа. Учитывая, что результаты реальных измерений свойств могут быть «зашумлены», можно использовать правило k ближайших соседей: если больше половины из k ближайших соседей принадлежат образу i, то и объект (точка) q относится к образу i.
2. Метод потенциальных функций Иногда в «голосовании» принимают участие все реализации обучающей выборки, но с разными весами, зависящими от расстояния. Смысл алгоритма заключается в излучении точками каждого класса потенциала, величина убывающего с расстоянием. Характер убывания может быть самым различным. В точке q определяется «притяжение» к каждому из классов. Если окажется, что какая-то точка распознается с ошибкой, то можно изменить картину потенциального поля. Доказана сходимость алгоритма к оптимальному при увеличении обучающей выборки и конечность числа шагов при не слишком сложной («вычурной») картине потенциального поля.
3. Алгоритм STOPL Сложность заключается в комбинаторном характере задачи - вечный поиск компромиса между требуемой скоростью и затратами памяти. Для сокращения перебора выбирают точки пограничные точки наибольшего риска. Пусть rin - расстояние до своей ближайшей точки, rout - до чужой. Тогда отношение W =rin / rout характеризует величину риска быть опознаной в качестве чужого образа. Среди точек каждого образа выбираются в качестве

прецедентов по одной точке с максимальным значением W. После этого распознаются все точки обучающей выборки с опорой на прецеденты по методу ближайшего соседа. Среди опознанных неправильно вновь выбирается точка с максимальным значением W и ею пополняется список прецедентов. Процесс повторяется до тех пор, пока все точки обучающей выборки не будут распознаваться правильно.

4. Метод дробящихся эталонов - алгоритм ДРЭТ Стремимся опять же к безошибочному распознаванию обучающей выборки, но для выбора прецедентов используется метод покрытий обучающей выборки каждого образа простыми фигурами (которые можно усложнять при необходимости). В качестве покрывающих фигур выбираются гиперсферы. Для каждого образа строится гиперсфера минимального радиуса, покрывающая все его точки (реализации). Значения радиусов гиперсфер и расстояния между центрами позволяет определить непересекающиеся гиперсферы. Это - эталоны первого поколения. Если сферы пересекаются, но пересечения пусты, то такие гиперсферы (центры и радиусы) также относятся к эталонам первого поколения. Приэтом область пересечения считается принадлежащей 21 гиперсфере меньшего радиуса. Эталоны второго поколения строятся только для пересечений, содержащих точки двух или более образов. Если эталоны второго поколения также пересекаются, то процесс продолжается до полной надежности распознавания обучающих выборок. Контрольные образцы классифицируются по попаданию в эталонные гиперсферы. Если контрольная точка не попадает в гиперсферу, то определяется ближайшие и далее используется метод ближайшего соседа.
5. Логические решающие правила. В задачах распознавания образов зачастую требуется, чтобы компактные точки в n-мерном пространстве признаков были компактными и несовпадающими в проекциях на координатные оси (гипотеза локальной компактности). Конечно, это не

всегда так. Например, все трехмерные геометрические фигуры (сферы, пирамиды, параллелепипеды) могут быть синими. Но здесь мы сталкиваемся с проблемой ситуативной информативности признаков. Очевидно, фигуры с геометрической точки зрения не будут классифицировать по цвету. Обычно, проекции на координатные оси пересекаются, но могут выглядеть по разному и это дает надежду, что комбинация несовпадающих проекций на несколько осей позволит построить эффективное решающее правило за счет сокращения размерности признакового пространства. Решающие правила, учитывающие разницу в проекциях разных образов имеют вид «Если-условие-то- следствие» и получили наименование логических решающих правил (ЛРП).

6. Алгоритм CORAL Выделяется подмножество значений Xjv ∈ Xj

признака Xj. Для сильных признаков – это интервал значений, для шкал порядка - ряд соседних порядковых позиций, для шкал наименований - одно или несколько имен. Обозначается факт, что некоторое значение признака Xj объекта ai принадлежит подмножеству Xjv как J(ai, Xjv), факт попадания объекта a в область v, образованную границами подмножеств Xjv, т.е. в гиперпараллелепипед , запишется в виде логического высказывания: n’ ≤ n, где n - размерность признакового пространства. Число n’ называется длиной высказывания. Логической закономерностью называется высказывание, удовлетворяющее двум условиям: где w - индекс объектов своего образа, w − - индекс всех чужих объектов, mw - число всех своих объектов, mw− - число чужих объектов, mws - число своих, удовлетворяющих высказыванию S, mws− - число чужих,

удовлетворяющих тому же высказыванию S, 23 α, β - некоторые величины в диапазоне от 0 до 1. Желательно, чтобы высказывание S отбирало больше своих объектов и поменьше чужих, т.е. чтобы α было как можно большим, а β - как можно меньшим. Набор закономерностей называется покрывающим для образа w, если для любой его реализации выполняется хотя бы одна закономерность из этого набора. Желательно, чтобы число закономерностей в наборе было минимальным. Поиск закономерностей начинается с больших значений α (например, α = 1) и малых значений β ≈ 0.02. Просматриваются все подмножества значений первого случайно выбранного признака и находится высказывание S, удовлетворяюшее условиям 1 и 2. Если таковое не находится, то процесс поиска продолжается при более низком пороге α вплоть до α = 0.5. Если и в этом случае нет результата, то увеличивается β в предельно допустимой доле «чужих среди своих». Если условия 1 и 2 не выполняются и при α = β = 0.5., то делается переход к рассмотрению второго признака, случайным образом выбранным из оставшихся. Если условия 1 и 2 на каком-то шаге выполняются, то объекты своего образа, удовлетворяющие высказыванию S, из дальнейшего рассмотрения исключаются. Для оставшихся объектов образа w длина высказывания увеличивается на единицу. Процесс продолжается до получения покрывающего набора закономерностей для всех объектов образа w. Аналогично строятся покрывающие наборы и для всех других распознаваемых образов. Можно потребовать, чтобы алгоритм делал для каждого образа не по одному, а по несколько покрывающих наборов, что соответствует высказыванию Р. Фейнмана о том, что можно говорить о понимании явления, если в состоянии объяснить его несколькими способами. С этой целью после получения первого покрытия исключают из рассмотрения первый признак, включенный в это покрытие, и процесс поиска закономерностей начинается с другого случайно выбираемого

признака. Распознавание контрольного объекта q с помощью покрывающих наборов закономерностей сводится к проверке того, каким высказываниям удовлетворяют его характеристики. Если таких высказываний одно или несколько и все они находятся в списке образа w, то объект q распознается в качестве реализации образа w. Если же объект q удовлетворяет закономерностям нескольких образов, то решение принимается в пользу того образа, которому принадлежит закономерность с наибольшим значением величины Pws. Анализ общего списка закономерностей может показать, что некоторые признаки из исходной системы X в них отсутствуют. Это означает, что они оказались неинформативными и в процессе принятия решения на них можно не обращать внимания. Для каждого i-го образа подмножество информативных признаков может оказаться разным. А это значит, что при проверке гипотезы о принадлежности объекта к тому или иному образу, нужно анализировать не все пространство признаков, а его информативное подпространство, что хорошо согласуется с интуитивными методами неформального распознавания.

7. Метод случайного поиска с адаптацией (алгоритм СПА) Единичный отрезок разбивается на g участков одинаковой длины (1/g). Каждому участку сопоставляется свой признак: первому - первый, второму - второй и т.д. Запускается датчик случайных чисел с равномерным распределениют в диапазоне 0..1. После n шагов работы выбирается n признаков. По числу ошибок оценивается качество распознавания. Такая 25 процедура проделывается r раз. В итоге будет получен список оценок L = (α1, ..., αr). Теперь можно упорядочить список L по возрастанию α, т.е. по убыванию качества распознавания, и ввести систему поощрений и наказаний. Участки, соответствующие признакам, дающим лучший результат, увеличиваются, «худшие» - уменьшаются, но так, чтобы суммарная длина по-прежнему была равна Испытывают r новых признаковых подсистем, но теперь вероятность

попадания на «лучшие» участки выше, чем на плохие. Продолжают процесс адаптации таким образом, что длина участков признаков, регулярно попадающих в самые информативные подсистемы, увеливается на величину $h < 1/g$, а для самых неинформативных длины их участков уменьшаются. После некоторого количества циклов поиска и адаптации, процесс стабилизируется. Алгоритм СПА был протестирован для систем, в которых возможен полный перебор сочетаний признаков.

Задание. Составить программу, реализующую один из предложенных алгоритмов.

Вариант 1. Написать программу, выполняющую поиск слова в строке при помощи алгоритма ближайших соседей.

Вариант 2. Написать программу, реализующую поиск слова в текстовом файле с помощью алгоритма ближайших соседей.

Вариант 3. Написать программу, реализующую поиск слова в текстовом файле с помощью алгоритма STOPL.

Вариант 4. Написать программу, реализующую поиск слова в текстовом файле с помощью алгоритма CORAL.

Вариант 5. Написать программу, реализующую поиск слова в текстовом файле с помощью метода случайного поиска с адаптацией.

Вариант 6. Написать программу, реализующую поиск слова в текстовом файле с помощью алгоритма ДРЭТ.

Вариант 7. Написать программу, реализующую поиск слова в текстовом файле с помощью алгоритма потенциальных функций.

Содержание отчета

1. Цель лабораторной работы
2. Краткое описание хода работы
3. Схема алгоритма метода распознавания прецендентов
4. Результатов работы программы распознавания прецендентов
5. Выводы по результатам работы

6. Ответы на вопросы

Вопросы

1. Перечислите существующие алгоритмы распознавания образов.
2. Описание алгоритма ближайших соседей.
3. Описание алгоритма потенциальных функций.
4. Описание алгоритма STOPL.
5. Описание алгоритма CORAL.
6. Описание метода случайного поиска с адаптацией. 7. Описание алгоритма ДРЭТ.

Лабораторная работа № 8. Фильтрация данных

Цель работы. Ознакомиться с методом фильтрации данных фильтром Калмана. Оценить возможность применения фильтра Калмана на практике.

Теоретическая часть. Фильтр Калмана применяют в разных областях – от радиотехники до экономики. В экономике, например, измеряемой величиной могут быть курсы валют, колебания цен. Каждыё день курс валют разный, т.е. каждый день “его измерения” дают нам разную величину. Измерения всегда идут с некоторой ошибкой. В простейшем случае описанное можно свести к следующему выражению: z=x+y, где x – истинное значение, которое нужно измерить, y – ошибка измерения, вносимая измерительным прибором, а z – измеряемая величина. Задача фильтра Калмана состоит в том, чтобы по измеренной z определить истинное значение x, т.е. необходимо отфильтровать (отсеять) из z истинное значение x – убрать из z искажающий шум y.

Для прогнозирования цены отдельного вида продукции предлагается рассмотреть следующую стохастическую нестационарную модель, описывающую процесс изменения цены:

$$p(t+\Delta t) - p(t) = f(t,p(t),z(t))\Delta t + F(t,p(t))\Delta\omega + c(v_1)V(\Delta t,\Delta v),$$

или в виде уравнения в дифференциальной форме:

$$p(t) = p_0 + \int_t f(t, p(t), z(t))dt + \int_t F(t, p(t))d\omega(t) + \int_t \sum_v c(v,)\delta(t - t,)dt$$

где p(t) – значение цены в момент времени t;

f(t,p(t),z(t)) – скорость изменения цен во времени – непрерывная функция по переменной t, определяющая тренд цены и восстанавливаемая по статистической информации;

F(t,p(t)) – среднеквадратичное отклонение;

ω – винеровский процесс;

v – случайная пуассоновская мера с параметром λ(t);

v_t – случайная величина, вызывающая приращение цены согласно закону P(c(v)), P(c(v)) = δ(v-a), a>0.

Первый элемент в правой части представленных выше уравнений, это тренд цены, зависящий от величины предложения и от потребности покупателей в данном товаре.

Кроме того, цены постоянно колеблются около своего тренда вследствие аддитивного воздействия на них случайных факторов, и будем считать, что приращения цены - независимые величины. Можно утверждать, что в краткосрочном периоде закон распределения приращений процесса изменения цен близок к нормальному. Также будем считать, что трендовая составляющая не оказывает влияния на случайную, и наоборот, случайные изменения не влияют на характер тренда. Второй элемент в правой части уравнений отражает именно случайную составляющую – незначительные колебания цен около своего тренда.

На практике часто приходится наблюдать резкие изменения цен - скачки. Их появление свидетельствует о нестабильности рынка или экономики в целом или об ее зависимости от внешних факторов: политических, спекулятивных и других. Причем появление таких скачков не связано с чисто сезонными колебания ми цены. Третий элемент в правой части уравнений отражает эту случайную составляющую, причем

предполагается, что величины скачков будут всюду положительными. Время появления скачков можно моделировать с помощью пуассоновского случайного процесса. Амплитуды скачков случайны и закон распределения их вероятности может быть восстановлен по тем же статистическим данным. Отсюда как следствие имеем, что промежутки между скачками характеризуются показательным распределением.

Любая реализация цены есть решение представленного выше дифференциального уравнения и имеет общий вид:

$$p(t) = p_0 + \int f(t, p(t), z(t))dt + \int F(t, p(t))d\omega(t) + \int \sum c(v_i)\delta(t - t_i)dt$$

Прогнозное значение, полученное с помощью модифицированного фильтра Калмана-Бьюси, будет оптимально с точки зрения принципа максимума апостериорной вероятности. Ниже изложена структура алгоритма для решения поставленной задачи.

2. Структура алгоритма прогнозирования на основе реализации модифицированного фильтра Калмана-Бьюси

Модифицированное уравнение для оптимальной оценки записывается в виде

$$\frac{dp^*(t)}{dt} = f(t, p^*(t), z(t)) + K(t)[y(t) - p^*(t)] + \int_{-T}^{t} \frac{M[\varphi(t)y(\tau_2)]}{M[y(t)y(\tau_1)]} H(t, \tau_2)y(\tau_2)d\tau_2.$$

где $p^*(t)$ – значение оценки; K(t) – коэффициент усиления фильтра, определяемый из следующего соотношения

$$K(t) = f(t, p(t), z(t))R(t)[R\varphi(t) + R(t)]^{-1},$$

в этом соотношении $R(t) = M[p^*(t)-p(t)]\,[p^*(t)-p(t)]$ – дисперсия ошибок оценки, для которой справедливо рекуррентное соотношение

$$R(t+1) = [f(t, p(t), z(t)) - K(t)]^2 R(t) + R_\eta(t)K^2(t) + R_\varphi(t+1) + 2M\left[\int_{-T}^{t} \frac{M[\varphi(t)y(\tau_2)]}{M[y(t)y(\tau_1)]} H(t, \tau_2)y(\tau_2)d\tau_2\right][p^*(t) - p(t)].$$

функции $R_\eta(t) = M[\eta(t)\,\eta(\tau)] = Q(t)\delta(t-\tau)$ и $R_\varphi(t+1)$ – известны.

Модификация связана с представлением уравнения состояния как

$$dp(t) = f(t, p(t), z(t))dt + F(t, p(t))d\omega(t) + \int c(v)v(dt, dv)$$

и уравнения наблюдения в виде

$$y(t) = p(t) + \eta(t),$$

где η(t) – гауссов белый шум; y(t) – наблюдаемое значение.

Структура алгоритма прогнозирования:

1. Задать начальные приближения из условия $p(\tau) = p^*(\tau)$; $p^*(\tau) = p_0$; $R(\tau) = 0$;
2. Вычислить коэффициент усиления K(t);
3. Вычислить по формуле значение R(t+l);
4. Получить прогноз цены $p^*(t+1)$ как численное решение дифференциального уравнения.

Производя операции 2-4, при $t = \tau, \tau + 1, \ldots$ получим траекторию предсказанных на шаг вперед значений цены.

Модель и алгоритм прогнозирования цены

Для практической реализации непрерывное время заменено дискретным;

уравнения модифицированной модели:

уравнение состояния: $x(k+l) = \Phi(k+1,k)x(k) + \Gamma(k+1,k)\omega(k)$,
уравнение измерения: $z(k+l) = H(k+l)x(k+l)+v(k+1)$,

где Ф – переходная матрица состояния размера n x n;

Г – переходная матрица возмущения размера n x p;

ω – p-вектор возмущения;

H – матрица измерения размера m x n;

v – m-вектор ошибки измерения.

Процесс {ω(k), k = 0,1, 2, ...} является p-мерной гауссовской белой последовательностью с неотрицательно определенной корреляционной матрицей Q(k) размером p x p. Процесс {v(k) , k = 0,1, 2, ...} является m-

мерной гауссовской белой последовательностью с неотрицательно определенной корреляционной матрицей R(k+1) размером m x m. Оба эти процесса взаимно независимы

Начальное состояние x(0) есть гауссовский случайный n-вектор с неотрицательно определенной корреляционной матрицей P(0) размера n x n.

Тогда алгоритм фильтра Калмана будет в виде:

1) $x(k+1|k) = \Phi(k+1,k)x(k|k)$;
2) $P(k+1|k) = \Phi(k+1,k)P(k|k)\Phi'(k+1,k) + \Gamma(k+1,k)Q(k)\,\Gamma'(k+1,k)$;
3) $K(k+1) = P(k+1|k)H'(k+1)[H(k+1)P(k+1|k)H'(k+1) + R(k+1)]^{-1}$;
4) $P(k+1|k+1) = [I - K(k+1)H(k+1)]P(k+1|k)$;
5) $x(k+1|k+1) = x(k+1|k) + K(k+1)[z(k+1) - H(k+1)x(k+1|k)]$.

Можно еще более упростить модель, взяв вместо векторов x, ω и v одиночные переменные, тогда матрицы Ф, Г и H превратятся в скаляр.

Ход работы

Из статистической информации выбираем временной ряд, отражающий динамику цен.

Таблица 1

Динамика цен

Дата	**Цена**
01.06.2016	3,7
02.06.2016	3,65
03.06.2016	3,7
04.06.2016	3,59
05.06.2016	3,7
06.06.2016	3,7
07.06.2016	3,58
08.06.2016	3,52
09.06.2016	3,52
10.06.2016	3,34

11.06.2016	3,52
12.06.2016	3,52
13.06.2016	3,45
14.06.2016	3,47
15.06.2016	3,52
16.06.2016	3,08
17.06.2016	3,09
18.06.2016	3,15
19.06.2016	2,95
20.06.2016	3,05
21.06.2016	3,11
22.06.2016	2,99
23.06.2016	3,1
24.06.2016	3,15
25.06.2016	3,09
26.06.2016	3,15
27.06.2016	3,15
28.06.2016	3,08
29.06.2016	3,37
30.06.2016	3,28

При анализе этого временного ряда формируем тренд изменения цены:

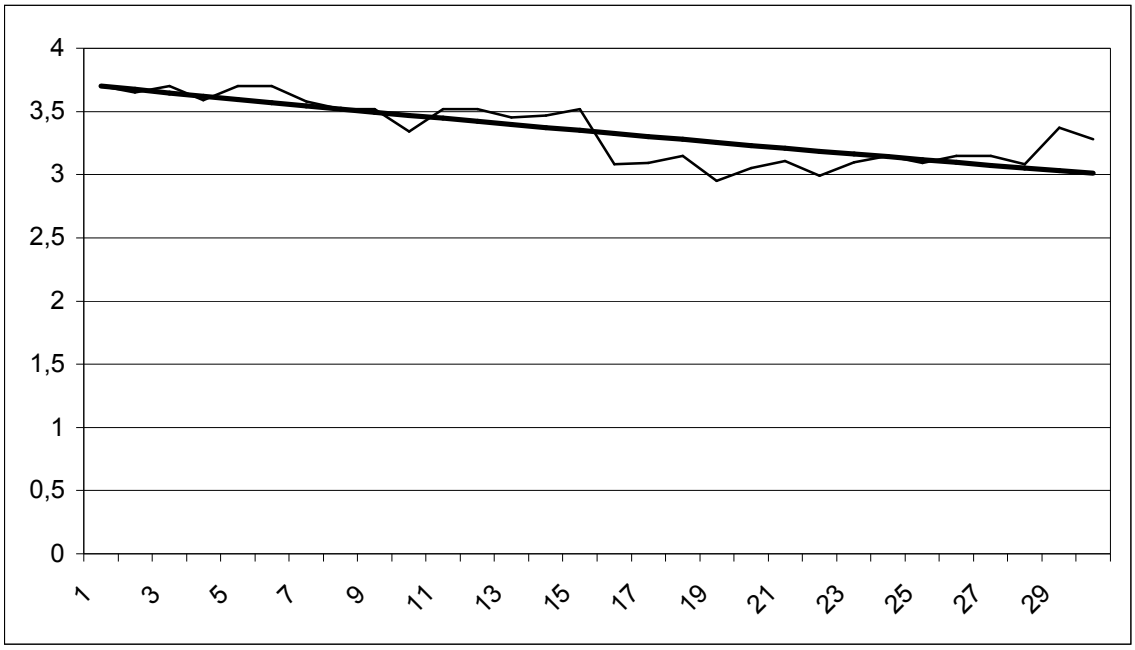


Рис. 1. Изменение цен (исходные данные и тренд)

Уравнение тренда получили в виде: x(k+1) = 0,9929x(k).

Затем формируем следующую модель:

уравнение состояния: x(k+l) = 0,9929x(k) + ω(k),

уравнение измерения: z(k+l) = x(k+l)+v(k+1),

где дисперсия ω(k) = Q; дисперсия v(k+1) = R.

Тогда уравнения фильтра примут вид:

1) x(k+1|k) = 0,9929x(k);
2) P(k+1|k) = P(k|k) + Q;
3) K(k+1) = P(k+1|k)[P(k+1|k) + R]$^{-1}$;
4) P(k+1|k+1) = R*K(k+1);
5) x(k+1|k+1) = x(k+1|k) + K(k+1)[z(k+1) – x(k+1|k)].

при начальном условии P(0|0) = 0; x(0|0) = z(0|0).

Исходный код программа, реализующей фильтр Калмана, на языке Си приведен в Приложении 1.

Таблица 2

Результаты работы алгоритма фильтра Калмана (Q=0,1;R=0,15)

Дата	Цена	Тренд	P(k+1|k+1)	x(k+1|k)	x(k+1|k+1)

01.06.2016	3,7	3,700000	0,060000	3,673730	3,664238
02.06.2016	3,65	3,673730	0,077419	3,638222	3,670108
03.06.2016	3,7	3,647647	0,081281	3,644050	3,614762
04.06.2016	3,59	3,621748	0,082082	3,589097	3,649785
05.06.2016	3,7	3,596034	0,082246	3,623871	3,665613
06.06.2016	3,7	3,570502	0,082279	3,639587	3,606902
07.06.2016	3,58	3,545151	0,082286	3,581293	3,547669
08.06.2016	3,52	3,519981	0,082287	3,522481	3,521120
09.06.2016	3,52	3,494989	0,082287	3,496120	3,410475
10.06.2016	3,34	3,470175	0,082288	3,386261	3,459628
11.06.2016	3,52	3,445536	0,082288	3,435065	3,481659
12.06.2016	3,52	3,421073	0,082288	3,456939	3,453132
13.06.2016	3,45	3,396783	0,082288	3,428615	3,451318
14.06.2016	3,47	3,372666	0,082288	3,426814	3,477934
15.06.2016	3,52	3,348720	0,082288	3,453241	3,248487
16.06.2016	3,08	3,324944	0,082288	3,225423	3,151132
17.06.2016	3,09	3,301337	0,082288	3,128759	3,140412
18.06.2016	3,15	3,277898	0,082288	3,118115	3,025890
19.06.2016	2,95	3,254625	0,082288	3,004406	3,029418
20.06.2016	3,05	3,231517	0,082288	3,007909	3,063915
21.06.2016	3,11	3,208573	0,082288	3,042161	3,013546
22.06.2016	2,99	3,185792	0,082288	2,992150	3,051315
23.06.2016	3,1	3,163173	0,082288	3,029650	3,095672
24.06.2016	3,15	3,140715	0,082288	3,073693	3,082639
25.06.2016	3,09	3,118416	0,082288	3,060752	3,109712
26.06.2016	3,15	3,096275	0,082288	3,087633	3,121847
27.06.2016	3,15	3,074291	0,082288	3,099682	3,088885
28.06.2016	3,08	3,052464	0,082288	3,066954	3,233200
29.06.2016	3,37	3,030791	0,082288	3,210244	3,248511

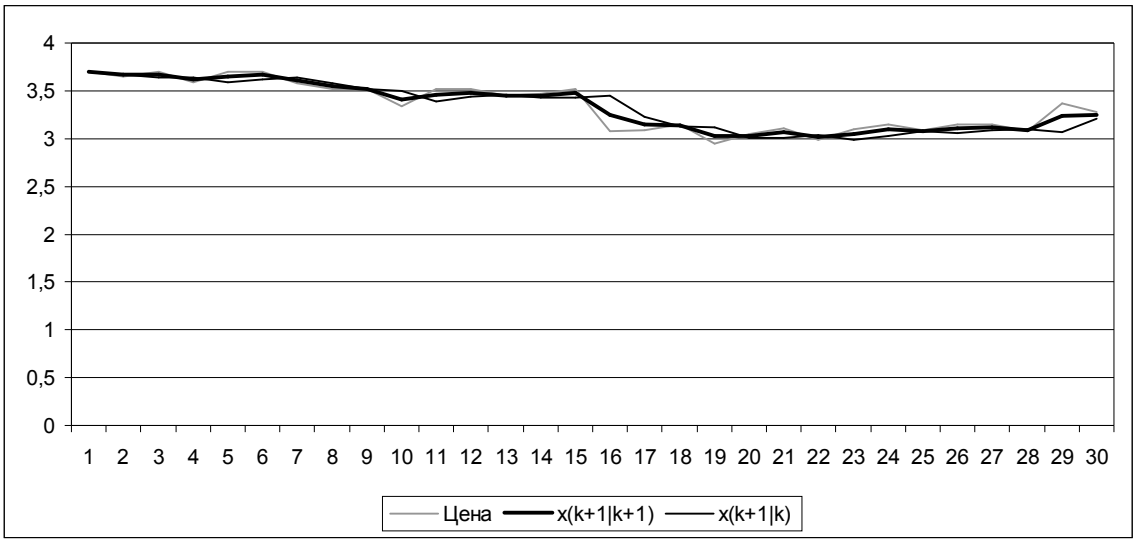


Рис. 2. Результат работы фильтра (Q=0,1; R=0,15)

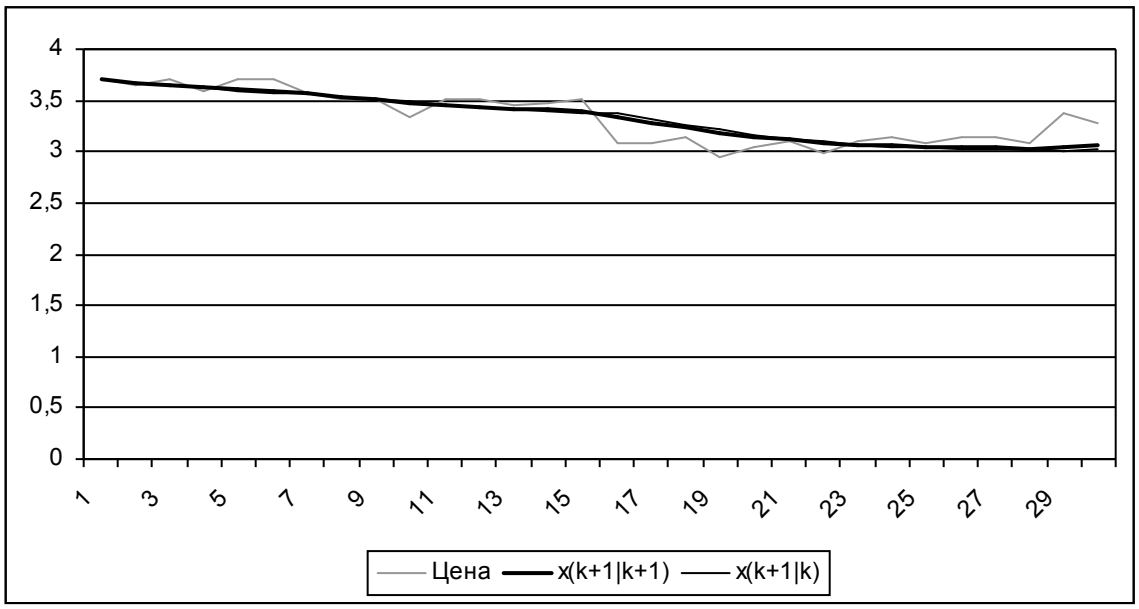


Рис. 3. Результат работы фильтра (Q=0,01; R=0,5)

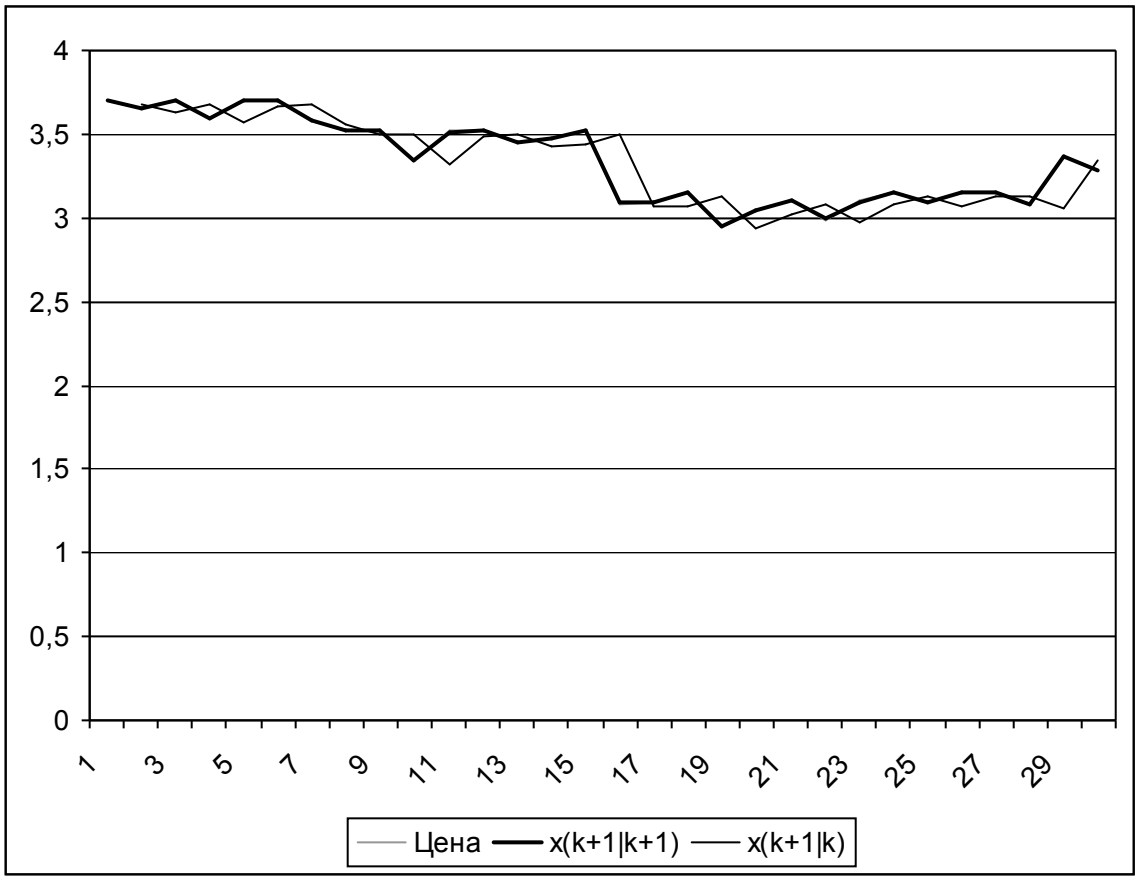


Рис. 4. Результат работы фильтра (Q=0,5; R=0,01)

Как видно из результатов работы алгоритма фильтра Калмана, если дисперсия ω(k) больше дисперсии v(k+1), то фильтр приближает значение оценки к измерениям. Если же дисперсия ω(k) меньше дисперсии v(k+1), то наоборот, скорректированная оценка будет ближе к модели. Также видно, что при начальном значении P(0|0) = 0, фильтр уже после обработки седьмого измерения, находится в установившемся состоянии. Кроме того, алгоритм был протестирован и при других значениях P(0|0), все равно после седьмого измерения фильтр приходил в устойчивое состояние.

С помощью фильтра Калмана, зная дисперсии случайных процессов ω(k) и v(k+1), можно получать достаточно точные оценки значений цены отдельного продукта.

Приложение1

```
1. #include <stdio.h>
2. #include <string.h>
```

```
3.  #include <ctype.h>
4.  int main(int argc, char *argv[]){
5.  FILE *input,*out;
6.  char buf[257];
7.  int k;
8.  float res,z,K,P,P1,Q,R,F,x;//косметика
9.  if(argc!=2){
10. printf("Синтаксис: %s <file>\n",argv[0]);
11.  printf("          где <file> - имя файла с результатами измерений...\n");
12.  printf("В результате работы программы создается файл \"result.txt\"\n");
13.  return(1);
14. } //есть ли параметр
```

Задание

Для различных объктов (продуктов или изданий по желанию студента) составить статистический ряд и используя фильтр Калмана выполнить прогноз цены. Выполнить исследования полученных результатов при различных значениях Q **и** R. Сделать выводы.

Содержание отчета

1. Цель лабораторной работы
2. Краткое описание хода работы
3. Файл статистического ряда исследуемого объекта
4. Результатов работы фильтра после прогноза в виде
5. Выводы по результатам работы
6. Ответы на вопросы

Вопросы:

1. В чем заключается суть работы фильтра Калмана?
2. Пояснить, что происходит с фильтром, если дисперсия ω(k) больше дисперсии v(k+1) и почему?

3. В чем смысл уравнение состояния?
4. Что представляет уравнение измерения?
5. Поясните возможность применения фильтра Калмана для нестанционарных процессов.

Лабораторная работа № 9. Парциальная обработка данных

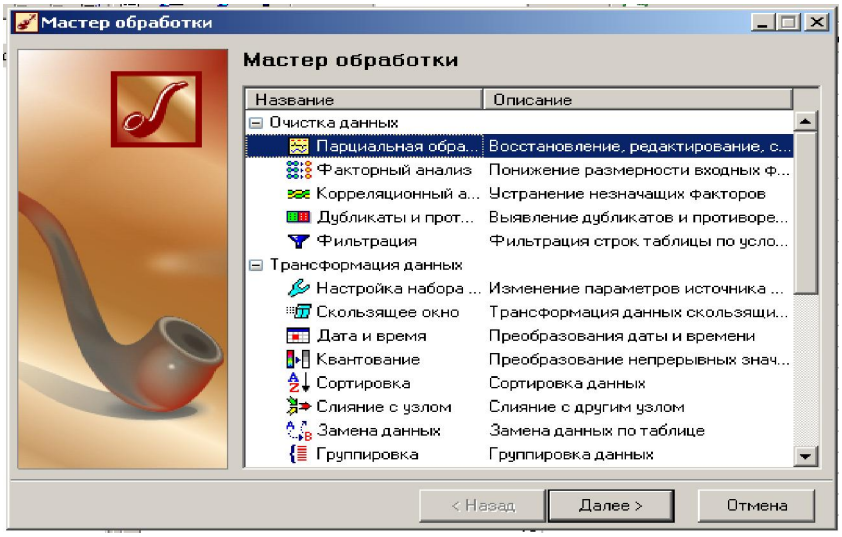
Цель работы. Изучить возможности АП процедур обработки данных.ых выполнить обработку данных выбранной предметной области.

Теоретическая часть. В процессе парциальной обработки восстанавливаются пропущенные данные, редактируются аномальные значения, проводится спектральная обработка. В Deductor Studio при этом используются алгоритмы, в которых 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, входят сглаживание, удаление шумов, редактирование аномальных значений, заполнение пропусков в рядах данных.

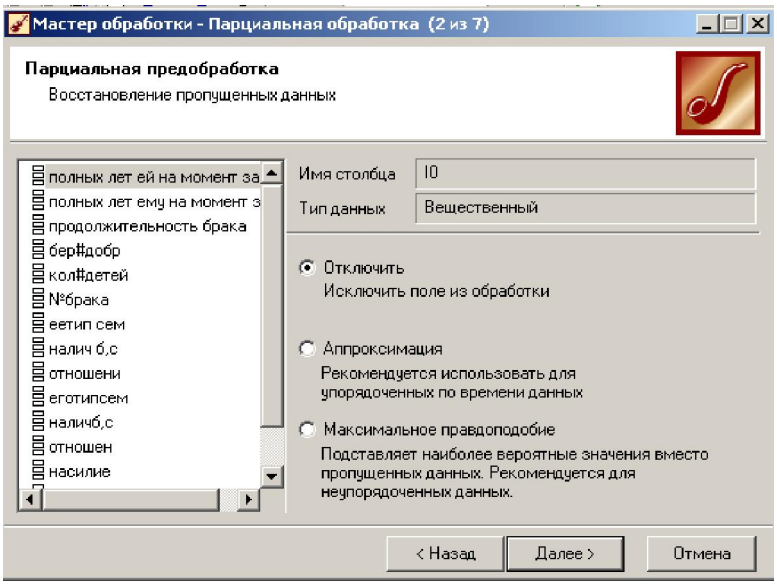
1) таблица с аномальными данными:

полных лет ей на момент аключени брака	олных лет ему на момент	жител брака	ер#доб	#де	№брака	еетип сем	налич б,с	тношен	типс	наличб,с	тноше	насилие	алк
100	100	2	нет	0	первый	полная	2	хорош	поли	1	хоро	нет	нет
30	35	5	нет	0	первый	полная	1	хорош	поли	0	хоро	нет	нет
29	31	12	нет	0	первый	полная	1	хорош	поли	1	хоро	нет	нет
28	32	4	да	1	второй	неполн	0	хорош	непо	1	хоро	да	нет
27	28	6	да	1	второй	полная	1	хорош	поли	0	хоро	нет	да
27	25	1,5	да	1	первый	неполн	2	хорош	поли	2	хоро	нет	нет
27	40	7	нет	0	первый	полная	1	хорош	поли	3	не оч	нет	нет
27	28	2	нет	1	первый	полная	1	хорош	непо	0	хоро	да	нет
26	25	17	нет	2	первый	полная	0	плохие	поли	0	хоро	да	да
26	22	8	да	1	первый	полная	0	хорош	поли	0	не оч	да	нет
26	30	1	да	2	первый	полная	0	хорош	поли	0	плохи	да	да
26	27	3	нет	1	первый	полная	2	хорош	поли	1	не оч	да	нет
26	26	11	нет	1	первый	полная	1	не оче	поли	1	хоро	да	нет
26	30	11	нет	0	первый	полная	1	хорош	поли	1	хоро	нет	нет

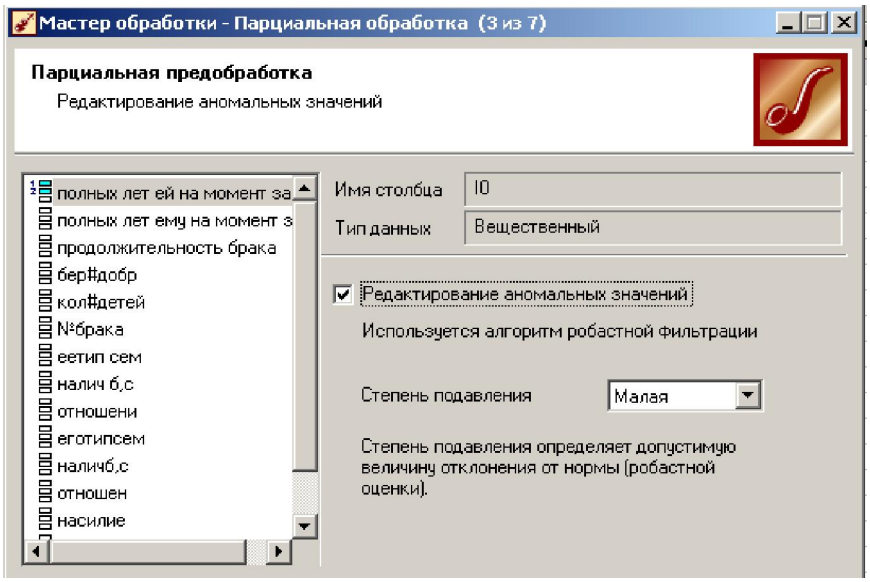
2) открываем мастер обработки и выбираем парциальную обработку:



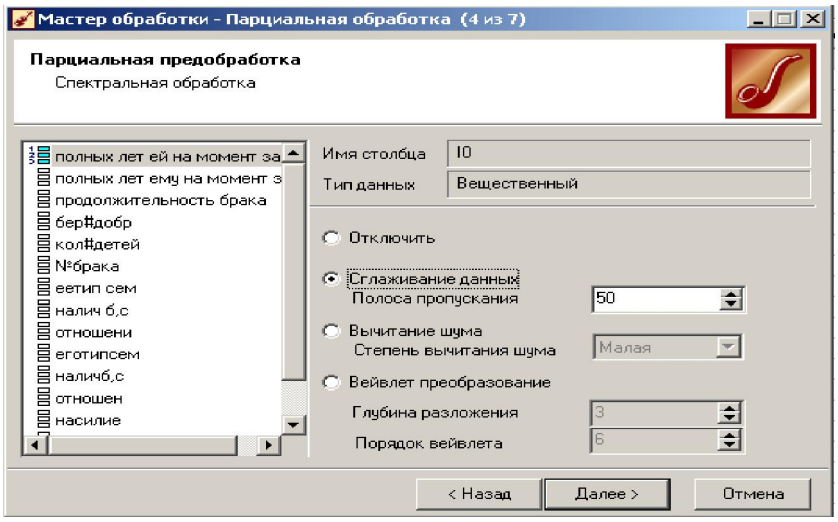
3) выбор операции восстановления пропущенных данных:



4) выбор степени подавления:



5) сглаживание данных возможно выполнить с помощью вейвлет-преобразования и вычитания шума:



6) полученная таблица:

полных лет ей на момент заключения брака	олны лет ему на момен	жител брака	ер#до	#де	№брака	еетип сем	налич б,с	отношени	отипсе	лич	отношен	асили	алкоголизм
19,8614971837076	30	13	нет	1	первый дл	полная	0	не очень х	полная	0	плохие	да	да
20,7632477464631	23	17	да	2	первый дл	полная	0	хорошие	неполн	1	не очень	да	да
21,3719146851208	24	11	нет	1	первый дл	полная	0	хорошие	полная	0	хорошие	да	да
20,3821882356354	23	12	нет	1	первый дл	полная	0	хорошие	полная	0	хорошие	нет	нет
19,5736272606974	22	14	нет	1	первый дл	полная до	0	хорошие	полная	3	плохие	нет	да
20,6027766202498	19	10	нет	1	первый дл	полная	0	хорошие	полная	0	хорошие	нет	нет
22,5543322481671	22	15	да	2	первый дл	полная	0	хорошие	полная	0	хорошие	нет	нет
24,009369451793	25	17	нет	2	первый дл	полная	0	плохие	полная	0	хорошие	да	да
24,8590407431192	24	9	да	1	первый дл	полная	0	не очень х	полная	0	не очень	да	да
24,9933531213713	23	5	нет	1	первый дл	полная	0	не очень х	полная	0	хорошие	да	нет
23,9055282093373	22	8	да	1	первый дл	полная	0	хорошие	полная	0	не очень	да	нет
22,6445951391282	40	3	нет	1	первый дл	полная	1	плохие	полная	3	плохие	да	нет
23,3919878087563	22	4	да	1	первый дл	полная	0	не очень х	полная	0	хорошие	да	да
25,8477646579785	30	1	да	2	первый дл	полная	0	хорошие	полная	0	плохие	да	да
26,6551971393871	28	6	да	1	второй для	полная	1	хорошие	полная	0	хорошие	нет	да
23,9191951190294	37	4	нет	1	первый дл	полная	0	хорошие	полная	0	хорошие	нет	нет
20,085655484726	20	4	нет	1	первый дл	полная	2	хорошие	полная	0	хорошие	нет	нет
18,7333854739885	19	11	нет	1	первый дл	полная	1	хорошие	полная	1	хорошие	да	да
20,1864738812745	23	4	нет	1	первый дл	полная	1	хорошие	полная	0	не очень	нет	нет

Полученная таблица отличается от первоначальной.

Задание: Извлечь данные выбранной предметной области из хранилища данных и выполнить парциальную обработку.

Содержание отчета

1. Цель лабораторной работы
2. Краткое описание хода работы
3. Файл первоначальных данных
4. Файл данных после парциальной обработки
5. Выводы по результатам работы
6. Ответы на вопросы

Вопросы:

1. Что такое парциальная обработка данных?
2. Зачем нужен спектральный анализ при обработке данных?
3. Приведите основные методики восстановления данных
4. Основное назначение вейвлет-преобразования при обработке данных
5. Укажите каким образом выбирается степень подавления шума

10. Список рекомендуемой литературы

1. Методы и модели анализа данных: OLAP и Data Mining. / А.А. Барсегян, М.С. Куприянов, В.В.Степаненко, И.И. Холод – СПб.: БХВ-Петербург, 2004.- 336 с.: ил.

2. Аналитика: методология, технология и организация информационно - аналитической работы /П.Ю. Конотопов, Ю.В. Курносов - М.: РУСАКИ, 2004. - 512 с.

3. Официальный сайт компании «BaseGroup Labs» [Электрон. ресурс]. Рязань, 1995-2010.- Режим доступа: http://www.basegroup.ru/

4. Data Mining и аналитическая платформа Deductor [Электрон. ресурс] : [статья]. М., 2008.- Режим доступа: http://sttc. ru/index.php?option=com_content&task=view&id=56&Itemid=90

5. Г.И. Просветов (МГУ им. М.В. Ломоносова). Дерево решений [Электрон. ресурс] : [статья] / Г.И. Просветов.- СПб, 2008.- Режим доступа: http://www.elitarium.ru/2008/04/09/derevo_reshenijj.html

6. Деревянко В.А. Поиск ассоциативных правил при интеллектуальном анализе данных [Электрон. ресурс] : [статья] / В.А. Деревянко.- 2009.- Режим доступа: http://www.rammus.ru/products/arda/article_lam_translation