МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
"Казанский (Приволжский) федеральный университет"
Елабужский институт (филиал) КФУ

УТВЕРЖДАЮ
Директор
Елабужского института КФУ
_____Е.Е. Мерзон
"22" 05 2024 г.

Программа дисциплины (модуля)
Машинное обучение

Направление подготовки/специальность: 15.03.06 Мехатроника и робототехника
Направленность (профиль) подготовки: Физические основы мехатроники и робототехники
Квалификация выпускника: бакалавр
Форма обучения: очно-заочная
Язык обучения: русский
Год начала обучения по образовательной программе: 2024

Содержание

Программу дисциплины разработал(а)(и) доцент, к.н. Анисимова Э.С. (кафедра математики и прикладной информатики, отделение математики и естественных наук, Елабужский институт КФУ)

1. Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с планируемыми результатами освоения ОПОП ВО

Обучающийся, освоивший дисциплину (модуль), должен обладать следующими компетенциями:

Шифр компетенции	Расшифровка приобретаемой компетенции
ПК-2	Способен разрабатывать, отлаживать, внедрять и сопровождать программное обеспечение мехатронных и робототехнических систем
ПК-2.1	Знать способы разработки, отладки и сопровождения программного обеспечения для мехатронных и робототехнических систем
ПК-2.2	Уметь разрабатывать, отлаживать и сопровождать программное обеспечение для мехатронных и робототехнических систем
ПК-2.3	Владеть навыками разработки, отладки и сопровождения программного обеспечения для мехатронных и робототехнических систем

Обучающийся, освоивший дисциплину (модуль):

Должен знать:
- классы задач в мехатронных и робототехнических системах, решаемых с помощью алгоритмов машинного обучения.

Должен уметь:
- применять алгоритмы машинного обучения при разработке, отладке и сопровождении программного обеспечения для мехатронных и робототехнических систем

Должен владеть:
- способностью реализовывать алгоритмы машинного обучения при разработке, отладке и сопровождении программного обеспечения для мехатронных и робототехнических систем.

2. Место дисциплины (модуля) в структуре ОПОП ВО

Данная дисциплина (модуль) включена в Блок 1 "Дисциплины (модули)" Б1.В.ДВ.02. основной профессиональной образовательной программы 15.03.06 «Мехатроника и робототехника» (Физические основы мехатроники и робототехники) и относится к дисциплинам по выбору, части, формируемой участниками образовательных отношений.

Осваивается на 2 курсе в 4 семестре.

3. Объем дисциплины (модуля) в зачетных единицах с указанием количества часов, выделенных на контактную работу обучающихся с преподавателем (по видам учебных занятий) и на самостоятельную работу обучающихся

Общая трудоемкость дисциплины составляет 4 зачетных(ые) единиц(ы) на 144 часа(ов).

Контактная работа - 16 часа(ов), в том числе лекции - 6 часа(ов), практические занятия - 0 часа(ов), лабораторные работы - 10 часа(ов), контроль самостоятельной работы - 0 часа(ов).

Самостоятельная работа - 128 часа(ов).

Контроль (зачёт / экзамен) - 0 часа(ов).

Форма промежуточного контроля дисциплины: зачет в 4 семестре.

4. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

4.1 Структура и тематический план контактной и самостоятельной работы по дисциплине (модулю)

N	Разделы дисциплины / модуля	Семестр	Виды и часы контактной работы, их трудоемкость (в часах)			Самостоятельная работа
			Лекции	Практические занятия	Лабораторные работы	
1.	Тема 1. Основные понятия машинного обучения	4	1	0	2	32
2.	Тема 2. Кластеризация	4	2	0	4	32
3.	Тема 3. Метод опорных векторов	4	1	0	2	32
4.	Тема 4. Деревья решений	4	2	0	2	32
	Итого: 144 часа		6	0	10	128

4.2 Содержание дисциплины (модуля)

Тема 1. Основные понятия машинного обучения

Понятие машинного обучения. Общая постановка задачи обучения по прецедентам. Способы машинного обучения. Задачи, решаемые с помощью машинного обучения.

Тема 2. Кластеризация

Понятие кластеризации. Постановка задачи кластеризации. Методы кластеризации. Метод k-means. Метод k-медиан. Дискриминантный анализ. Генетический алгоритм.

Тема 3. Метод опорных векторов

Обучение с учителем. Задача классификации. Задача регрессии. Разделяющая гиперплоскость. Применение метода опорных векторов в задаче классификации.

Тема 4. Деревья решений

Понятие дерева решений. Типология деревьев. Обучение дерева решений.

5. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине (модулю)

Самостоятельная работа обучающихся выполняется по заданию и при методическом руководстве преподавателя, но без его непосредственного участия. Самостоятельная работа подразделяется на самостоятельную работу на аудиторных занятиях и на внеаудиторную самостоятельную работу. Самостоятельная работа обучающихся включает как полностью самостоятельное освоение отдельных тем (разделов) дисциплины, так и проработку тем (разделов), осваиваемых во время аудиторной работы. Во время самостоятельной работы обучающиеся читают и конспектируют учебную, научную и справочную литературу, выполняют задания, направленные на закрепление знаний и отработку умений и навыков, готовятся к текущему и промежуточному контролю по дисциплине.

Организация самостоятельной работы обучающихся регламентируется нормативными документами, учебно-методической литературой и электронными образовательными ресурсами, включая:

Порядок организации и осуществления образовательной деятельности по образовательным программам высшего образования – программам бакалавриата, программам специалитета, программам магистратуры (утвержденный приказом Министерства науки и высшего образования Российской Федерации от 6 апреля 2021 года № 245)

Порядок организации и осуществления образовательной деятельности по образовательным программам высшего образования - программам бакалавриата, программам специалитета, программам магистратуры (утвержден приказом Министерства образования и науки Российской Федерации от 5 апреля 2017 года №301)

Письмо Министерства образования Российской Федерации №14-55-996ин/15 от 27 ноября 2002 г. "Об активизации самостоятельной работы студентов высших учебных заведений"

Устав федерального государственного автономного образовательного учреждения "Казанский

(Приволжский) федеральный университет"

Правила внутреннего распорядка федерального государственного автономного образовательного учреждения высшего профессионального образования "Казанский (Приволжский) федеральный университет"

Локальные нормативные акты Казанского (Приволжского) федерального университета

6. Фонд оценочных средств по дисциплине (модулю)

Фонд оценочных средств по дисциплине (модулю) включает оценочные материалы, направленные на проверку освоения компетенций, в том числе знаний, умений и навыков. Фонд оценочных средств включает оценочные средства текущего контроля и оценочные средства промежуточной аттестации.

В фонде оценочных средств содержится следующая информация:

- соответствие компетенций планируемым результатам обучения по дисциплине (модулю);

- критерии оценивания сформированности компетенций;

- механизм формирования оценки по дисциплине (модулю);

- описание порядка применения и процедуры оценивания для каждого оценочного средства;

- критерии оценивания для каждого оценочного средства;

- содержание оценочных средств, включая требования, предъявляемые к действиям обучающихся, демонстрируемым результатам, задания различных типов.

Фонд оценочных средств по дисциплине находится в Приложении 1 к программе дисциплины (модулю).

7. Перечень литературы, необходимой для освоения дисциплины (модуля)

Освоение дисциплины (модуля) предполагает изучение основной и дополнительной учебной литературы. Литература может быть доступна обучающимся в одном из двух вариантов (либо в обоих из них):

- в электронном виде - через электронные библиотечные системы на основании заключенных КФУ договоров с правообладателями;

- в печатном виде - в Научной библиотеке Елабужского института КФУ. Обучающиеся получают учебную литературу на абонементе по читательским билетам в соответствии с правилами пользования Научной библиотекой.

Электронные издания доступны дистанционно из любой точки при введении обучающимся своего логина и пароля от личного кабинета в системе "Электронный университет". При использовании печатных изданий библиотечный фонд должен быть укомплектован ими из расчета не менее 0,25 экземпляра каждого из изданий основной и дополнительной литературы на каждого обучающегося из числа лиц, одновременно осваивающих данную дисциплину.

Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины (модуля), находится в Приложении 2 к рабочей программе дисциплины. Он подлежит обновлению при изменении условий договоров КФУ с правообладателями электронных изданий и при изменении комплектования фондов Научной библиотеки Елабужского института КФУ.

8. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)

Проектирование систем искусственного интеллекта - https://www.intuit.ru/studies/courses/1122/167/info

Машинное обучение (курс лекций, К.В.Воронцов) - https://edu.mmcs.sfedu.ru/mod/url/view.php?id=22249

Введение в машинное обучение: полное руководство - https://www.machinelearningmastery.ru/machine-learning-introduction-a-comprehensive-guide-af6712cf68a3/

9. Методические указания для обучающихся по освоению дисциплины (модуля)

Вид работ	Методические рекомендации
лекции	Лекционные занятия проводятся с использованием интерактивных технологий и предполагают активное участие студентов. Для подготовки к занятиям рекомендуется выделять в материале проблемные вопросы, затрагиваемые преподавателем в лекции, и группировать информацию вокруг них. Желательно выделять в используемой литературе постановки вопросов, на которые разными авторам могут быть даны различные ответы. На основании постановки таких вопросов следует собирать аргументы в пользу различных вариантов решения поставленных проблем.

Вид работ	Методические рекомендации
лабораторные работы	Лабораторные занятия - это одна из разновидностей практического занятия, являющаяся эффективной формой учебных занятий в организации высшего образования. Лабораторные занятия имеют выраженную специфику в зависимости от учебной дисциплины, углубляют и закрепляют теоретические знания. На этих занятиях студенты осваивают конкретные методы изучения дисциплины, обучаются экспериментальным способам анализа, умению работать с приборами и современным оборудованием. Лабораторные занятия дают наглядное представление об изучаемых явлениях и процессах, студенты осваивают постановку и ведение эксперимента, учатся умению наблюдать, оценивать полученные результаты, делать выводы и обобщения. Отчёт по итогам выполненных лабораторных работ выполняется на листах белой бумаги формата А4 в печатном или рукописном виде. При оформлении отчёта используется сквозная нумерация страниц, считая титульный лист первой страницей. Номер страницы на титульном листе не ставится. Номера страницы ставятся по центру вверху. При оформлении отчёта в печатном виде желательно соблюдать следующие требования. Для заголовков: полужирный шрифт, 14 пт, центрированный. Для основного текста: нежирный шрифт, 14 пт, выравнивание по ширине. Во всех случаях тип шрифта - Times New Roman, отступ абзаца 1.25 см, полуторный междустрочный интервал. Поля: левое - 3 см, правое - 1 см, верхнее и нижнее - 2 см. Отчет должен содержать следующие элементы: 1) Титульный лист с обязательным указанием варианта; 2) Цель работы; 3) Задание; 4) Основная часть; 5) Вывод
самостоя-тельная работа	Самостоятельная работа студентов по дидактической сути представляет собой комплекс условий обучения, организуемых преподавателем и направленных на самоподготовку учащихся. Учебная деятельность протекает без непосредственного участия преподавателя и заключается в проработке лекционного материала, подготовке к лабораторным занятиям; изучении учебной литературы из основного и дополнительного списка.
зачет	Зачет является формой оценки качества освоения студентом образовательной программы по дисциплине. По результатам зачета студенту выставляется оценка "зачтено" или "не зачтено". Зачет может проводиться в форме устного опроса по билетам (вопросам) или без билетов, с предварительной подготовкой или без подготовки, по усмотрению кафедры. Преподаватель может проставить зачет без опроса или собеседования тем студентам, которые активно участвовали на лабораторных занятиях.

10. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень программного обеспечения и информационных справочных систем (при необходимости)

Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень программного обеспечения и информационных справочных систем, представлен в Приложении 3 к рабочей программе дисциплины (модуля).

11. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)

Материально-техническое обеспечение образовательного процесса по дисциплине (модулю) включает в себя следующие компоненты:

Учебная аудитория для проведения учебных занятий лекционного типа, семинарского типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации № 61

Комплект мебели для преподавателя – 1 шт., посадочные места для обучающихся – 30 шт., одноместные столы – 12 шт., компьютерные столы – 18 шт., компьютеры – 19 шт., интерактивная панель – 1 шт., меловая доска настенная – 1 шт., выход в интернет, внутривузовская компьютерная сеть, доступ в электронную информационно-образовательную среду.

Помещение для самостоятельной работы № 10

Посадочные места для пользователей – 28 шт., металлические двусторонние стеллажи для книг – 11 шт., книжный шкаф открытый – 5 шт., проектор – 1 шт., ноутбуки для пользователей – 11 шт., шкаф каталожный – 8 шт., шкаф для одежды – 1 шт., ксерокс – 1 шт., рабочий стол библиотекаря – 1 шт., компьютер библиотекаря – 1 шт., вешалка для одежды – 1 шт., жалюзи рулонные «Омега» с фотопечатью – 4 шт., стенд настенный (бронированное стекло) – 4 шт., шкаф-витрина встроенный в арку – 2 шт., шкаф-витрина стеклянный – 2 шт., стеллаж трубчатый с деревянными полками – 2 шт., рабочий стол для инвалидов и лиц с ОВЗ – 2 шт., стол СИ-1 рабочий для инвалидов-колясочников – 1 шт., компьютер– 2 шт., наушники – 2 шт., устройство «Говорящая книга» (тифлоплеер) – 2 шт., видеоувеличитель – 2 шт., радиокласс – 1 шт., портативный тактильный дисплей - 1 шт.,

сканирующая читающая машина - 1 шт., сканер – 1 шт., веб-камера – 1 шт., выход в интернет, внутривузовская компьютерная сеть, доступ в электронную информационно-образовательную среду.

12. Средства адаптации преподавания дисциплины к потребностям обучающихся инвалидов и лиц с ограниченными возможностями здоровья

При необходимости в образовательном процессе применяются следующие методы и технологии, облегчающие восприятие информации обучающимися инвалидами и лицами с ограниченными возможностями здоровья:

- создание текстовой версии любого нетекстового контента для его возможного преобразования в альтернативные формы, удобные для различных пользователей;

- создание контента, который можно представить в различных видах без потери данных или структуры, предусмотреть возможность масштабирования текста и изображений без потери качества, предусмотреть доступность управления контентом с клавиатуры;

- создание возможностей для обучающихся воспринимать одну и ту же информацию из разных источников - например, так, чтобы лица с нарушениями слуха получали информацию визуально, с нарушениями зрения - аудиально;

- применение программных средств, обеспечивающих возможность освоения навыков и умений, формируемых дисциплиной, за счёт альтернативных способов, в том числе виртуальных лабораторий и симуляционных технологий;

- применение дистанционных образовательных технологий для передачи информации, организации различных форм интерактивной контактной работы обучающегося с преподавателем, в том числе вебинаров, которые могут быть использованы для проведения виртуальных лекций с возможностью взаимодействия всех участников дистанционного обучения, проведения семинаров, выступления с докладами и защиты выполненных работ, проведения тренингов, организации коллективной работы;

- применение дистанционных образовательных технологий для организации форм текущего и промежуточного контроля;

- увеличение продолжительности сдачи обучающимся инвалидом или лицом с ограниченными возможностями здоровья форм промежуточной аттестации по отношению к установленной продолжительности их сдачи:

- продолжительности сдачи зачёта или экзамена, проводимого в письменной форме, - не более чем на 90 минут;

- продолжительности подготовки обучающегося к ответу на зачёте или экзамене, проводимом в устной форме, - не более чем на 20 минут;

- продолжительности выступления обучающегося при защите курсовой работы - не более чем на 15 минут.

Программа составлена в соответствии с требованиями ФГОС ВО и учебным планом по направлению 15.03.06 «Мехатроника и робототехника» и профилю подготовки "Физические основы мехатроники и робототехники".

Приложение №1
к рабочей программе дисциплины (модуля)
Б1.В.ДВ.02.02 Машинное обучение

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
"Казанский (Приволжский) федеральный университет"
Елабужский институт (филиал) КФУ

Фонд оценочных средств по дисциплине
Машинное обучение

Направление подготовки: 15.03.06 Мехатроника и робототехника
Профиль подготовки: Физические основы мехатроники и робототехники
Квалификация выпускника: бакалавр
Форма обучения: очно-заочная
Язык обучения: русский
Год начала обучения по образовательной программе: 2024

СОДЕРЖАНИЕ

1. Соответствие компетенций планируемым результатам обучения по дисциплине (модулю)

Код и наименование компетенции	Индикаторы достижения компетенций для данной дисциплины	Оценочные средства текущего контроля и промежуточной аттестации
ПК-2 Способен разрабатывать, отлаживать, внедрять и сопровождать программное обеспечение мехатронных и робототехнических систем	Знать классы задач в мехатронных и робототехнических системах, решаемых с помощью алгоритмов машинного обучения Уметь применять алгоритмы машинного обучения при разработке, отладке и сопровождении программного обеспечения для мехатронных и робототехнических систем Владеть способностью реализовывать алгоритмы машинного обучения при разработке, отладке и сопровождении программного обеспечения для мехатронных и робототехнических систем	**Текущий контроль:** Реферат по темам Тема 1. Основные понятия машинного обучения Тема 2. Кластеризация Тема 3. Метод опорных векторов Тема 4. Деревья решений Лабораторные работы по темам Тема 1. Основные понятия машинного обучения Тема 2. Кластеризация Тема 3. Метод опорных векторов Тема 4. Деревья решений **Промежуточная аттестация:** *Зачёт*

2. Критерии оценивания сформированности компетенций

Компетенция	Зачтено			Не зачтено
	Высокий уровень (отлично) (86-100 баллов)	**Средний уровень (хорошо) (71-85 баллов)**	**Низкий уровень (удовлетворительно) (56-70 баллов)**	**Ниже порогового уровня (неудовлетворительно) (0-55 баллов)**
ПК-2	Знает классы задач в мехатронных и робототехнических системах, решаемых с помощью алгоритмов машинного обучения	Знает классы задач в мехатронных и робототехнических системах, решаемых с помощью алгоритмов машинного обучения, допуская незначительные ошибки в рассуждениях и выводах	Знает классы задач в мехатронных и робототехнических системах, решаемых с помощью алгоритмов машинного обучения, допуская типичные ошибки в рассуждениях и выводах	Не знает классы задач в мехатронных и робототехнических системах, решаемых с помощью алгоритмов машинного обучения.
	Умеет применять алгоритмы машинного обучения при разработке, отладке и сопровождении программного обеспечения для мехатронных и робототехнических систем	Умеет применять алгоритмы машинного обучения при разработке, отладке и сопровождении программного обеспечения для мехатронных и робототехнических систем, допуская незначительные ошибки.	Умеет применять алгоритмы машинного обучения при разработке, отладке и сопровождении программного обеспечения для мехатронных и робототехнических систем, допуская типичные ошибки	Не умеет применять алгоритмы машинного обучения при разработке, отладке и сопровождении программного обеспечения для мехатронных и робототехнических систем
	Владеет способностью реализовывать алгоритмы машинного обучения при разработке, отладке и сопровождении	Владеет способностью реализовывать алгоритмы машинного обучения при разработке, отладке и сопровождении программного	Владеет способностью реализовывать алгоритмы машинного обучения при разработке, отладке и сопровождении	Не владеет способностью реализовывать алгоритмы машинного обучения при

	программного обеспечения для мехатронных и робототехнических систем	обеспечения для мехатронных и робототехнических систем, допуская незначительные ошибки	программного обеспечения для мехатронных и робототехнических систем, допуская типичные ошибки	разработке, отладке и сопровождении программного обеспечения для мехатронных и робототехнических систем

3. Распределение оценок за формы текущего контроля и промежуточную аттестацию

4 семестр:
Текущий контроль:
Реферат
Тема 1. Основные понятия машинного обучения
Тема 2. Кластеризация
Тема 3. Метод опорных векторов
Тема 4. Деревья решений
Максимальное количество баллов по БРС - 10.

Лабораторные работы
Тема 1. Основные понятия машинного обучения
Тема 2. Кластеризация
Тема 3. Метод опорных векторов
Тема 4. Деревья решений

Максимальное количество баллов по БРС - 40.
Итого 10+40= 50 баллов

Промежуточная аттестация - зачет- 50 баллов.
Промежуточная аттестация проводится после завершения изучения дисциплины или ее части в форме, определяемой учебным планом образовательной программы с целью оценить работу обучающегося, степень усвоения теоретических знаний, уровень сформированности компетенций.
Преподаватель, принимающий зачет обеспечивает случайное распределение вариантов зачетных заданий между обучающимися с помощью билетов и/или с применением компьютерных технологий; вправе задавать обучающемуся дополнительные вопросы и давать дополнительные задания помимо тех, которые указаны в билете.
Зачет проводится по билетам. В каждом билете оценочные средства одного вида: устный или письменный ответ на вопрос.
Устный или письменный ответ – 50 баллов.
Итого 50 баллов.
Общее количество баллов по дисциплине за текущий контроль и промежуточную аттестацию: 50+50=100 баллов.

Соответствие баллов и оценок:
Для зачета:

56-100 – зачтено
0-55 – не зачтено

4. Оценочные средства, порядок их применения и критерии оценивания
4.1. Оценочные средства текущего контроля
4.1.1. Реферат
Тема 1. Основные понятия машинного обучения
Тема 2. Кластеризация
Тема 3. Метод опорных векторов
Тема 4. Деревья решений

4.1.1.1. Порядок проведения.
Обучающиеся самостоятельно пишут работу на заданную тему и сдают преподавателю в письменном виде. В работе производится обзор материала в определённой тематической области либо предлагается собственное решение определённой теоретической или практической проблемы. Оцениваются проработка источников,

изложение материала, формулировка выводов, соблюдение требований к структуре и оформлению работы, своевременность выполнения. В случае публичной защиты реферата оцениваются также ораторские способности

Требования к реферату

При оформлении текста реферата следует придерживаться следующих параметров:

поля: левое – 35 мм, правое – 15 мм, верхнее – 25 мм, нижнее – 25 мм;

ориентация страницы: книжная;

шрифт: TimesNewRoman;

кегль: 14 пт (пунктов);

красная строка:1 мм;

междустрочный интервал: полуторный;

выравнивание основного текста и сносок: по ширине.

Иллюстрации в виде рисунков, фотоснимков, схем и т.п. могут располагаться органично с текстом (возможно ближе к иллюстрируемой части) либо на отдельных листах. В любом случае выполняется нумерация (сквозная для всех разделов), которая располагается вверху. Подрисуночную нумерацию и надпись располагать внизу.

Заканчивается пояснительная записка библиографическим списком источников, к которым обращался студент во время работы над разрабатываемой темой.

Объем информационно-технологической документации не регламентируется – он диктуется достаточностью для практического применения. Карточки задания для самоконтроля (если таковы имеются) вкладываются в прозрачные файлы.

Реферат по своему структурному содержанию должен содержать следующие элементы:

- титульный лист;
- содержание;
- введение;
- базовое понятия;
- историческая справка (особенности зарождения и развития, основоположники и т.д.);
- классификация (виды, формы и т.д.);
- общее и частное положения по применению в учебно-воспитательном процессе;
- глоссарий;
- список использованных источников
- приложения

4.1.1.2 Критерии оценивания

9-10 баллов ставится, если обучающийся:

Тему раскрыл полностью. Продемонстрировал превосходное владение материалом. Использовал надлежащие источники в нужном количестве. Структура работы соответствует поставленным задачам. Степень самостоятельности работы высокая.

7-8 баллов ставится, если обучающийся:

Тему в основном раскрыл. Продемонстрировал хорошее владение материалом. Использовал надлежащие источники. Структура работы в основном соответствует поставленным задачам. Степень самостоятельности работы средняя.

5-6 баллов ставится, если обучающийся:

Тему раскрыл слабо. Продемонстрировал удовлетворительное владение материалом. Использованные источники и структура работы частично соответствуют поставленным задачам. Степень самостоятельности работы низкая.

0-4 баллов ставится, если обучающийся:

Тему не раскрыл. Продемонстрировал неудовлетворительное владение материалом. Использованные источники недостаточны. Структура работы не соответствует поставленным задачам. Работа несамостоятельна.

4.1.1.3. Содержание оценочного средства

1. Векторные представления текстов и графов.
2. Адаптивные системы прогнозирования.
3. Генетические алгоритмы.
4. Нелинейная регрессия.
5. Поиск ассоциативных правил.
6. Самоорганизующиеся карты Кохонена.
7. Многомерная линейная регрессия
8. Обучение без учителя.
9. Методы понижения размерности.
10. Иерархические методы кластеризации.
11. Data Science.
12. Градиентный спуск.
13. Переобучение модели.
14. Логистическая регрессия.

15. Алгоритм AdaBoost.
16. Методы прогнозирования.
17. Машинное обучение в обработке текстов.
18. Машинное обучение в транспорте.

4.1.2. Лабораторные работы

Тема 1. Основные понятия машинного обучения
Тема 2. Кластеризация
Тема 3. Метод опорных векторов
Тема 4. Деревья решений

4.1.2.1. Порядок проведения.

В аудитории, оснащённой соответствующим оборудованием, обучающиеся проводят учебные эксперименты и тренируются в применении практико-ориентированных технологий. Оцениваются знание материала и умение применять его на практике, умения и навыки по работе с оборудованием в соответствующей предметной области.

Лабораторные работы проводятся преподавателем согласно разработанному и утвержденному на кафедре рабочей программе. Каждая лабораторно-практическая работа выполняется по определенной теме программы в соответствии с заданием.

Перед выполнением каждой работы студенты-бакалавры должны проработать соответствующий материал, используя конспекты теоретических занятий, периодические издания, учебно-методические пособия и учебники

На каждом занятии студенты выполняют работу в соответствии с ее содержанием и методическими указаниями.

По окончании занятий студенты оформляют отчет по каждой работе, соблюдая следующую форму:

- Наименование темы;
- Цель работы;
- Задание и содержание выполненной работы,
- Письменные ответы на контрольные вопросы.
- Выводы по проделанной работе.
- Список использованных источников.

4.1.2.2 Критерии оценивания

35-40 баллов ставится, если обучающийся:
Правильно выполнил все задания. Продемонстрировал высокий уровень владения материалом. Проявлены превосходные способности применять знания и умения к выполнению конкретных заданий.
29-34 баллов ставится, если обучающийся:
Правильно выполнил большую часть заданий. Присутствуют незначительные ошибки. Продемонстрирован хороший уровень владения материалом. Проявлены средние способности применять знания и умения к выполнению конкретных заданий.
23-28 баллов ставится, если обучающийся:
Задания выполнил более чем наполовину. Присутствуют серьёзные ошибки. Продемонстрирован удовлетворительный уровень владения материалом. Проявлены низкие способности применять знания и умения к выполнению конкретных заданий.
0-22 баллов ставится, если обучающийся:
Задания выполнил менее чем наполовину. Продемонстрирован неудовлетворительный уровень владения материалом. Проявлены недостаточные способности применять знания и умения к выполнению конкретных заданий.

4.1.2.3. Содержание оценочного средства

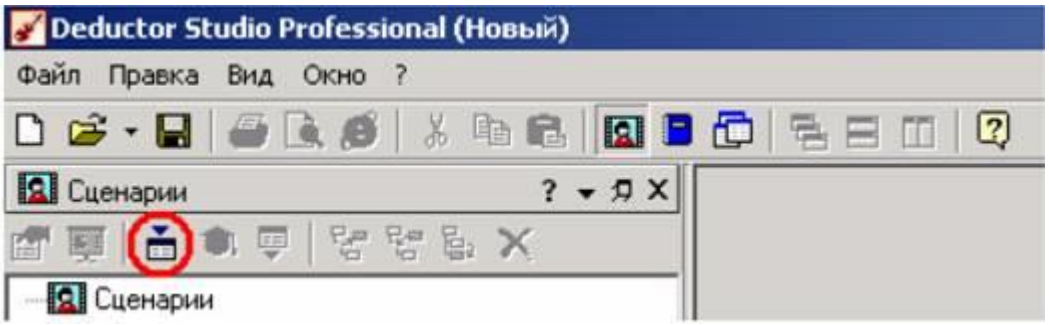
Лабораторная работа 1. Предварительная обработка данных

Задание

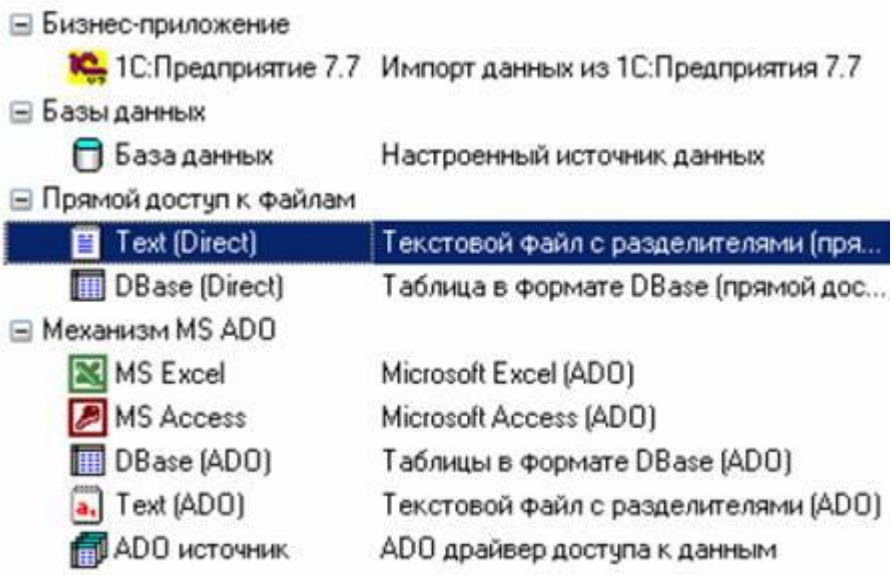
Ознакомиться с возможностями аналитического пакета Deductor, выполнив приведённые ниже задания.

Импорт данных

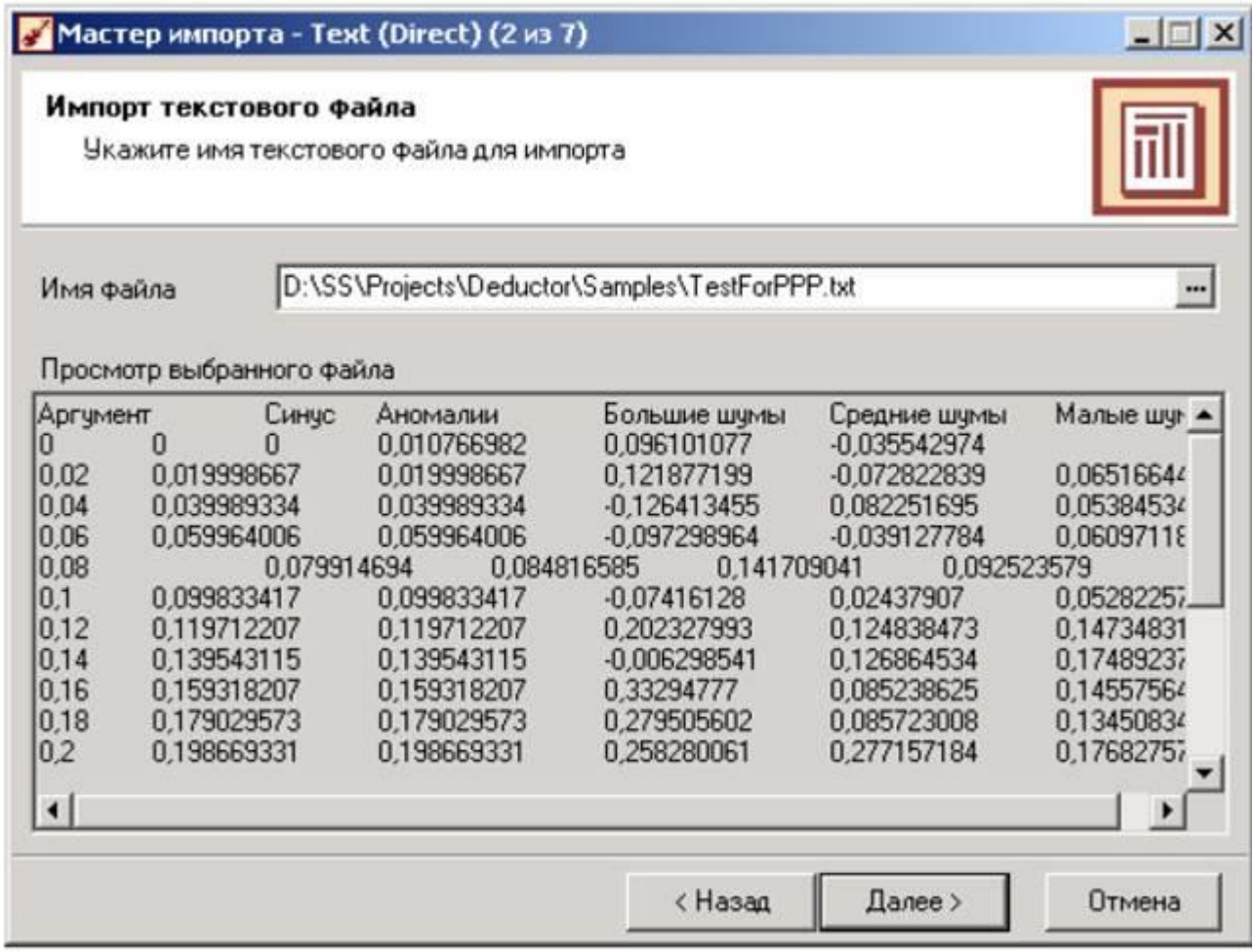
Импорт данных является отправной точкой анализа данных. Импорт в Deductor может осуществляться из популярных форматов хранения данных, таких как Excel, Access, MS SQL, Oracle, Текстовый файл и прочих. Кроме того, имеется универсальный доступ к любому источнику данных посредством ADO или ODBC.



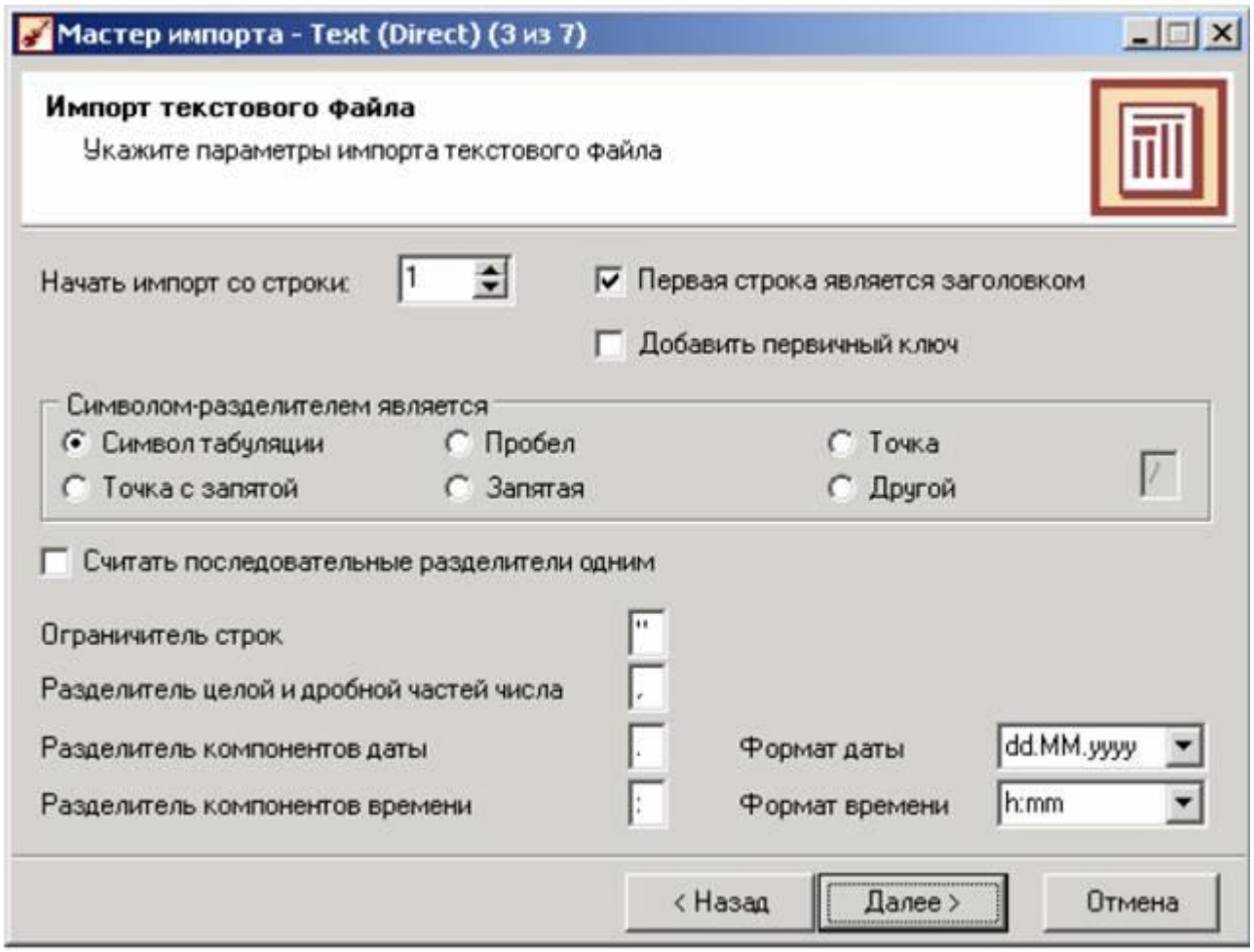
Импорт данных из текстового файла с разделителями осуществляется путем вызова мастера импорта на панели «Сценарии».



После запуска мастера импорта укажем тип импорта “Текстовый файл с разделителями” и перейдем к настройке импорта. Укажем имя файла, из которого необходимо получить данные (пример для парциальной обработки). В окне просмотра выбранного файла можно увидеть содержание данного файла.



Далее перейдем к настройке параметров импорта. На этой странице мастера предоставляется возможность указать, с какой строки следует начать импорт, указать, то, что первая строка является заголовком, возможность добавить первичный ключ. Указать, что является символом-разделителем столбцов, а также указать ограничитель строк, разделитель целой и дробной части вещественного числа, разделитель компонентов даты и ее формат.



В данном случае параметры по умолчанию на этой странице мастера установлены правильно, а именно: начать импорт с первой строки, первая строка является заголовком, разделителем между столбцами является знак табуляции, разделителем целой и дробной частей является запятая. Далее перейдем к настройке свойств полей.

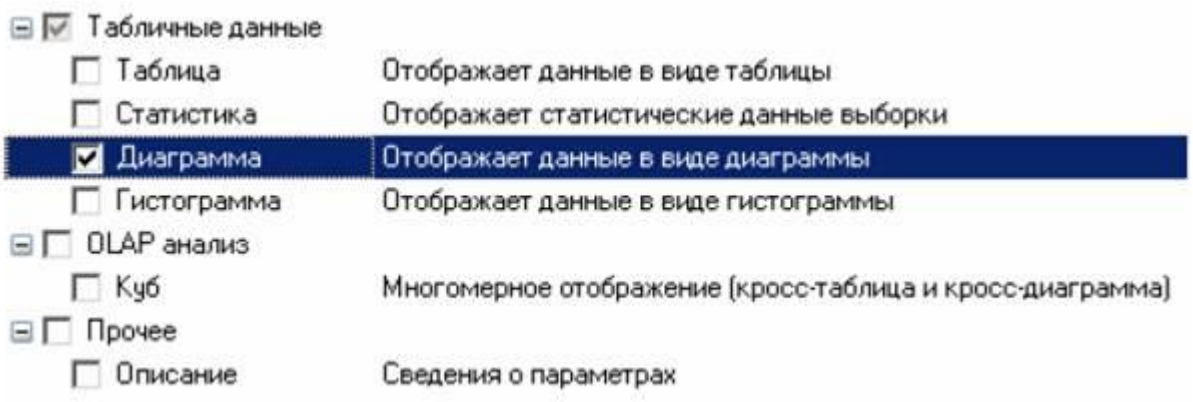
На этом шаге мастера предоставляется возможность настроить имя, название (метку), размер, тип данных, вид данных и назначение. Некоторые свойства (например, тип данных) можно задавать для выделенного набора столбцов. Вид данных определяет – конечный ли это набор (дискретные) или бесконечный (непрерывные). Назначение столбцов определяет характер их использования в алгоритмах обработки (при импорте можно оставить значение по умолчанию).

Для правильного импорта данных необходимо изменить тип данных у первых трех столбцов («АРГУМЕНТ», «СИНУС», «АНОМАЛИИ»). Тип данных по умолчанию неверный, поскольку программа определяет его,

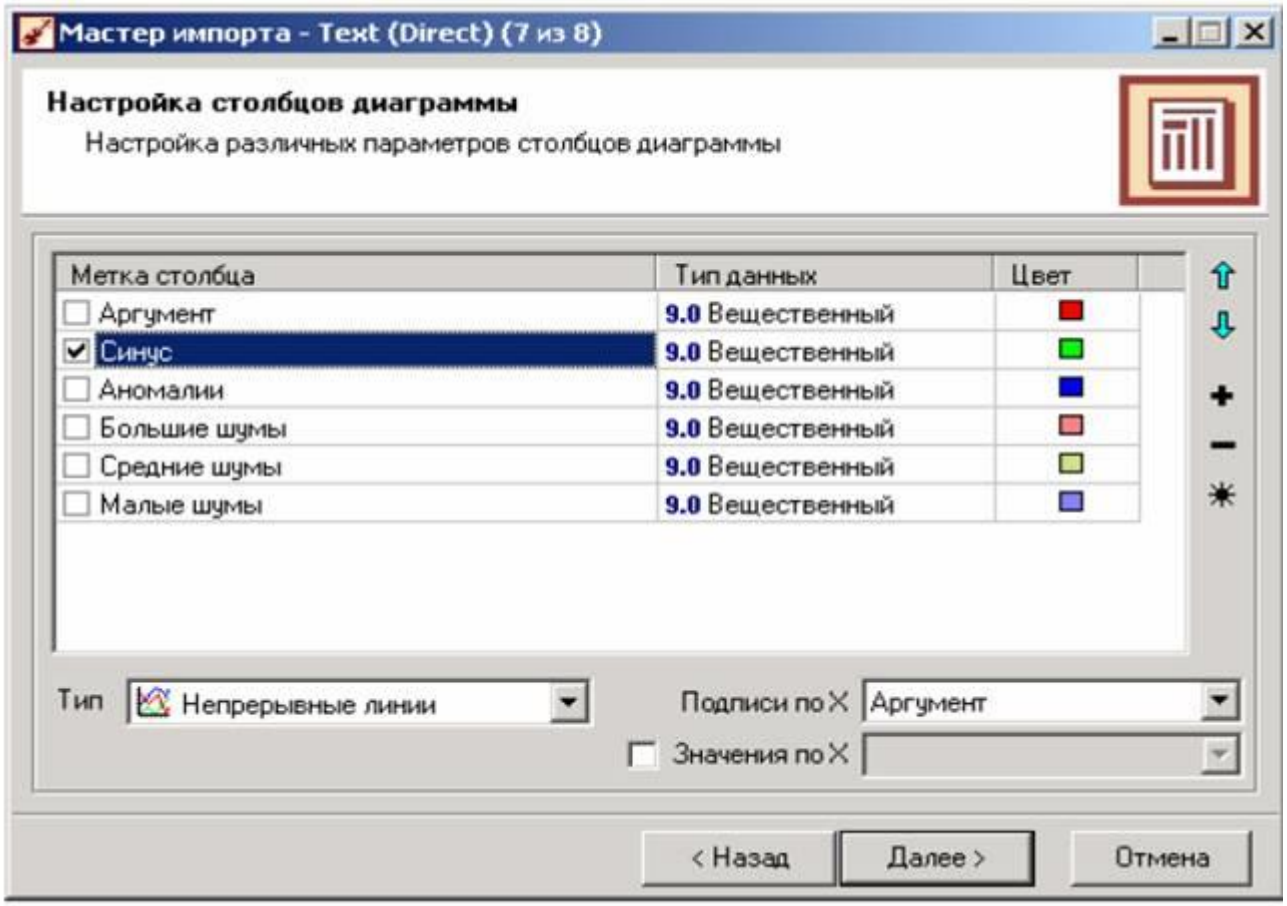
основываясь на значениях первой строки данных. В данном случае там находятся нули – целые числа. Поэтому программа определила, что столбец содержит целочисленные значения.

Выделим их с помощью мыши и укажем им тип данных – «Вещественный». Далее осталось только выполнить импорт данных, нажав на кнопку «Пуск» на следующем шаге мастера импорта.

После импорта данных на следующем шаге мастера необходимо выбрать способ отображения данных. В данном случае самым информативным является диаграмма, выберем ее.



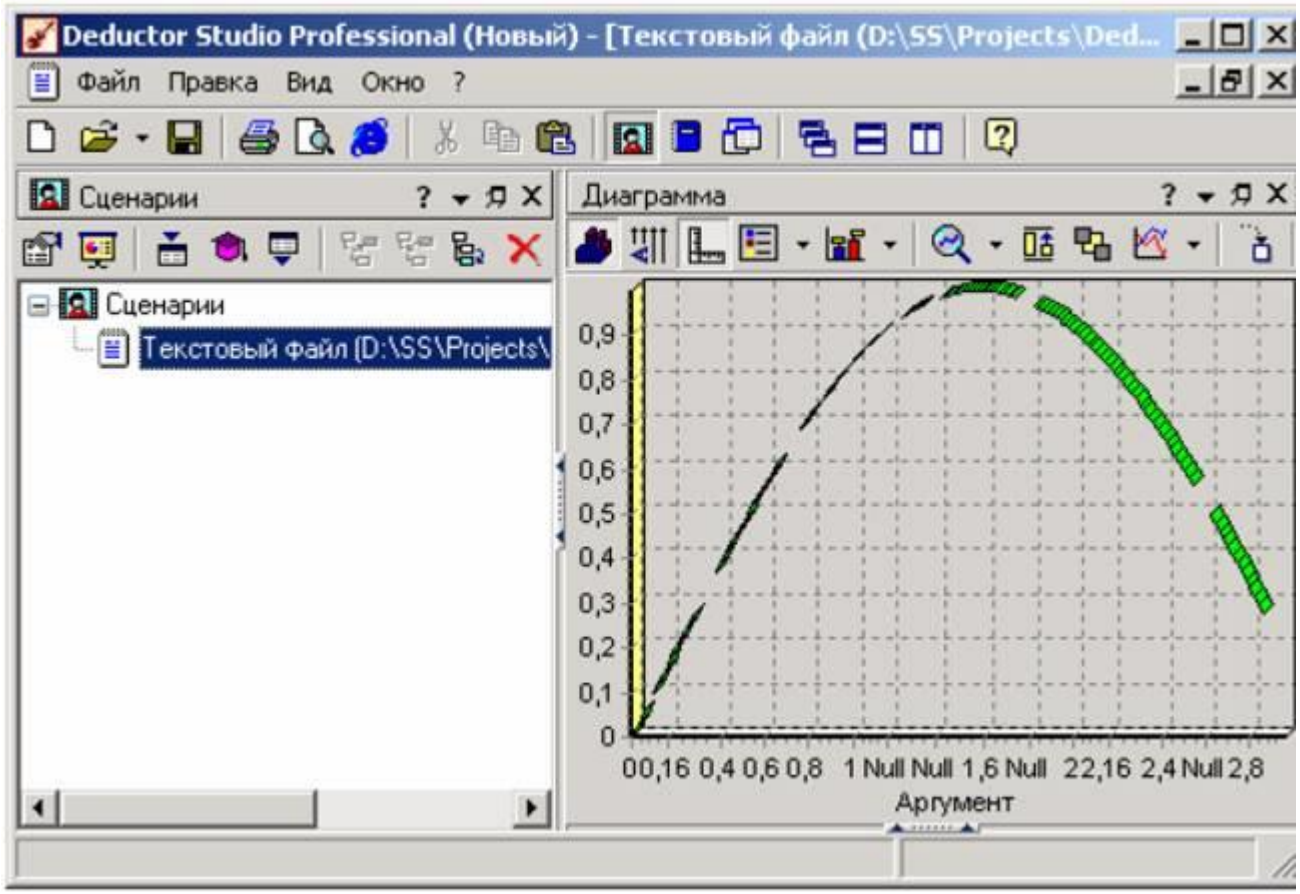
От того, какие способы отображения будут выбраны на этом этапе, зависят последующие шаги мастера. В данном случае необходимо настроить, какие столбцы диаграммы следует отображать и как именно.



Выберем для отображения поле «СИНУС» и тип диаграммы «Линии».

На последнем шаге мастера необходимо указать название ветки в дереве сценариев.

Напишем в поле заголовка окна «Импорт примера для демонстрации парциальной обработки» и нажмем «Готово». На этом работа мастера импорта заканчивается. Теперь в дереве сценариев появится новый узел с необходимыми данными. В главном окне программы представлены все выбранные отображения данных этого узла. В данном случае только диаграмма.



Примеры предобработки данных

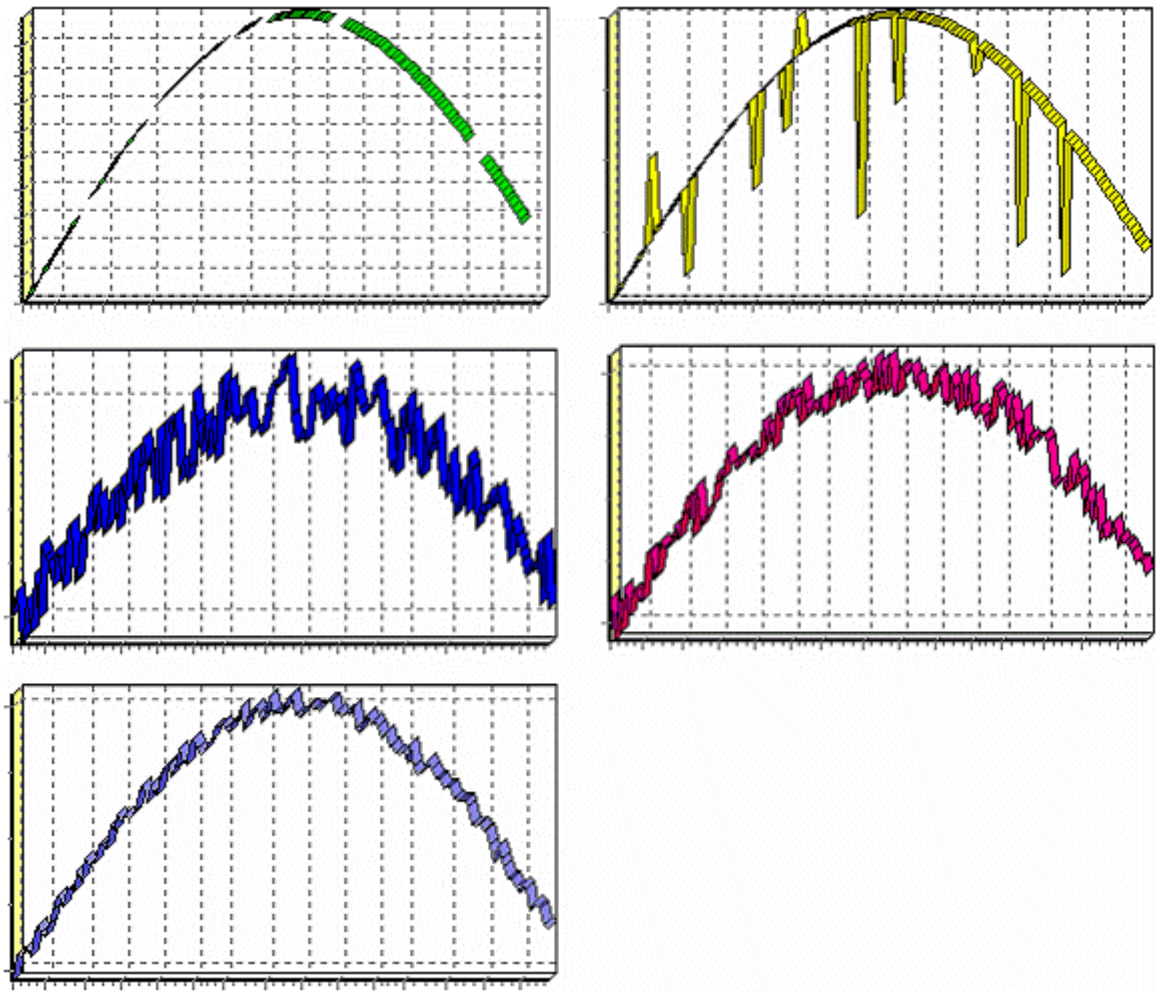
Часто исходные данные для анализа не годятся, а качество данных влияет на качество результатов. Так что вопрос подготовки данных для последующего анализа является очень важным. Обычно «сырые» данные содержат в себе различные шумы, за которыми трудно увидеть общую картину, а также аномалии – влияние случайно, либо редко происходивших событий. Очевидно, что влияние этих факторов на общую модель необходимо минимизировать, т. к. модель, учитывающая их, получится неадекватной.

Парциальная предобработка

Парциальная предобработка служит для восстановления пропущенных данных, редактирования аномальных значений и спектральной обработке данных (например, сглаживания данных). Именно этот шаг часто проводится в первую очередь.

Исходные данные

Рассмотрим применение обработки на примере данных из файла «TestForPPP. txt». Он содержит таблицу со следующими полями: «АРГУМЕНТ» – аргумент, «СИНУС» – значения синуса аргумента (некоторые значения пустые), «АНОМАЛИИ» – синус с выбросами, «БОЛЬШИЕ ШУМЫ» – значения синуса с большими шумами, «СРЕДНИЕ ШУМЫ» – значения синуса со средними шумами, «МАЛЫЕ ШУМЫ» – значения синуса с малыми шумами. Все данные можно увидеть на диаграмме после импорта из текстового файла.



а) Столбец с пропущенными данными

б) Столбец с аномалиями (выбросами)

в) Столбец с большими шумами

г) Столбец со средними шумами

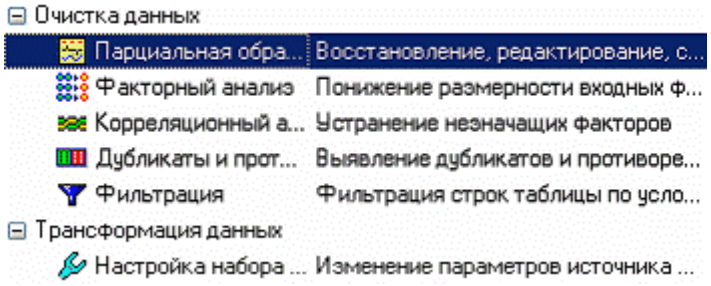
д) Столбец с малыми шумами

Восстановление пропущенных данных

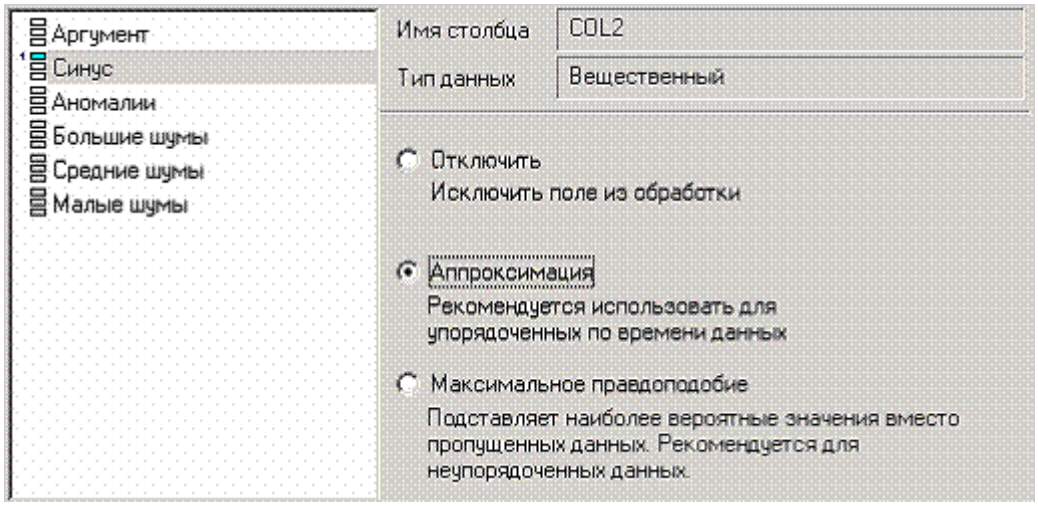
Часто бывает так, что в столбце некоторые данные отсутствуют в силу каких либо причин (данные не известны, либо их забыли внести и т. п.). Обычно из–за этого пришлось бы убрать из обработки все строки, которые содержат пропущенные данные. Но механизмы Deductor Studio позволяют решить эту проблему. Один из шагов парциальной обработки как раз отвечает за восстановление пропущенных значений. Если данные упорядочены

(например, по времени), то рекомендуется в качестве восстановления пропущенных значений использовать аппроксимацию. Алгоритм сам подберет значение, которое должно стоять на месте пропущенного значения, основываясь на близлежащих данных. Если же данные не упорядочены, то следует использовать режим максимального правдоподобия, когда алгоритм подставляет вместо пропущенных данных наиболее вероятные значения, основываясь на всей выборке.

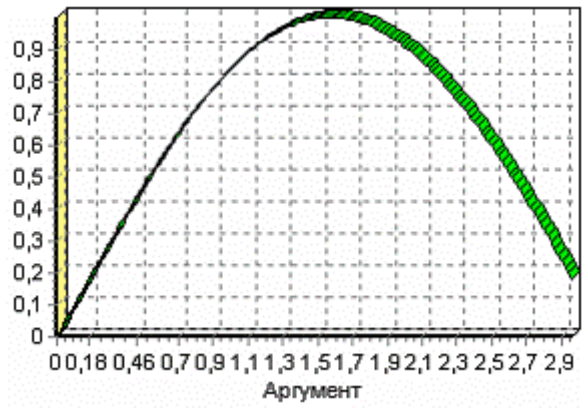
Для демонстрации воспользуемся мастером парциальной обработки. Импортировав файл можно увидеть, что в столбце «СИНУС» содержатся пустые значения. На диаграмме выше видно, что некоторые значения синуса пропущены. Для дальнейшей обработки необходимо из восстановить. Для этого следует запустить мастер парциальной обработки.



Поскольку данные в исходном наборе упорядочены, на следующем шаге мастера обработки выделим поле «СИНУС» и укажем для него тип обработки «Аппроксимация». Так как в данном случае больше ничего не требуется, то остальные параметры обработки оставляем отключенными. Перейдя на страницу запуска процесса обработки, выполняем ее, нажав на пуск, и далее выбираем тип визуализации обработанных данных (как в примере импорта).

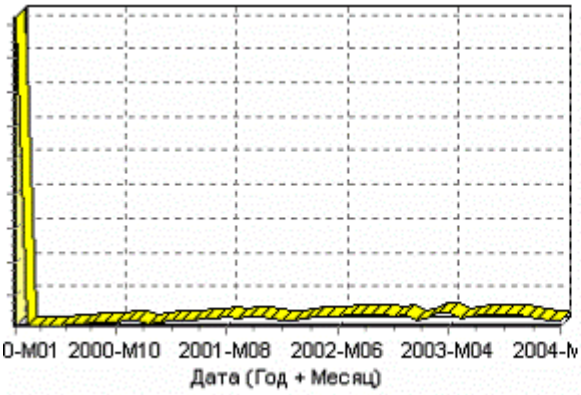


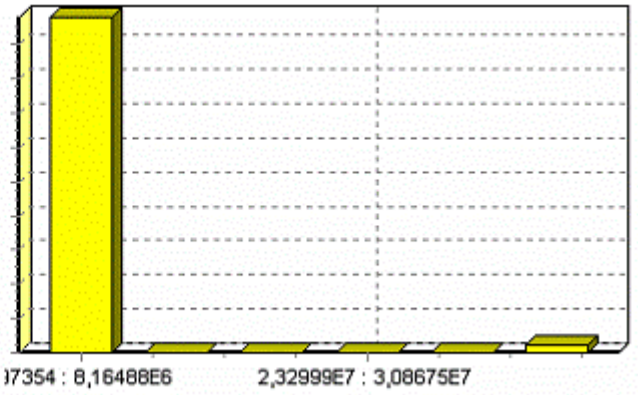
После выполнения процесса обработки на диаграмме видно, что пропуски в данных исчезли, что и было необходимо сделать.



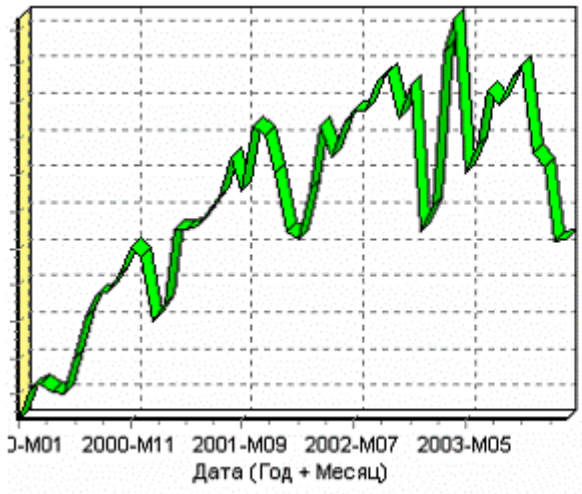
Удаление аномалий

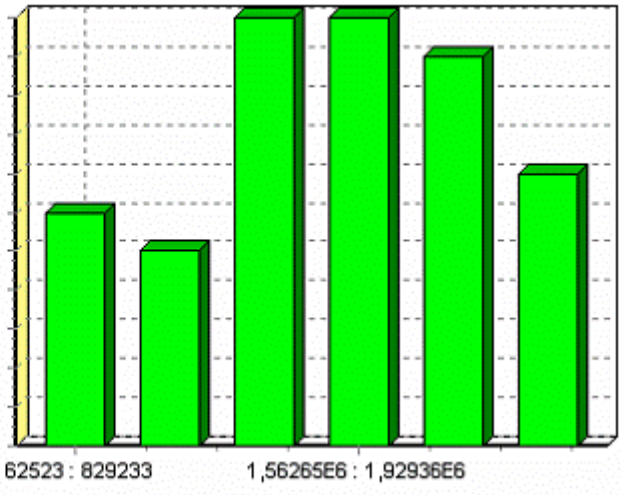
Аномалии встречаются в «сырых» данных не реже шумов. По существу они вообще не должны оказывать никакого влияния на результат. Если же они присутствуют при построении модели, то оказывают на нее весьма большое влияние. Т. е. предварительно их необходимо устранить. Также они портят статистическую картину распределения данных. К примеру, вот как выглядят данные с аномалиями, а также гистограмма их распределения:



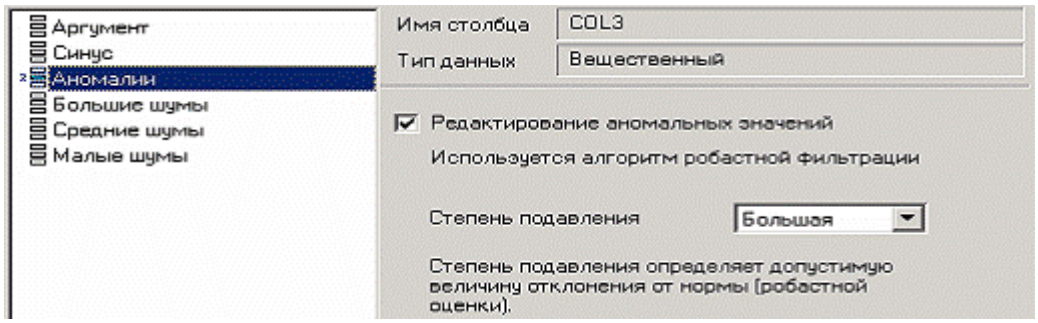


Очевидно, что аномалии не позволяют определить как характер самих данных, так и статистическую картину. После устранения аномалий те же данные представляются в следующем виде:

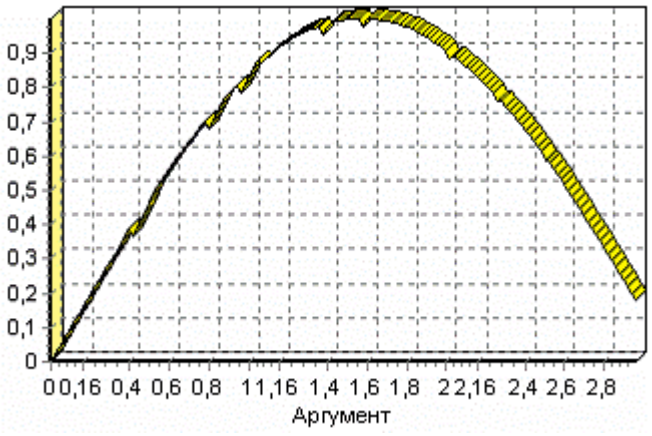




Этот пример еще раз подчеркивает необходимость проведения парциальной обработки данных перед анализом. Вернемся к примеру с удалением аномалий из поля «АНОМАЛИИ» импортированной таблицы.



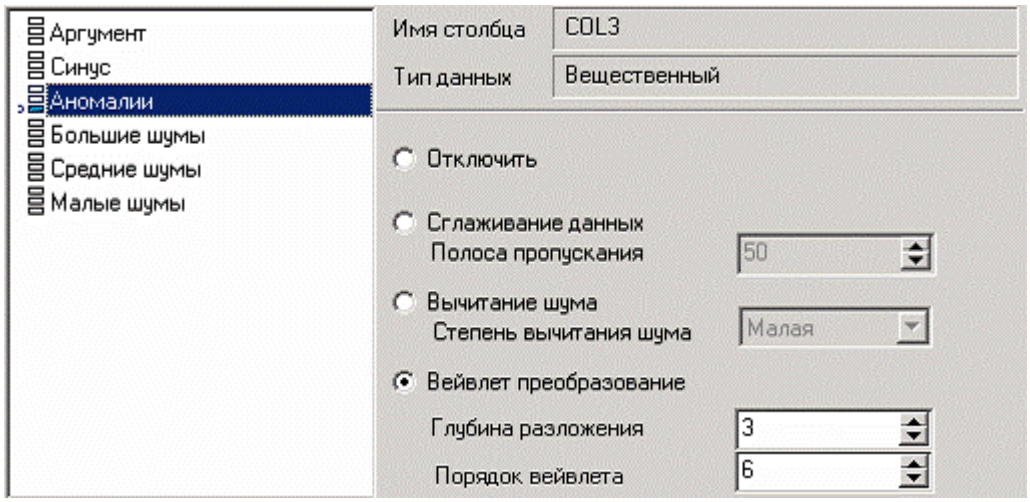
В мастере парциальной предобработки на третьем шаге выбираем поле «АНОМАЛИИ» и указываем ему тип обработки «Удаления аномальных явлений», степень подавления «Большая». Так как больше никаких обработок не планировалось, то переходим на шаг запуска процесса обработки и нажимаем «Пуск». После выполнения процесса обработки на диаграмме видно, что выбросы исчезли, остались лишь небольшие возмущения, которые легко сгладить при помощи спектральной обработки.



Спектральная обработка.

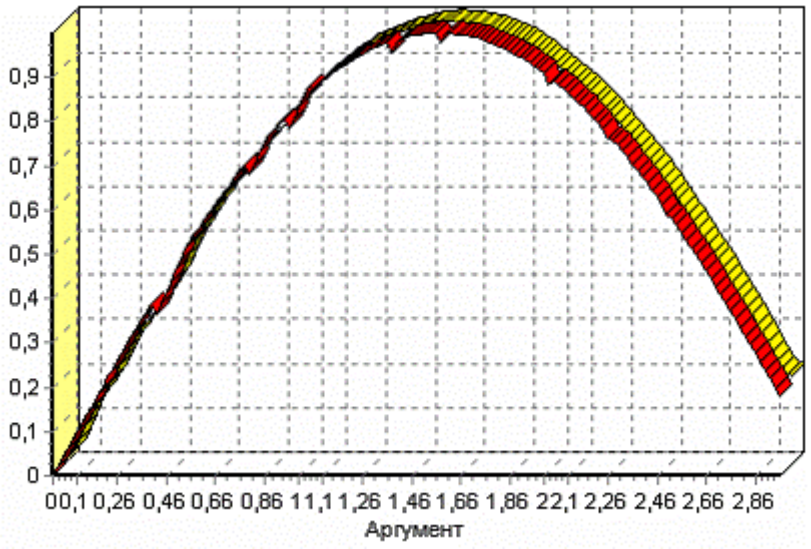
Данные, как мы видим из предыдущего примера, бывает необходимо сгладить. Сглаживание данных применяется для удаления шумов из исходного набора, (что будет продемонстрировано позднее) а также для выделения тенденции, трудно видимой в исходном наборе. Платформа Deductor Studio предлагает несколько видов спектральной обработки: сглаживание данных путем указания полосы пропускания, вычитание шума путем указания степени вычитания шума и вейвлет преобразование путем указания глубины разложения и порядка вейвлета.

Продемонстрируем такой метод спектральной обработки, как вейвлет преобразование. Для этого продолжим работу с данными, полученными в предыдущем примере. Как видно на рисунке, аномалии были устранены, однако небольшие возмущения остались. Сгладим их при помощи парциальной обработки. Для этого после удаления аномалий вновь запустим мастер парциальной обработки. В нем на четвертом шаге выберем поле «АНОМАЛИИ» и укажем ему тип обработки «Вейвлет преобразование» с параметрами по умолчанию (глубина разложения 3, порядок вейвлета 6).



Так как больше ничего не планировалось, то перейдем с шагу запуска процесса обработки и выполним ее. В качестве визуализатора укажем диаграмму. После обработки можно убедиться на диаграмме в отсутствии выбросов и сравнить результат с эталонным значением синуса (столбец «СИНУС»). На рисунке красный (темный) график – значения

синуса, желтый (светлый) – значения сглаженного синуса после устранения аномалий.



Удаление шумов

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

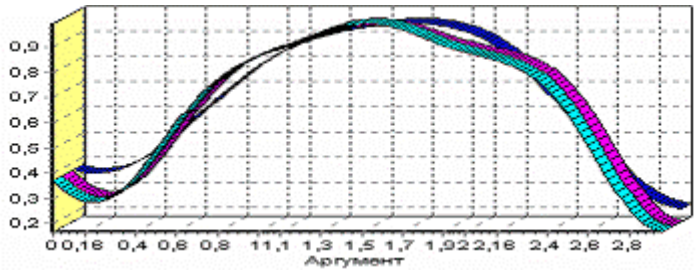
В примере по парциальной обработке, как было показано ранее, есть 3 столбца с шумами: «БОЛЬШИЕ ШУМЫ», «СРЕДНИЕ ШУМЫ», и «МАЛЫЕ ШУМЫ» - соответственно синус с большими, средними и малыми шумами. Ясно, что для дальнейшей работы с данными эти шумы необходимо устранить.

Спектральная обработка, как говорилось ранее, позволяет сделать это с помощью указания для этих полей в качестве типа обработки «Вычитание шума». Настройки обладают определенной гибкостью. Так, существует большая, средняя и малая степень вычитания шума. Аналитик может подобрать степень, устраивающую его.

Удаление больших, малых и средних шумов.

Таким образом, в мастере парциальной обработки на четвертом шаге выберем по очереди поля «БОЛЬШИЕ ШУМЫ», «СРЕДНИЕ ШУМЫ» и «МАЛЫЕ ШУМЫ» , зададим тип обработки «Вычитание шума» и укажем степень подавления – «большая», «средняя» и «малая» соответственно.

После выполнения обработки на диаграмме можно просмотреть полученные результаты.

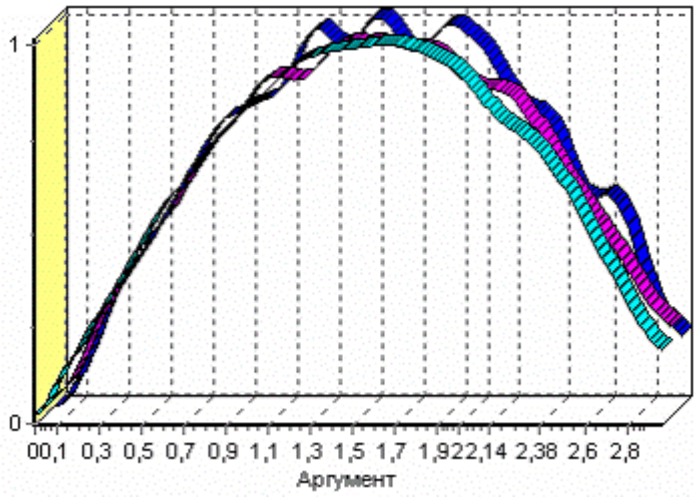


Сглаживание больших, малых и средних шумов

В некоторых случаях неплохие результаты удаления шумов дает вейвлет преобразование. Покажем, какие результаты показывает на этих же данных этот вид спектральной обработки.

В мастере парциальной обработки выберем поля «БОЛЬШИЕ ШУМЫ», «СРЕДНИЕ ШУМЫ» и «МАЛЫЕ ШУМЫ», укажем тип обработки «Вейвлет преобразование», оставив параметры обработки по умолчанию (глубина разложения – 3, порядок вейвлета – 6).

На диаграмме можно убедиться в том, что данные сгладились. Синий график – сглаженные большие шумы, красный – сглаженные средние и желтый – сглаженные малые шумы. Повысить качество сглаживания шумов таким способом можно, путем подбора удовлетворительных параметров обработки.

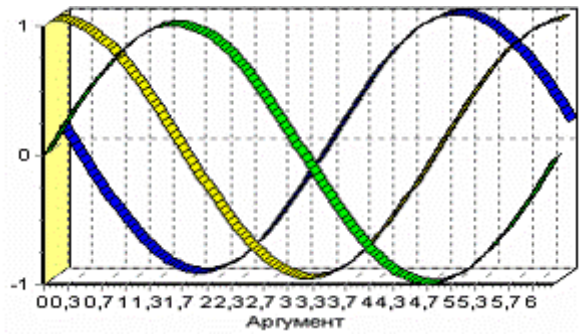


Факторный анализ

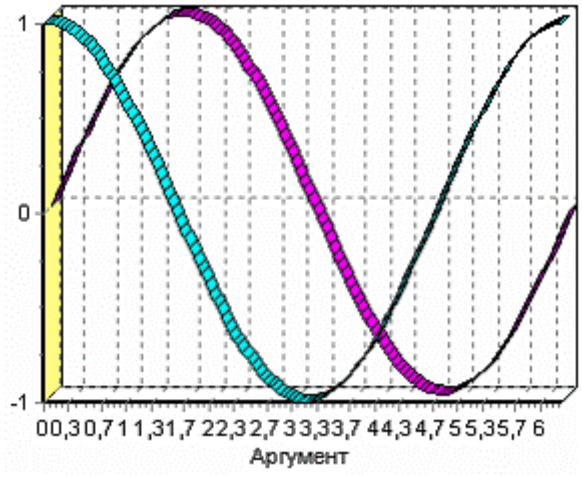
Факторный анализ служит для понижения размерности пространства входных факторов. Обработку можно выполнять как в автоматическом режиме (с указанием порога значимости), так и самостоятельно (основываясь на значениях матрицы значимости).

Исходные данные

Рассмотрим применение обработки на примере данных из файла «TestForCPP. txt». Он содержит таблицу со следующими полями: «АРГУМЕНТ» – аргумент, «ФАКТОР1», «ФАКТОР2», «ФАКТОР3» – входные значения, «РЕЗУЛЬТАТ1», «РЕЗУЛЬТАТ2» – выходные значения.

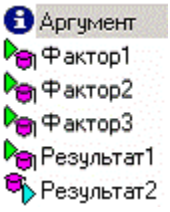


а) Входные факторы



б) Выходы

Понижение размерности пространства входных факторов
В мастере факторного анализа зададим «ФАКТОР1», «ФАКТОР2», «ФАКТОР3» входными полями, «РЕЗУЛЬТАТ1», «РЕЗУЛЬТАТ2» - выходными, а поле «АРГУМЕНТ» – непригодным.

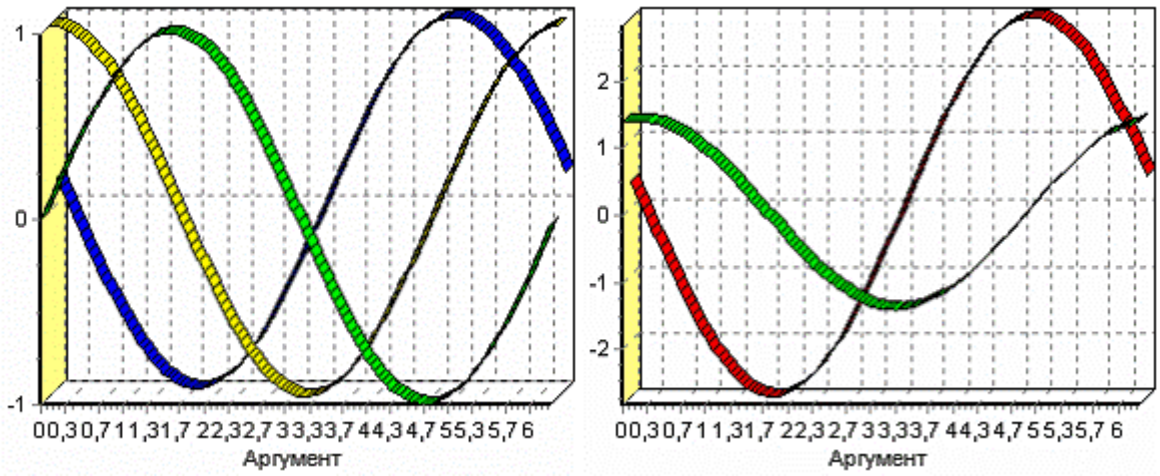


Следующий шаг предлагает запустить процесс понижения размерности пространства входных факторов. После завершения процесса на следующем шаге предлагается выбрать, какие из полученных в результате обработки факторы оставить для дальнейшей работы. Это делается путем указания необходимого порога значимости (по умолчанию порог значимости равен 90%, не будем его менять).

Главные компоненты	Собственное значение	Вклад в результат	Суммарный вклад
☑ Значение 1	2,000	66,66 %	66,66 %
☑ Значение 2	1,000	33,34 %	100,00 %
☐ Значение 3	0,000	00,00 %	

Порог значимости (%) 90

Теперь необходимо перейти на следующий шаг и выбрать способ визуализации: просмотрим результаты на диаграмме.



а) Исходные входные факторы б) Полученные входные факторы.

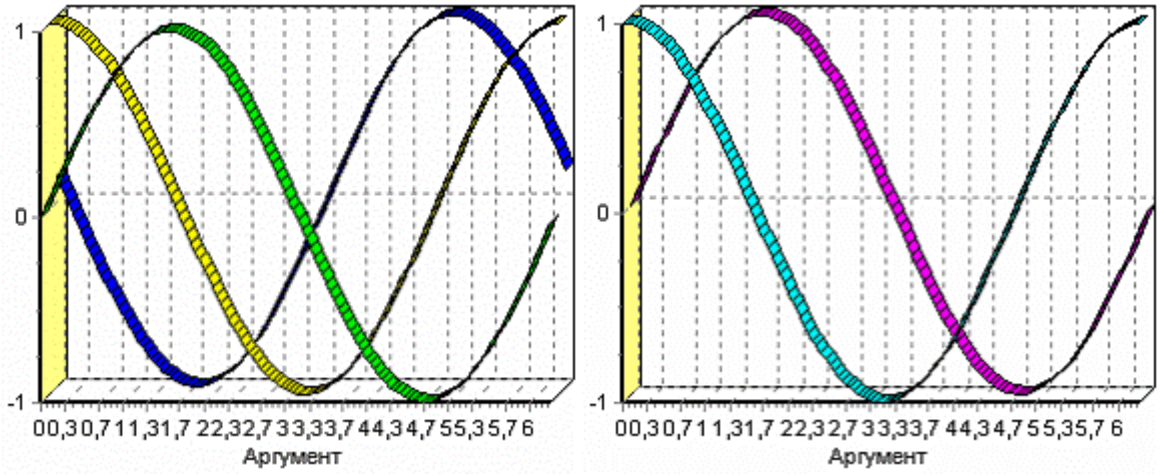
После обработки в наборе данных вместо трех исходных входных полей появились два новых поля – «Фактор1» и «Фактор2» – это результат понижения размерности (было 3 входных фактора, стало 2). На диаграмме видно, что «Фактор2» – близок к полю «ФАКТОР3», следовательно, «Фактор1» – это преобразованные факторы «ФАКТОР1» и «ФАКТОР2».

Корреляционный анализ

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

Исходные данные

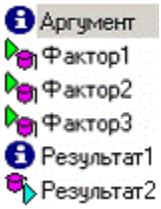
Рассмотрим применение обработки на примере данных из файла «TestForCPP. txt». Он содержит таблицу со следующими полями: «АРГУМЕНТ» – аргумент, «ФАКТОР1», «ФАКТОР2», «ФАКТОР3» – входные значения, «РЕЗУЛЬТАТ1», «РЕЗУЛЬТАТ2» – выходные значения. В данном примере определим степень влияния входных факторов на один из выходов – «РЕЗУЛЬТАТ2» и оставим только значимые факторы.



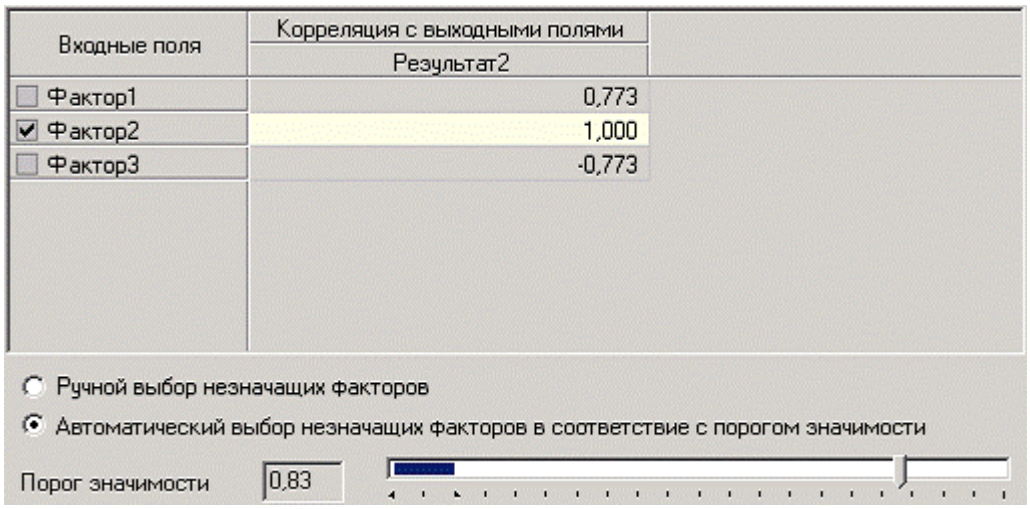
а) Входные факторы б) Выходы

Устранение незначащих входных факторов

В мастере корреляционного анализа зададим «ФАКТОР1», «ФАКТОР2», «ФАКТОР3» входными полями, «РЕЗУЛЬТАТ2» - выходными, а поля «АРГУМЕНТ» и «РЕЗУЛЬТАТ1» – информационным.



Следующий шаг предлагает запустить процесс корреляционного анализа. После завершения процесса на следующем шаге предлагается выбрать, какие факторы оставить для дальнейшей работы. Это делается либо вручную, основываясь на значениях матрицы ковариации, либо путем указания порога значимости (по умолчанию порог значимости равен 0.05). Из рассчитанной матрицы ковариации видно, что выходное поле «РЕЗУЛЬТАТ2» напрямую зависит от поля «ФАКТОР2» (вообще, значение коэффициента, равное 1.000 говорит о том, что эти поля идентичны), и в меньшей степени от остальных факторов. В данном случае без потери полезной информации можно исключить из дальнейшего рассмотрения «Фактор1» и «Фактор3».



Теперь необходимо перейти на следующий шаг и выбрать способ визуализации: просмотрим результаты на диаграмме (например, можно убедиться в идентичности полей «Фактор2» и «Результат2»). Таким образом, корреляционный анализ позволил проанализировать влияние входных факторов на результат и исключить незначащие факторы из дальнейшего анализа.

Трансформация данных

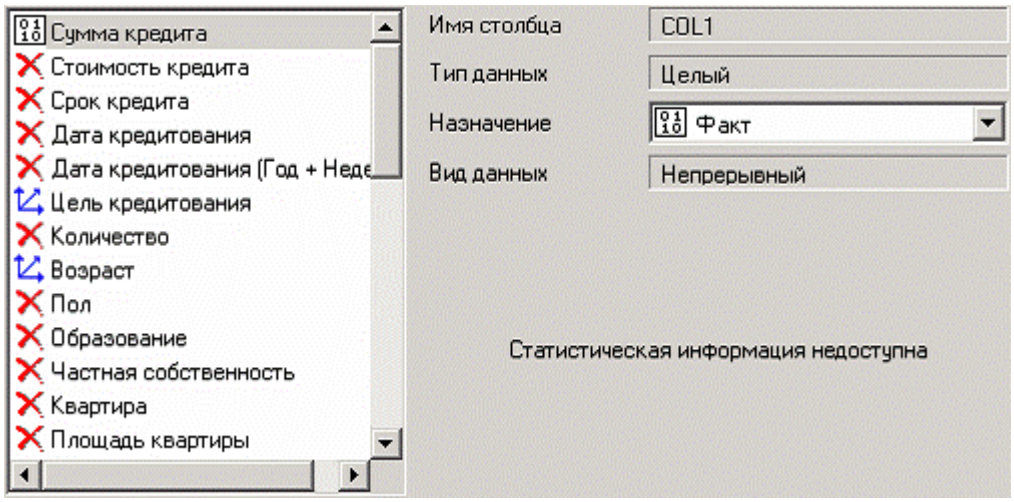
Часть примеров, входящих в группу трансформация данных показывает, как с помощью инструментов Deductor Studio можно добиться тех или иных промежуточных задач, касающихся сбора аналитической информации, либо разбиения данных на какие-либо группы по определенным критериям.

Разбиение данных на группы

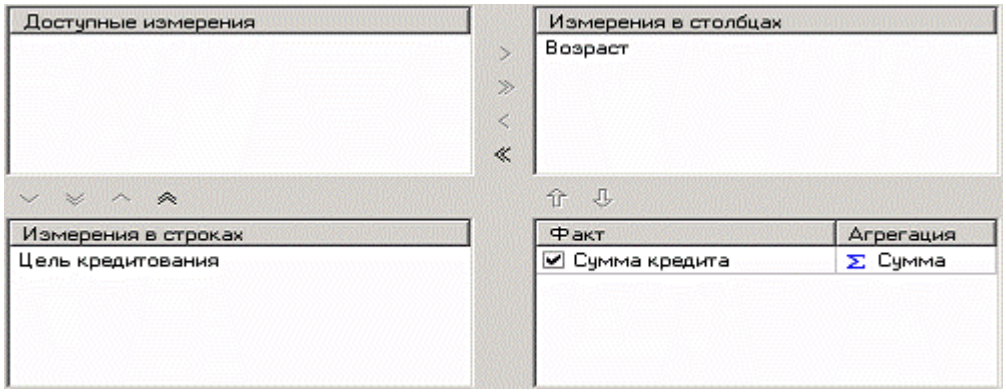
Часто для проведения анализа или построения модели прогноза приходится разбивать данные на группы, исходя из определенных критериев. В первом случае такая необходимость возникает, если аналитик желает просмотреть, к примеру, информацию не по всей совокупности данных, а по определенным группам (например, какую сумму кредита берут на те или иные цели, либо кредиторы того или иного возраста). Во втором случае (прогнозирование) аналитику необходимо учитывать тот факт, что определенные группы (в данном случае группы кредиторов) ведут себя по разному, и что модель прогноза, построенная на всех данных не будет учитывать нюансов, возникающих в этих группах. Т. е. лучше построить несколько моделей прогноза, например, в зависимости от суммовой группы кредита и строить прогноз на них, нежели построить одну модель прогноза. Исходя из этого и не только, в Deductor Studio предоставляется широкий набор инструментов, тем или иным способом позволяющие разбивать исходные данные на группы, группировать любым способом всевозможные показатели и т. п.

Рассмотрим разбиение данных на группы на примере данных по рискам кредитования физических лиц (Файл «Credit. txt»). Интересующие нас столбцы: «СУММА КРЕДИТА», «ДАТА КРЕДИТОВАНИЯ», «ЦЕЛЬ КРЕДИТОВАНИЯ» и «ВОЗРАСТ».

После импорта данных из текстового файла наиболее информативно просмотреть данные можно с помощью визуализатора «Куб», выбрав в качестве измерений столбцы «ВОЗРАСТ» и «ЦЕЛЬ КРЕДИТОВАНИЯ», а в качестве факта – столбец «СУММА КРЕДИТА». Остальные столбцы установить как непригодные.



На следующем шаге настройки куба следует указать измерение «ЦЕЛЬ КРЕДИТОВАНИЯ» как измерение в сроках, а измерение «ВОЗРАСТ» как измерение в столбцах, перетащив их с помощью мыши в соответствующие окна из области доступных измерений.



В итоге, на кросс диаграмме (одна из закладок визуализатора куб) можно просмотреть исходные данные.

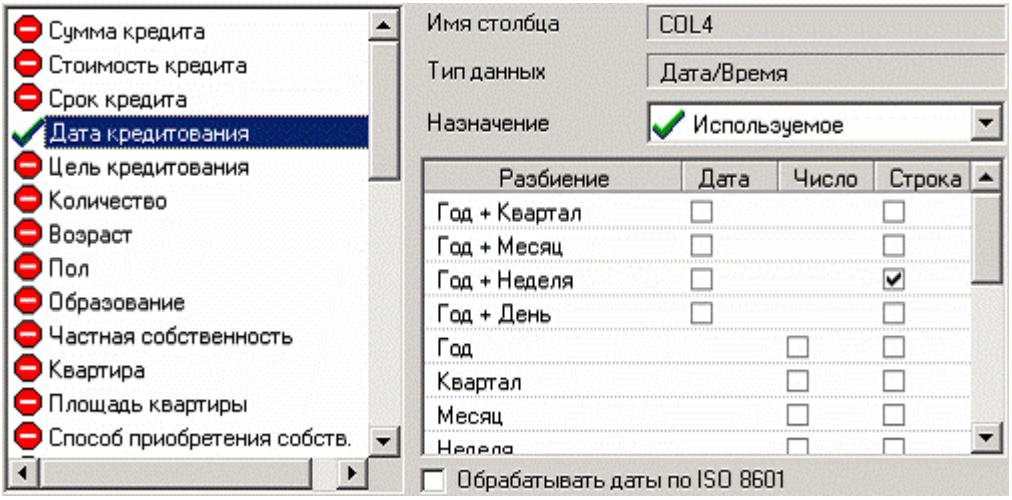
	Возраст		
Цель кредитования	19	20	21
Иное	50 000,00	17 000,00	8 500,00
Оплата за образование		17 500,00	29 500,00
Оплата услуг (мед., юрид. и т.п.)			
Покупка и ремонт недвижимости	78 000,00		13 000,00
Покупка товара	46 500,00	73 500,00	76 500,00
Турпоездки, развлечения и т.п.		30 500,00	
Итого	174 500,00	138 500,00	127 500,00

Разбиение даты (по неделям)

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

Пусть нам необходимо получить данные по суммам взятых кредитов по неделям (в файле «Credit. txt» содержится информация за первые две недели 2003 года).

Для этого в мастере обработки «Дата и Время» на втором шаге выберем поле «ДАТА КРЕДИТОВАНИЯ» используемым, в появившейся после этого таблице настроек выберем назначение



«Используемое» в столбце «Строка» напротив строки «Год + Неделя». Больше никакие настройки не понадобятся, поэтому перейдем далее к выбору типа визуализации. Выберем в качестве визуализаторов «Таблицу» и «Куб», поставив галочки в соответствующих позициях. В мастере настройки полей куба выберем в качестве измерения появившийся после обработки столбец «ДАТА КРЕДИТОВАНИЯ_YWStr (Год + Неделя)» и столбец «ЦЕЛЬ КРЕДИТОВАНИЯ», а в качестве факта – «СУММА КРЕДИТА». Остальные поля сделаем неиспользуемыми.

На следующем шаге перенесем одно измерение из области «доступных» в область «Измерения в строках», а другое – в область «Измерения в столбцах». Таким образом, на кросс диаграмме имеем суммы взятых кредитов по неделям (за первые две недели года) в разрезе целей кредитования.

	Дата кредитования (Год + Неделя)		
Цель кредитования	2003-W01	2003-W02	Итого
Иное	358 000,00	137 000,00	495 000,00
Оплата за образование	62 000,00	312 000,00	374 000,00
Оплата услуг (мед., юрид. и т.п.)	110 500,00	191 000,00	301 500,00
Покупка и ремонт недвижимости	404 000,00	538 000,00	942 000,00
Покупка товара	642 000,00	643 500,00	1 285 500,00
Турпоездки, развлечения и т.п.	35 500,00	113 500,00	149 000,00
Итого	1 612 000,00	1 935 000,00	3 547 000,00

В таблице с данными видно, что новое поле - «ДАТА КРЕДИТОВАНИЯ_YWStr (Год + Неделя)» содержит одинаковые значения (дата начала недели) для строк, которые попадают в одну и ту же неделю (дата начала недели или номер недели с начала года).

	Срок кредита	Дата кредитования	Дата кредитования	Цель кредитования
	24	05.01.2003	2003-W01	Покупка товара
	12	05.01.2003	2003-W01	Покупка товара
	30	05.01.2003	2003-W01	Иное
	36	06.01.2003	2003-W02	Покупка и ремонт недвижимости
	12	06.01.2003	2003-W02	Оплата за образование
	18	06.01.2003	2003-W02	Иное
	6	06.01.2003	2003-W02	Покупка товара

Квантование возраста кредиторов на 5 интервалов

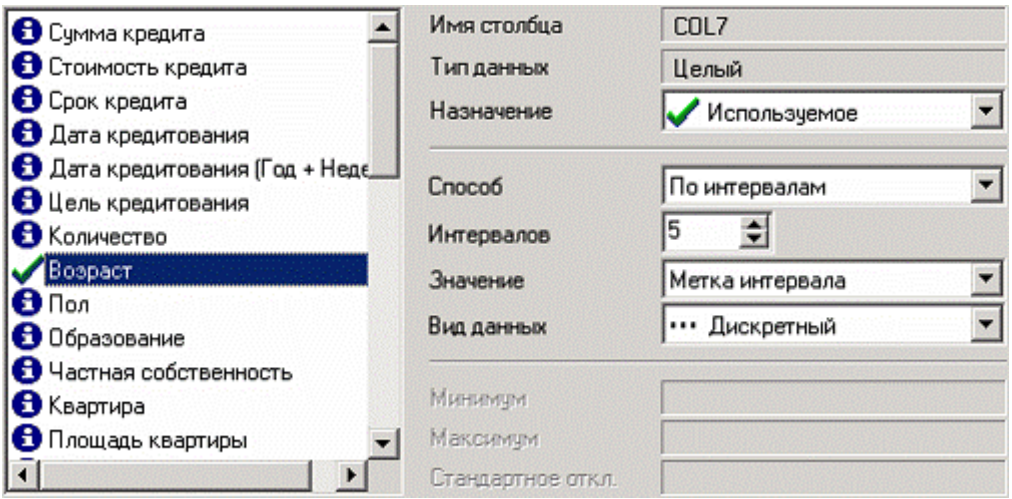
Часто аналитику необходимо отнести непрерывные данные (например, количество продаж) к какому-либо конечному набору (например, всю совокупность данных о количестве продаж необходимо разбить на 5 интервалов – от 0 до 100, от 100 до 200 и т. д., и отнести каждую запись исходного набора к какому – то конкретному интервалу) для анализа или фильтрации исходя именно из этих интервалов. Для этого в Deductor Studio применяется инструмент квантования (или дискретизации).

Квантование предназначено для преобразования непрерывных данных в дискретные. Преобразование может проходить как по интервалам (данные разбиваются на заданное количество интервалов одинаковой длины), так и по квантилям (данные разбиваются на интервалы разной длины так, чтобы в каждом интервале находилось одинаковое количество данных). В качестве значений результирующего набора данных могут выступать номер интервала, нижняя или верхняя граница интервала, середина интервала, либо метка интервала (значения определяемые аналитиком).

Примером использования данного инструмента может служить разбиение данных о возрасте кредиторов на 5 интервалов (до 30 лет, от 30 до 40, от 40 до 50, от 50 до 60, старше 60 лет).

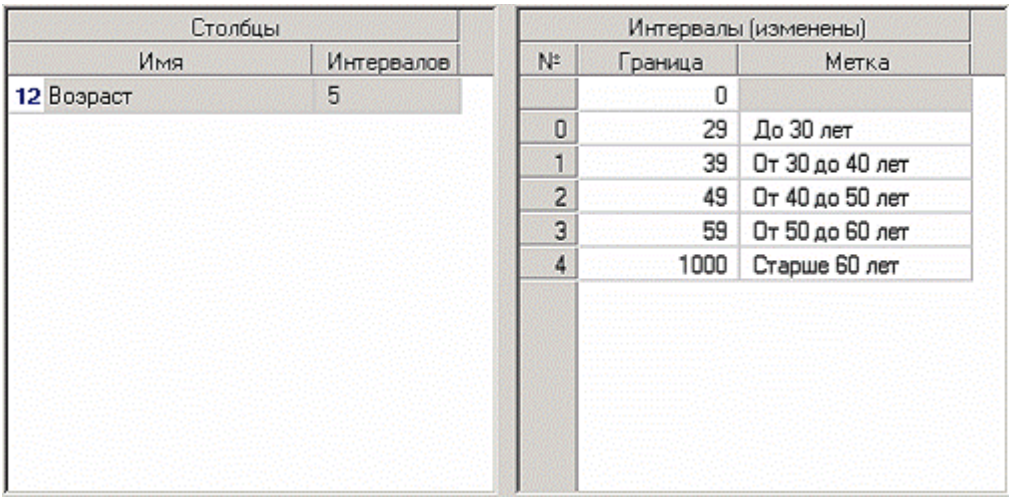
Исходные данные распределятся по пяти интервалам именно так, поскольку, согласно статистике, минимальное значение возраста кредитора 19, а максимальное 69 лет. Это необходимо аналитику для оценки кредиторской активности разных возрастных групп, с целью принятия решения о стимулировании кредиторов в группах с низкой активностью (например, уменьшение стоимости кредита для этих групп) и, быть может, увеличение прибыли в возрастных группах кредиторов с высоким риском (путем предложения дополнительных платных услуг). Причем аналитик желает видеть данные в разрезе по неделям (поэтому продолжим работу на последних полученных данных предыдущего примера).

Воспользуемся мастером квантования.



В нем выберем назначение поля «Возраст» используемым, укажем способ разбиения «По интервалам», зададим количество интервалов равное 5, в качестве значения выберем «Метку интервала».

На следующем шаге мастера определим сами метки соответственно возраста кредиторов: «до 30 лет», «от 30 до 40 лет» и т. д.



Столбцы	
Имя	Интервалов
12 Возраст	5

Интервалы (изменены)		
№	Граница	Метка
	0	
0	29	До 30 лет
1	39	От 30 до 40 лет
2	49	От 40 до 50 лет
3	59	От 50 до 60 лет
4	1000	Старше 60 лет

После обработки выберем в качестве способа отображения «Куб». В мастере укажем «СУММА КРЕДИТА» в качестве факта, «ВОЗРАСТ» и поле «ДАТА КРЕДИТОВАНИЯ (Год + Неделя)» в качестве измерения, остальные поля укажем неиспользуемыми.

Далее перенесем «ВОЗРАСТ» из доступных измерений в «Измерения в строках», а «ДАТА КРЕДИТОВАНИЯ (Год + Неделя)» в «Измерения в столбцах». На кросс диаграмме теперь видна информация о том, какие суммы кредитов берут кредиторы определенных возрастных групп в разрезе по неделям.

	Дата кредитования (Год + Неделя)		
Возраст	2003-W01	2003-W02	Итого
До 30 лет	721 500,00	795 000,00	1 516 500,00
От 30 до 40 лет	375 000,00	499 000,00	874 000,00
От 40 до 50 лет	195 000,00	362 500,00	557 500,00
От 50 до 60 лет	79 000,00	218 000,00	297 000,00
Старше 60 лет	241 500,00	60 500,00	302 000,00
Итого	1 612 000,00	1 935 000,00	3 547 000,00

Теперь аналитик, получив такие данные, может дать рекомендации о снижении стоимости кредита для лиц, старше 50 лет, либо о применении каких – нибудь других мер, способных привлечь большее количество кредиторов этих групп, либо мер, направленных на то, чтобы кредиторы брали кредит на большие суммы.

Фильтрация данных

Почти всегда исходный набор данных, или набор данных после обработки аналитику необходимо отфильтровать. Фильтрация бывает необходима для разбиения данных на какие либо группы (например, товарные группы) для последующей обработки или анализа данных уже отдельно по каждой группе. Также некоторые данные могу не подходить, или наоборот, подходить для дальнейшего анализа в силу накладываемых условий (например, если на каком – либо этапе обработки данных были выявлены противоречивые записи, то их необходимо исключить из последующей обработки). Здесь тоже возникает необходимость фильтрации.

Фильтрация позволяет из базового набора данных получить набор данных, удовлетворяющий определенным аналитиком условиям. В Deductor Studio механизм построения условий фильтрации прост для понимания. В окне мастера можно определить несколько элементарных условий фильтрации (<ПОЛЕ> <ОТНОШЕНИЕ> <ЗНАЧЕНИЕ>), последовательно связанных логическими операциями (И, ИЛИ).

Рассмотрим ситуацию, когда аналитику необходимо спрогнозировать кредитоспособность потенциального кредитора. Предполагается, что кредиторы, берущие суммы разного диапазона ведут себя по-разному, следовательно, модели прогноза должны свои для каждой группы. Т. е. для дальнейшего построения моделей прогноза кредитоспособности определенных аналитиком категорий необходимо использовать фильтрацию.

Определим, для примера группу кредиторов, взявших кредит менее 10000 руб. Воспользуемся данными предыдущего примера. Для этого, находясь на узле импорта данных из текстового файла, запустим мастер обработки. В нем в качестве метода обработки выберем фильтрацию. На втором шаге мастера можно видеть одно неопределенное условие фильтрации (при необходимости их можно добавлять или удалять соответствующими кнопками на форме). Поскольку необходимо отфильтровать данные только по кредиторам, взявших кредит менее 10000, то в графе «Имя поля» выбираем поле «СУММА КРЕДИТА», в графе «Условие» выбираем знак меньше, в графе «Значение» пишем «10000».

Операция	Имя поля	Условие	Значение
	12 Сумма кредита	<	10000

Больше никаких условий не требуется, поэтому переходим на следующий шаг мастера и запускаем процесс фильтрации. После выполнения обработки можно манипулировать уже только с данными по кредиторам выбранного кредитного диапазона. В правильности выполненной операции можно легко убедиться, выбрав в качестве визуализации данных статистику и просмотрев значения минимального и максимального значения поля «СУММА КРЕДИТА».

	Метка столбца	Минимум	Максимум
1	12 Сумма кредита	2000	9500
2	12 Стоимость кред...	400	1900
3	12 Срок кредита	6	6
4	Дата кредитова...	01.01.2003	11.01.2003

Калькулятор

Иногда возникает необходимость на каком-либо этапе обработки данных получить на их основе новые (производные) данные. Возможно, аналитику требуется вычислить процентное отклонение значения одного поля относительно другого, либо подсчитать сумму, разность полей, получить на основе данных показатель и уже его использовать для дальнейшей обработки, в зависимости от значения полей вычислить те или иные выражения. В Deductor Studio такую возможность предоставляет инструмент «Калькулятор». Он позволяет создавать новые поля, вычисляющие заданные аналитиком выражения. Т. е. калькулятор служит для получения производных данных на основе имеющихся в исходном наборе. Мастер предоставляет широкий набор функций различного направления. В мастере представлен список новых выражений, где добавляются необходимые аналитику выражения, список доступных функций с кратким описанием каждой, список доступных операций и также список доступных столбцов, которые можно задействовать при создании выражения.

Замечание: Реализованный в Deductor Studio конструктор выражений при построении использует не метки (Сумма, Количество, Цена...), а имена полей таблицы, заданные в источнике данных (Summ, Count, Price…). При импорте в некоторых случаях (напр. из текстового файла) можно задать как метки, так и имена импортируемых полей. В следующем примере метками являются «АРГУМЕНТ1», «АРГУМЕНТ2», «АРГУМЕНТ3», а именами соответственно «COL1», «COL2», «COL3». При желании как метки, так и имена полей можно изменить на более информативные, используя обработчик «Настройка набора данных».

Исходные данные

Рассмотрим применение на примере данных из файла «Calculate. txt». В нем содержится таблица с полями «АРГУМЕНТ1», «АРГУМЕНТ2», «АРГУМЕНТ3» – набор аргументов.

Для начала необходимо импортировать данный файл в программу. Для просмотра исходных данных в данном случае удобнее использовать визуализатор «Таблица».

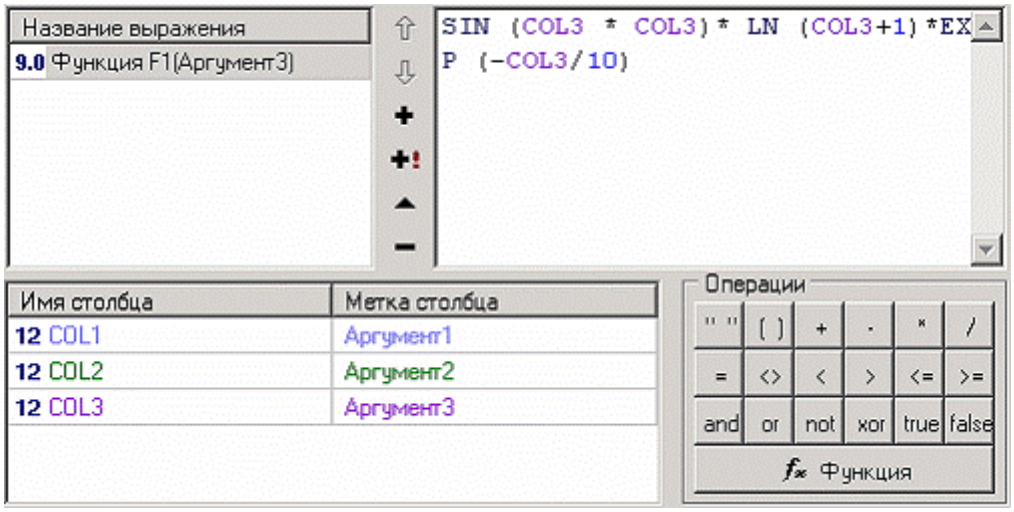
	Аргумент1	Аргумент2	Аргумент3
	0	4	4
	0	5	5
	0	6	6
	0	7	7
	0	8	8

Пусть необходимо на основе аргументов рассчитать некоторые математические функции. Пусть это будут две функции одного аргумента (АРГУМЕНТ3), одна функция от двух аргументов, одна кусочно-заданная функция и функция, показывающая относительное отклонение (АРГУМЕНТ1 + 1 от АРГУМЕНТ2 + 1). Предполагается, что все эти функции будут использоваться для последующей обработки.

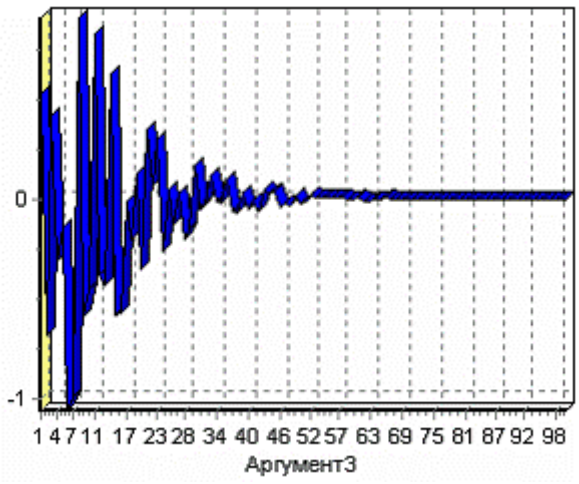
Функции F1(АРГУМЕНТ3), F2(АРГУМЕНТ3)

Рассчитаем значение функций SIN (АРГУМЕНТ 3 * АРГУМЕНТ 3)* LN (АРГУМЕНТ 3+1)*EXP (- АРГУМЕНТ 3/10) и 10*SIN(АРГУМЕНТ 3 * АРГУМЕНТ 3/100)/ (АРГУМЕНТ 3+1)*EXP(- АРГУМЕНТ 3/10).

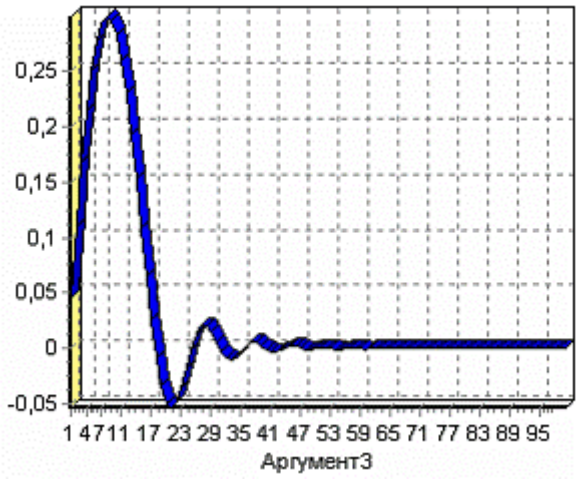
Для этого, находясь на узле импорта, запустим мастер обработки. Выберем в качестве обработчика калькулятор. На втором шаге мастера в списке выражений в первой строке в графе «Название выражения» вместо надписи «Выражение» напишем F1(АРГУМЕНТ3). В поле редактора выражения (в верхней части мастера) напишем «SIN (COL3 * COL3)* LN (COL3+1)*EXP (-COL3/10)».



Таким образом, мы создали новый столбец, задали ему название «F1(АРГУМЕНТ3)» и также определили, какие значения будут принимать записи этого поля. На этом создание вычисляемого значения окончено, поэтому переходим на следующий шаг мастера, где предлагается выбрать способ отображения данных. Самым информативным в данном случае является диаграмма, которую и следует выбрать. Далее, выбрав в мастере настроек диаграммы в качестве отображаемого поля «F1(АРГУМЕНТ3)», в качестве типа графика «Линии», в качестве подписей по оси X значения поля «АРГУМЕНТ3» можно увидеть график вычисленной функции.



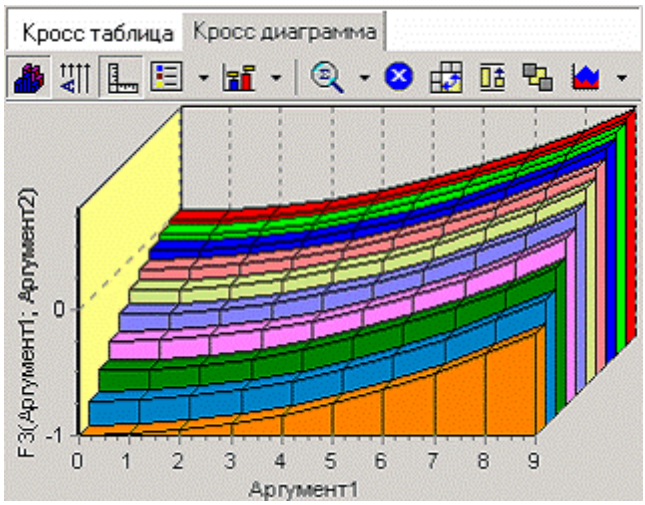
Сложная функция F2(АРГУМЕНТ3) отличается только видом функции («10*SIN(COL3 * COL3/100)/(COL3+1)*EXP(-COL3/10)»).



Функция от двух аргументов F3(АРГУМЕНТ1; АРГУМЕНТ2)

Данная функция интересна тем, что для ее просмотра в трех измерениях можно использовать визуализатор «Куб». Зададим название выражения «F3 (АРГУМЕНТ1; АРГУМЕНТ2)», в поле вычисляемого выражения напишем «COL1*COL1/100 – COL2*COL2/100». Выберем визуализатор «Куб» и настроим его так, что «АРГУМЕНТ1 и «АРГУМЕНТ2» являлись бы измерениями, F3 (АРГУМЕНТ1; АРГУМЕНТ2) – фактом, а «АРГУМЕНТ3» – неиспользуемым.

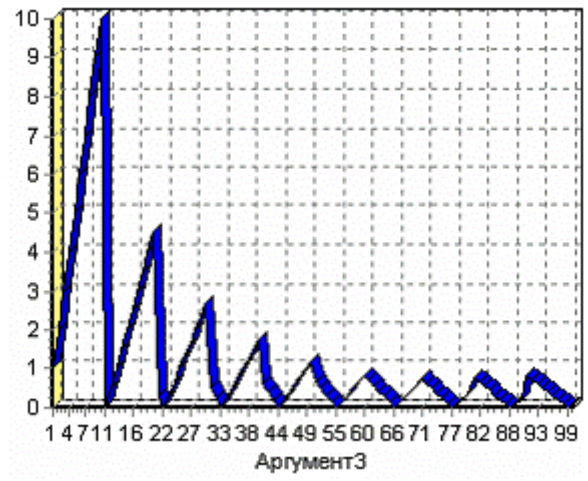
Выбрав «АРГУМЕНТ1» измерением в столбцах, а «АРГУМЕНТ2» – измерениям в строках перейдем к просмотру Кросс-диаграммы. Для более наглядного просмотра установим тип диаграммы «области». Теперь можно посмотреть вычисленную функцию в объемном виде.



Вычисление отклонения АРГУМЕНТ1+1 от АРГУМЕНТ2+1

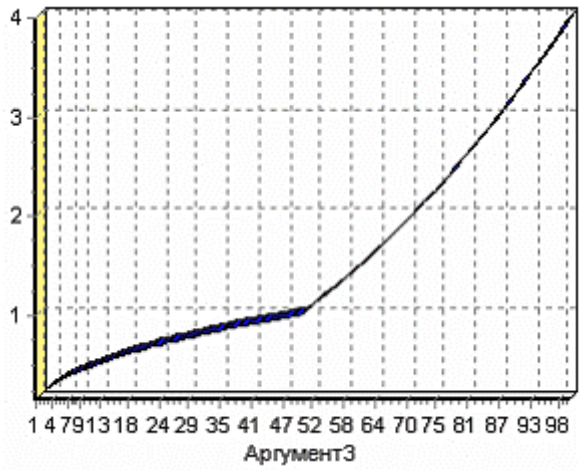
Покажем пример применения одной из встроенных функций – вычисление долевого отклонения одного аргумента от другого (RELDEV). Список всех встроенных функций вместе с описанием можно посмотреть в мастере нажав на кнопку «Функция».

Задав в качестве вычисляемого выражения RELDEV(COL1 + 1; COL2 + 1) можно на диаграмме увидеть данное отклонение.



Пример кусочно-заданной функции.

Пусть функция принимает значения SQRT(АРГУМЕНТ3/50) (квадратный корень) при значениях АРГУМЕНТ3 от 0 до 50 и значения АРГУМЕНТ3*АРГУМЕНТ3/2500 при остальных. Для вычисления подобной функции необходимо воспользоваться имеющейся в наличии функцией IFF(арумент1; аргумент2; аргумент3), которая позволяет в зависимости от логического значения первого аргумента получить второй или третий аргумент. Согласно примеру, если значение аргумента больше нуля и меньше 50 необходимо получить выражение SQRT(АРГУМЕНТ3/50), в противном случае – выражение АРГУМЕНТ3*АРГУМЕНТ3/250. Таким образом, в поле построения выражения необходимо написать «IFF((COL3 > 0) AND (COL3 < 50); SQRT(COL3/50); COL3*COL3/2500)». Сделав это в мастере обработки «Калькулятор», и выбрав далее визуализатор «Диаграмма», и также выбрав в мастере настройки диаграммы поле со значениями кусочно-заданной функции, можно посмотреть на требуемый результат.



Группировка данных

Сложно делать выводы на основе необработанной первичной информации. Аналитику для принятия решения почти всегда нужна сводная информация. Совокупные данные намного более информативны, тем более, если их можно получить в различных разрезах. В Deductor Studio предусмотрен инструмент, реализующий сбор сводной информации – «Группировка». Группировка позволяет объединять записи по полям - измерениям и агрегируя данные в полях-фактах для дальнейшего анализа.

Исходные данные

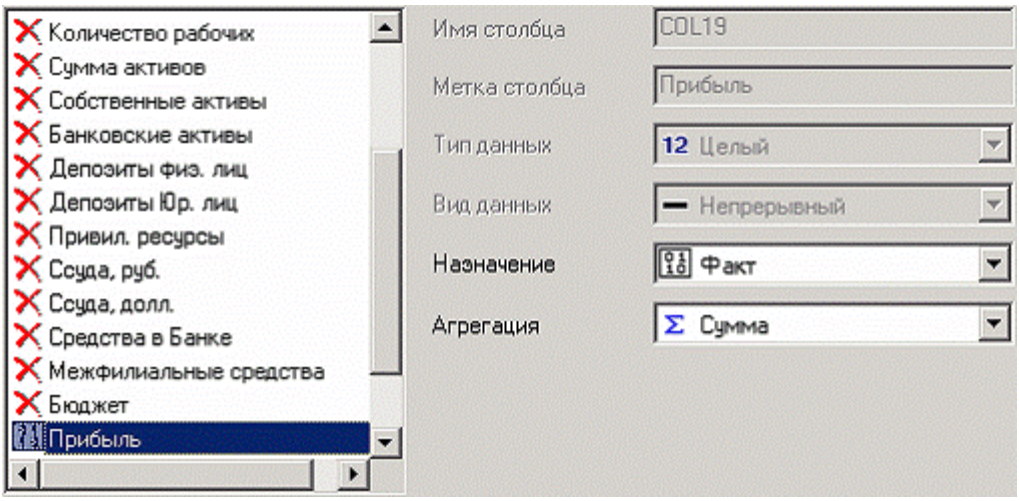
Допустим, что у аналитика имеется статистика по банкам России за определенный период. Она находится в файле «banks. txt». Перед ним стоит задача выявления ряда городов, в которых прибыль банков самая большая для использования этих данных в дальнейшем. Для этого аналитик должен обратить внимание на следующие поля таблицы из файла: «БАНК», «ФИЛИАЛЫ», «ГОРОД», «ПРИБЫЛЬ». Т. е. информация о названии банка, городе, в котором он находится (филиалы банка могут находиться в разных городах – следовательно, по одному и тому же банку может быть несколько записей с данными по разным городам) и прибыль банка.

Ясно, что для решения поставленной задачи первым делом необходимо найти суммарную прибыль всех банков в каждом городе. Для этого и необходима группировка.

Для начала следует импортировать данные по банкам из текстового файла. Просмотреть исходную информацию можно в виде куба, где по строкам будут названия банков, а по столбцам – города. С помощью визуализатора «Куб» также можно получить требуемую информацию, выбрав в качестве измерения поле «ГОРОД», а в качестве факта «ПРИБЫЛЬ». Но нам необходимо получить эти данные для последующей обработки, следовательно, необходимо сделать аналогичную группировку.

Группировка по городам

Находясь в узле импорта, запустим мастер обработки. Выберем в качестве обработки группировку данных. На втором шаге мастера установим назначение поля «ГОРОД» как измерение, а назначение поля «ПРИБЫЛЬ» как факт. В качестве функции агрегации у поля «ПРИБЫЛЬ» следует указать Сумму.



Таким образом, после обработки получим суммарные данные по прибыли всех банков по каждому городу. Их можно просмотреть, используя таблицу. Теперь аналитику можно выполнять следующий этап обработки данных.

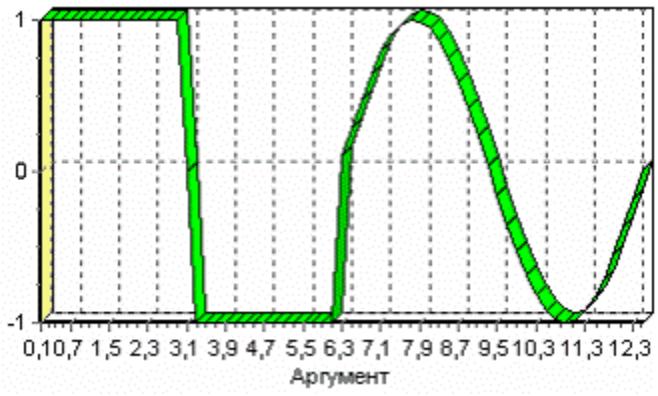
Город	Прибыль
Санкт-Петербург	128 038
Владивосток	17 152
Вологда	35 144
Екатеринбург	125 126
Казань	68 576
Краснодар	26 991

Преобразование данных к скользящему окну
Когда требуется прогнозировать временной ряд, тем более, если налицо его периодичность (сезонность), то лучшего результата можно добиться, учитывая значения факторов не только в данный момент времени, но и, например, за аналогичный период прошлого года. Такую возможность можно получить после трансформации данных к скользящему окну. То есть, например, при сезонности продаж с периодом 12 месяцев, для прогнозирования количества продаж на месяц вперед можно в качестве входного фактора указать не только значение количества продаж за предыдущий месяц, но и за 12 месяцев назад.

Обработка создает новые столбцы путем сдвига данных исходного столбца вниз и вверх (глубина погружения, горизонт прогноза).

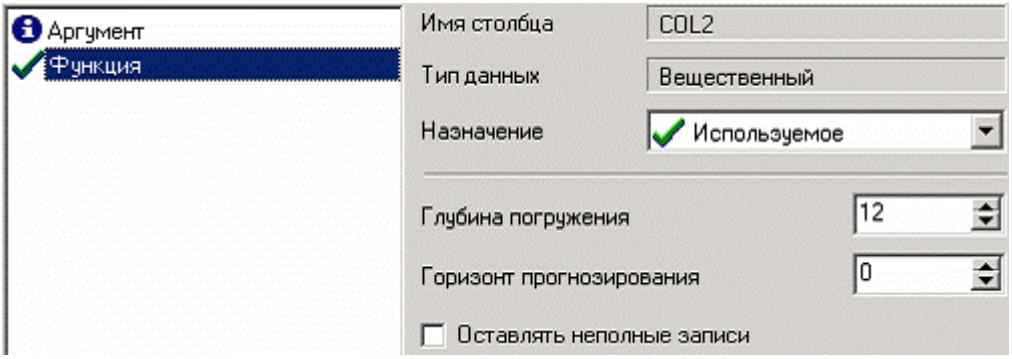
Исходные данные

Продемонстрируем сам принцип трансформации данных, используя данные из файла «Sliding. txt». В нем всего 2 поля – «АРГУМЕНТ» - аргумент (время), «ФУНКЦИЯ» – временной ряд. Импортируем данные из файла (необходимо указать тип полей – вещественный) и построим диаграмму.

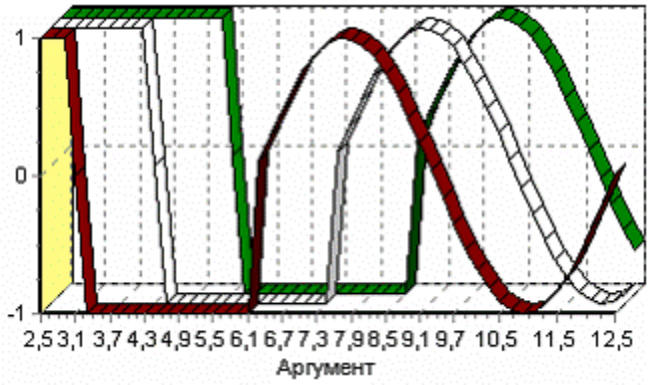


Преобразование скользящим окном

В мастере преобразования укажем назначение столбца «ФУНКЦИЯ» используемым, установим для него глубину погружения 12.



После трансформации были получены новые столбцы – «ФУНКЦИЯ - 12», ... «ФУНКЦИЯ - 2», «ФУНКЦИЯ - 1» на основе столбца «ФУКЦИЯ». Если на диаграмме посмотреть несколько таких столбцов, то видно, что данные в них сдвинуты относительно друг друга.



Настройка набора данных

Настройка набора данных применяется, когда необходимо изменить имя, метку, размер, тип, вид и назначение полей текущей таблицы данных для более удобного дальнейшего использования.

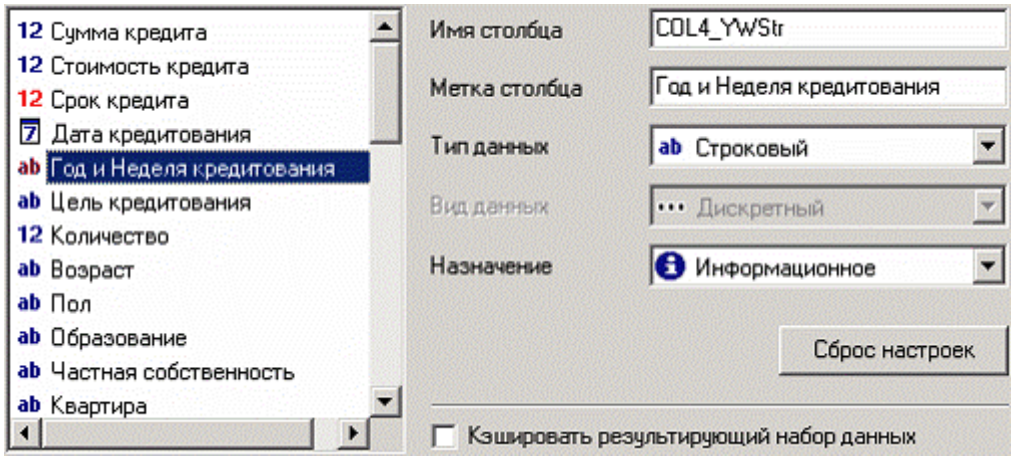
Замечание: Данный обработчик аналогичен шагу мастера настройки полей при импорте данных в программу, рассмотренному выше.

Исходные данные

Продемонстрируем использование настройки полей, используя данные, полученные после квантования возраста кредиторов на интервалы из примера выше. Пусть необходимо изменить метку поля «Дата кредитования (Год + Неделя)» на более информативную метку при подготовке отчетности - «Год и неделя кредитования». Пусть также, для дальнейшего использования необходимо установить размер поля «Цель кредитования» 30 символов и необходимо использовать поле «Срок кредита» как дискретное.

Выполнение настройки

В мастере настройки выделим столбец «Дата кредитования (Год + Неделя)» и укажем ему новую метку. Подобные действия по изменению произведем и с другими полями.



После настройки полей, полученный отчет, представленный в виде кросс-таблицы, будет выглядеть следующим образом:

		Давать кредит		
Год и Неделя кредитования	Срок кредита	Да	Нет	Итого
2003-W01	6	194 500,00	7 500,00	202 000,00
	12	244 500,00	176 500,00	421 000,00
	18	48 000,00	249 000,00	297 000,00
	24		195 500,00	195 500,00
	30		151 000,00	151 000,00
	36		229 000,00	229 000,00
	42		58 500,00	58 500,00
	48		58 000,00	58 000,00
	Итого	487 000,00	1 125 000,00	1 612 000,00
2003-W02	6	183 500,00	25 500,00	209 000,00
	12	109 000,00	373 500,00	482 500,00

Слияние

Обработчик "Слияние" предназначен для объединения двух таблиц по нескольким одинаковым полям. Обработчик применяется, например, для добавления в таблицу с данными о продажах данных по остаткам за те же месяца. Различают две таблицы: исходная и присоединяемая. К исходной таблице добавляются новые поля, значения которых берутся из присоединяемой таблицы. Количество строк исходной таблицы остается неизменным.

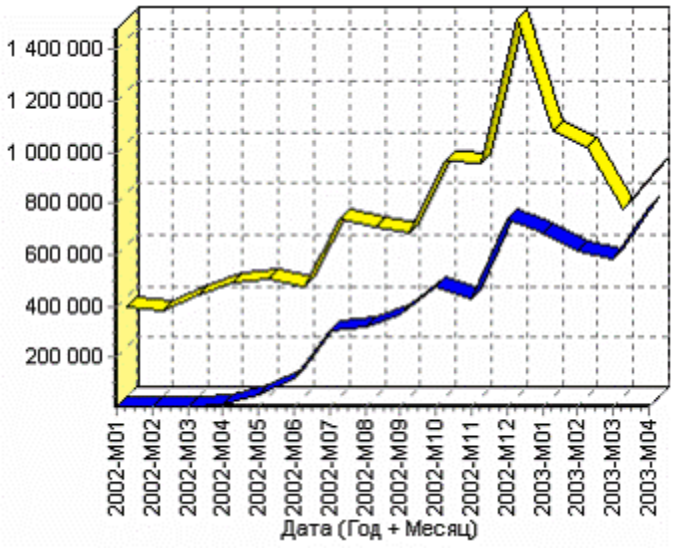
Исходные данные

Продемонстрируем использование слияния, используя данные по продажам и остаткам (файлы «TradeSales. txt» и «TradeRest. txt» соответственно). Добавим к данным по продажам данные по остаткам. Для этого сначала импортируем данные из файла, содержащего данные по продажам, а затем запустим мастер обработки и выберем обработчик «Слияние».

Выполнение слияния
В мастере слияния сначала необходимо выбрать источник данных для слияния. Данные шаги аналогичны шагам мастера импорта данных. Так что импортируем данные из текстового файла с остатками «TradeRest. txt». Далее необходимо установить связь между наборами данных, а именно, указать соответствие импортируемого поля имеющемуся полю (измерение - для связи двух таблиц) и указать, какие новые поля добавить при слиянии (факт с указанием способа агрегации).

Импортируемое поле	Назначение	Поле/Агрегировать
ab Дата (Год + Месяц)	Измерение	Дата (Год + Месяц)
9.0 Остаток (количество)	Факт	Σ Сумма

После указания параметров полей, как показано на рисунке выше, необходимо перейти на следующий шаг мастера и запустить процесс слияния. Полученные результаты, представленные в виде диаграммы будут выглядеть следующим образом:



Как видно, при помощи слияния удалось объединить объем продаж с объемом остатков.

Выявление дубликатов и противоречий

Бывают ситуации, когда проблема неочищенных данных не позволяет построить хорошую модель прогнозирования вообще. Такое происходит, если в наборе данных для прогноза содержатся строки с одинаковыми входными факторами, но разными выходными. В такой ситуации непонятно, какое результирующее значение верное – налицо противоречие. Если противоречивые использовать для построения модели прогноза, то модель окажется неадекватной. Поэтому противоречивые данные, чаще всего, лучше вообще исключить из исходной выборки. Также в данных могут встречаться записи с одинаковыми входными факторами и одинаковыми выходными, т. е. дубликаты. Таким образом, данные несут избыточность. Присутствие дубликатов в анализируемых данных можно рассматривать как способ повышения «значимости» дублирующейся информации. Иногда они даже необходимы, например, если при построении модели нужно особо выделить некоторые наборы значений. Но все равно, включение в выборку дублирующей информации должно происходить осознанно: в большинстве случаев дубликаты в данных являются следствием ошибок при подготовке данных.

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

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

Исходные данные

Рассмотрим механизм выявления дубликатов и противоречий на примере данных файла «MultTable. txt». В нем находится таблица умножения двух целых аргументов в диапазоне от 1 до 10. Таблица имеет четыре поля: «АРГУМЕНТ1», «АРГУМЕНТ2» – аргументы, «ПРОИЗВЕДЕНИЕ», «ПРОИЗВЕДЕНИЕ С ПРОТИВОРЕЧИЯМИ» – произведение аргументов, содержащее противоречия. Данные подготовлены следующим образом: сначала идет 100 строк таблицы умножения (от 1*1 до 10*10), причем в поле «ПРОИЗВЕДЕНИЕ С ПРОТИВОРЕЧИЯМИ» в некоторых строках содержатся неверный результат умножения (например, «АРГУМЕНТ1» = 1, «АРГУМЕНТ2»=5, «ПРОИЗВЕДЕНИЕ» = 5, «ПРОИЗВЕДЕНИЕ С ПРОТИВОРЕЧИЯМИ» = 10). Следующие 50 строк дублируют первые 50, причем значения поля «ПРОИЗВЕДЕНИЕ С ПРОТИВОРЕЧИЯМИ» содержат верный результат умножения. Таким образом, данные содержат ряд строк с одинаковыми входными значениями, но разными выходными и строки с одинаковыми входными и выходными значениями. Т. е. присутствуют дубликаты и противоречия. Остается только обнаружить их.

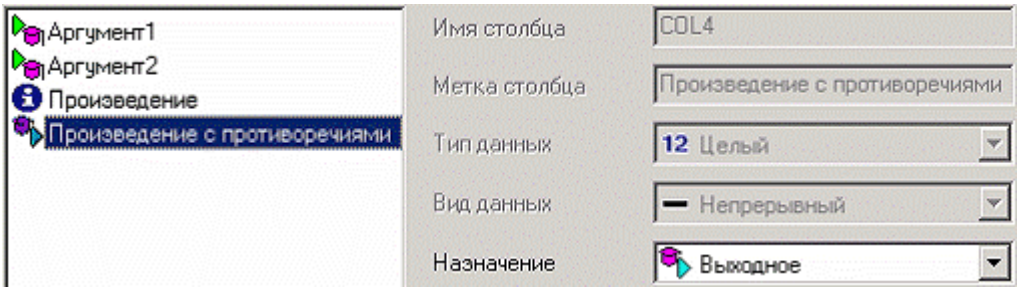
Импортируем данные из текстового файла и посмотрим их в виде таблицы.

	Аргумент1	Аргумент2	Произведение	Произведение с противоречиями
▶	1	1	1	1
	1	2	2	2
	1	3	3	3
	1	4	4	4
	1	5	5	10
	1	6	6	6

Поиск дубликатов и противоречий

Для выявления дубликатов и противоречий запустим мастер обработки. В нем выберем тип обработки «Дубликаты и противоречия».

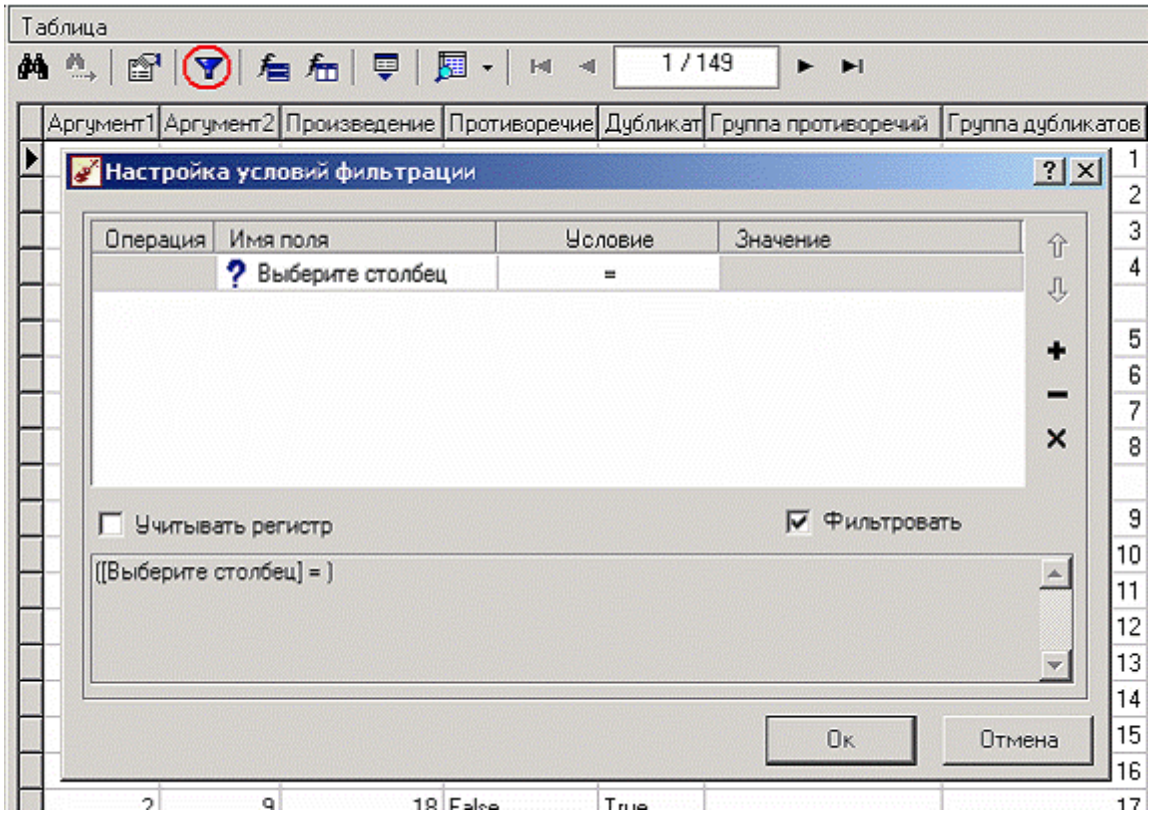
На втором шаге мастера необходимо настроить назначение полей. В данном случае входными полями являются «АРГУМЕНТ1» и «АРГУМЕНТ2», а выходным «ПРОИЗВЕДЕНИЕ С ПРОТИВОРЕЧИЯМИ».



На следующем шаге необходимо запустить процесс обработки. После завершения выявления дубликатов и противоречий просмотрим результат в виде таблицы.

Аргумент1	Аргумент2	Произведение	Противоречие	Дубликат	Группа противоречий	Группа дубликатов
1	1	1	False	True		1
1	2	2	False	True		2
1	3	3	False	True		3
1	4	4	False	True		4
1	5	5	True	False	1	
1	6	6	False	True		5
1	7	7	False	True		6
1	8	8	False	True		7
1	9	9	False	True		8
1	10	10	True	False	2	
2	1	2	False	True		9

В четырех новых столбцах как раз и находится интересующая нас информация: какие записи являются дубликатами, какие – противоречиями, к какой группе дубликатов или противоречий относятся. Аналитик может также отфильтровать данные в таблице для просмотра только дубликатов или только противоречий. Покажем, как это можно сделать. Нажав на кнопку фильтрации таблицы, появится мастер настроек условий фильтра (аналогичный обработчику «Фильтрация», рассмотренного ранее).



Для просмотра только дубликатов необходимо задать условие «Дубликат истина»

После ввода условия необходимо нажать на кнопку «Ок» и в таблице будут только дубликаты.

	Аргумент1	Аргумент2	Произведение	Противоречие	Дубликат	Группа противоречий	Группа дубликатов
▶	1	1	1	False	True		1
	1	2	2	False	True		2
	1	3	3	False	True		3
	1	4	4	False	True		4
	1	6	6	False	True		5

Аналогично отфильтруем только противоречия.

	Аргумент1	Аргумент2	Произведение	Противоречие	Дубликат	Группа противоречий	Группа дубликатов
▶	1	5	5	True	False	1	
	1	10	10	True	False	2	
	1	5	5	True	False	1	
	1	10	10	True	False	2	

Примеры анализа данных
Основное направление программы Deductor Studo – анализ, прогнозирование, классификация и кластеризация данных. Предыдущие примеры в основном касались только подготовки данных для последующего анализа. Программа предоставляет следующие механизмы анализа: нейронные сети, линейный регрессионный анализ, построение деревьев решений, самоорганизующиеся карты Кохонена, прогнозирование временного ряда, обнаружение дубликатов и противоречий.

Рассмотрим принцип работы каждого из этих механизмов на последующих примерах.

Прогнозирование умножения с помощью нейронных сетей

Нейросети – механизм, который используют для прогнозирования и решения задач классификации. Они применяются в основном там, где существует нелинейные зависимости результата от входных факторов.

Исходные данные

Рассмотрим прогнозирование с помощью нейронных сетей на примере прогнозирования результата умножения двух чисел – файл «multi. txt»

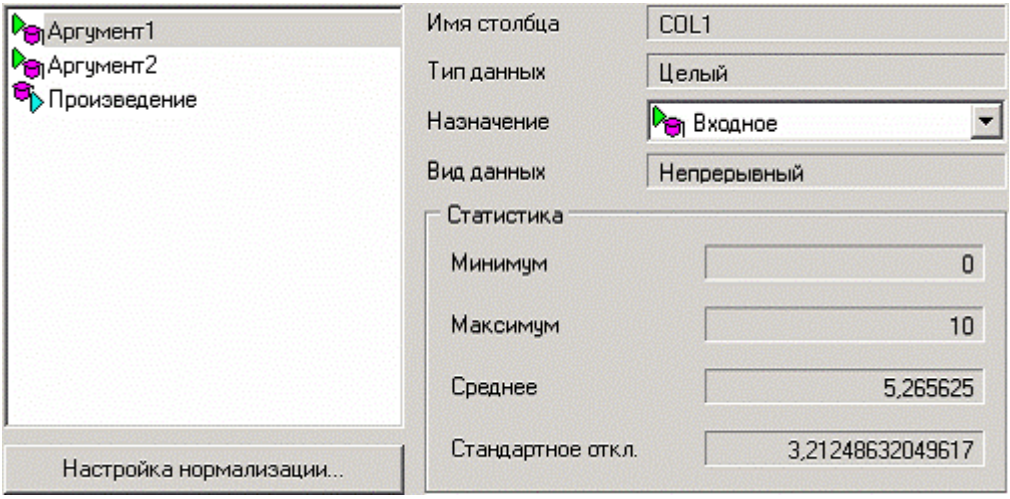
В нем содержится таблица со следующими полями: «АРГУМЕНТ1», «АРГУМЕНТ2» – множители, «ПРОИЗВЕДЕНИЕ» – их произведение.

Импортировав данные из файла, можно посмотреть результат умножения, используя таблицу.

	Аргумент1	Аргумент2	Произведение
▶	1	0	0
	0	1	0
	3	0	0
	0	3	0
	5	0	0

Прогнозирование результата умножения

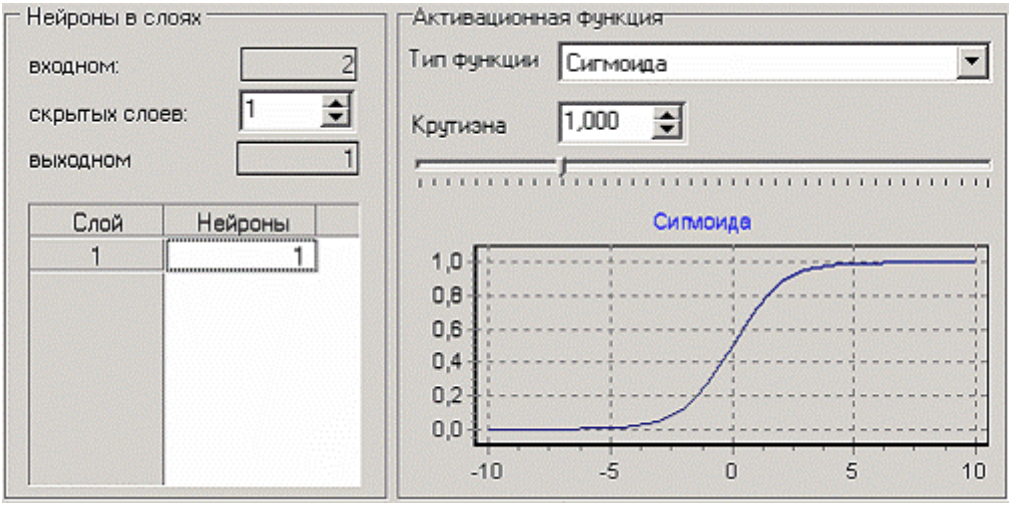
Пусть необходимо построить модель прогноза умножения, подавая на вход которой два множителя получать на выходе их произведение. Для этого необходимо, находясь на узле импорта, открыть мастер обработки. В нем выбрать в качестве обработки нейронную сеть и перейти к следующему шагу мастера. На втором шаге мастера необходимо установить назначение полей «АРГУМЕНТ1» и «АРГУМЕНТ2» как входные, а поле «ПРОИЗВЕДЕНИЕ» – как выходное.



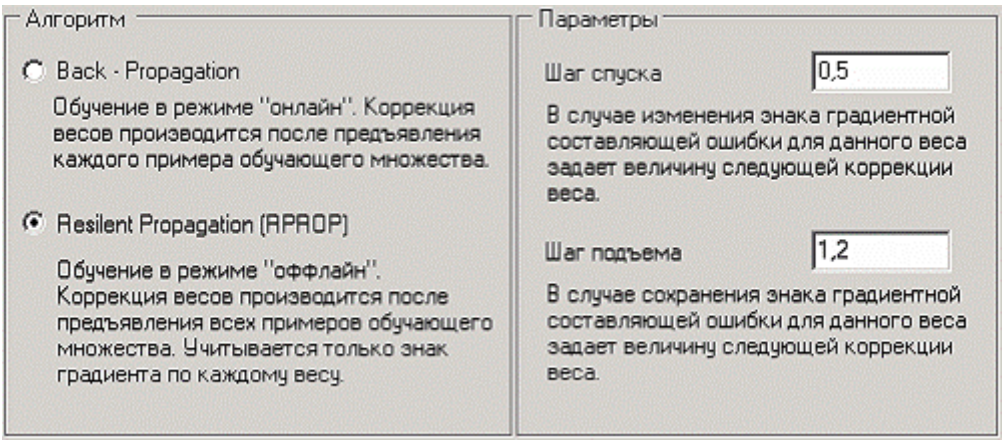
На следующем шаге предлагается настроить разбиение исходного множества данных на обучающее тестовое и валидационное. Здесь необходимо только указать способ разбиения исходного множества данных «Случайно».

Множество	Размер		Порядок сортировки
	В процентах	В строках	
☑ Обучающее	95,00	61	По возрастанию
☑ Тестовое	5,00	3	По возрастанию
☐ Валидационное	0,00	0	По возрастанию
ИТОГО:	100,00	64	

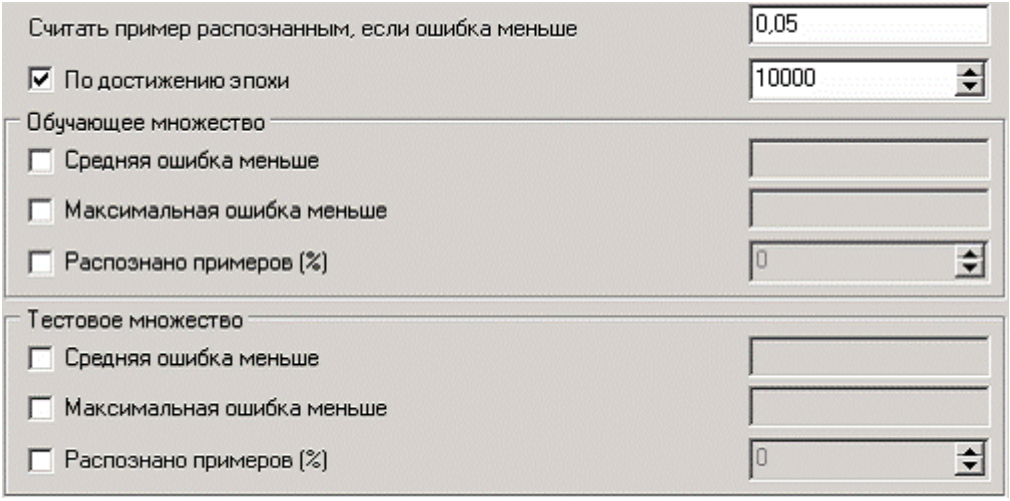
На следующем шаге необходимо указать количество нейронов в скрытом слое – 1, остальное можно оставить по умолчанию.



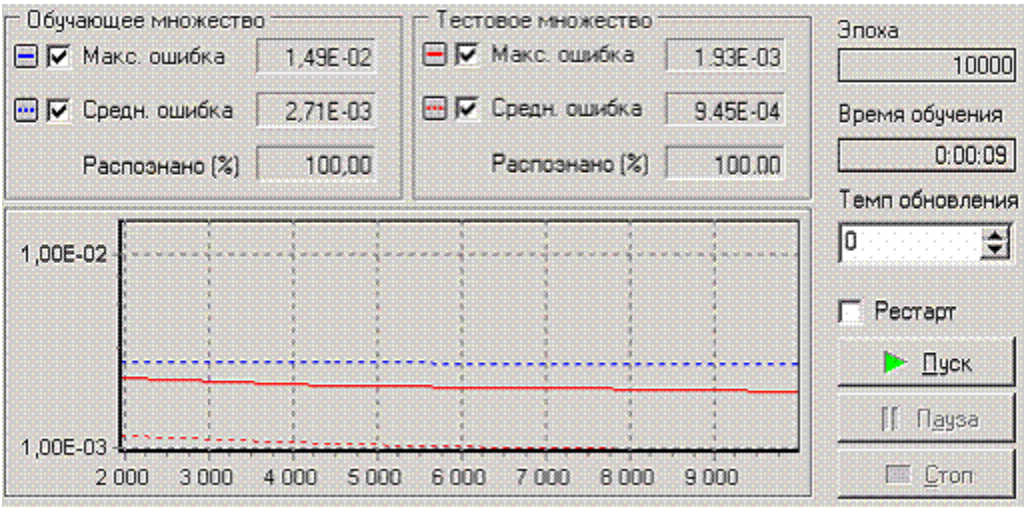
Следующий шаг предлагает выбрать алгоритм обучения и его параметры. Здесь тоже ничего менять не нужно.



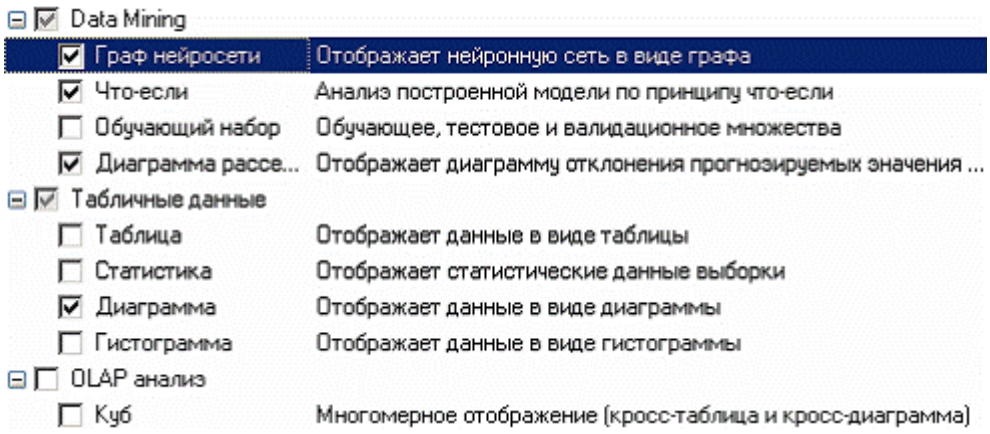
Следующий шаг предлагает настроить условия остановки обучения. Укажем, что следует считать пример распознанным, если ошибка меньше 0.005, и также укажем условие остановки обучения при достижении эпохи 10000.



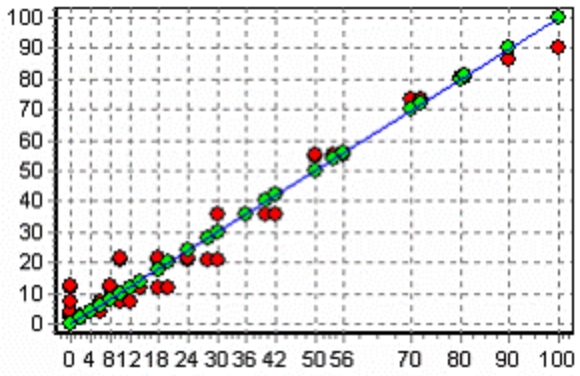
Следующий шаг мастера предлагает запустить процесс обучения и наблюдать в процессе обучения величину ошибки, а также процент распознанных примеров. Параметр «Частота обновления» отвечает за то, через какое количество эпох обучения выводится данная информация.



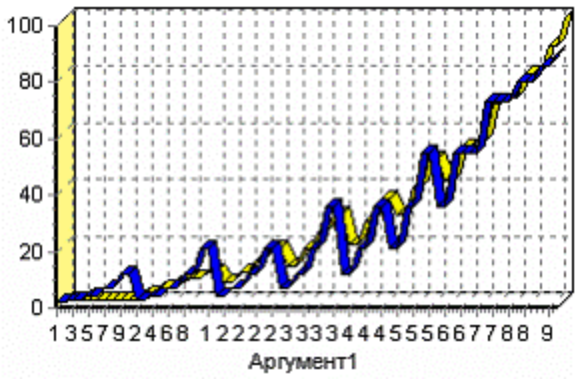
После обучения сети, в качестве визуализаторов выберем Диаграмму, Диаграмму рассеяния, Граф нейросети, Что-если.



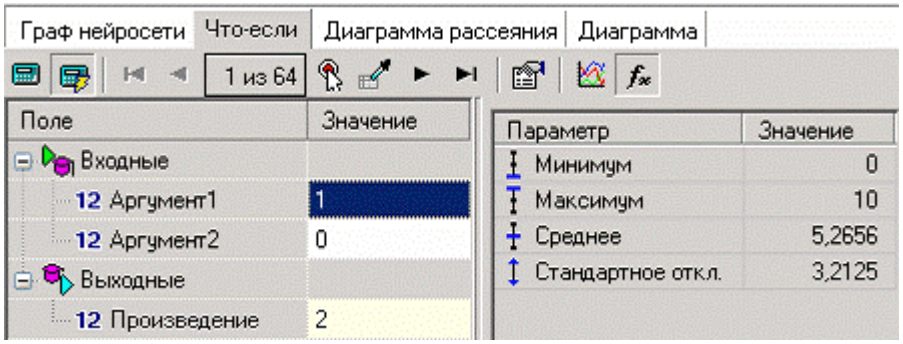
Результаты наглядно видны на диаграмме рассеяния, которая показывает рассеяние прогнозируемых данных относительно эталонных.



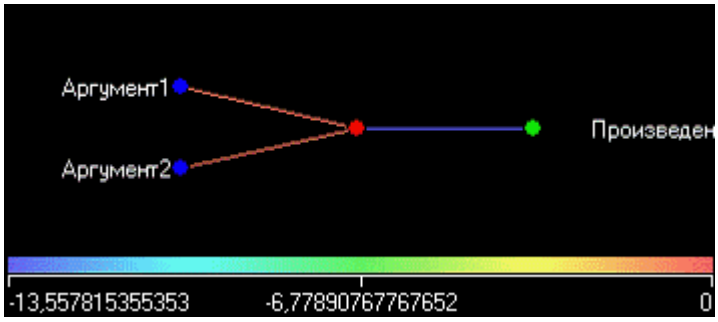
Также можно сравнить эталонные данные с прогнозируемыми, выбрав на обычной диаграмме два поля – «ПРОИЗВЕДЕНИЕ» и «ПРОИЗВЕДЕНИЕ_OUT».



Визуализатор «Что-если» позволит провести эксперимент, введя любые значения множителей АРГУМЕНТ1 и АРГУМЕНТ2 и рассчитав результат их произведения.



Вид построенной сети можно посмотреть, выбрав визуализатор “Граф нейронной сети”.



Выводы

Данный пример показал, как можно построить модель прогноза, используя нейронную сеть. Пример показал, что для построения нет необходимости в строгой математической спецификации модели, что особенно ценно при анализе плохо формализуемых процессов. А большинство бизнес задач плохо формализуется. Это означает, что наличие достаточно развитых и удобных инструментальных программных средств позволяет аналитику при построении модели прогнозируемого процесса руководствоваться такими понятиями, как опыт и интуиция.

Настройки мастера позволяют увидеть широкие возможности Deductor Studio касательно структуры сети, способов обучения и т. д. Аналитику предоставляется широкие возможности по настройке нормализации столбцов, разбиения данных на обучающее и тестовое множество, определения структуры сети, количества слоев и нейронов в каждом слое, выборе функции активации и ее параметров, выборе различных алгоритмов обучения и настройки их параметров. Все это позволяет построить модель описывающую практически любые закономерности. Также было показано, как можно спрогнозировать результат, введя любые значения входных факторов, используя визуализатор «Что-если». Понятно, что этап построения модели стоит на завершающих позициях анализа данных и
перед тем, как его провести, необходимо должным образом подготовить данные, что позволяет сделать широкий набор инструментов Deductor Studio. Качество подготовки данных для модели, а также качество самой модели аналитик может оценить разными способами: посмотреть диаграмму рассеяния, провести ряд экспериментов при помощи «Что-если», построить гистограмму распределения ошибки и т. п.

Лабораторная работа 2 Кластеризация с помощью самоорганизующейся карты Кохонена

Самоорганизующаяся карта Кохонена является разновидностью нейронной сети. Она применяется, когда необходимо решить задачу кластеризации, т. е. распределить данные по нескольким кластерам. Алгоритм определяет расположение кластеров в многомерном пространстве факторов. Исходные данные будут относиться к какому-либо кластеру в зависимости от расстояния до него. Многомерное пространство трудно для представления в графическом виде. Механизм же построения карты Кохонена позволяет отобразить многомерное пространство в двумерном, которое более удобно и для визуализации и для интерпретации результатов аналитиком.

Также с помощью построенной карты Кохонена можно решить и задачу прогнозирования. В этом случае результирующее поле (то, которое необходимо спрогнозировать) в построении карты не участвует. После

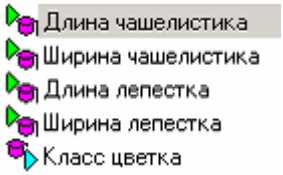
кластеризации используя диаграмму «Что-если» можно провести эксперимент. Алгоритм определяет точку пространства, где расположены введенные для прогноза данные, затем определяет, к какому кластеру принадлежит данная точка и подсчитывает среднее по результирующему полю всех точек этого кластера, что и будет результатом прогноза (для дискретных данных результатом прогноза является значение, больше всего встречающееся в результирующем поле всех ячеек кластера).

Исходные данные

Рассмотрим механизм кластеризации путем построения самоорганизующейся карты, основываясь на типичных характеристиках цветков. Исходная таблица находится в файле «Iris. txt». Она содержит следующие параметры цветов: «ДЛИНА ЧАШЕЛИСТИКА», «ШИРИНА ЧАШЕЛИСТИКА», «ДЛИНА ЛЕПЕСТКА», «ШИРИНА ЛЕПЕСТКА», «КЛАСС ЦВЕТКА». Задача состоит в том, чтобы определить по различным параметрам цветка его класс. Предполагается, что цветы одного класса имеют схожие параметры, поэтому они должны находиться в одном кластере.

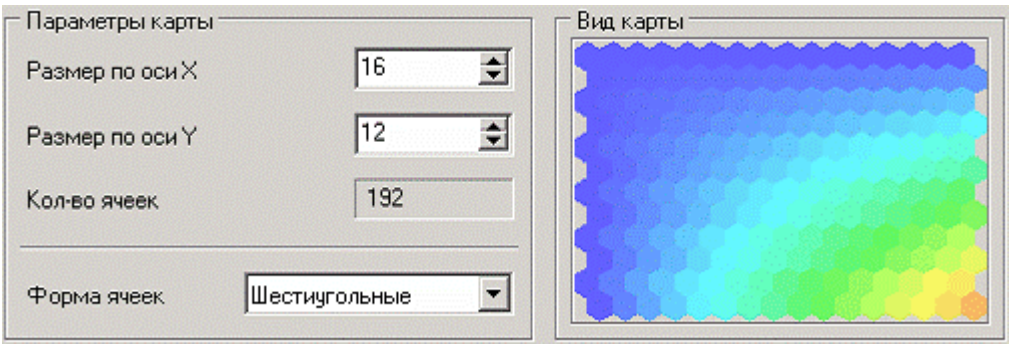
Кластеризация ирисов

Для начала необходимо импортировать данные из файла. После этого запустим, мастер обработки и выберем из списка метод обработки «Карта Кохонена». На втором шаге мастера настроим назначения столбцов. Укажем столбцу «КЛАСС ЦВЕТКА» назначение «Выходной», а остальным – «Входной». Т. е. на основе данных о цветке будем относить его к тому или иному классу.

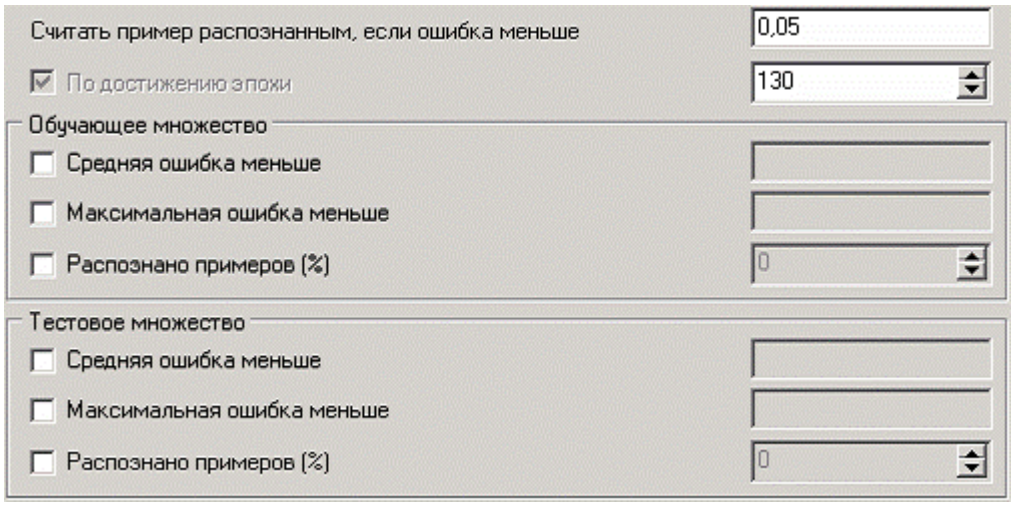


На третьем шаге мастера необходимо настроить способ разделения исходного множества данных на тестовое и обучающее, а также количество примеров в том и другом множестве. Укажем, что данные обоих множеств берутся случайным образом, зададим размер тестового множества равным десяти примерам, путем изменения значения столбца «Размер в строках» строки «Тестовое множество».

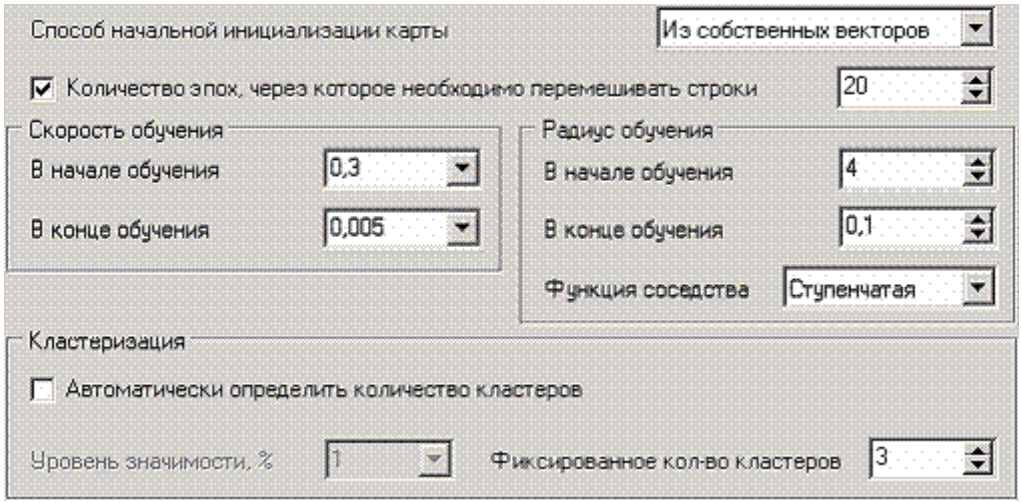
Следующий шаг предлагает настроить параметры карты (количество ячеек по X и по Y, их форму) и параметры обучения (способ начальной инициализации, тип функции соседства, перемешивать ли строки обучающего множества и количество эпох, через которые необходимо перемешивание). Значения по умолчанию вполне подходят.



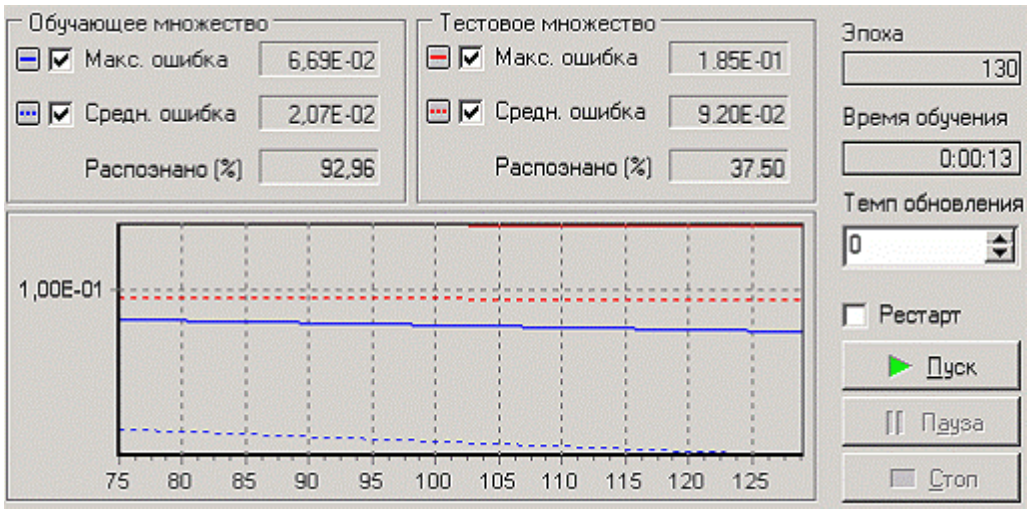
На пятом шаге мастера необходимо настроить параметры остановки обучения. Оставим параметры по умолчанию.



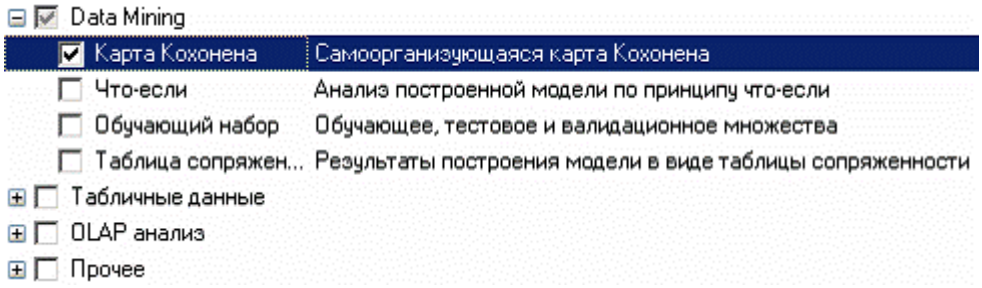
На шестом шаге настраиваются остальные параметры обучения – способ начальной инициализации, тип функции соседства и также параметры кластеризации – автоматическое определение числа кластеров с соответствующим уровнем значимости либо фиксированное количество кластеров предоставляется возможность настроить интервалы обучения. Каждый интервал задается количеством эпох, радиусом обучения и скоростью обучения. Укажем фиксированное количество кластеров, равное трем.



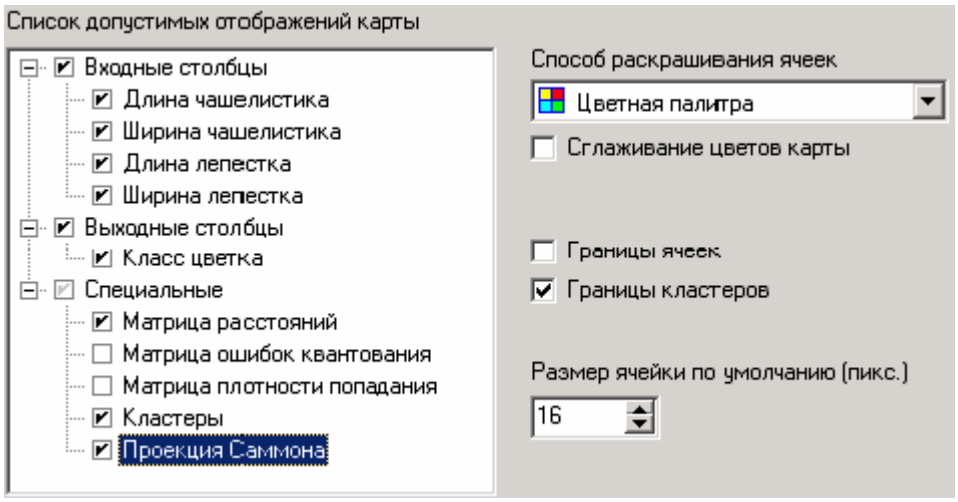
На седьмом шаге предлагается запустить сам процесс обучения. Во время обучения можно посмотреть количество распознанных примеров и текущие значения ошибок. Здесь необходимо нажать на кнопку пуск и дождаться завершения процесса обработки.



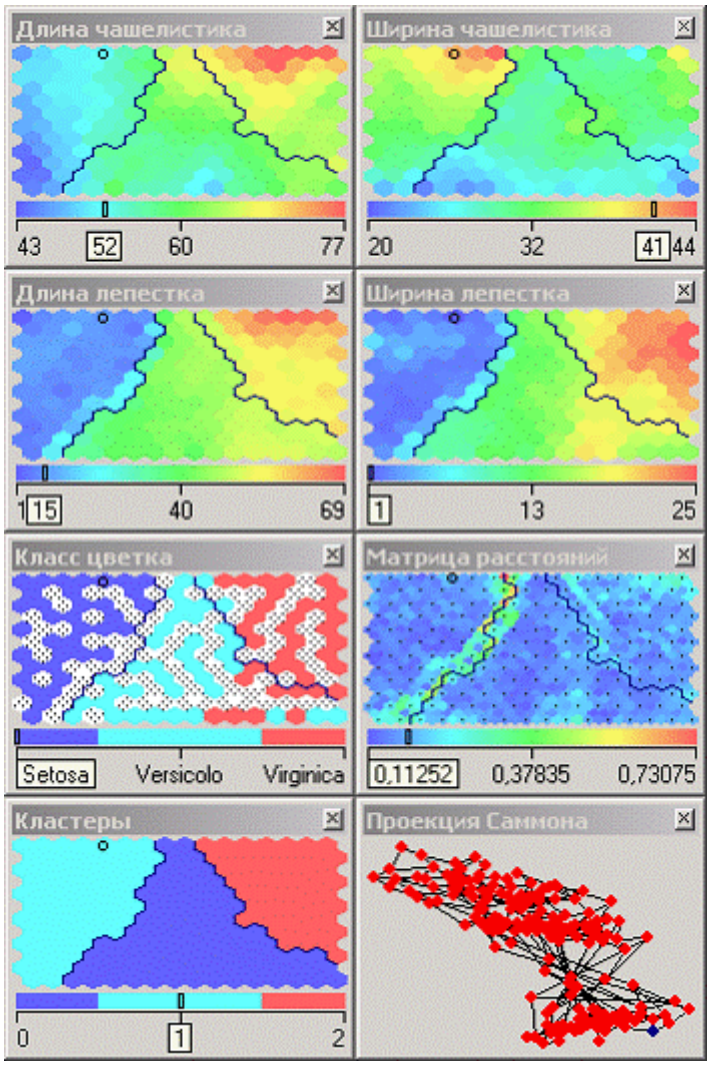
После этого необходимо в списке визуализаторов выбрать появившуюся теперь «Карту Кохонена» для просмотра результатов кластеризации, а также визуализатор «Что-если» для прогнозирования класса цветка.



Далее, в мастере настройки отображения карты Кохонена необходимо указать, чтобы отображались все поля, также следует установить количество кластеров равным трем и поставить флажок «Границы кластеров».



После этого можно увидеть полученные результаты.



Качество кластеризации можно оценить, просмотрев карту «КЛАСС ЦВЕТКА». На ней видно, что большинство цветов были классифицированы правильно. Заметим, что все цветы класса Setosa попали в один кластер. Это говорит о значительном отличии параметров цветов этого класса от других. Явное различие наблюдается по длине и ширине лепестка. То, что часть примеров Virginica попала в класс Versicolo и наоборот говорит о меньшем различии этих классов. На картах, в отличие от Setosa не видны резкие отличия параметров цветов этих двух классов. Этим как раз и объясняется «проникновение» некоторой части примеров в другой кластер.

Выводы

Данный пример показал область применения самоорганизующихся карт. Изначально имелось многомерное (четырехмерное) пространство входных факторов. Алгоритм представил его в двумерном виде, которое удобнее анализировать. Также исходные данные были отнесены к трем кластерам, по типу цветка – «Setosa», «Versicolo», «Virginica». Основным визуализатором после построения является «Самоорганизующаяся карта». Здесь в первую очередь следует обратить внимание на матрицу расстояний и проекцию Саммона. На них явно видны расстояния между отдельными ячейками карты, т. е. четкие границы различных скоплений данных. Мастер предоставляет широкий набор настройки параметров обучения: настройка нормализации столбцов, настройка разбиения на тестовое и обучающее множество, настройка условий остановки обучения, настройка параметров карты и параметров обучения, настройка интервалов обучения.

Лабораторная работа 3. Машины опорных векторов

Цель работы: Ознакомиться с алгоритмом классификации с помощью машин опорных векторов.

Постановка задачи

Часто в алгоритмах машинного обучения возникает необходимость классифицировать данные. Каждый объект данных представлен как вектор (точка) в p-мерном пространстве (последовательностьрчисел). Каждая из этих точек принадлежит только одному их двух классов. Нас интересует, можем ли мы разделить точки гиперплоскостью размерностью "p−1". Это типичный случай линейной разделимости. Таких гиперплоскостей может быть много. Поэтому вполне естественно полагать, что максимизация зазора между классами способствует более уверенной классификации. То есть можем ли мы найти такую гиперплоскость, чтобы расстояние от нее до ближайшей точки было максимальным. Это бы означало, что расстояние между двумя ближайшими точками, лежащими по разные стороны гиперплоскости, максимально. Если такая гиперплоскость существует, то она нас будет интересовать больше всего, она называется оптимальной разделяющей гиперплоскостью, а соответствующий ей линейный классификатор называется оптимально разделяющим классификатором.

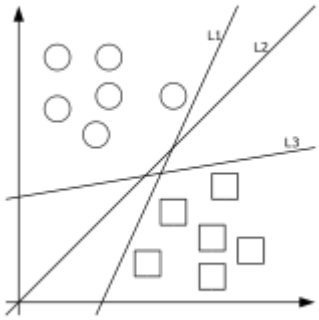


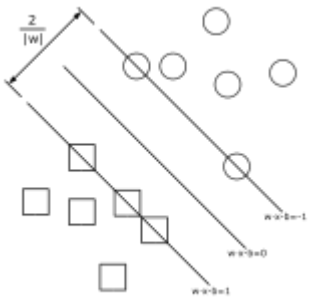
Рис. 1Несколько классифицирующих разделяющих прямых (гиперплоскостей). Но только одна достигает оптимального разделения.

Формальное описание задачи

Мы полагаем, что точки имеют вид::

$$\{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \ldots, (\mathbf{x}_n, c_n)\}$$

где ciпринимает значение 1 или −1, в зависимости от того, какому классу принадлежит точка $\mathbf{x}_i$. Каждое $\mathbf{x}_i$ этор-мерный вещественный вектор, обычно нормализованный значениями [0,1] или [-1,1]. Если точки не будут нормализованы, то точка с большими отклонениями от средних значений координат точек слишком сильно повлияет на классификатор. Мы можем рассматривать это, как учебную коллекцию, в которой для каждого элемента уже задан класс, к которому он принадлежит. Мы хотим, чтобы алгоритм метода опорных векторов классифицировал их таким же образом. Для этого мы строим разделяющую гиперплоскость, которая имеет вид:



Оптимальная разделяющая гиперплоскость для метода опорных векторов, построенная на точках из двух классов. Ближайшие к параллельным гиперплоскостям точки называются опорными векторами.

$$\mathbf{w} \cdot \mathbf{x} - b = 0.$$

Вектор $\mathbf{w}$- перпендикуляр к разделяющей гиперплоскости. Параметрbзависит от кратчайшего расстояния гиперплоскости до начала координат. Если параметрbравен нулю, гиперплоскость проходит через начало координат, что ограничивает решение.

Так как нас интересует оптимальное разделение, нас интересуют опорные вектора и гиперплоскости параллельные оптимальной и ближайшие к опорным векторам двух классов. Можно показать, что эти параллельные гиперплоскости могут быть описаны следующими уравнениям (с точностью до нормировки).

$$\mathbf{w} \cdot \mathbf{x} - b = 1,$$

$$\mathbf{w} \cdot \mathbf{x} - b = -1.$$

Если учебная коллекция линейно разделима, то мы можем выбрать гиперплоскости таким образом, чтобы между ними не лежала ни одна точка обучающей выборки и затем максимизировать расстояние между гиперплоскостями. Ширину полосы между ними легко найти из соображений геометрии, она равна $\frac{2}{\|\mathbf{w}\|}$[1]., таким образом наша задача минимизировать $\|\mathbf{w}\|$. Чтобы исключить все точки из полосы, мы должны убедиться для всехi, что

$$\left[\begin{array}{l} \mathbf{w} \cdot \mathbf{x_i} - b \geq 1, \ c_i = 1 \\ \mathbf{w} \cdot \mathbf{x_i} - b \leq -1, \ c_i = -1 \end{array} \right.$$

Это может быть также записано в виде:

$$c_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1, \quad 1 \leq i \leq n. \qquad (1)$$

Случай линейной разделимости

Проблема построения оптимальной разделяющей гиперплоскости сводится к минимизации $\|\mathbf{w}\|$, при условии (1). Это задача квадратичной оптимизации, которая имеет вид:

$$\left\{ \begin{array}{l} \|\mathbf{w}\|^2 \to \min \\ c_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1, \quad 1 \leq i \leq n. \end{array} \right.$$

По теореме Куна-Таккера эта задача эквивалентна двойственной задаче поиска седловой точки функции Лагранжа

$$\left\{ \begin{array}{l} \mathbf{L}(\mathbf{w}, \mathbf{b}; \lambda) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \lambda_\mathbf{i}(c_i((\mathbf{w} \cdot \mathbf{x_i}) - b) - 1) \to \min_{w,b} \max_{\lambda} \\ \lambda_\mathbf{i} \geq 0, \quad 1 \leq i \leq n \end{array} \right. \qquad (2)$$

где $\lambda = (\lambda_\mathbf{1}, \ldots, \lambda_\mathbf{n})$- вектор двойственных переменных.

Сведем эту задачу к эквивалентной задаче квадратичного программирования, содержащую только двойственные переменные:

$$\left\{ \begin{array}{l} -\mathbf{L}(\lambda) = -\sum_{i=1}^{n} \lambda_\mathbf{i} + \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_\mathbf{i}\lambda_\mathbf{j} c_i c_j (\mathbf{x_i} \cdot \mathbf{x_j}) \to \min_{\lambda} \\ \lambda_\mathbf{i} \geq 0, \quad 1 \leq i \leq n \\ \sum_{i=1}^{n} \lambda_\mathbf{i} c_i = 0 \end{array} \right. \qquad (3)$$

Допустим мы решили данную задачу, тогда $\mathbf{w}$и $\mathbf{b}$можно найти по формулам:

$$\mathbf{w} = \sum_{i=1}^{n} \lambda_\mathbf{i} c_i \mathbf{x_i}$$

$$\mathbf{b} = \mathbf{w} \cdot \mathbf{x_i} - c_i, \quad \lambda_i > 0$$

В итоге алгоритм классификации может быть записан в виде:

$$a(x) = sign\left(\sum_{i=1}^{n} \lambda_\mathbf{i} c_i \mathbf{x_i} \cdot \mathbf{x} - b\right) \quad (4)$$

Обратим внимание, что суммирование идет не по всей выборке, а только по опорным векторам, для которых $\lambda_\mathbf{i} \neq 0$.

Случай линейной неразделимости

Для того, чтобы алгоритм мог работать в случае, если классы линейно неразделимы, позволим ему допускать ошибки на учебной коллекции. Введем набор дополнительных переменных $\xi_i \geq 0$, характеризующих величину ошибки на объектах $\mathbf{x}_i, \quad 1 \leq i \leq n$. Возьмем за отправную точку (2), смягчим ограничения неравенства, так же введем в минимизируемый функционал штраф за суммарную ошибку:

$$\begin{cases} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i \to \min_{w,b,\xi_i} \\ c_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1 - \xi_i, \quad 1 \leq i \leq n \\ \xi_i \geq 0, \quad 1 \leq i \leq n \end{cases}$$

Коэффициент C - параметр настройки метода, который позволяет регулировать отношение между максимизацией ширины разделяющей полосы и минимизацией суммарной ошибки.

Аналогично, по теореме Куна-Таккера сводим задачу к поиску седловой точки функции Лагранжа:

$$\begin{cases} \mathbf{L}(\mathbf{w}, \mathbf{b}, \xi; \lambda, \eta) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \lambda_\mathbf{i}(c_i(\mathbf{w} \cdot \mathbf{x_i}) - 1) - \sum_{i=1}^{n} \xi_\mathbf{i}(\lambda_\mathbf{i} + \eta_\mathbf{i} - C) \to \mathrm{mi} \\ \xi_\mathbf{i} \geq 0, \lambda_\mathbf{i} \geq 0, \eta_\mathbf{i} \geq 0, \quad 1 \leq i \leq n \\ \left[\begin{array}{l} \lambda_\mathbf{i} = 0 \\ c_i(\mathbf{w} \cdot \mathbf{x_i} - b) = 1 - \xi_i, \\ \eta_\mathbf{i} = 0 \\ \xi_\mathbf{i} = 0, \end{array}\right. \begin{array}{l} \\ 1 \leq i \leq n \\ \\ 1 \leq i \leq n \end{array} \end{cases}$$

По аналогии сведем эту задачу к эквивалентной:

$$\begin{cases} -\mathbf{L}(\lambda) = -\sum_{i=1}^{n} \lambda_\mathbf{i} + \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_\mathbf{i}\lambda_\mathbf{j} c_i c_j (\mathbf{x_i} \cdot \mathbf{x_j}) \to \min_\lambda \\ 0 \leq \lambda_\mathbf{i} \leq \mathbf{C}, \quad 1 \leq i \leq n \\ \sum_{i=1}^{n} \lambda_\mathbf{i} c_i = 0 \end{cases}$$

На практике для построения машины опорных векторов решают именно эту задачу, а не (3), так как гарантировать линейную разделимость точек на два класса в общем случае не представляется возможным. Этот вариант алгоритма называют алгоритмом с мягким зазором (soft-margin SVM), тогда как в линейно разделимом случае говорят о жёстком зазоре (hard-margin SVM).

Для алгоритма классификации сохраняется формула (4), с той лишь разницей, что теперь ненулевыми λ_i обладают не только опорные объекты, но и объекты-нарушители. В определённом смысле это недостаток, поскольку нарушителями часто оказываются шумовые выбросы, и построенное на них решающее правило, по сути дела, опирается на шум.

Константу C обычно выбирают по критерию скользящего контроля. Это трудоёмкий способ, так как задачу приходится решать заново при каждом значении C.

Если есть основания полагать, что выборка почти линейно разделима, и лишь объекты-выбросы классифицируются неверно, то можно применить фильтрацию выбросов. Сначала задача решается при некотором C, и из выборки удаляется небольшая доля объектов, имеющих наибольшую величину ошибки ξ_i. После этого задача решается заново по усечённой выборке. Возможно, придётся проделать несколько таких итераций, пока оставшиеся объекты не окажутся линейно разделимыми.

Ядра

Алгоритм построения оптимальной разделяющей гиперплоскости, предложенный в 1963 году Владимиром Вапником — алгоритм линейной классификации. Однако в 1992 году Бернхард Босер, Изабель Гуйон и Вапник предложили способ создания нелинейного классификатора, в основе которого лежит переход от скалярных произведений к произвольным ядрам, так называемый kernel trick (предложенный впервые М. А. Айзерманом, Э. М. Броверманом и Л. В. Розоноером для метода потенциальных функций), позволяющий строить нелинейные разделители. Результирующий алгоритм крайне похож на алгоритм линейной классификации, с той лишь разницей, что каждое скалярное произведение в приведённых выше формулах заменяется нелинейной функцией ядра (скалярным произведением в пространстве с большей размерностью). В этом пространстве уже может существовать оптимальная разделяющая гиперплоскость. Так как размерность получаемого пространства может быть больше размерности исходного, то преобразование, сопоставляющее скалярные произведения, будет нелинейным, а значит функция, соответствующая в исходном пространстве оптимальной разделяющей гиперплоскости, будет также нелинейной.

Стоит отметить, что если исходное пространство имеет достаточно высокую размерность, то можно надеяться, что в нём выборка окажется линейно разделимой.

Рассмотрим наиболее распространенные ядра:

Полиномиальное (однородное): $k(\mathbf{x}, \mathbf{x}') = (\mathbf{x} \cdot \mathbf{x}')^d$

Полиномиальное (неоднородное): $k(\mathbf{x}, \mathbf{x}') = (\mathbf{x} \cdot \mathbf{x}' + 1)^d$

Радиальная базисная функция: $k(\mathbf{x}, \mathbf{x}') = \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2)$, для $\gamma > 0$

Радиальная базисная функция Гаусса: $$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right)$$

Сигмоид: $k(\mathbf{x}, \mathbf{x}') = \tanh(\kappa \mathbf{x} \cdot \mathbf{x}' + c)$, для почти всех $\kappa > 0$ и $c < 0$

Пример

Загрузите данные из встроенной базы данных MATLAB для классической задачи классификации ирисов Фишера. БД содержит 150 примеров с 5 признаками

load fisheriris

Создайте переменную data, в которой в 1 столбце хранятся измерения sepal length , во 2 - sepal width для 150 ирисов.

data = [meas(:,1), meas(:,2)];

Из вектора species, создайте новый вектор-столбец groups, чтобы классифицировать данные в две группы: Setosa и non-Setosa.

groups = ismember(species,'setosa');

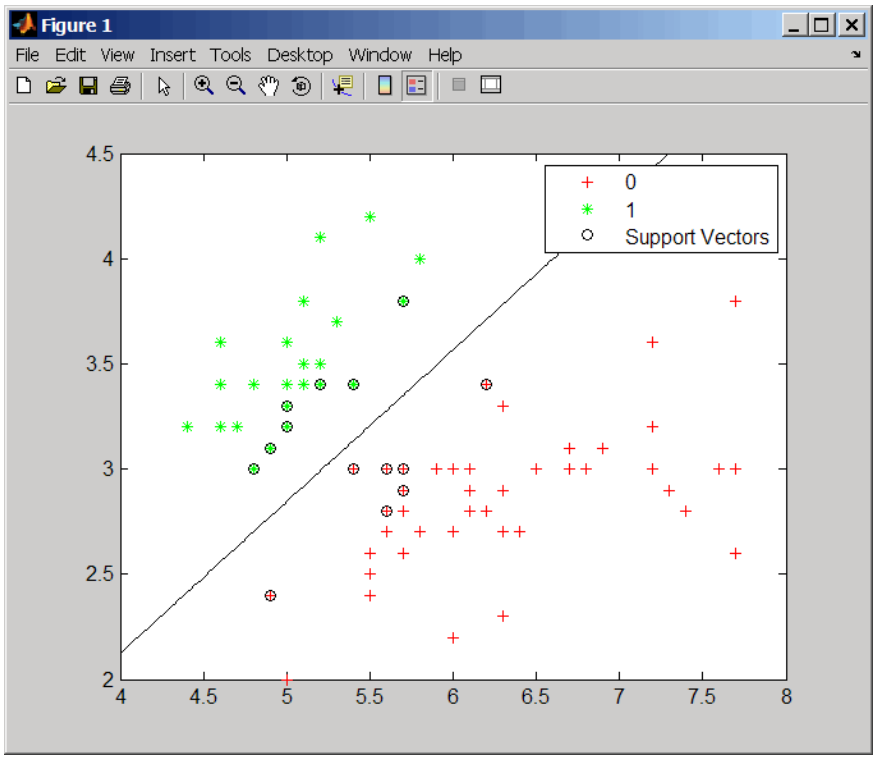
Наугад выберите обучающее и тестовое множества.

[train, test] = crossvalind('holdOut',groups);

cp = classperf(groups);

Обучите SVM – классификтор, используя линейную функцию ядра и отобразите сгруппированные данные.

svmStruct = svmtrain(data(train,:),groups(train),'showplot',true);

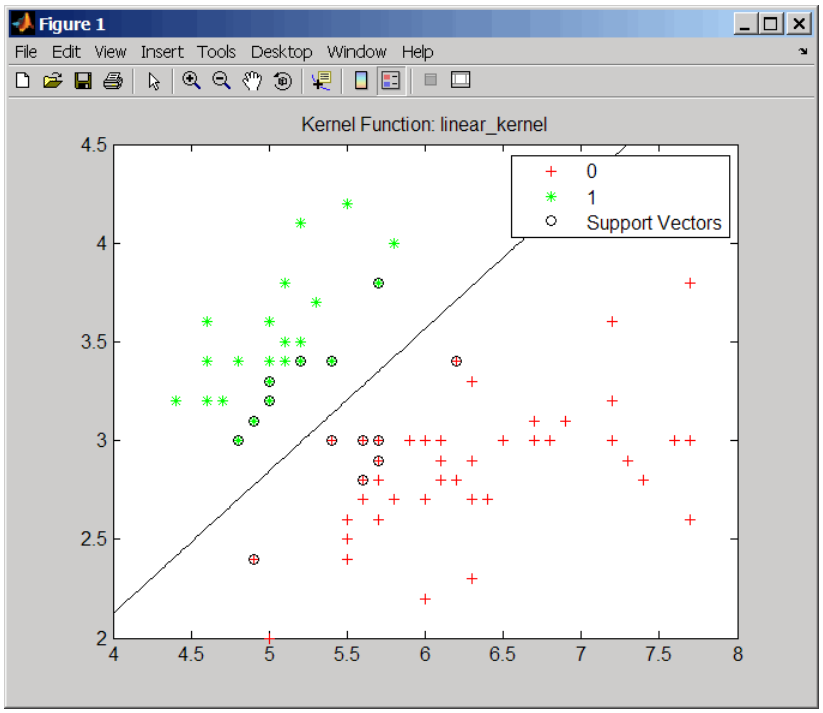


Добавьте заголовок к графику, используя поле svmStruct.KernelFunction в качестве значения.

title(sprintf('Kernel Function: %s',...

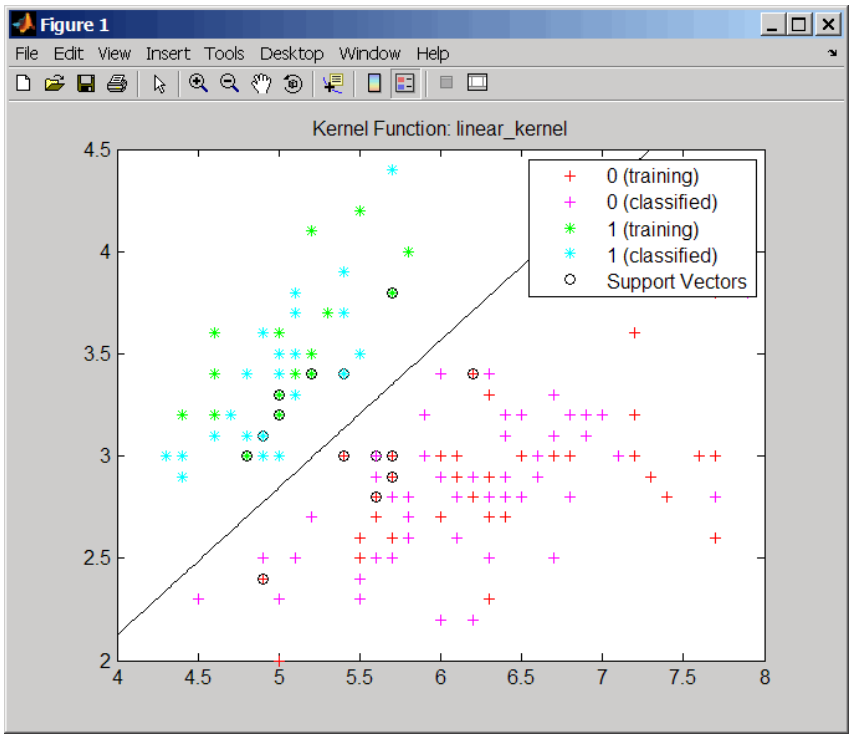
func2str(svmStruct.KernelFunction)),...

'interpreter','none');



Используйте функцию svmclassify для классификации тестового множества.

```
classes = svmclassify(svmStruct,data(test,:),'showplot',true);
```



Вычислите точность классификации.

```
classperf(cp,classes,test);
```

```
cp.CorrectRate
```

```
ans =
```

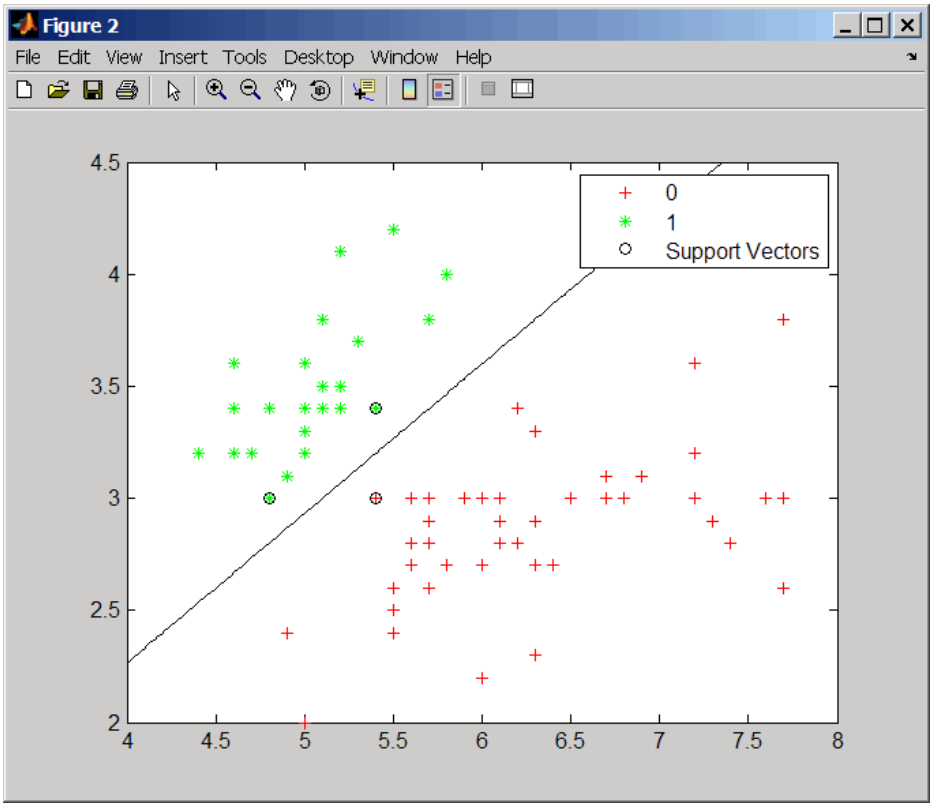
0.9867

Используйте one-norm, hard margin support vector machine классификатор для изменения свойства boxconstraint.

figure

svmStruct = svmtrain(data(train,:),groups(train),...

'showplot',true,'boxconstraint',1e6);



Задание:

Выполните пример, описанный в лабораторной работе

Сравните алгоритмы классификации с помощью радиально-базисных сетей, карт Кохонена и машины опорных векторов

Лабораторная работа 4
Построение и оптимизация параметров деревьев решений в аналитической платформе Deductor

Цель работы. Изучения процесса построения, оптимизации и оценки эффективности деревьев решений в аналитическом приложении Deductor Studio.

Теоретическая часть. Наряду с нейронными сетями деревья решений являются еще одной популярной технологией Data Mining, широко используемой для решения задач численного предсказания и классификации. Так же, как и нейронные сети, деревья решений являются моделями, строящимися на основе обучения с учителем. Но структура их существенно отличается. Деревья решений представляют собой древовидные иерархические структуры, состоящие из решающих правил вида «Если…то…». В основе работы деревьев решений лежит процесс рекурсивного разбиения исходного множества наблюдений или объектов на подмножества, ассоциированные с классами. Разбиение производится с помощью решающих правил, в которых осуществляется проверка значений

атрибутов по заданному условию. Рекурсивными называются алгоритмы, которые на каждом последующем шаге используются результаты, полученные на предыдущем.

Деревья решений состоят из объектов 2-х типов – узлов и листьев. В узлах располагаются правила, а в листьях – результат, т.е. наблюдения исходного набора данных, имеющих одинаковую переменную класса. Таким образом, каждый лист в дереве связан (или, как часто говорят в литературе – ассоциирован) с определенным классом. Пример структуры дерева решений представлен на рис. 1.

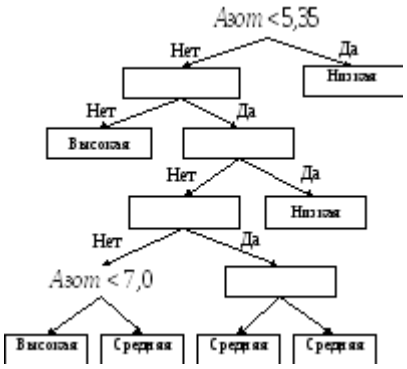


Рис. 1. Дерево решений, для классификации уровня урожайности на основе данных агрохимического обследования почв.

Данное дерево решений решает задачу классификации по урожайности (высокая, средняя, низкая) земельных участков на основе данных агрохимического обследования почв по содержанию азота, калия и фосфора (в мг/100 г.). Визуально узлы и листья в дереве хорошо различимы: в узлах указываются правила, разбивающие содержащиеся в нем наблюдения, и производится дальнейшее ветвление. В листьях правил нет, они помечаются меткой класса, объекты которого попали в данный лист. Кроме этого, ветвление в листьях не производится и они заканчивают собой ветвь дерева (поэтому их иногда называют терминальными узлами).

Обратите внимание, что дерево решений является линейным классификатором, т.е. производит разбиение объектов в многомерном пространстве признаков плоскостями. Для двумерного случая это поясняется рис. 2.

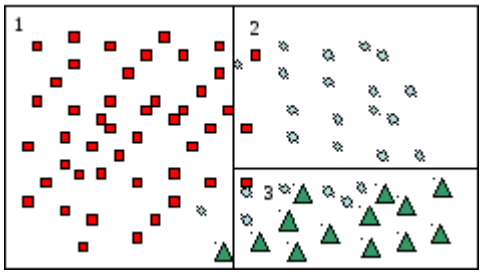


Рисунок 2. Дерево решений как линейный классификатор.

Пусть прямоугольники обозначают участки с низкой урожайностью, круги – со средней, а треугольники – с высокой. Расположение объектов на рисунке качественно отражает их расположение в пространстве признаков. Мы разделили линиями пространство на три подмножества и ассоциировали их с классами: 1- «Высокая», 2 – «Средняя» и 3 – «Низкая». Эти же подмножества будут соответствовать и трем исходам классификации. Обратите внимание, что во всех 3-х подмножествах имеются нераспознанные примеры, т.е. примеры, попавшие в подмножества, ассоциированные с другим классом.

Теоретически, алгоритм может генерировать новые разбиения до тех пор, все примеры не будут распознаны правильно. Однако это приводит к усложнению дерева: большое число ветвлений усложняет его структуру и ухудшает его интерпретацию пользователем. Действительно, при использовании нейронных сетей аналитика интересует только реакция модели на определенное входное воздействие, а сама структура сети и значения ее весов никакой смысловой нагрузки не несут. В то же время, правила в деревьях решений формируются практически на естественном языке, что делает объясняющую способность очень высокой. Иными словами, большой интерес представляют не только сами деревья, как модели, делающие предсказания, но и правила, автоматически формируемые при их построении. Например, первое правило в дереве решений на

рис. $Азот < 5{,}35$, формирующие лист «Низкая», указывает, что низкое содержание азота в почве связано с низкой урожайностью. Таким образом, правила, извлеченные из деревьев решений, имеют собственную ценность с точки зрения принятия управленческих решений. Поэтому если дерево решений оказывается слишком сложным и содержит много правил, то разбираться в них становится сложение и объясняющая способность модели падает.

Таким образом, вопрос о том, когда следует остановить рост дерева решений, это компромисс между точностью и объясняющей способностью модели, а также вычислительными затратами, связанными с ее построением и использованием. На практике, ограничение роста, или, как часто говорят, упрощение, дерева осуществляется соответствующими настройками алгоритма обучения. Используются два подхода:

1. Отсечение ветвей (стрижка). Сначала строится полное дерево, распознающее все примеры. Как правило, оно получается слишком сложным и трудно интерпретируемым. Затем производится автоматическое удаление узлов и листьев, имеющих малую значимость, т.е. тех, в которые попало лишь небольшое количество примеров. Понятно, что решающие правила, относящиеся к очень узкому кругу наблюдений, практически не имеют ценности. Наоборот, правила, позволяющие классифицировать большое число наблюдений, являются наиболее значимыми. В качестве параметра отсечения ветвей можно использовать показатель, называемый достоверностью:

$$C_k = N_{нерасп} / N_k ,$$

где k - номер узла, $N_{нерасп}$ - число нераспознанных примеров в узле, общее число примеров, распределенных в k-й узел. Он определяется как отношение числа неправильно распознанных примеров в листе к общему числу примеров. Следовательно, чем меньше число ошибок в листе, тем выше достоверность правила. Поэтому, определив пороговый уровень достоверности, можно заставить алгоритм отсечь все листья, достоверность в которых ниже заданного порога.

2. Ранняя остановка. В конечных узлах дерева, как правило, оказывается мало примеров. Правила, в таких узлах имеют низкую значимость и объясняющую способность. Поэтому, задав минимально допустимое количество примеров в узле дерева, можно ограничить его рост, запретив создавать новые узлы.

Оба подхода имеют свои преимущества и недостатки. Отсечение ветвей требует дополнительных временных затрат на построение полного дерева. Ранняя остановка экономит время и вычислительные затраты, но может привести к тому, что потенциально интересные правила могут быть простой не найдены.

Замечание. Алгоритм построения деревьев решений, реализованный в АП Deductor, позволяет решать только задачи классификации, поэтому выходная переменная может быть только дискретной (метка класса).

Таблица сопряженности. Для быстрой и эффективной оценки надежности классификации с помощью дерева решений является таблица сопряженности, которая является аналогом диаграммы рассеяния для задачи классификации. Пример таблицы сопряженности представлен в таблице 1.

	Классифицировано			
Фактически	Высокая	Низкая	Средняя	Итого
Высокая	8			8
Низкая		30		30
Средняя	4		14	18
Итого	12	30	14	56

Ячейки, расположенные на главной диагонали таблицы и стоящие на пересечении одноименных строк и столбцов содержат число примеров, распознанных для данного класса правильно. Например, число 30 на пересечении столбца и строки «Низкая» указывает, что 30 примеров, фактически имеющих класс «Низкая» классифицированы соответствующим образом. Любое число в ячейке, расположенной вне главной диагонали, т.е. на пересечении разноименных строки и столбца, показывают число неправильно распознанных примеров для данного класса. Например, число 4, расположенное в ячейке на пересечении строки «Средняя» и столбца «Высокая», указывает на то, что 4 примера, фактически имеющих класс «Средняя», были ошибочно распознаны деревом как имеющие класс «Высокая».

Порядок выполнения работы.

1. Запустить аналитическое приложение Deductor Studio и загрузить набор данных по указанию преподавателя».

2. Открыть Мастер обработки и в разделе Data Mining в списке доступных методов обработки выбрать пункт «Дерево решений».

3. На 2-м шаге Мастера обработки установить назначение поле набора данных.

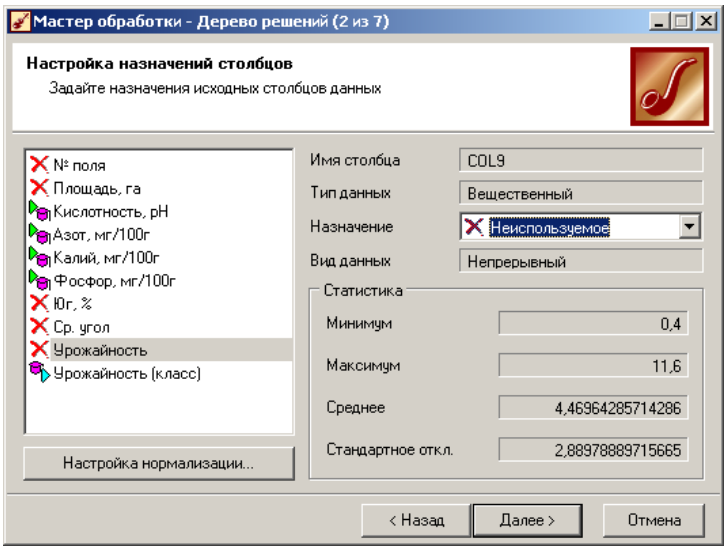


Рисунок 3. Настройка назначения полей.

4. На 3-м шаге установить размеры обучающего и тестового множеств 90% и 10% соответственно, способ отбора – случайно (рис. 4).

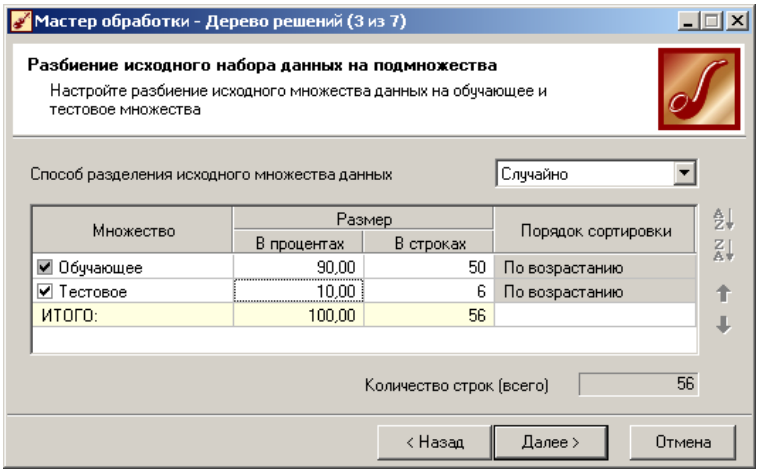


Рисунок 4. Настройка размеров обучающего и тестового множеств.

5. На 4-м шаге Мастера настроить параметры обучения дерева решений:

- параметры ранней остановки: минимальное количество примеров в узле, при котором будет создан новый узел – 2, флажок «Строить дерево с более достоверными правилами в ущерб компактности» - установить;

- отсечение узлов дерева – отключить, сбросив соответствующий флажок в разделе Параметры отсечения.

Таким образом, мы настроили алгоритм обучения на построение «полного» дерева, содержащее максимальное число правил.

6. На 5-м шаге запустить процесс обучения и проконтролировать его результаты (рис. 5): зафиксировать процент распознанных примеров на обучающем и тестовом множестве, сделать вывод о результатах обучения; зафиксировать число правил и узлов в построенном дереве. Темп обновления установит равным 1.

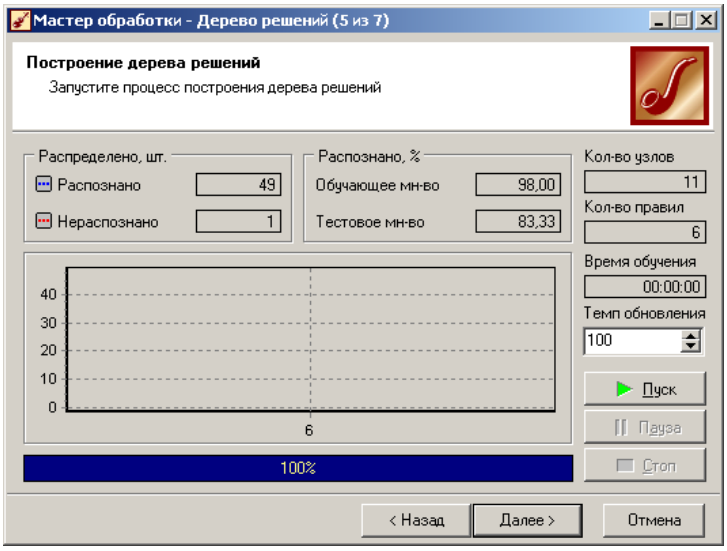


Рисунок 5. Запуск процесса обучения и контроль его результатов.

7. На 6-м шаге Мастера обработки выбрать способы визуализации: Дерево решений, Обучающий набор, Правила, Таблица сопряженности.

8. По внешнему виду дерева решений визуально оценить его сложность и интерпретируемость. Схематично зарисовать полученное дерево решений и для каждого узла зафиксировать число примеров каждого класса, попавшее в него. Для этого выделить узел щелчком мыши и щелкнуть по кнопке  , в результате чего внизу откроется список попавших в данный узел объектов. Сделать вывод, какое из правил является самым эффективным (то, которое распределит в узел наибольшее число примеров одного класса). По таблице сопряженности определить характер ошибки.

9. По правилам, извлеченным из дерева решений (вкладка «Правила» в верхней части окна), сделать вывод о влиянии тех или иных показателей на урожайность, например, какая урожайность связана с низкой кислотностью, а какая – с высоки содержанием калия и т.д.).

10. По таблице сопряженности определить, сколько ошибок классификации допустила модель и в чем они заключаются.

11. Построить зависимость числа распознанных примеров и узлов дерева от минимально допустимого числа примеров в узле. Для этого повторить пункты 5 -7, изменяя минимально допустимое число примеров в узле от 2 до 10 и каждый раз регистрируя число правильно и не правильно распознанных примеров. Результаты свести в таблицу вида:

Минимально допустимое число примеров Nmin	2	3	4	5	6	7	8	9	10
Количество узлов в дереве, K									
Правильно распознанных примеров, Nрасп									
Неправильно распознанных примеров Nнерасп									
Относительная ошибка распознавания Nнерасп/N									

По графикам сделать вывод о влиянии минимально допустимого числа примеров в узле на количество узлов в дереве и ошибку распознавания.

4.2. Оценочные средства промежуточной аттестации

По дисциплине предусмотрен зачет. Зачет проходит по билетам. В каждом билете два теоретических вопроса. Зачет проводится в устной, письменной или компьютерной форме. Оценивается владение материалом, его системное освоение, способность применять нужные знания, навыки и умения при анализе проблемных ситуаций.

4.2.1. Устный или письменный ответ на вопрос

4.2.1.1. Порядок проведения.

Устный или письменный ответ на вопрос направлен на проверку знаний основных разделов по дисциплине «Машинное обучение».

4.2.1.2. Критерии оценивания.

43-50 баллов ставится, если обучающийся:

В ответе качественно раскрыл содержание темы. Ответ хорошо структурирован. Прекрасно освоен понятийный аппарат. Продемонстрирован высокий уровень понимания материала. Превосходное умение формулировать свои мысли, обсуждать дискуссионные положения.

36-42 баллов ставится, если обучающийся:

Основные вопросы темы раскрыл. Структура ответа в целом адекватна теме. Хорошо освоен понятийный аппарат. Продемонстрирован хороший уровень понимания материала. Хорошее умение формулировать свои мысли, обсуждать дискуссионные положения.

28-35 баллов ставится, если обучающийся:

Тему частично раскрыл. Ответ слабо структурирован. Понятийный аппарат освоен частично. Понимание отдельных положений из материала по теме. Удовлетворительное умение формулировать свои мысли, обсуждать дискуссионные положения.

0-27 баллов ставится, если обучающийся:

Тему не раскрыл. Понятийный аппарат освоен неудовлетворительно. Понимание материала фрагментарное или отсутствует. Неумение формулировать свои мысли, обсуждать дискуссионные положения.

4.2.1.3. Оценочные средства.

Вопросы для устного или письменного ответа

1. Понятие машинного обучения.
2. Общая постановка задачи обучения по прецедентам.
3. Способы машинного обучения.
4. Задачи, решаемые с помощью машинного обучения.
5. Понятие кластеризации.
6. Постановка задачи кластеризации.
7. Методы кластеризации.
8. Метод k-means.
9. Метод k-медиан.
10. Дискриминантный анализ.
11. Генетический алгоритм.
12. Обучение с учителем.
13. Задача классификации.
14. Задача регрессии.
15. Разделяющая гиперплоскость.
16. Метод опорных векторов.
17. Применение метода опорных векторов в задаче классификации.
18. Понятие дерева решений.
19. Типология деревьев.
20. Обучение дерева решений.

Приложение 2
к рабочей программе дисциплины (модуля)
Б1.В.ДВ.02.02 Машинное обучение

Перечень литературы, необходимой для освоения дисциплины (модуля)

Направление подготовки: 15.03.06 Мехатроника и робототехника

Профиль подготовки: Физические основы мехатроники и робототехники

Квалификация выпускника: бакалавр

Форма обучения: очно-заочная

Язык обучения: русский

Год начала обучения по образовательной программе: 2024

Основная литература:

1. Осипов, Г. С. Методы искусственного интеллекта: монография / Г. С. Осипов. - Москва : Физматлит, 2011. - 296 с. - ISBN 978-5-9221-1323-6. - Текст: электронный. - URL: https://znanium.com/catalog/product/544787 – Режим доступа: по подписке.
2. Сергеев, Н. Е. Системы искусственного интеллекта. Часть 1: Учебное пособие / Сергеев Н.Е. - Таганрог:Южный федеральный университет, 2016. - 118 с.: ISBN 978-5-9275-2113-5. - Текст: электронный. - URL: https://znanium.com/catalog/product/991954 . – Режим доступа: по подписке.
3. Теоретические основы информатики / Царев Р.Ю., Пупков А.Н., Самарин В.В [и др.]. - Красноярск: СФУ, 2015. - 176 с.: ISBN 978-5-7638-3192-4. - Текст: электронный. - URL: https://znanium.com/catalog/product/549801. – Режим доступа: по подписке.

1. Вышегуров, С. Х. Информатика [Электронный ресурс] : учеб. пособие / Новосиб. гос. аграр. ун-т. Агроном. фак.; сост.: И.И. Некрасова, С.Х. Вышегуров. - Новосибирск: Золотой колос, 2014. - 105 с. - Текст: электронный. - URL: https://znanium.com/catalog/product/516070. – Режим доступа: по подписке.
2. Ермакова, А.Н. Информатика [Электронный ресурс]: учебное пособие / А.Н. Ермакова, С.В. Богданова. - Ставрополь: Сервисшкола, 2013. - 184 с. - Текст: электронный. - URL: https://znanium.com/catalog/product/514863 – Режим доступа: по подписке.
3. Каймин, В. А. Информатика: Учебник / Каймин В. А. - 6-е изд. - Москва: НИЦ ИНФРА-М, 2015. - 285 с. (Высшее образование: Бакалавриат) ISBN 978-5-16-010876-6. - Текст: электронный. - URL: https://znanium.com/catalog/product/504525. – Режим доступа: по подписке.

Приложение 3
к рабочей программе дисциплины (модуля)
Б1.В.ДВ.02.02 Машинное обучение

Перечень информационных технологий, используемых для освоения дисциплины (модуля), включая перечень программного обеспечения и информационных справочных систем

Направление подготовки: 15.03.06 Мехатроника и робототехника

Профиль подготовки: Физические основы мехатроники и робототехники

Квалификация выпускника: бакалавр

Форма обучения: очно-заочная

Язык обучения: русский

Год начала обучения по образовательной программе: 2024

Освоение дисциплины (модуля) предполагает использование следующего программного обеспечения и информационно-справочных систем:

Программное обеспечение: операционная система Windows, Microsoft office, PyCharm, Kaspersky Free для Windows

Электронная библиотечная система «ZNANIUM.COM»

Электронная библиотечная система Издательства «Лань»

Электронная библиотечная система «Консультант студента»