Федеральное агентство связи

Федеральное государственное образовательное бюджетное учреждение высшего профессионального образования

ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ

ЭЛЕКТРОННАЯ БИБЛИОТЕЧНАЯ СИСТЕМА

Самара

Федеральное государственное бюджетное образовательное
учреждение высшего профессионального образования

**Поволжский государственный университет
телекоммуникаций и информатики**

Кафедра экономических и информационных систем

ПРОЕКТИРОВАНИЕ БАЗ ДАННЫХ И БАЗ ЗНАНИЙ

**Методические указания
для выполнения курсового проекта**

Составитель: Жданова Е.И.

Самара
ИУНЛ ПГУТИ
2011 г.

УДК 004.89

Рецензент

Кандидат технических наук, ст.преподаватель каф. «Прикладная математика» Оренбургского государственного университета Болдырев П.А.

Жданова Е.И. Проектирование баз данных и баз знаний: Методические указания для выполнения курсового проекта. – Самара: ПГУТИ, 2011. – 31 с., ил.

В методических указаниях даются теоретические и практические основы построения нечетких деревьев решений, а также нечеткой экспертной системы в программном пакете CubiCalc v. 2.0.

Методические указания для выполнения курсового проектирования подготовлены на кафедре "Экономические и информационные системы", предназначены для студентов дневной формы обучения специальности 220601 (“Управление инновациями”) и являются руководством к выполнению их студентами. Также они могут быть полезны преподавателям смежных дисциплин и разработчикам программного обеспечения.

Содержание

Введение

Нечеткое управление оказывается особенно полезным, когда технологические процессы являются слишком сложными для анализа с помощью общепринятых количественных методов, или когда доступные источники информации интерпретируются качественно, неточно или неопределенно. Нечеткая логика, в основном, обеспечивает эффективные средства отображения неопределенностей и неточностей реального мира. Наличие математических средств отражения нечеткости исходной информации позволяет построить модель, адекватную реальности.

Нечеткие деревья решений применяются как для решения задач классификации, так и для решения задачи регрессии, когда необходимо знать степени принадлежности к тому или иному исходу. Они могут быть использованы в различных областях: в банковском деле, в медицине, в промышленности и так далее.

Безусловным достоинством данного подхода является высокая точность классификации, достигаемая за счет сочетания достоинств нечеткой логики и деревьев решений. Процесс обучения происходит быстро, а результат прост для интерпретации. Так как алгоритм способен выдавать для нового объекта не только класс, но и степень принадлежности к нему, это позволяет управлять порогом для классификации.

Цели и задачи работы

Цель курсового проектирования – применение на практике знаний, полученных в процессе изучения курса "Проектирование баз данных и баз знаний", и приобретение практических навыков при проектировании и создании нечеткой экспертной системы.

1. Теоретические сведения

1.1 Определение лингвистической переменной

Способность человека оценивать информацию играет существенную роль в определении сложных явлений. По своей природе оценка является приближением. Во многих случаях достаточна весьма приближенная характеристика набора данных, поскольку в большинстве задач, решаемых человеком, не требуется высокая точность.

Способность человека оценивать информацию наиболее ярко проявляется в использовании естественных языков. Каждое слово *t* можно рассматривать как сжатое описание нечеткого подмножества *M(t)* полного множества области рассуждений *X*. Таким образом *M(t)* есть значение *t*. В этом смысле весь язык можно рассматривать как систему, в соответствии с которой нечетким подмножествам множества *X* приписываются элементарные или составные символы, т.е. слова, группы слов и предложения.

При описании объектов и явлений с помощью нечетких множеств используется понятие нечеткой и лингвистической переменных.

Нечеткая переменная характеризуется тройкой *{a, X, A}*,

где *a* – имя переменной,

X – универсальное множество (область определения *a*),

A – нечеткое множество на *X*, описывающее ограничение (то есть *mA(x)*) на значение нечеткой переменной a.

Нечеткие системы основаны на правилах продукционного типа, где в качестве посылки и заключения в правиле используются *лингвистические переменные.* Данное понятие введено Лотфи Заде в 1973 году.

Лингвистическая переменная – переменная, которая может принимать значения фраз из естественного или искусственного языка.

Фразы, значение которых принимает переменная, в свою очередь, являются именами нечетких переменных и описываются нечетким множеством.

Лингвистической переменной называется пятерка

$$\{x, T(x), X, G, M\},$$

где *x* - имя переменной;

T(x) - множество имен лингвистических значений переменной *x* (терм-множество, количество термов в лингвистической переменной редко превышает 7), каждое из которых является нечетким множеством на базовом терм-множестве *X*;

G есть синтаксическое правило для образования имен значений *x*;

M есть семантическое правило для ассоциирования каждой величины значения с ее понятием.

Пример. Рассмотрим лингвистическую переменную с именем:

x: «температура в комнате».

Тогда оставшуюся четверку *{T(x),X,G,M}* можно определить так:

- универсальное множество *T(x)*=[5, 35];

– терм-множество X={"холодно", "комфортно", "жарко"} с такими функциями принадлежностями:

$$\mu_{\text{холодно}} = \frac{1}{1+\left(\frac{\text{T(x)-10}}{7}\right)^{12}};$$

$$\mu_{\text{комфортно}} = \frac{1}{1+\left(\frac{\text{T(x)-20}}{3}\right)^{6}};$$

$$\mu_{\text{жарко}} = \frac{1}{1+\left(\frac{\text{T(x)-30}}{6}\right)^{10}}.$$

– синтаксические правила G, порождающие новые термы с использованием квантификаторов "не", "очень" и "более-менее";
– семантические правила M, в виде таблицы 1.

Табл. 1 – Правила расчета функций принадлежности

Квантификатор	Функция принадлежности
не t	$1-\mu_t$
очень t	μ_t^2
более-менее t	$\sqrt{\mu_t}$

В общем случае значение лингвистической переменной есть составной терм $t=t_1t_2...t_n$, который представляет собой сочетание элементарных термов $t_1t_2...t_n$.

Эти элементарные термы можно разбить на 4 категории:

1. первичные термы, которые являются символами нечетких подмножеств области рассуждения (например, *молодой, старый*);
2. отрицание *не* и союзы *и, или*;
3. лингвистические неопределенности типа *очень, много, слабо, более или менее* и т.д., которые дают возможность модифицировать значения элементарных и составных терминов и служат для увеличения области значений лингвистической переменной;
4. маркеры, такие, как скобки, вводные слова.

Одним из важных применений нечеткой логики выступают нечеткие экспертные системы (НЭС), в которых логические правила вывода оперируют с нечеткими операциями.

Основой для проведения операции нечеткого логического вывода является база правил, содержащая нечеткие высказывания в форме «*Если-То*» и функции принадлежности для соответствующих лингвистических термов.

При этом должны соблюдаться следующие условия:

1) Существует хотя бы одно правило для каждого лингвистического терма выходной переменной.

2) Для любого терма входной переменной имеется хотя бы одно правило, в котором этот терм используется в качестве предпосылки (левая часть правила).

Например, вполне работоспособная система кондиционирования может быть описана правилами:

«Если температура в комнате высокая, То скорость вращения вентилятора высокая»;

«Если температура в комнате низкая, то скорость вращения вентилятора низкая».

Подобные правила вывода используются в нечетких экспертных системах. Как правило, они имеют вид:

Если *цена велика* и *спрос низкий*, То *оборот мал*, (а)

где *цена* и *спрос* – входные переменные;

оборот – выходное значение;

велика, низкий и *мал* – функции принадлежности (нечеткие множества), определенные на множествах значений *цены, спроса* и *оборота* соответственно.

Благодаря возможностям нечеткой логики можно решить проблему, когда классифицировать объект по тому или иному признаку довольно трудно, и тогда говорят не просто о принадлежности к кому-то классу, признаку, атрибуту, а о её степени.

При использовании *нечетких деревьев решений (fuzzy decision trees)* не теряются знания о том, что объект может обладать свойствами как одного признака, так и другого в той или иной мере.

Главной идеей в таком подходе является сочетание возможностей деревьев решений и нечеткой логики.

1.2 Алгоритм построения нечеткого дерева решений

Деревья решений как способ представления классифицирующего правила существуют уже давно.

Предположим, имеется набор N признаков, принимающих значения из множеств $D_1,...,D_N$, соответственно. Обозначим как x_i реализацию i-го признака. В зависимости от конкретного приложения, под x_i может пониматься, например, случайная величина или нечеткое подмножество области значений признака. Пусть для каждого i-го признака задано некоторое семейство $\Delta(D_i)$,

каждый элемент которого есть некоторое подмножество D_i (возможно нечеткое). Это может быть $2^D i$ – множество всех четких подмножеств D_i, или σ-алгебра, если в качестве реализации признака x_i рассматривается случайная величина. С каждой вершиной дерева решений связывается один единственный признак. Мы будем говорить, что данная вершина проверяет этот признак. Веса дуг, исходящих из некоторой вершины нечеткого дерева, связанной с i-м признаком, есть элементы $\Delta(D_i)$ (рисунок 1).

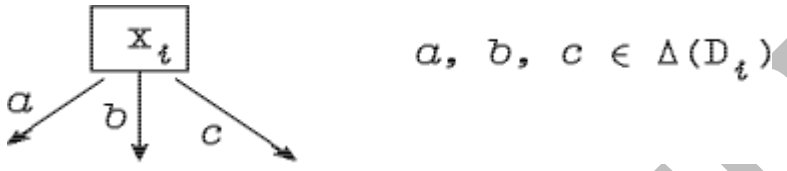

Рис.1 – Вершина дерева

Отличительной чертой деревьев решений является то, что каждый пример определенно принадлежит конкретному узлу. В нечетком случае это не так. Для каждого атрибута необходимо выделить несколько его лингвистических значений и определить степени принадлежности примеров к ним. Вместо количества примеров конкретного узла нечеткое дерево решений группирует их степень принадлежности. Коэффициент – это соотношение примеров $D_j \in S^N$ узла N для целевого значения i, вычисляемый как:

$$P_i^N = \sum_{S^N} \min(\mu_N(D_j), \mu_i(D_j)), \qquad (1)$$

где $\mu_N(D_j)$ – степень принадлежности примера D_j к узлу N,
$\mu_i(D_j)$ – степень принадлежности примера относительно целевого значения i,
S^N – множество всех примеров узла N.

Затем находим коэффициент $P^N = \sum_i P_i^N$, обозначающий общие характеристики примеров узла N.

В стандартном алгоритме дерева решений определяется отношение числа примеров, принадлежащих конкретному атрибуту, к общему числу примеров.

Для нечетких деревьев используется отношение $\frac{P_i^N}{P^N}$, для расчета которого учитывается степень принадлежности.

Выражение

$$E(S^N) = -\sum_i \frac{P_i^N}{P^N} \cdot \log_2 \frac{P_i^N}{P^N} \qquad (2)$$

даёт оценку среднего количества информации для определения класса объекта из множества P^N.

Действительно, для получения $E(S^N)$ каждое значение $logp_i$ (логарифм вероятности i-го состояния) со знаком минус множится на вероятность этого состояния и всё такие произведения складываются. Естественно каждое отдельное слагаемое – $logp_i$ рассматривать как частную информацию, получаемую от отдельного сообщения, состоящего в том, что система X находится в состоянии x_i.

На следующем шаге построения нечеткого дерева решений алгоритм вычисляет энтропию для разбиения по атрибуту A со значениями a_j:

$$E(S^N, \mathrm{A}) = \sum_j \frac{P^{N|_j}}{P^N} \cdot E(S^{\mu_j}), \quad (3)$$

где узел $N|j$ – дочерний для узла N.

Алгоритм выбирает атрибут $A^{\times}$ с максимальным приростом информации:

$$G(S^N, \mathrm{A}) = E(S^N) - E(S^N, \mathrm{A}), \quad (4)$$

$$A^{\times} = \arg\max_A G(S, A). \quad (5)$$

Узел N разбивается на несколько подузлов $N|j$. Степень принадлежности примера D_k узла $N|j$ вычисляется пошагово из узла N как

$$\mu_{N|j}(e^k) = \min(\mu_{N|j}(D_k), \mu_{N|j}(D_k, \mathrm{a_j})), \quad (6)$$

где $\mu_i(D_k,a_j)$ показывает степень принадлежности D_k к атрибуту a_j. Подузел $N|j$ удаляется, если все примеры в нем имеют степень принадлежности, равную нулю.

Алгоритм повторяется до тех пор, пока все примеры узла не будут классифицированы, либо пока не будут использованы для разбиения все атрибуты.

Принадлежность к целевому классу для новой записи находится по формуле

$$\delta_j = \frac{\sum_l \sum_k P_k^l \cdot \mu_l(D_j) \cdot \chi_k}{\sum_l (\mu_l(D_j) \cdot \sum_k P_k^l)}, \quad (7)$$

где P_k^l – коэффициент соотношения примеров листа дерева l для значения целевого класса k, $\mu_l(D_j)$ – степень принадлежности примера к узлу l, χ_k – принадлежность значения целевого класса k к положительному значению исхода классификации.

1.3 Построение нечеткой экспертной системы в CubiCalc v. 2.0

Нечеткие экспертные системы используют представление знаний в форме нечетких продукций и лингвистических переменных. Основу представления лингвистической переменной составляет терм с функцией принадлежности.

Способ обработки знаний в НЭС – это логический вывод по нечетким продукциям. Особенностью НЭС является способ извлечения функций принадлежности, который сводится либо к статистическим методам построения, либо к методу экспертных оценок.

Нечеткие правила вывода образуют базу правил. Важно то, что в НЭС, в отличие от традиционной работают все правила одновременно, но степень их влияния на выход может быть различной. Принцип вычисления суперпозиции многих влияний на окончательный результат лежит в основе нечетких экспертных систем.

Процесс обработки нечетких правил вывода в экспертной системе состоит из 4 этапов;

- вычисление степени истинности левых частей правил (между "Если" и "То") – определение степени принадлежности входных значений нечетким подмножествам, указанным в левой части правил вывода;
- модификация нечетких подмножеств, указанных в правой части правил вывода (после "То"), в соответствии со значениями истинности, полученными на первом этапе;
- объединение (суперпозиция) модифицированных подмножеств;
- скаляризация результата суперпозиции - переход от нечетких подмножеств к скалярным значениям.

Для определения степени истинности левой части каждого правила нечеткая экспертная система вычисляет значения функций принадлежности нечетких подмножеств от соответствующих значений входных переменных. Например, для правила (а) определяется степень вхождения конкретного значения переменной *цена* в нечеткое подмножество *велика***,** то есть истинность предиката "цена велика". К вычисленным значениям истинности могут применяться логические операции. Наиболее часто используются следующие определения операций нечеткой логики:

truth(НЕ x) = 1 - *truth(x)*,
truth(x И y) = min{truth(x), truth(y)},
truth(x ИЛИ y) = max{truth(x), truth(y)},

где *x* и *y* – высказывания; *truth(z)* – степень истинности высказывания *z*. Полученное значение истинности используется для модификации нечеткого множества, указанного в правой части правила. Для выполнения такой модификации используют один из двух методов: "минимума" (correlation-min encoding) и "произведения" (correlation-product encoding). Первый метод ограничивает функцию принадлежности для множества, указанного в правой части правила, значением истинности левой части. Результат выполнения правила – нечеткое множество.

CubiCalc v. 2.0 – программа фирмы HyperLogic, является одной из наиболее мощных экспертных систем на основе нечёткой логики. Пакет содержит

интерактивную оболочку для разработки нечётких экспертных систем и систем управления.

Фактически пакет CubiCalc 2.0 представляет собой своего рода экспертную систему, в которой пользователь задает набор правил типа «если-то», а система пытается на основе этих правил адекватно реагировать на параметры текущей ситуации. Отличие состоит в том, что вводимые правила содержат нечёткие величины, т.е. имеют вид «Если *X* принадлежит *A*, То *Y* принадлежит *B*», где *A* и *B* – нечёткие множества. Аппарат нечёткой логики, заложенный в CubiCalc, дает возможность оперировать нечёткими понятиями как точными и строить на их основе целые логические системы.

Процесс выполнения проекта в CubiCalc состоит из нескольких этапов:

1. инициализация (Initialization);
2. ввод данных (Input);
3. предобработка (Preprocessing);
4. выполнение правил (Rules Execution);
5. постобработка (Postprocessing);
6. вывод данных (Output);
7. моделирование (Simulation).

Процесс разработки нечёткой экспертной системы в пакете CubiCalc начинается с определения *переменных*, которые будут использоваться в проекте.

В CubiCalc доступны несколько типов переменных:

1) Fuzzy Input – входная переменная, на области значения которой задаются нечёткие множества, используемые в левой части правил вывода.

2) Fuzzy Output – результат работы системы нечёткого логического вывода, на интервале ее изменения определяются нечёткие множества, используемые в правой части правил вывода.

3) Constant – переменная с фиксированным значением.

4) Temporary – переменная, принимающая действительные значения (данному типу переменных соответствует тип «float»).

На шаге инициализации производится присвоение начальных значений переменным, а также выполнение действий, предусмотренных пользователем, для подготовки нечеткой экспертной системы к работе. Последующие этапы повторяются циклически до тех пор, пока не будет выполняться условие завершения обработки. На шаге ввода данных CubiCalc, если требуется, получает значения из входного файла или берет значения, введенные пользователем. В системе реализован произвольный доступ к файлам, состоящим из записей. Предобработка используется для преобразования введенных данных, изменения значений переменных.

Этап выполнения правил нечеткого вывода – основа функционирования CubiCalc. Он предназначен для вычисления выхода базы правил в зависимости от текущих значений входных переменных. Последовательность действий, выполняемых на этом этапе, традиционна для подобных систем. CubiCalc позволяет выбрать методы суперпозиции и скаляризации нечетких множеств, которые будут использованы в процессе логического вывода.

На стадии постобработки реализуются операции преобразования результатов вывода, присвоения значений внутренним переменным, используемым для записи результатов в файл или представления их в графическом виде на этапе вывода данных. Кроме того, на этой стадии могут модифицироваться условия окончания выполнения проекта. Этап моделирования используется для получения отклика внешней системы на данные, генерируемые нечеткой экспертной системой.

2. Практические применения

2.1 Построение нечеткого дерева решений

В таблице 2 представлены данные о семи клиентах банка: проживание в регионе (в годах), доход (в денежных единицах) и рейтинг выдачи ему кредита (определяется экспертом).

Необходимо построить нечеткое дерево решений, с помощью которого определить рейтинг выдачи кредита для клиента, который проживает в регионе 25 лет, и доход его составляет 32 000 ден.единиц.

Табл. 2 – Данные о клиентах банка

№	Проживание в регионе	Доход	Рейтинг
D1	0	10 000	0,0
D2	10	15 000	0,0
D3	15	20 000	0,1
D4	20	30 000	0,3
D5	30	25 000	0,7
D6	40	35 000	0,9
D7	40	50 000	1,0

Определим лингвистические переменные.

1. x_1: «проживание в регионе»;
   X: [0, 100];
   T(x): «временно», «продолжительно», «постоянно»;
   G: «достаточно», «недостаточно»;
   M: задано таблично (таблица 3).
2. x_2: «доход»;
   X: [0, 100 000];
   T(x): «малый», «средний», «высокий»;
   G: «достаточно», «недостаточно»;
   M: задано таблично (таблица 4).

Табл. 3 – Табличное представление семантического правила для x_1

N	Проживание в регионе		
	временно	продолжительно	постоянно
D1	1,0	0,0	0,0
D2	0,8	0,2	0,0
D3	0,5	0,5	0,0
D4	0,2	0,8	0,0
D5	0,0	0,5	0,5
D6	0,0	0,0	1
D7	0,0	0,0	1

Табл. 4 – Табличное представление семантического правила для x_2

N	Доход		
	малый	средний	высокий
D1	1,0	0,0	0,0
D2	0,6	0,4	0,0
D3	0,1	0,9	0,0
D4	0,0	1,0	0,0
D5	0,0	1,0	0,0
D6	0,0	0,6	0,4
D7	0,0	0,0	1,0

Общий вид функции принадлежности лингвистических переменных показан на рисунке 2.

Необходимо найти значение общей энтропии:

$P_{да} = 0 + 0 + 0{,}1 + 0{,}3 + 0{,}7 + 0{,}9 + 1{,}0 = 3,$

$P_{нет} = 1 + 1 + 0{,}9 + 0{,}7 + 0{,}3 + 0{,}1 + 0 = 4,$

$P = P_{да} + P_{нет} = 3 + 4 = 7.$

Рассчитаем $E(S^N)$, воспользовавшись формулой (2):

$$E(S^N) = -\frac{3}{7}\log_2\frac{3}{7} - \frac{4}{7}\log_2\frac{4}{7} \approx 0{,}985 \text{ бит.}$$

Теперь рассчитаем $E(S^N$, *проживание в регионе*$)$.

$P_{да}^{временно}$ =min(0;1)+min(0;0,8)+min(0,1;0,5)+min (0,3;0,2)+
+min(0,7;0)+min(0,9;0)+min(1;0)=0+0+0,1+0,2+
+ 0 + 0 + 0 = 0,3.

$P_{нет}^{временно}$ =min(1;1)+min(1;0,8)+min(0,9;0,5)+min(0,7;0,2)+
+min(0,3;0)+min(0,1;0)+min(0;0)=1+0,8+0,5+0,2
+ 0 + 0 + 0 = 2,5.

$P^{временно} = 0{,}3 + 2{,}5 = 2{,}8.$

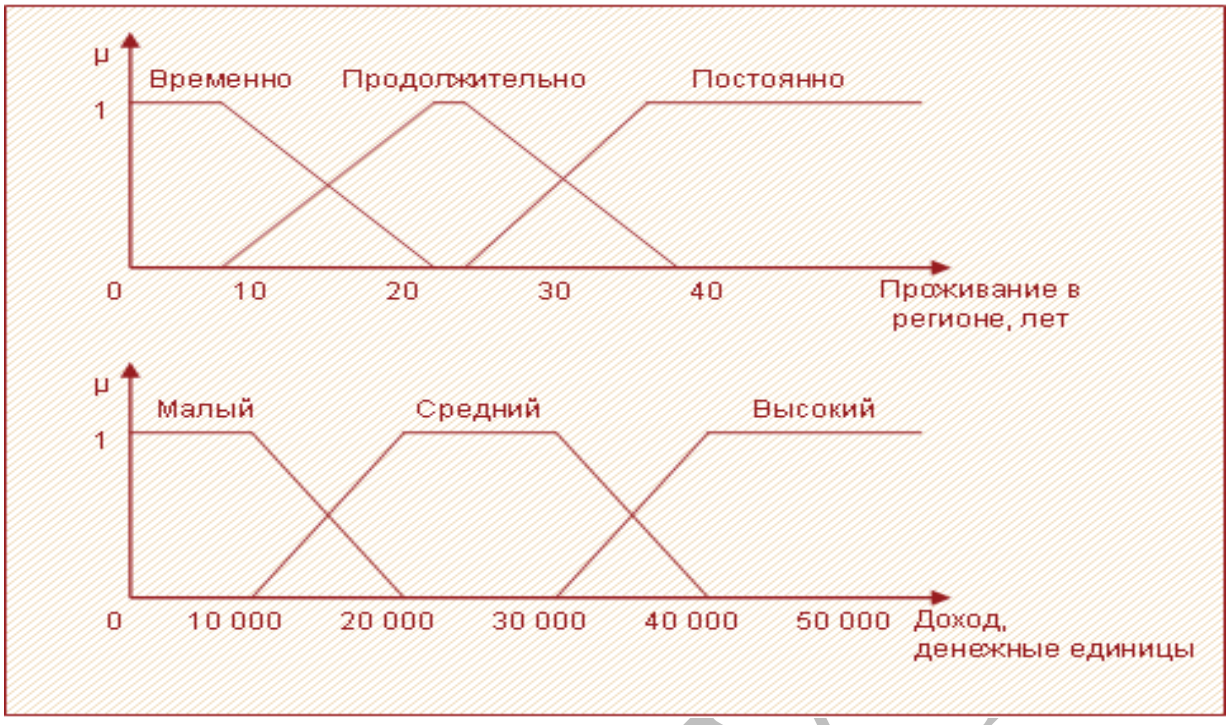

Рис. 2 – Графики функций принадлежности

$$E(\text{проживание\_в\_регионе}, \text{временно}) =$$

$$= -\frac{0,3}{2,8}\log_2\frac{0,3}{2,8} - \frac{2,5}{2,8}\log_2\frac{2,5}{2,8} \approx 0,491 \text{ бит.}$$

Для продолжительного и постоянного проживания в регионе проводятся аналогичные вычисления. Результат сведем в таблицу 5.

Табл. 5 – Итоги расчетов для x_1

	временно	продолжительно	постоянно
$P_{да}$	0,3	0,9	2,4
$P_{нет}$	2,5	1,7	0,4
E, бит	0,491	0,931	0,592

Найдем энтропию, воспользовавшись формулой (3):

$$E(S^N, \text{проживание\_в\_регионе}) =$$

$$= \frac{2,5}{7}\cdot 0,491 + \frac{1,7}{7}\cdot 0,931 + \frac{0,4}{7}\cdot 0,592 \approx 0,486 \text{ бит.}$$

Рассчитаем прирост информации для данного атрибута.

$$G(S^N, \text{проживание\_в\_регионе}) =$$

= 0,985 – 0,486 = 0,499 бит.

Проводя подобные вычисления для лингвистической переменной "доход", получаем (таблица 6).

Табл. 6 – Итоги расчетов для x_2

	малый	средний	высокий
$P_{да}$	0,1	1,7	1,4
$P_{нет}$	1,7	2,4	0,1
E, бит	0,310	0,979	0,353

$E(S^N, доход) = 0{,}416$ бит,

$G(S^N, доход) = 0{,}569$ бит.

Максимальный прирост информации обеспечивает атрибут "доход", следовательно, разбиение начнется с него.

На следующем шаге алгоритма необходимо для каждой записи рассчитать степень принадлежности к каждому новому узлу по формуле (6). Результат представлен в таблице 7.

К узлам [проживание в регионе = временно и доход = высокий], [проживание в регионе = продолжительно и доход = высокий], [проживание в регионе = постоянно и доход = средний] и [проживание в регионе = постоянно и доход = малый] не принадлежит ни одна запись, поэтому они удаляются из дерева.

Для каждого узла находятся коэффициенты P_i^N.

1) Узел [проживание в регионе = временно и доход = малый]:

$P_{да}$= min(0;1)+min(0;0,6)+min(0,1;0,1)+min (0,3;0)+
min(0,7;0)+min(0,9;0)+min(1;0) =
= 0 + 0 + 0,1 + 0 + 0 + 0 + 0 = 0,1.

$P_{нет}$= min(1;1)+min(1;0,6)+min(0,9;0,1)+min(0,7;0)+
+min(0,3;0)+min(0,1;0)+min(0;0)=
= 1 + 0,6 + 0,1 + 0 + 0 + 0 + 0 = 1,7.

2) Узел [проживание в регионе = продолжительно и доход = малый]:

$P_{да}$= min(0;0)+min(0;0,2)+min(0,1;0,1)+min (0,3;0)+
min(0,7;0)+min(0,9;0)+min(1;0) =
= 0 + 0 + 0,1 + 0 + 0 + 0 + 0 = 0,1.

$P_{нет}$= min(1;0)+min(1;0,2)+min(0,9;0,1)+min(0,7;0)+
+min(0,3;0)+min(0,1;0)+min(0;0)=
= 0 + 0,2 + 0,1 + 0 + 0 + 0 + 0 = 0,3.

3) Узел [проживание в регионе = временно и доход = средний]:

$P_{да}$= min(0;0)+min(0;0,4)+min(0,1;0,5)+min (0,3;0,2)+
min(0,7;0)+min(0,9;0)+min(1;0) =
= 0 + 0 + 0,1 + 0,2 + 0 + 0 + 0 = 0,3.

$P_{нет}$= min(1;0)+min(1;0,4)+min(0,9;0,5)+min(0,7;0,2)+
+min(0,3;0)+min(0,1;0)+min(0;0)=
= 0 + 0,4 + 0,5 + 0,2 + 0 + 0 + 0 = 1,1.

4) Узел [проживание в регионе = продолжительно и доход = средний]:

$P_{да}$= min(0;0)+min(0;0,2)+min(0,1;0,5)+min (0,3;0,8)+
min(0,7;0,5)+min(0,9;0)+min(1;0) =

= 0 + 0 + 0,1 + 0,3 + 0,5 + 0 + 0 = 0,9.

$P_{нет}$= min(1;0)+min(1;0,2)+min(0,9;0,5)+min(0,7;0,8)+
+min(0,3;0,5)+min(0,1;0)+min(0;0)=
= 0 + 0,2 + 0,5 + 0,7 + 0,3 + 0 + 0 = 1,7.

5) Узел [проживание в регионе = постоянно и доход = высокий]:

$P_{да}$= min(0;0)+min(0;0)+min(0,1;0)+min (0,3;0)+
min(0,7;0)+min(0,9;0,4)+min(1;1) =
= 0 + 0 + 0 + 0 + 0 + 0,4 + 1 = 1,4.

$P_{нет}$= min(1;0)+min(1;0)+min(0,9;0)+min(0,7;0)+
+min(0,3;0)+min(0,1;0,4)+min(0;1)=
= 0 + 0 + 0 + 0 + 0 + 0,1 + 0 = 0,1.

Полученное дерево представлено на рисунке 3.

Теперь определим кредитный рейтинг для клиента, проживающего в регионе 25 лет, и с доходом 30 000.

За положительный исход в данной задаче принято одобрение в выдаче кредита, поэтому $\chi_{да} = 1,0$, $\chi_{нет} = 0,0$.

Табл. 7 – Принадлежность записей новым узлам дерева

Доход	малый			средний			высокий		
Проживание в регионе	временно	продолжительно	постоянно	временно	продолжительно	постоянно	Временно	продолжительно	постоянно
D1	1	0	0	0	0	0	0	0	0
D2	0,6	0,2	0	0,4	0,2	0	0	0	0
D3	0,1	0,1	0	0,5	0,5	0	0	0	0
D4	0	0	0	0,2	0,8	0	0	0	0
D5	0	0	0	0	0,5	0	0	0	0
D6	0	0	0	0	0	0	0	0	0,4
D7	0	0	0	0	0	0	0	0	1

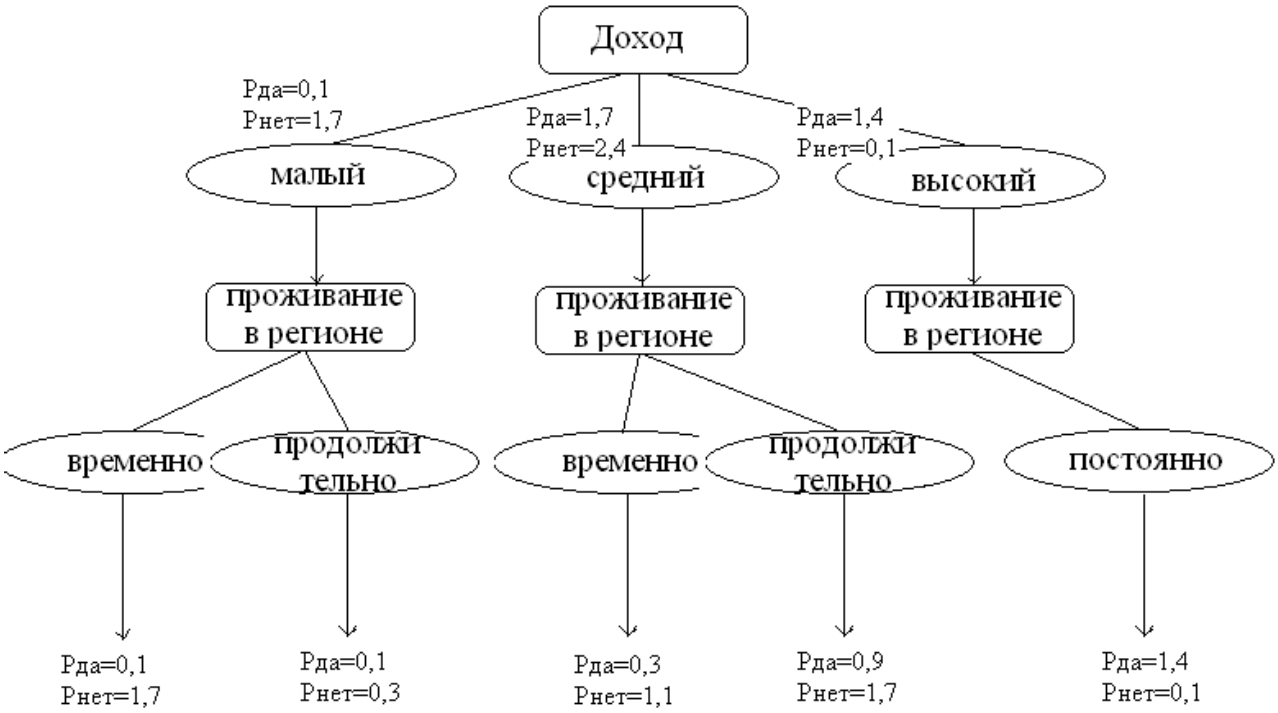

Рис. 3 – Графическая иллюстрация полученного нечеткого дерева решений

Новый клиент принадлежит к двум узлам: [проживание в регионе = продолжительно и доход = средний] и [проживание в регионе = постоянно и доход = высокий], со степенями 0,8 и 0,2 соответственно. Подставляя полученные значения в формулу (7), рассчитываем кредитный рейтинг:

$$\delta = \frac{0{,}9 \cdot 0{,}8 \cdot 1{,}0 + 1{,}7 \cdot 0{,}8 \cdot 0{,}0 + 1{,}4 \cdot 0{,}2 \cdot 1{,}0 + 0{,}1 \cdot 0{,}2 \cdot 0{,}0}{(0{,}9 + 1{,}7) \cdot 0{,}8 + (1{,}4 + 0{,}1) \cdot 0{,}2} = 0{,}353 .$$

В итоге мы получили кредитный рейтинг, равный 0,353. Он означает, что степень принадлежности записи к тому, что кредит клиенту будет выдан, равна 0,353, а к невыдаче – 0,647. Следовательно, этому клиенту банком будет отказано.

2.2 Построение нечеткой экспертной системы в CubiCalc 2.0

Процесс разработки нечёткой экспертной системы в пакете CubiCalc начинается с определения *переменных*, которые будут использоваться в проекте *project→variables→new*.

Объявляем первую переменную x_1 как *Prozh* (проживание в регионе), задаем атрибуты переменной: выберем ее тип (Fuzzy Input), зададим диапазон изменения ее значений [0, 50] (Range Low – нижняя граница диапазона, Range High – верхняя граница), а также зададим начальное значение Initial Value равное 0.

Объявляем вторую переменную x_2 как *Doxod* (доход) (рисунок 4), задаем атрибуты переменной: выберем ее тип (Fuzzy Input), зададим диапазон

изменения ее значений [0, 50000], а также зададим начальное значение Initial Value равное 0.

Объявляем третью (выходную) переменную *Ранг* как *Rang*, задаем атрибуты переменной: выберем ее тип (Fuzzy Output), зададим диапазон изменения ее значений [0, 1], а также зададим начальное значение Initial Value равное 0.

Далее необходимо задать характеристики всех трех переменных *Project→ Adjective editor*.

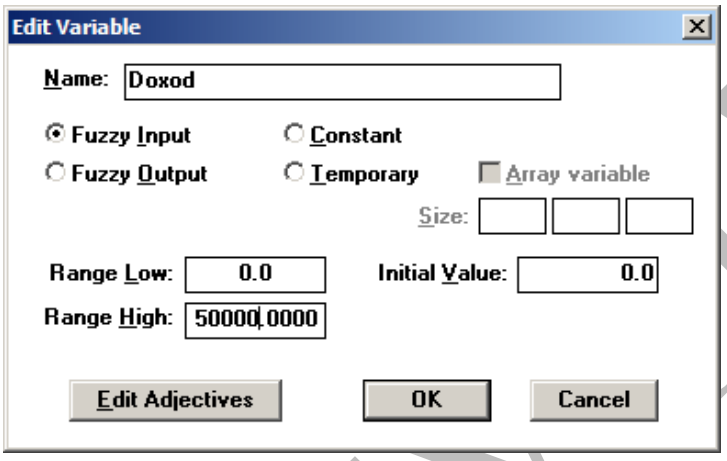

Рис. 4 – Определение переменной *Doxod*

Выбрав переменную *Doxod* в окне *«Adjectives for variables»*, нажмите кнопку *Edit*. Здесь каждой входной и выходной переменной поставим в соответствие набор функций принадлежности *Adjective→ Change List→New*. В появившемся окне *«Create Adjective(s)»* зададим следующие параметры: количество функций принадлежности (Number) равное 3; вид функции принадлежности (Shape) – Trapezoid; ширина основания (base width) равная 45,0 (рисунок 5).

В окне *«Edit Adjective List»* присвоим наименования – *Small, Medium, Large* соответственно малому, среднему и высокому доходу (рисунок 6).

Для переменной *Prozh* присвоим наименования *Vrem* (временно), *Prod* (продолжительно), *Post* (постоянно).

Для переменной *Rang* присвоим наименования *Small* (низкий), *Medium* (средний), *Large* (высокий).

Далее необходимо определить набор правил, которые связывают входные переменные с выходными. Для этого в редакторе правил вывода *Project→Rules* определим:

IF *Prozh* is *Vrem* AND *Doxod* is *Small*
THEN *Rang* is *Small*;
IF *Prozh* is *Prod* AND *Doxod* is *Large*
THEN *Rang* is *Large*;
IF *Prozh* is *Prod* AND *Doxod* is *Medium*
THEN *Rang* is *Medium*;
IF *Prozh* is *Post* AND *Doxod* is *Small*
THEN *Rang* is *Medium*;

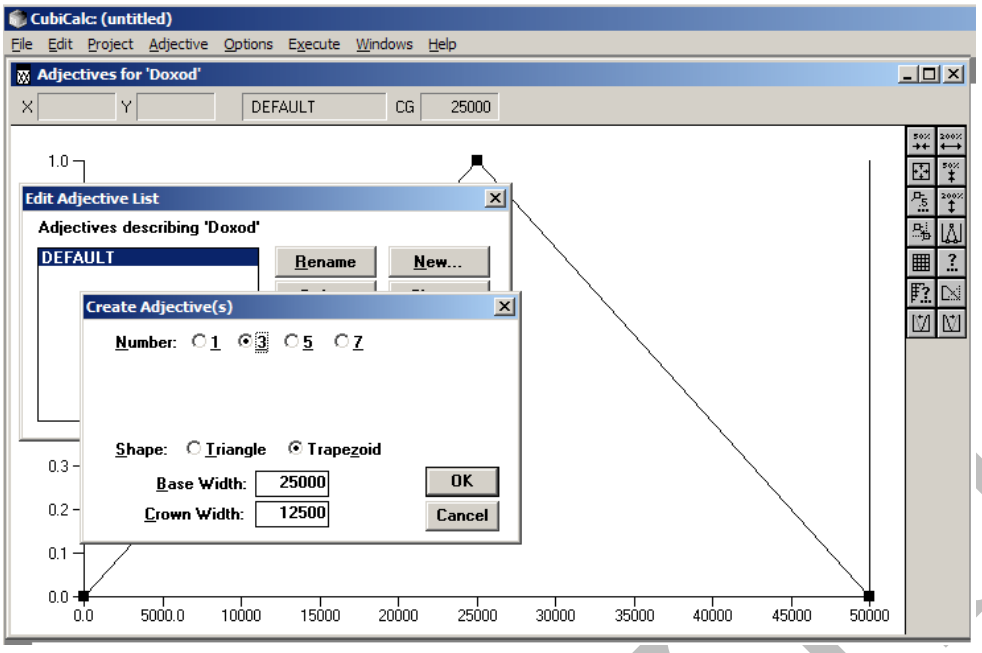

Рис. 5

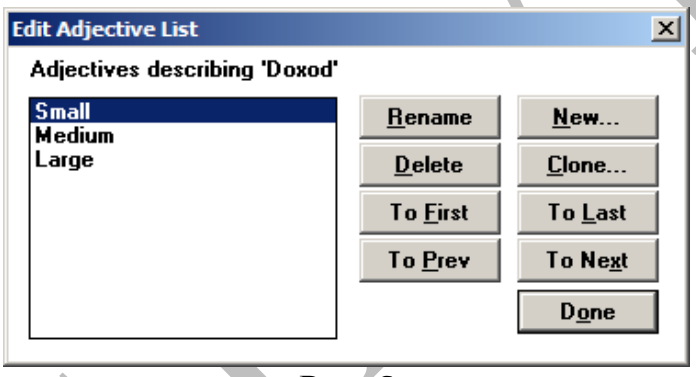

Рис. 6

Следующими этапами выполнения проекта в CubiCalc являются предобработка и постобработка. Предобработка (*Project→Preprocessing*) задает входные данные. Для этого используется функция field**()**, которая возвращает данные из отдельного поля текущей записи.

Откройте *Preprocessing Editor* (Редактор предобработки) и вставьте нижеследующие инструкции в его окно редактора:

Prozh = field(1);
Doxod = field(2);
check = field(3);

В постобработке (*Project→Postprocessing*) задается метод автоматической оценки эффективности системы. Необходимо посчитать количество сделанных системой ошибок и вывести сумму в конце запуска. Чтобы обнаружить ошибку классификации в постобработке, необходимо сравнить выходную переменную с информацией *check* в файле данных.

Чтобы это сделать, вставьте нижеследующие инструкции в окно редактора:

```
If (Rang <> check) errors += 1;end
If (endoffile=TRUE) call message(errors); end
```

В процессе настройки системы мы использовали переменные, отличные от входных и выходных. Для обнаружения ошибок классификации мы использовали переменную *check* и переменную *errors* для подсчета количества ошибок классификации. Теперь мы должны создать эти переменные.

Откройте *Variables Editor* с помощью меню *Variables*. Появится диалоговое окно, содержащее имена уже созданных входных и выходных переменных. Нажмите кнопку *New* для создания новой переменной.

В окне *Edit Variable* определите переменные: *check* (тип переменной – Temporary, начальное значение 0.0); *errors* (тип переменной – Temporary, начальное значение 0.0).

Настройка входного файла

Для присоединения входного файла к проекту, необходимо создать файл с расширением .txt (рисунок 7), задать имя файла (*File→Input File*) и некоторые его атрибуты (рисунок 8).

Укажите *Text* для определения файла как текстового файла, содержащего печатаемые данные. Задайте количество *Fields per Record* (Полей на запись), равное 3 (каждая запись содержит значения для *Prozh, Doxod* и *check*) и количество *Lines per Record* (Строк на запись), равное 1 (каждая запись занимает одну текстовую строку).

Рис. 7

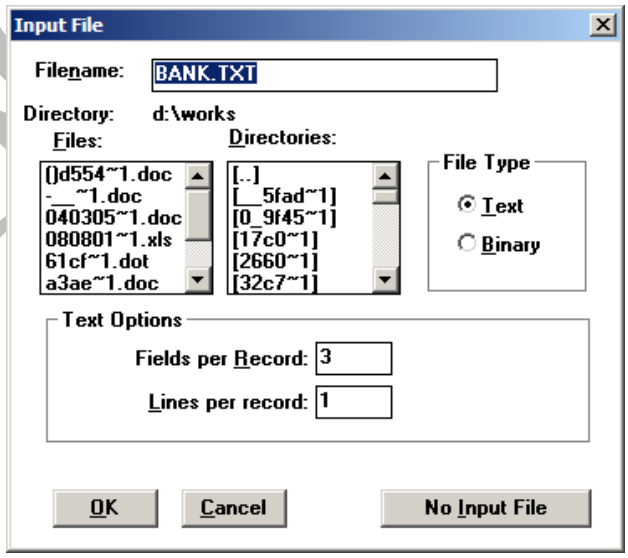

Рис. 8

Далее необходимо определить, какая информация должна отображаться при запуске системы (*Project→Log Format*). Выберем нажатием кнопки *«Add»* из левого списка переменные *Prozh, Doxod, Rang* и *check*. Убедимся в

непротиворечивости переменных (*Execute→Check Definitions*), снимем флажки *Initialization, Simulation.*

Проверим работу системы, запустив сценарий *Execute→Run.* Появится окно, содержащее значения переменных и их порядок, выбранные в *Log Format Editor*. Через некоторое время CubiCalc дойдет до конца входного файла и появится итоговое сообщение, заданное в предобработке (рисунок 9).

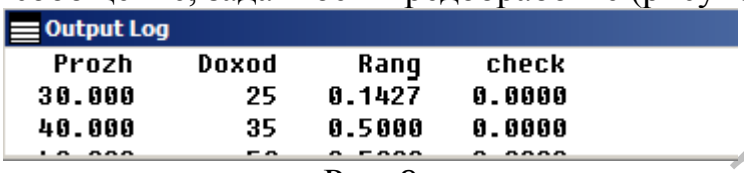

Рис. 9

Выполните *Execute→Terminate* для остановки сценария.

3. Структура пояснительной записки к курсовому проекту

Титульный лист
Лист рецензии
Введение
Лист задания
Анализ предметной области

1. Построение нечеткого дерева решений (согласно варианту, выданному преподавателем).
    1.1 Определение лингвистических переменных.
    1.2 Построение функций принадлежности.
    1.3 Расчет $E(S^N)$, $G(S^N)$.
    1.4 Расчет степеней принадлежности к каждому новому узлу.
    1.5 Расчет принадлежности новой записи к целевому классу.

2. Построение нечеткой экспертной системы в программном пакете CubiCalc 2.0 (согласно варианту, выданному преподавателем).
    2.1 Определение переменных Fuzzy Input, Output.
    2.2 Построение функций принадлежности.
    2.3 Определение набора правил, связывающих входные переменные с выходными.
    2.4 Настройка входного файла.
    2.5 Проверка работы системы.

Заключение
Список использованных источников
Приложения

4. Варианты заданий на курсовое проектирование

Задача (результат работы системы нечёткого логического вывода) – выбор наиболее подходящей альтернативы.

Вариант 1 Автомойка. Руководству компании необходимо сформировать ценовую политику компании.

Вариант 2 Банк. Руководством банка разрабатываются кредитные программы для различных категорий клиентов.

Вариант 3 Букмекерская контора. Руководство фирмы рассматривает кандидатов на замещение вакантной должности бухгалтера.

Вариант 4 Туристическое агентство. Руководство фирмы оценивает различные туристические направления.

Вариант 5 Риэлтерское агентство. Менеджеру компании необходимо оценить недвижимость в зависимости от различных параметров.

Вариант 6 Пассажирское автопредприятие. Руководству необходимо определить степень риска банкротства компании.

Вариант 7 Инвестиционная компания. Руководству необходимо оценить инвестиционный риск.

Вариант 8 Адвокатская контора. Руководство компании приняло решение о приобретении поддержанного автомобиля.

Вариант 9 Магазин спортивных товаров. Руководству компании необходимо определить место организации магазина представленного направления.

Вариант 10 Строительная компания. Руководству компании необходимо выбрать инвестора строительного объекта.

Вариант 11 Логистическая компания. Необходимо категоризировать клиентов компании по возможным вариантам предоставления отсрочки оплаты товаров.

Вариант 12 Банк. Руководству необходимо определить возможность выдачи кредита для клиента.

Вариант 13 Спортивная секция. Необходимо спрогнозировать результаты футбольных матчей команд секции.

Вариант 14 Кинотеатр. Необходимо оценить возможное время кинопоказов в зависимости от параметров фильма/мультфильма.

Вариант 15 Риэлтерское агентство. Необходимо оценить диапазон прибыли, на который можно рассчитывать в следующем году.

Список рекомендуемой литературы

1. Андрейчиков А.В., Андрейчикова О.Н. Анализ, синтез, планирование решений в экономике. – М.:Финансы и статистика, 2002. – 368 с.: ил.
2. Жданова, Е.И. Методические указания к выполнению лабораторных работ по дисциплине «Проектирование баз данных и баз знаний» / Е.И.Жданова, Ю.В.Трошин, Р.Р.Халимов. – ПГУТИ, 2011.
3. Жиров В.Г. Графическое представление и анализ нечеткой модели логического вывода в базе знаний информационной системы. – Самара, 2010.
4. Матвеев М.Г., Свиридов А.С., Алейникова Н.А. Модели и методы искусственного интеллекта. Применение в экономике. – М.: Инфра-М, 2008. – 448 с.: ил.
5. Пономарев, А.С. Нечёткие множества в задачах автоматизированного управления и принятия решений: Учебное пособие. – Харьков, 2005.
6. Рутковская, Д. Нейронные сети, генетические алгоритмы и нечёткие системы/ Д. Рутковская, М. Пилиньский, Л. Рутковский; пер. с польск. И. Д. Рудинского.- М.: Горячая линия-Телеком, 2004.- 452 с.: ил.
7. Штовба, С.Д. Введение в теорию нечетких множеств и нечеткую логику. Сообщество пользователей Matlab и Simulink. Режим доступа: http//matlab.exponenta.ru, свободный.
8. Ярушкина, Н. Г. Основы теории нечётких и гибридных систем: учеб. пособие для вузов/ Н. Г. Ярушкина.- М.: Финансы и статистика, 2004.- 320 с.: ил.
9. Яхъяева, Г. Э. Нечёткие множества и нейронные сети: учеб. пособие/ Г.Э.Яхъяева. – М.: Интернет Ун-т Информ. Технологий: БИНОМ. Лаб. знаний, 2006.- 316 с.: ил.

Методические материалы

Жданова Е.И.

ПРОЕКТИРОВАНИЕ БАЗ ДАННЫХ И БАЗ ЗНАНИЙ

Методические указания для выполнения курсового проекта