Министерство образования Республики Беларусь
УО «Полесский государственный университет»

В.Н. ШТЕПА, Р.Е. КОТ

РАСПРЕДЕЛЕННЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ. НЕЙРОСЕТЕВЫЕ ТЕХНОЛОГИИ

Методические рекомендации по выполнению лабораторных работ для студентов по специальности 1-40 05 01 «Информационные системы и технологии (по направлениям)»

Пинск
ПолесГУ
2017

УДК 004.8(075.8)
ББК 32.973.202я73
Ш89

Р е ц е н з е н т ы:

кандидат технических наук, доцент Ю.М. Вишняков;
кандидат технических наук, доцент Д.В. Кузёмкин

У т в е р ж д е н о
научно-методическим советом ПолесГУ

Ш89 Штепа, В.Н.
Распределённые информационные системы. Нейросетевые технологии : методические рекомендации по выполнению лабораторных работ / В.Н. Штепа, Р.Е. Кот. – Пинск : ПолесГУ, 2017. – 40 с.

ISBN 978-985-516-505-8

Пособие предназначено для студентов специальности 1-40 05 01 «Информационные системы и технологии» (по направлениям).

УДК 339.138(076.5)
ББК 65.291.3я73

ISBN 978-985-516-505-8

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

С точки зрения системного анализа, распределённые информационные системы (РИС) – это сложные технические системы, которые функционируют в условиях действия случайных факторов, при активном взаимодействии с внешней средой и наличии негативных воздействий различной природы, высокой стоимости последствий возможных нарушений или ошибок в работе системы. РИС имеют свою цель функционирования, большое количество взаимодействующих элементов, блоков, подсистем, сложную иерархическую систему управления. Они характеризуются также сложностью поведения и функций, которыми выполняются, постоянным ростом количества пользователей и подключенного оборудования, постоянной модификацией составляющих, что приводит к фактической невозможности построения адекватной математической модели для исчерпывающего описания функционирования системы. Рост сложности структуры и функционирования РИС обусловливает появление таких свойств, как естественная избыточность, адаптивность, надежность, устойчивость к отказам, устойчивость к внешним воздействиям, живучесть.

Интенсификация процессов информационного взаимодействия, усиленный развитие территориально распределенных систем, широкое использование ресурсов сети Интернет выдвигает новые требования к технологиям работы в РИС, технологий разработки таких систем, обеспечения безопасности их функционирования. Это требует развития безопасных технологий работы с информационными ресурсами, внедрение методов и средств повышения живучести РИС с применением решений систем искусственного интеллекта, способных работать в условиях неопределённости и размытости данных.

Таким функциональным требованиям соответствует математический аппарат нейронных сетей (НС). Под термином «нейронные сети» понимают вычислительные структуры, моделирующие простые биологические процессы, конечно ассоциированные с процессами человеческого мозга. Они являют собой распараллеленные системы, способные к обучению путем анализа информации. Элементарным преобразователем в этих сетях является искусственный нейрон или просто нейрон, названный так по аналогии с биологическим прототипом.

Термин «нейронные сети» сформировался в 40-х годах XX в. среди исследователей, изучавших принципы организации и функционирования биологических нейронных сетей. Основные результаты, полученные в этой сфере, связаны с именами американских исследователей В. Маккалоха, Д. Хебба, Ф. Розенблатта, М. Минского, Дж. Хопфилда.

Нейронные сети применяются для решения ряда практических задач:

1. Классификация образов.
2. Кластеризация/категоризация.
3. Аппроксимация функций.
4. Предсказания / прогноз.
5. Оптимизация.
6. Управление.

1 ТЕОРЕТИЧЕСКИЕ ОСНОВЫ НЕЙРОННЫХ СЕТЕЙ

Искусственный нейрон – это составная часть искусственной нейронной сети (рис. 1.1). В состав нейрона входят умножители (синапсы), сумматор и нелинейный преобразователь. Синапсы осуществляют связь между нейронами и умножают входной сигнал на число, которое характеризует силу связи – вес синапса.

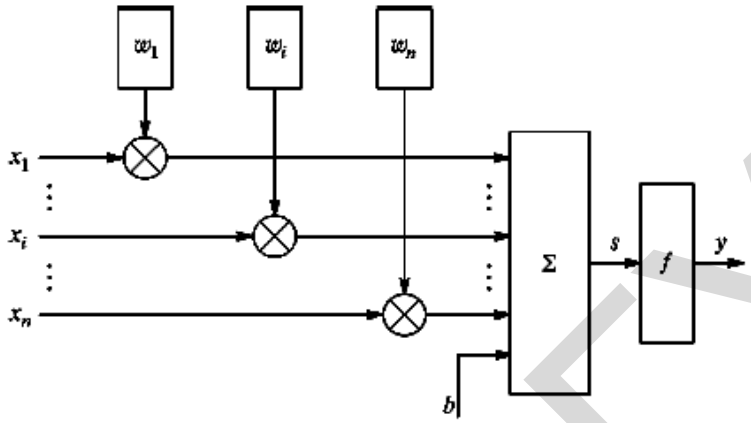


Рис. 1.1 – Структура искусственного нейрона

Сумматор составляет сигналы, поступающие по синаптических связях от других нейронов и внешних входных сигналов. Нелинейный преобразователь реализует нелинейную функцию одного аргумента – выхода сумматора. Эта функция называется «функция активации», или «передаточная функция» нейрона. Нейрон, в целом, реализует скалярную функцию векторного аргумента. Математическая модель нейрона описывается следующим соотношением:

$$s = \sum_{i=1}^{n} \omega_i \cdot x_i + b, \tag{1.1}$$

где ω_i – вес синапса;
$(i = 1,...,n)$;
b – значение смещения;
x_i – компонент входного вектора (входной сигнал) $(i = 1,...,n)$;
s – результат сложения;
y – выходной сигнал нейрона;
n – количество входов нейрона;
f – нелинейное преобразование (функция активации, или передаточная функция).

В общем случае входной сигнал, весовые коэффициенты и значения смещения могут принимать действительные значения. Выход (y) определяется видом функции активации и может быть как реальным, так и целым (**табл. 1.1**). Во многих практических задачах входы, вес и смещения могут принимать только некоторые фиксированные значения.

Синаптические связи с положительными весами называют возбуждающими, с отрицательными массами – тормозящими. Таким образом, нейрон полностью описывается своими весами и передаточной функцией *f (s)*. Получив набор чисел (вектор), нейрон выдает некоторое число на выходе.

Описанный вычислительный элемент (1.1) можно считать упрощенной математической моделью биологических нейронов – клеток, из которых состоит нервная

система человека и животных. Чтобы подчеркнуть отличие нейронов биологических и математических, их иногда называют нейроноподибнимы элементами или формальными нейронами.

Таблица 1.1. Перечень функций активации нейронов

Название	**Формула**	**Диапазон значений**
Пороговая	$f(s)=\begin{cases}0, s<\theta\\1, s\geq\theta\end{cases}$	(0, 1)
Знаковая (сигнатурная)	$f(s)=\begin{cases}1, s>0\\-1, s\leq 0\end{cases}$	(-1, 1)
Сигмоидальная (логистическая)	$f(s)=\frac{1}{1+e^{-s}}$	(0, 1)
Полулинейная	$f(s)=\begin{cases}s, s>0\\0, s\leq 0\end{cases}$	(0, ∞)
Линейная	$f(s)=s$	(-∞, ∞)
Радиально базисная (гаусовская)	$f(s)=e^{-S^2}$	(0, 1)
Гиперболический тангенс	$f(s)=\frac{e^{S}-e^{-S}}{e^{S}+e^{-S}}$	(-1, 1)

Как отмечалось, искусственная нейронная сеть (НС) – набор нейронов, соединенных между собой. Как правило, передаточные (активационные) функции всех нейронов в сети фиксированы, а вес является параметром сети и может меняться.

Некоторые входы нейронов помечены как внешние входы сети, а некоторые выходы как внешние выходы сети. Подавая любые числа на входы, мы получаем некий набор чисел на выходах. Таким образом, работа нейросети состоит в преобразовании входного вектора *X* в выходной вектор *Y*, причем это преобразование задается весами сети.

Практически любую задачу можно свести к такой, которую можно решить с помощью нейросети.

Вопрос построения НС, как правило, решается в два этапа:

1. Выбор типа (архитектура) сети.
2. Подбор весов (обучение) сети.

На первом этапе следует выбрать следующее:

– какие нейроны мы хотим использовать (количество входов, передаточные функции);

– как следует соединить их между собой;

– что взять как входы и выходы сети.

Эта задача, на первый взгляд, кажется неразрешимой, но нам не обязательно придумывать нейросеть «с нуля», ведь существует несколько десятков различных нейросетевых архитектур, причем эффективность многих из них доказана математически.

Самые популярные архитектуры – многослойный персептрон, нейросеть с общей регрессией, сети Кохонена.

На втором этапе нам следует «научить» выбранную сеть, то есть подобрать такие значения ее весов, чтобы сеть работала должным образом. Для многих архитектур разработаны специальные алгоритмы, которые позволяют настроить веса сети.

В зависимости от функций, которые выполняют нейроны в сети, можно выделить три их типа:

– входные нейроны – это нейроны, на которые подается входной вектор, кодирующий входное воздействие или образ внешней среды; в них обычно не осуществляется вычислительных процедур, информация передается со входа на выход нейрона путем изменения его активации;

– выходные нейроны – это нейроны, исходные значения которых представляют выход сети;

– промежуточные нейроны – это нейроны, составляющие основу искусственных нейронных сетей.

В большинстве нейронных моделей тип нейрона связан с его размещением в сети. Если нейрон имеет только выходные связи, то это входной нейрон, если наоборот – выходной. Однако может встретиться случай, когда выход топологически внутреннего нейрона рассматривается как часть выхода сети. В процессе функционирования (эволюции состояния) сети осуществляется преобразование входного вектора в выходной, т.е. некоторая обработка информации.

Вообще говоря, большая часть прикладных задач может быть сведена к реализации некоторого сложного многомерного функционального преобразования. В результате построения такого отображения необходимо добиться того, чтобы:

– обеспечивалось формирование правильных выходных сигналов в соответствии со всеми примерами обучающей выборки;

– обеспечивалось формирование правильных выходных сигналов в соответствии со всеми возможными входными сигналами, которые не вошли в обучающую выборку.

Второе требование в значительной мере усложняет задачу формирования обучающей выборки. В общем виде эта задача сейчас еще не решена, однако во всех известных случаях было найдено отдельное решение. Дальнейшие рассуждения предполагают, что обучающая выборка уже сформирована.

Отметим, что теоретической основой для построения нейронных сетей является теорема о полноте.

Утверждение. *Для любого количества пар входных-выходных векторов произвольной размерности* $\{X_k , Y_k\}$, $k = 1...N$ *существует двухслойная однородная нейронная сеть с последовательными связями, с Сигмоидальными передаточными функциями и конечным количеством нейронов, которая для каждого входного вектора* X_k *формирует соответствующий ему выходной вектор* Y_k.

Таким образом, для представления многомерных функций многих переменных может быть использована однородная нейронная сеть с сигмоидальными передаточными функциями нейронов, имеющая всего один скрытый слой.

В процессе функционирования нейронная сеть формирует выходной сигнал Y в соответствии с сигналом X, реализуя функцию $Y = G(X)$. Если архитектура сети задана, то вид функции G определяется значениями синаптических весов и смещений сети.

Пусть решением некоторой задачи является функция $Y = F(X)$, заданная парами входных-выходных данных (X_1, Y_1), (X_2, Y_2), ..., (X_N, Y_N), для которых $Y_K = F(X_k)$ ($k = 1, 2, ..., N$). Обучение состоит в поиске (синтезе) функции G, близкой к F в смысле некоторой функции ошибки E. Если выбранные множества учебных примеров – это пара (X_k, Y_K) (где $k = 1,2, ..., N$), а способ вычисления функции ошибки E, то обучение нейронной сети превращается в задачу многомерной оптимизации, имеет очень большую размерность при

этом, поскольку функция E может иметь произвольный вид, обучение в общем случае – многоэкстремальная невыпуклая задача оптимизации.

Для решения этой задачи могут быть использованы такие (итерационные) алгоритмы:

– алгоритмы локальной оптимизации с вычислением частных производных первого порядка;

– алгоритмы локальной оптимизации с вычислением частных производных первого и второго порядка;

– стохастические алгоритмы оптимизации;

– алгоритмы глобальной оптимизации.

К первой группе относятся градиентный алгоритм (метод найскорейшего спуска), методы с одномерной и двумерной оптимизацией целевой функции в направлении антиградиента, метод сопряженных градиентов, методы, учитывающие направление антиградиента на нескольких шагах алгоритма.

Ко второй группе относятся метод Ньютона, квазиньютоновские методы, метод Гаусса-Ньютона, метод Левенберг-Марквардта и др.

Стохастическими методами является поиск в случайном направлении, имитация отжига, метод Монте-Карло (численный метод статистических испытаний).

Задачи глобальной оптимизации решаются с помощью перебора значений переменных, от которых зависит целевая функция (функция ошибки E).

Алгоритм обратного распространения используется для обучения многослойных нейронных сетей с последовательными связями. Нейроны в таких сетях делятся на группы с общим входным сигналом – слои, при этом на каждый нейрон первого слоя подаются все элементы внешнего входного сигнала, а все выходы нейронов того слоя подаются на каждый нейрон слоя $(q+1)$. Нейроны выполняют взвешенное (с синаптическими весами) суммирование элементов входных сигналов; к этой сумме добавляется смещение нейрона. Над полученным результатом выполняется активационной функцией нелинейное преобразование. Значение функции активации – выход нейрона.

В многослойных сетях оптимальные выходные значения нейронов всех слоев, кроме последнего, как правило, неизвестны, и трех- или более слойный персептрон уже невозможно научить, руководствуясь только величинами ошибок на выходах НС. Наиболее приемлемым вариантом обучения в таких условиях оказался градиентный метод поиска минимума функции ошибки с рассмотрением сигналов ошибки от выходов НС к ее входов в направлении, обратном прямому распространению сигналов в обычном режиме работы. Этот алгоритм обучения НС получил название процедуры обратного распространения.

В таком алгоритме функция ошибки является суммой квадратов ошибки сети желаемого выхода и реального. При исчислении элементов вектора градиента был использован своеобразный вид производных функций активации сигмоидального типа. Алгоритм действует циклически (итеративно), и его циклы принято называть *эпохами*.

На каждую эпоху на вход сети поочередно подаются все учебные наблюдения, исходные значения сети сравниваются с целевыми значениями и рассчитывается ошибка. Значение ошибки, а также градиента поверхности ошибок, используется для корректировки весов, после чего все действия повторяются. Начальная конфигурация сети выбирается случайным образом, и процесс обучения прекращается, когда пройдена определенное количество эпох, или когда ошибка достигнет определенного уровня малости, или когда ошибка перестанет уменьшаться (пользователь может сам выбрать нужное условие остановки).

Приведем описание алгоритма.

Шаг 1. Весам сети присваиваются небольшие начальные значения.

Шаг 2. Выбирается очередная обучающая пара (*X, Y*); вектор *X* подается на вход сети.

Шаг 3. Вычисляется выход сети.

Шаг 4. Вычисляется разница между выходом сети, требуется (целевым, *Y*), и реальным (вычисленным).

Шаг 5. Вес сети корректируется так, чтобы минимизировать ошибку.

Шаг 6. Шаги со 2-го по 5-й повторяются для каждой пары обучающего множества до тех пор, пока ошибка на всем множестве не достигнет приемлемой величины.

Шаги 2 и 3 подобные тем, которые выполняются в уже обученной сети. Вычисления в сети выполняются послойно. На шаге 3 каждый из выходов сети вычитается из соответствующей компоненты целевого вектора с целью получения ошибки. Эта ошибка используется на шаге 5 для коррекции весов сети. Шаги 2 и 3 можно рассматривать как «проход вперед», поскольку сигнал распространяется сетью от входа к выходу. Шаги 4 и 5 составляют «обратный проход», поскольку здесь исчисляемый сигнал ошибки распространяется назад сетью и используется для подстройки весов.

По мере того как сеть учится, ошибки уменьшаются, и пока обучение уменьшает действительную функцию ошибок, ошибки на контрольном множестве также будет уменьшаться. Если же контрольная ошибка перестала уменьшаться или даже стала расти, это указывает на то, что сеть начала очень близко аппроксимировать данные и обучение следует остановить. Это явление слишком точной аппроксимации в процессе обучения называется *переобучением*. Если такое случилось, то обычно советуют уменьшить количество скрытых элементов или слоев, ибо сеть является очень мощной для этой задачи. Если же, наоборот, взята недостаточно сложная сеть для того, чтобы моделировать такую зависимость, то переобучения, скорее всего, не произойдет, и обе ошибки – обучения и проверки – не достигнут достаточного уровня малости.

Описанные проблемы с локальными минимумами и выбором размера сети приводят к тому, что во время практической работы с нейронными сетями, как правило, приходится экспериментировать с большим количеством различных сетей, порой учить каждую из них по несколько раз (чтобы не быть введенным в заблуждение локальными минимумами) и сравнивать полученные результаты. Главным показателем качества результата является контрольная ошибка. Согласно общенаучному принципу: при прочих равных условиях следует предпочесть более простую модель, из двух сетей (с примерно равными ошибками контроля) имеет смысл выбрать ту ошибку, которая меньше.

Классический метод обратного распространения относится к алгоритмам с линейной сходимостью. Для увеличения скорости сходимости необходимо использовать матрицы других производных функции ошибки.

Обучение без учителя. Алгоритм обучения нейронной сети, с помощью процедуры обратного распространения, подразумевает наличие некого внешнего звена, предоставляя сети, кроме входных, также и целевые выходные образы. Алгоритмы, которые пользуются подобной концепцией, называются алгоритмами обучения с учителем. Для их успешного функционирования необходимо наличие экспертов, создающих на предыдущем этапе для каждого входного образа эталонный выходной. Поскольку создание искусственного интеллекта идет по пути копирования природных прообразов, ученые не прекращают спор на тему, можно ли считать алгоритмы обучения с учителем натуральными или же они полностью искусственны. Например, обучение человеческого мозга, на первый взгляд, происходит без учителя: на зрительные, слуховые, тактильные и др. рецепторы поступает информация извне, а внутри нервной системы происходит самоорганизация.

Однако нельзя отрицать и того, что в жизни человека немало учителей (и в прямом, и в переносном смысле), координирующих внешние воздействия.

Главное, что делает обучение без учителя привлекательным – это его «самостоятельность». Процесс обучения, как и за обучение с учителем, заключается в подстройке веса сети. Некоторые алгоритмы, правда, меняют и структуру сети, т.е. количество нейронов и их взаимосвязь, но такие преобразования правильнее назвать более широким термином – *самоорганизация*.

2 ПРИМЕНЕНИЕ НЕЙРОННЫХ СЕТЕЙ ДЛЯ РЕШЕНИЯ ПРАКТИЧЕСКИХ ЗАДАЧ

Для использования НС при решении практических задач применяем специализированный программный продукт *STATISTICA*, который предоставляет мощные и удобные в использовании инструменты для статистического и графического анализа, прогнозирования, data mining, создания собственных пользовательских приложений, интеграции, совместной работы, web-доступа.

STATISTICA Neural Networks – один из передовых и эффективных нейросетевых продуктов на рынке.

Преимущества использования STATISTICA Neural Networks:

– пре- и пост-процессирование, включая выбор данных, кодирование номинальных значений, шкалирование, нормализацию, удаление пропущенных данных с интерпретацией для классификации, регрессию и задачи временных рядов;

– исключительная простота в использовании плюс непревзойденная аналитическая мощность; так, например, не имеющий аналогов инструмент автоматического нейросетевого поиска *Автоматизированная нейронная сеть (АНС)* проведет пользователя через все этапы создания различных нейронных сетей и выберет наилучшую (в противном случае эта задача решается длительным путем "проб и ошибок" и требует серьезного знания теории);

– самые современные, оптимизированные и мощные алгоритмы обучения сети (включая методы сопряженных градиентов, алгоритм Левенберга-Марквардта, BFGS, алгоритм Кохонена); полный контроль над всеми параметрами, влияющими на качество сети, такими как функции активации и ошибок, сложность сети;

– поддержка ансамблей нейросетей и нейросетевых архитектур практически неограниченного размера;

– богатые графические и статистические возможности, которые облегчают интерактивный исследовательский анализ;

– полная интеграция с системой STATISTICA; все результаты, графики, отчеты и т. д. могут быть в дальнейшем модифицированы с помощью мощных графических и аналитических инструментов STATISTICA (например, для проведения анализа предсказанных остатков, создания подробного отчета и т. п.);

– полная интеграция с мощными автоматическими инструментами STATISTICA; запись полноценных макросов для любых анализов; создание собственных нейросетевых анализов и приложений с помощью STATISTICA Visual Basic, вызов STATISTICA Neural Networks из любого приложения, поддерживающего технологию COM (например, автоматическое проведение нейросетевого анализа в таблице MS Excel или объединение нескольких пользовательских приложений, написанных на языках C, C++, C#, Java и т.д.);

– выбор наиболее популярных сетевых архитектур, включая Многослойные персептроны, Радиальные базисные функции и Самоорганизующиеся карты признаков;

– имеется инструмент Автоматического Сетевого Поиска, позволяющий в автоматическом режиме строить различные нейросетевые архитектуры и регулировать их сложность;

– сохранение наилучших нейронных сетей.

– поддержка различного рода статистического анализа и построение прогнозирующих моделей, включая регрессию, классификацию, временные ряды с непрерывной и категориальной зависимой переменной, кластерный анализ для снижения размерности и визуализации.

– поддержка загрузки и анализа нескольких моделей.

– опциональная возможность генерации исходного кода на языках C, C++, C#, Java, PMML (Predictive Model Markup Language), который может быть легко интегрирован во внешнюю среду для создания собственных приложений.

2.1 ПОНИЖЕНИЕ РАЗМЕРНОСТИ

Теоретические сведения

Имеются данные геологоразведки, *требуется* определить, какие факторы значимо влияют на добычу полезных ископаемых, а какие можно исключить.

Практическое задание

Этапы выполнения упражнения

Структура данных

В данных имеется зависимая переменная Y и несколько десятков независимых (предикторов) (рис. 2.1).

Данные: Таблица данных2* (45v * 1673c)

	1 No	2 category_Пл	3 F1	4 F2	5 F3	6 F4	7 F5
1	1	A	1024	2435	711,5	16,055	16,94
2	2	A	1011	2459,6	774,6	8,625	8,855
3	3	B	1117	2599	804,4	21,865	22,915
4	4	A	945	2367	533	28,05	29,93
5	5	A	959	2376	350	25,34	25,785
6	6	B	1008	2436	855,5	10,355	10,51
7	7	B	1025,2	2449	550	30,21	32,375
8	8	A	1023	2439,6	350	5,705	5,745
9	9	A	1006	2424	844,4	18,15	18,455
10	10	B	1013,2	2427	750	66,07	72,395
11	11	A	996	2396	533	40,42	43,485
12	12	A	1047	2456	650	4,955	5,11
13	13	A	1109	2528	850	27,63	29,42
14	14	B	1016	2441	657,1	30,035	33,22
15	15	A	1061	2495	450	3,72	4,11
16	16	A	940	2364	362,3	0,895	0,92
17	17	A	1149	2565	459,9	50,935	52,57

Рис 2.1 – Фрагмент исходного файла данных

Зависимая переменная Y является непрерывной и характеризует добычу полезных ископаемых; к независимым переменным относятся характеристики проб, взятых на различной глубине.

Данные носят модельный характер. Это типичные данные, возникающие в геологоразведке; задача состоит в том, чтобы по назначению предикторов предсказать значение отклика.

Отметим, в рассматриваемой задаче имеется большое число предикторов (около 50), поэтому вначале нужно понизить размерность, т.е. уменьшить количество независимых переменных и на их основе построить предсказательную модель.

Построение модели

Применим методы понижения размерности, доступные в нейронных сетях *STATISTICA*.

Шаг 1. Запускаем модель *Нейронные сети STATISTICA*. Стартовое окно показано на рис. 2.2.

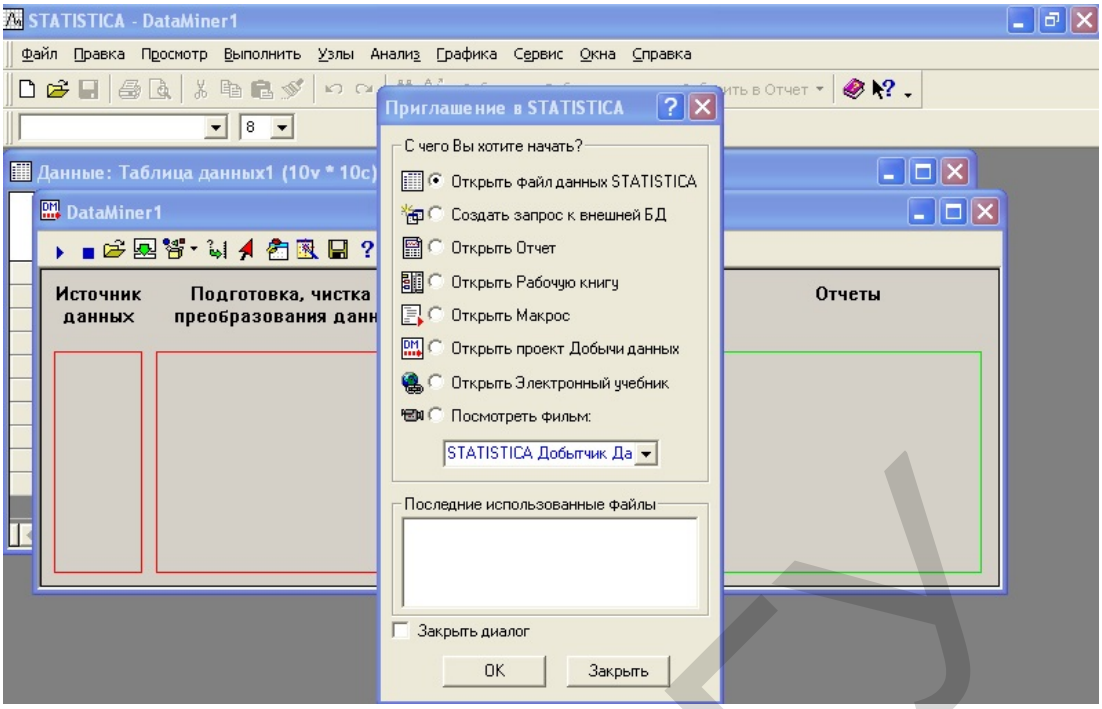


Рис. 2.2 Стартовое окно *STATISTICA*

Закрываем все окна. На панели задач нажимаем *Файл→Создать.* Откроется меню (рис. 2.3).

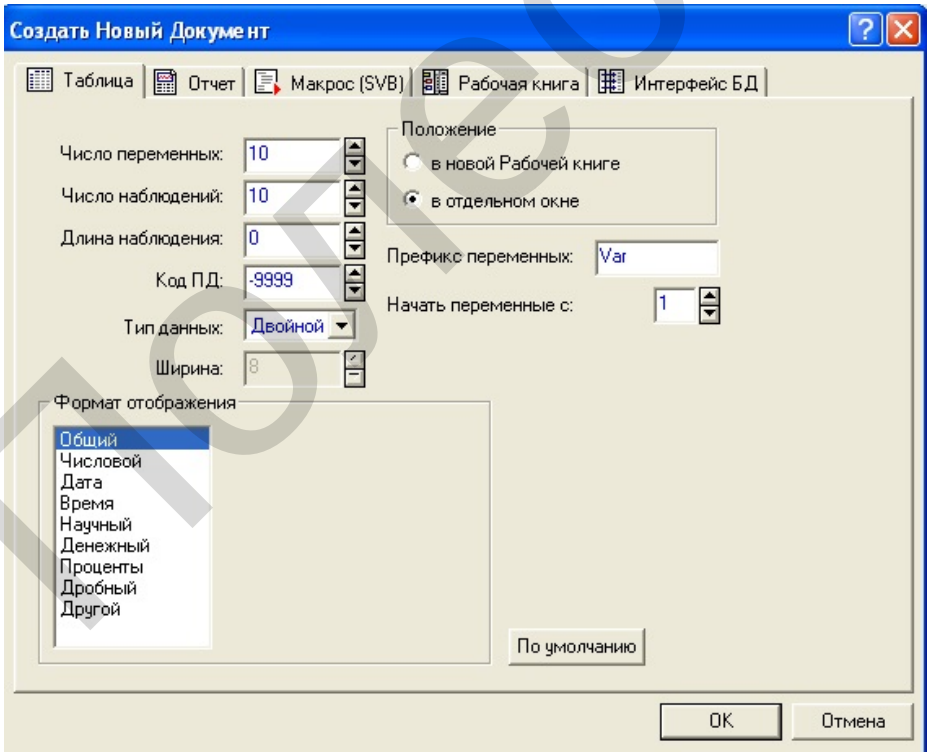


Рис. 2.3 – Окно создания документа

Выбираем нужные для нас данные и нажимаем *ОК.*

Далее на панели выбираем *Анализ→Нейронные сети.* Перед нами появится стартовое окно (рис. 2.4).

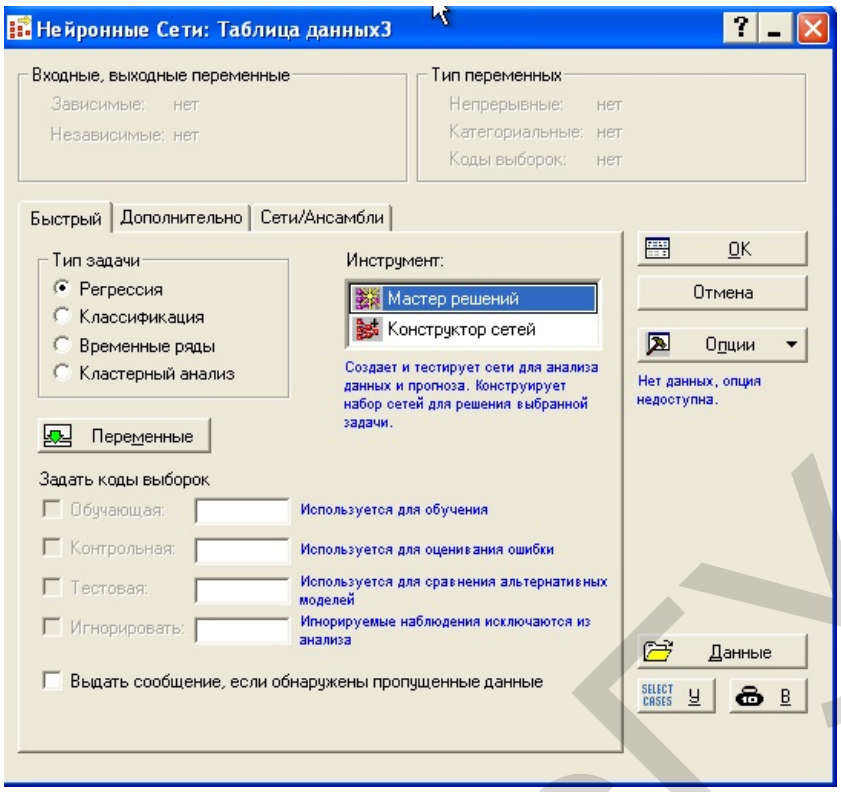


Рис. 2.4 – Стартовое окно

В стартовом окне выбираем тип задачи: *Регрессия* (зависимая переменная непрерывная).

Далее укажем переменные для анализа. Для этого в стартовом окне модуля нажимаем кнопку *Переменные*. В появившемся диалоговом окне (рис. 8.1.5) выбираем переменные. В данном примере имеется выходная (зависимая) переменная и 46 входных переменных.

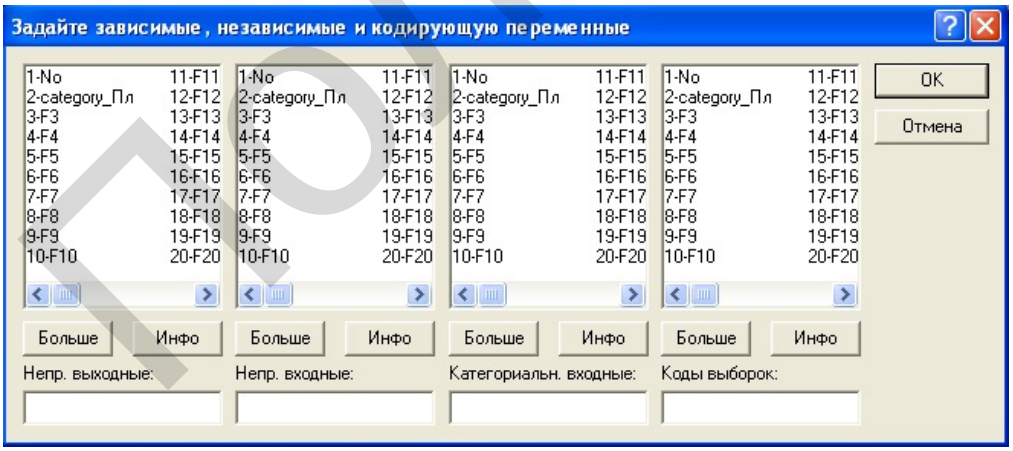


Рис. 2.5 – Диалоговое окно задания переменных

В левом окне выбираем непрерывную выходную переменную – это переменная Y с номером 46.

Указываем входные переменные – это независимые переменные, которые предсказывают отклик. В нашем примере имеется только одна категориальная входная переменная – category_Пл., остальные – непрерывные.

После выбора переменных нажимаем *ОК*.

В стартовом окне переходим на вкладку *Дополнительно* и выбираем инструмент *Понижение размерности* (рис. 2.6).

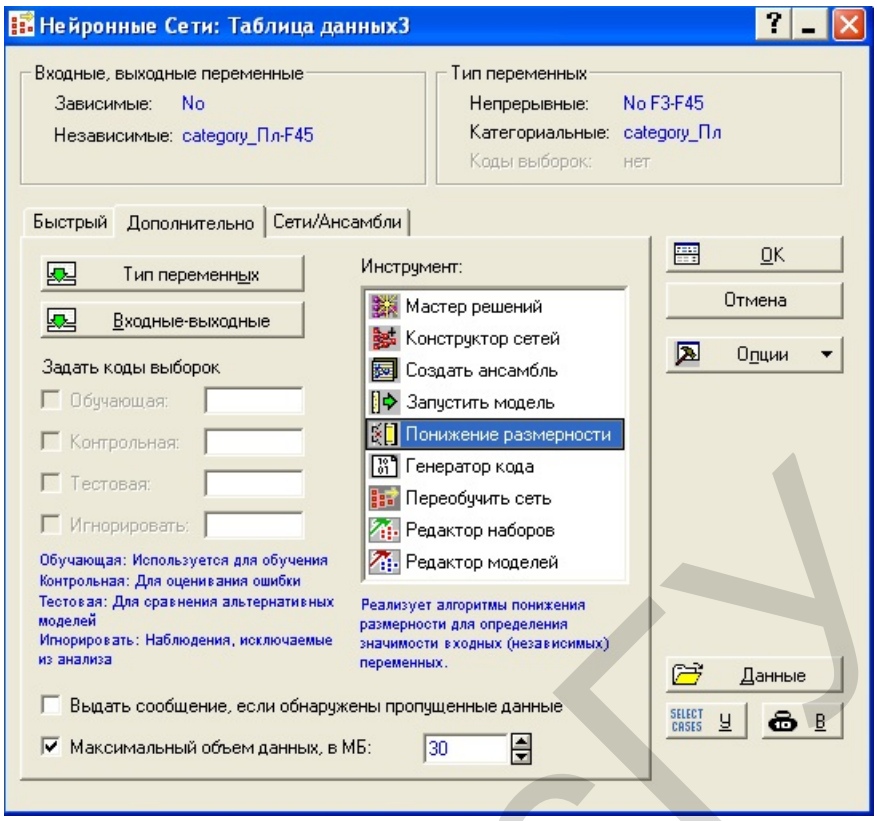


Рис. 2.6 – Стартовое окно модуля, вкладка *Дополнительно*

Шаг 2. На экране появляется диалоговое окно *Понижение размерности.* Доступны следующие методы: последовательный с включением, последовательный с исключением, генетический алгоритм отбора признаков.

Вначале получим результаты с помощью алгоритма с включением. Для этого на вкладке *Быстрый* выбираем алгоритм *Последовательный с включением* и нажимаем кнопку *ОК* (рис. 2.7).

Результат работы алгоритма представляется в таблице (рис. 2.8). На первом этапе переменные по отдельности включаются в модель, находится переменная, которая дает наименьшую ошибку. Затем модель начинает процедуру поиска второй переменной, которая уменьшает значение ошибки. И так до тех пор, пока включение новых переменных уменьшает ошибку. Итоговый результат записывается в последнюю строчку таблицы (рис. 2.9). В качестве значимых предикторов выделены: F9, F13, F24, F26, F27, F32, F34, F38, F42, F43.

Из-за того, что для работы не было предоставлено всех данных, результаты будет отличаться от тех, которые должны получиться.

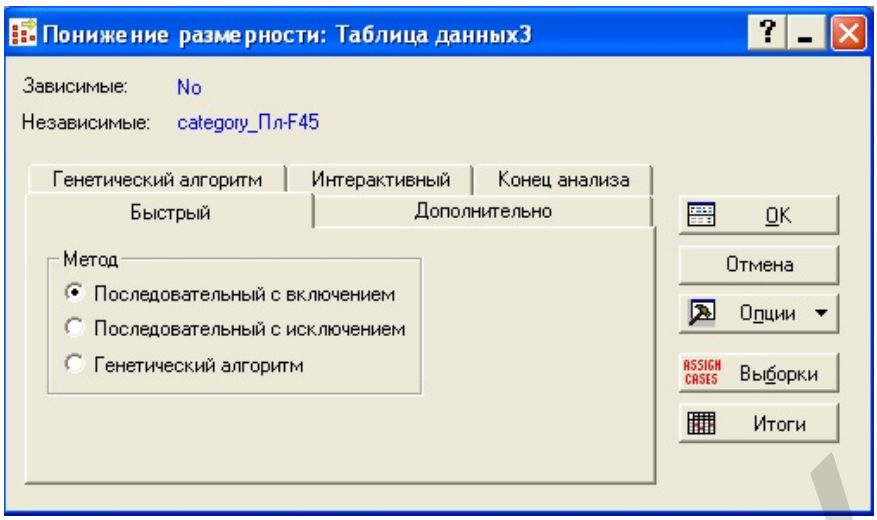


Рис. 2.7 – Окно задания параметров для алгоритма понижения размерности

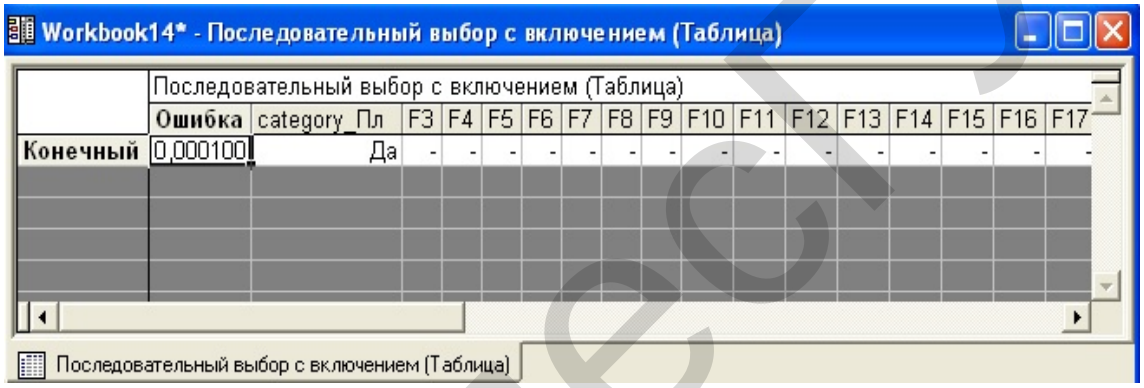


Рис. 2.8 – Начальная часть таблицы результатов алгоритма *Последовательный с включением*

Шаг 3. Теперь аналогичную задачу решим с помощью алгоритмов с исключением. Для этого возвращаемся к окну задания параметров для алгоритмов снижения размерности и выбираем *Последовательный с исключением.*

Нажимаем *ОК* и переходим к анализу результатов.

Программа стартовала с модели, в которую включены все предикторы, и на каждом шаге исключала незначимые (рис. 2.10).

Значимые, с точки зрения модели с исключением факторы: F6, F10-F14, F24, F26, F29, F30, F34, F42, F43.

Из-за того, что для работы не было предоставлено всех данных, результаты будет отличаться от тех, которые должны получиться.

Шаг 4. Снова возвращаемся к окну (см. рис. 2.5) и выбираем *Генетический алгоритм.* В результате работы генетического алгоритма: выделяются предикторы F4, F6, F10, F16, F20-F22, F24, F26- F30, F38, F41-F43.

Увеличим значение штрафа за элемент в 5 раз (до 0,0005) и снова запустим генетический алгоритм (изменение параметра *Штраф за элемент* производится в вкладке *Дополнительно*). Выделяются следующие факторы: F4, F16, F22, F29, F30, F42. Результаты отличаются, поскольку в каждом случае строится отдельная нейронная сеть, для каждой сети подгоняются коэффициенты.

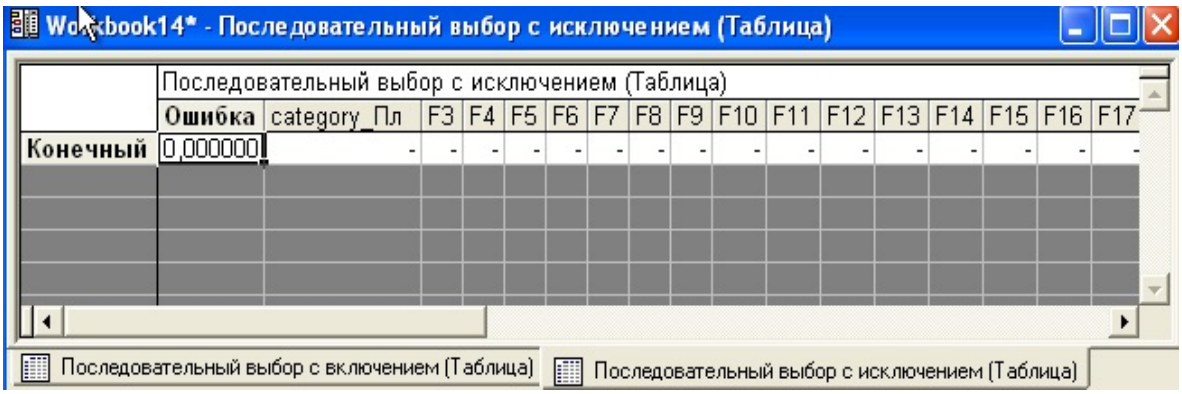


Рис.2.10 – Начальная часть таблицы результатов алгоритма *Последовательный с исключением*

Экспериментируя с разными алгоритмами и штрафами за элемент можно выделить несколько наборов значимых предикторов. Это рабочие модели, с которыми можно работать далее

Мы существенно сократили число предикторов, теперь можно построить несколько регрессионных моделей и по предсказанному отклику сравнить их качество.

Сделаем комментарий по поводу выбора штрафов. Штрафы задаются в вкладке *Дополнительно*. Параметр, указанный в поле *Штраф за элемент,* умножается на число выбранных входных переменных и добавляется к контрольной ошибке сети после того, как она обучена и протестирована. Ненулевой штраф за элемент поощряет меньшие сети и часто улучшает производительность сети.

Если параметр слишком велик, число переменных становится более важным, чем качество сети. Это может заставить алгоритм отключить все ходы.

Небольшое значение рекомендуется, чтобы скомпенсировать шум в процессе выбора наблюдений. Типичные значения лежат в диапазоне 0,001-0,005.

Шаг 5. В программе *STATISTICA* предусмотрена возможность проведения анализа чувствительности сети к входным переменным.

Процедура позволяет сделать вывод об относительной важности входных переменных для конкретной нейронной сети и, при необходимости, удалить входы с низкими показателями чувствительности. Анализ чувствительности можно использовать либо с сугубо информационными целями, либо для удаления лишних входов.

Имеется ряд моментов, на которые следует обратить внимание при использовании процедуры.

В общем случае входные переменные не являются независимыми. Анализ чувствительности располагает их в порядке, который соответствует степени ухудшения качества модели при исключении из нее соответствующей переменной. Каждой переменной присваивается определенный рейтинг. Однако при наличии зависимостей между входными переменными мы не можем быть уверены, что одиночный рейтинг правильно отражает реальную ситуацию.

Тем не менее, процедура оказывается чрезвычайно полезной на практике. Если исследуется целый ряд моделей, то имеет смысл выделять ключевые переменные, которые всегда важны для отклика и имеют высокий показатель чувствительности, определить переменные с низкой чувствительностью и получить информацию о «сомнительных» переменных, которые меняют свой рейтинг и, возможно, содержат избыточную информацию.

Нейронные сети *STATISTICA* проводят анализ чувствительности, исключая последовательно входные переменные. Каждая модель определяет процедуру замены пропущенных данных (ПД), которая используется, чтобы осуществить прогноз при пропущенных значениях одной или более входных переменных.

Чтобы определить чувствительность данной переменной *x*, мы сначала прогоним сеть на наборе тестовых наблюдений и получим ошибку сети. Затем прогоним сеть на тех же наблюдениях, заменив наблюдаемые значения *x* на значения, оцененные с помощью процедуры замены ПД, и вновь получим ошибку сети.

Мы удалили часть информации, которую использует сеть (одну из входных переменных), поэтому следует ожидать некоторое увеличение ошибки. Основное значение чувствительности – это отношение ошибки с заменой ПД к исходной ошибке.

Чем чувствительнее сеть к переменной, тем больше данное отношение. Если отношение ≤ 1, тогда отключение переменной либо не влияет на производительность сети, либо улучшает ее.

После того как чувствительности подсчитаны для всех переменных, их можно упорядочить. *STATISTICA* проводит ранжировку для удобства интерпретации чувствительностей.

Анализ чувствительности запускается из окна результатов нейросети, например, после построения регрессионной модели. Результаты анализа показаны на рис. 2.11.

Из-за того, что для работы не было предоставлено всех данных, результаты будет отличаться от тех, которые должны получиться.

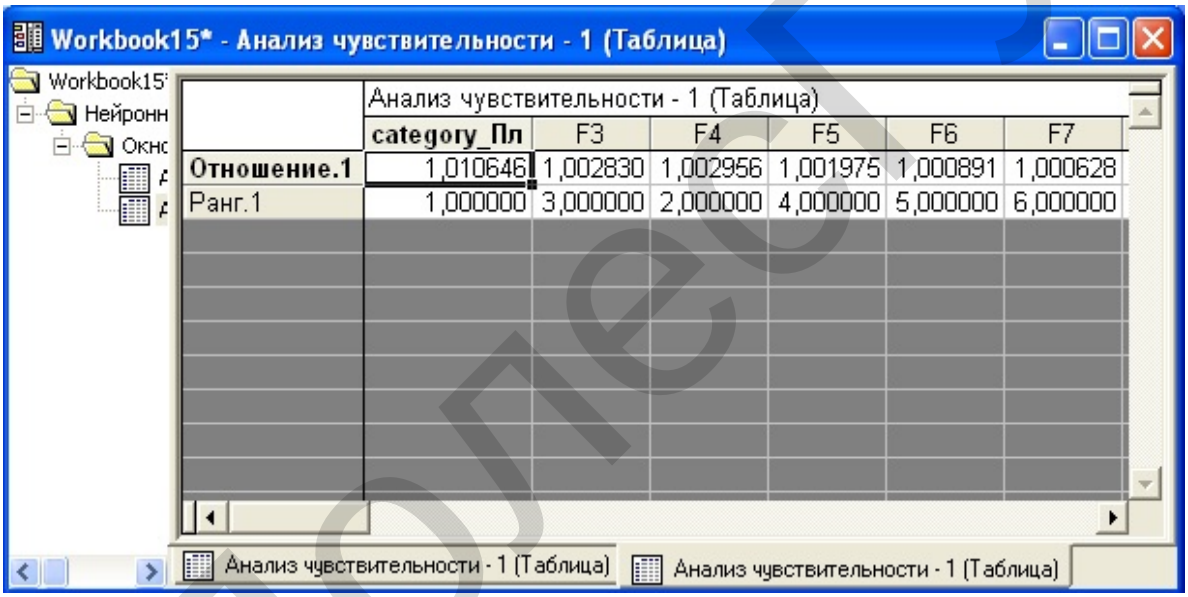


Рис. 2.11 – Результаты анализа чувствительности

Пустые ячейки таблицы означают, что данные переменные были удалены еще на этапе обучения сети.

2.2 РАСПОЗНАВАНИЕ ОБРАЗОВ

Теоретические сведения

Покажем, как нейронные сети можно использовать в распознавании образов. Данные имитируют «цифры», высвечивающиеся на экране неисправного калькулятора. Наблюдаемые классы зависимой переменной *DIGIT* соответствовали цифрам 0-9, которые вводились с клавиатуры калькулятора.

В задаче имеется семь категориальных предикторов: *VAR1-VAR7* по числу линий, образующих цифру. Уровень категориального предиктора (0 – отсутствует; 1 – присутствует) показывает, высвечивалась ли на экране соответствующая ему одна из семи линий (три горизонтальных и четыре вертикальных).

Описание предикторных переменных: *VAR1* – верхняя горизонтальная, *VAR2* – верхняя левая горизонтальная, *VAR3* – верхняя правая горизонтальная, *VAR4* – средняя

горизонтальная, *VAR5* – нижняя левая вертикальная, *VAR6* – нижняя правая вертикальная, *VAR7* – нижняя горизонтальная.

Калькулятор неисправен, поэтому при нажатии какой-либо кнопки на цифровой клавиатуре на экране не всегда высвечивается правильная комбинация линий.

Практическое задание

Этапы выполнения упражнения

Структура данных

На рис. 2.12 приведены первые 10 наблюдений файла данных.

Весь выбор данных, состоящий из 500 наблюдений, содержится в файле примеров *Digit.sta.*

Данные: Таблица1 (8v * 10c)

	1 DIGIT	2 VAR2	3 VAR3	4 VAR4	5 VAR5	6 VAR6	7 VAR7	8 VAR8
1	seven	ONE	ZERO	ONE	ZERO	ZERO	ONE	ZERO
2	one	ZERO	ZERO	ONE	ZERO	ZERO	ONE	ZERO
3	four	ZERO	ONE	ONE	ONE	ZERO	ONE	ZERO
4	two	ONE	ONE	ONE	ONE	ONE	ZERO	ZERO
5	eight	ZERO	ONE	ONE	ONE	ONE	ONE	ONE
6	one	ZERO	ZERO	ONE	ZERO	ZERO	ONE	ZERO
7	five	ONE	ONE	ZERO	ONE	ZERO	ONE	ONE
8	six	ONE	ZERO	ZERO	ONE	ONE	ONE	ONE
9	two	ONE	ZERO	ONE	ONE	ONE	ZERO	ONE
10	eight	ONE	ONE	ONE	ONE	ZERO	ONE	ONE

Рис. 2.12 – Фрагмент исходного файла данных

Результаты распознавания записаны в первой переменной (*DIGIT*). В переменные *VAR1-VAR7* заключены уровни независимых категориальных предикторов.

Построение модели

В данном примере необходимо построить модель распознавания цифр, выдаваемых реальным калькулятором.

Разумно сформулировать следующие требования к нейронной сети: 1) сеть должна обладать возможностью экстраполировать за область обучающих данных (т.е. давать правильный прогноз при комбинациях предикторов, которые сильно отличаются от обучающего множества); 2) требовать небольшого времени для прогноза (это диктуется требованиями практического применения).

Указанным условиям соответствует архитектура многослойного персептрона. Число элементов на скрытом слое выберем равным 5 (классификация проводится в 10 классов). Обычно, в первом приближении при построении модели на скрытом слое выбирается число элементов, но в данной задаче выбор еще обусловлен количеством классов.

Шаг 1. Из меню *Анализ →Нейронные сети* запускаем модуль *Нейронные сети.*

Выбираем задачу: *Классификация.*

Заходим в диалог выбора *Переменных: DIGIT* – зависимая категориальная переменная. *VAR1-VAR7* – независимые категориальные предикторы (рис. 2.13). Нажмите *OK.*

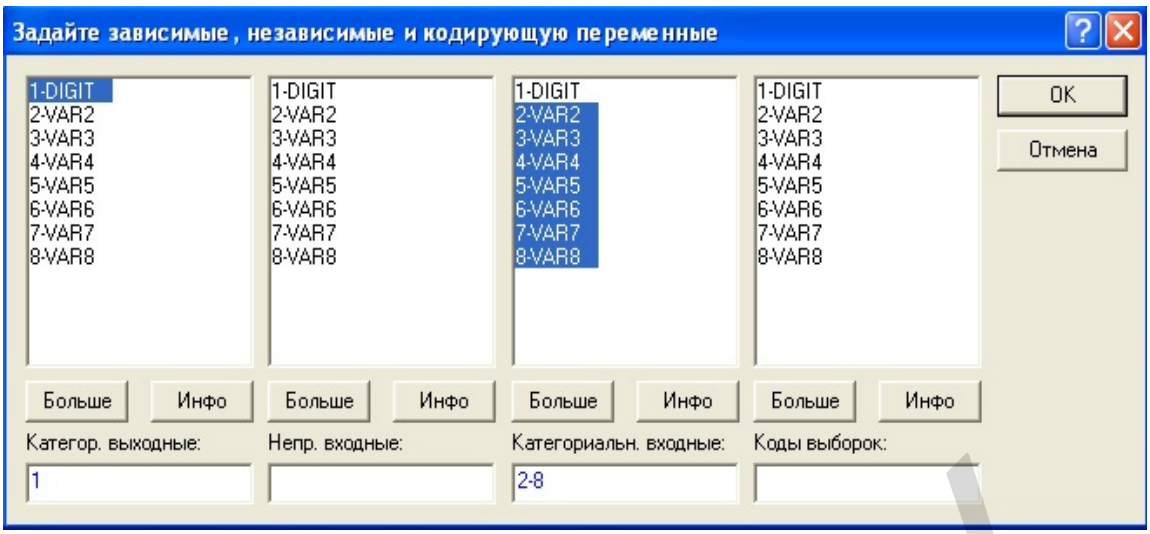


Рис. 2.13 – Диалог выбора переменных

Шаг 2. На стартовом окне в качестве инструмента построения сети выбираем *Конструктор сетей.* Нажимаем *ОК.*

Шаг 3. В следующем окне можно выбрать тип сети (рис. 2.14). Указываем *Многослойный персептрон.*

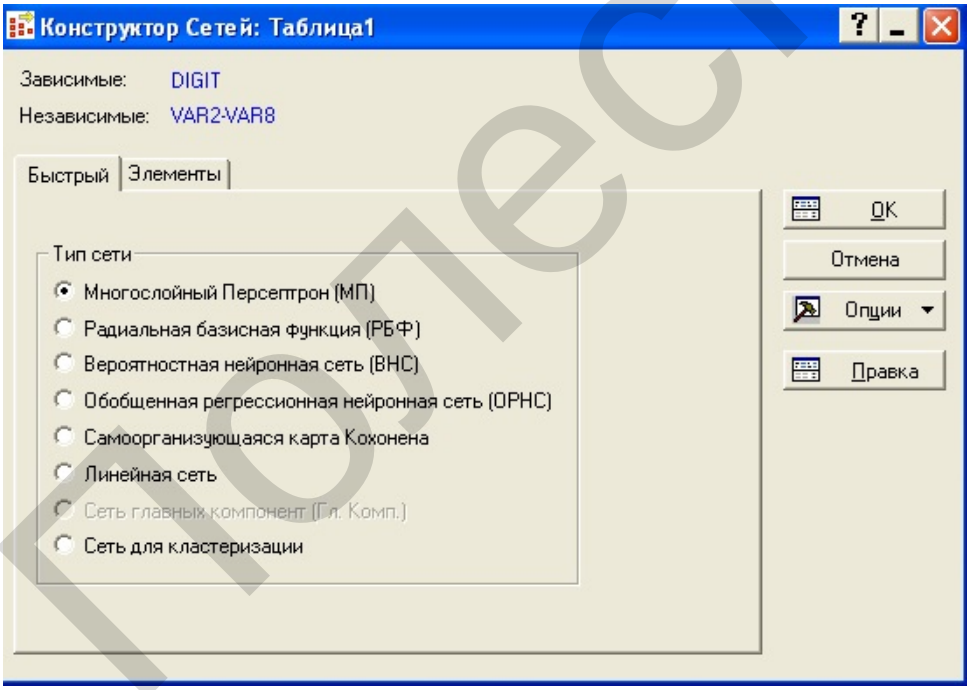


Рис. 2.14 – Диалог выбора типа сети

Далее необходимо задать число элементов на скрытом слое, для чего переходим на вкладку *Элементы сети* (рис. 2.15). На вкладке *Элементы* задаем число элементов на скрытом слое равным 5.

Итак, архитектура сети определена. Нажимаем кнопку *ОК* и переходим к диалогу *Обучение многослойного персептрона.*

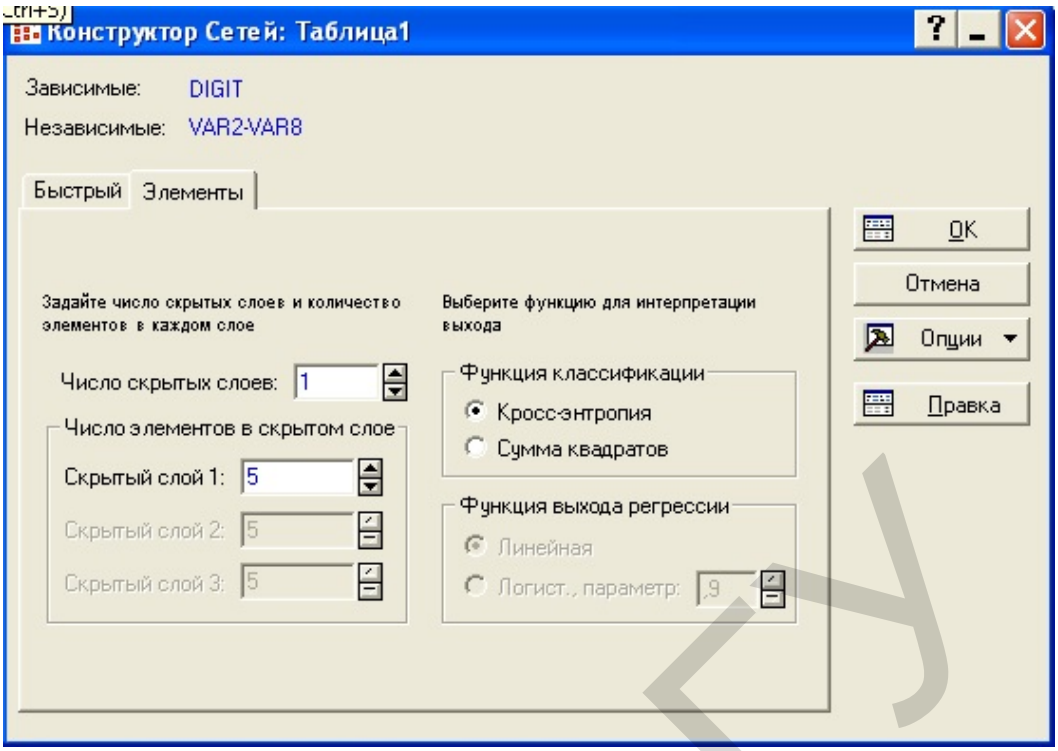


Рис. 2.15 – Диалог задания параметров сети

Шаг 4. Обучим сеть с помощью алгоритма обратного распространения ошибки (*Количество эпох = 500, Скорость обучения = 0,01).*

Этот алгоритм можно использовать с большинством сетей пакета *STATISTICA Neural Networks,* однако, в наибольшей степени он приспособлен для обучения многослойных персептронов. По сравнению с другими алгоритмами он требует меньше памяти и обычно довольно быстро достигает приемлемого уровня ошибки, хотя к точному минимуму ошибки может сходиться довольно медленно. Под числом эпох понимается максимальное число итераций в процессе обучения. Обычно количество эпох меняется в пределах от 100 до 1000. Управляющий параметр «скорость обучения» управляет величиной шага при последовательной корректировке весов. Установка по умолчанию для данного параметра в большинстве задач приводит к устойчивому обучению и не требует корректировки (рис. 2.16).

Отметим, что довольно часто приходится повторять процедуру обучения, пока не будет достигнут глобальный минимум ошибки.

Нажимаем *ОК,* запуская процедуру обучения.

Шаг 5. Теперь необходимо проанализировать результаты. Производительность классификатора на контрольном и тестовом множествах равняется приблизительно 71–72. В окне результатов просмотрим описательные статистики (рис. 2.17).

Нажимаем кнопку *Описательные статистики* в левом нижнем углу.

Таблица описательных статистик содержит подробную информацию о количестве правильно и неправильно классифицированных наблюдений по каждому классов (рис. 2.18).

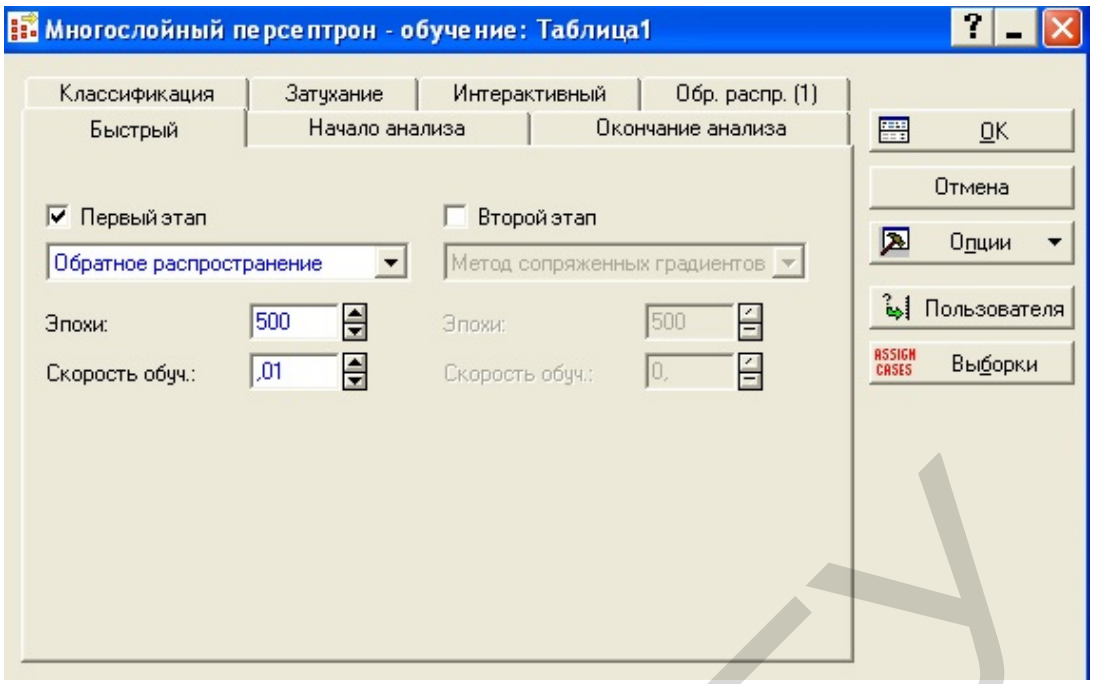


Рис. 2.16 – Диалог выбора алгоритма обучения

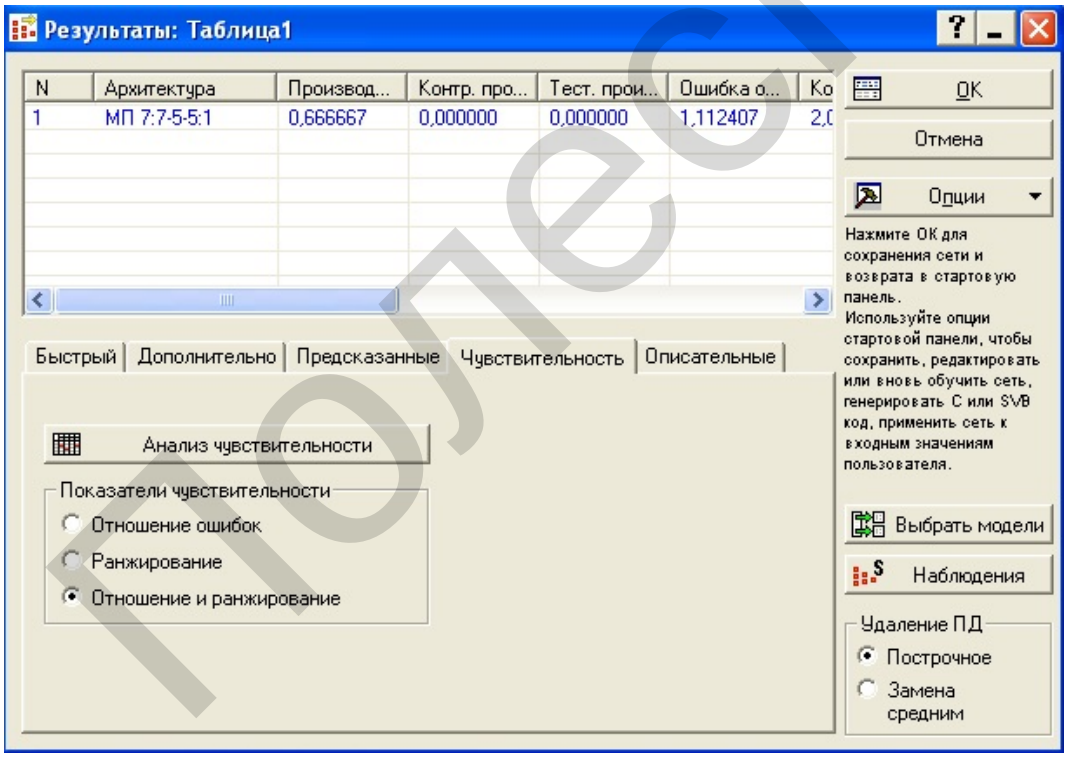


Рис. 2.17 – Окно результатов

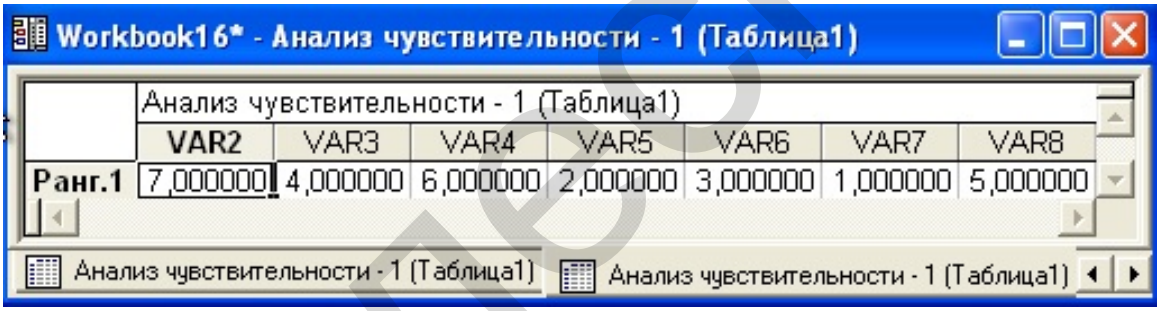
Workbook16* - Классификация (1) (Таблица1)

	Классификация (1) (Таблица1)					
	DIGIT.seven.1	DIGIT.one.1	DIGIT.four.1	DIGIT.two.1	DIGIT.eight.1	DIGI
Всего	1,0000	2,0000	1,0000	2,00000	2,00000	
Правильно	1,0000	0,0000	1,0000	1,00000	1,00000	
Ошибочно	0,0000	2,0000	0,0000	1,00000	1,00000	
Неизвестно	0,0000	0,0000	0,0000	0,00000	0,00000	
% правильных	100,0000	0,0000	100,0000	50,00000	50,00000	
% ошибочных	0,0000	100,0000	0,0000	50,00000	50,00000	1
% неизвестно	0,0000	0,0000	0,0000	0,00000	0,00000	

Рис. 2.18 – Фрагмент окна таблицы описательных статистик

Посмотрим на таблицу. Видно, что больше всего ошибок допущено при распознавании цифры 9 (см. левый столбец).

Далее запустим анализ чувствительности к предикторам. Нажимаем вкладку *Чувствительность* в окне *Результаты.* Анализ чувствительности позволяет сделать вывод об относительной важности входных переменных для конкретной нейронной сети и при необходимости удалить входы с низкими показателями чувствительности. Результаты анализа показаны в таблице (рис. 2.19).



Workbook16* - Анализ чувствительности - 1 (Таблица1)

	Анализ чувствительности - 1 (Таблица1)						
	VAR2	VAR3	VAR4	VAR5	VAR6	VAR7	VAR8
Ранг.1	7,000000	4,000000	6,000000	2,000000	3,000000	1,000000	5,000000

Анализ чувствительности - 1 (Таблица1) Анализ чувствительности - 1 (Таблица1)

Рис. 2.19 – Таблица результатов анализа чувствительности

Ранг 1 соответствует наибольшей чувствительности к данному предиктору. Переменные *VAR1* и *VAR2* (верхняя горизонтальная, верхняя левая вертикальная) вносят наибольший вклад в определение цифр.

Результаты модели на обучающем, контрольном и тестовом множествах можно получить с помощью кнопки *Предсказанные.*

Таблица предсказанных значений для первых десяти наблюдений приведена на рис. 2.20.

Архитектуру построенной сети удобно представить в графическом виде.

STATISTICA позволяет отображать с помощью цвета уровни активации нейронов на входных, выходных и скрытых слоях.

Для построения графа необходимо перейти на вкладку *Дополнительно* и нажать кнопку *Архитектура сети.* На экране появится график, представленный на рис. 2.21.

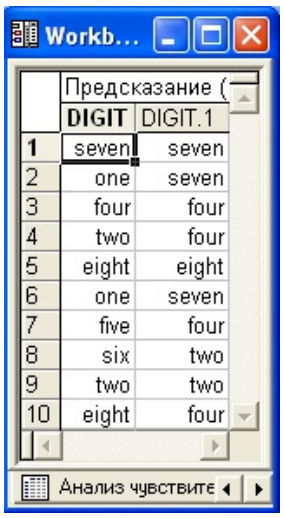


Рис. – 2.20 Фрагмент таблицы

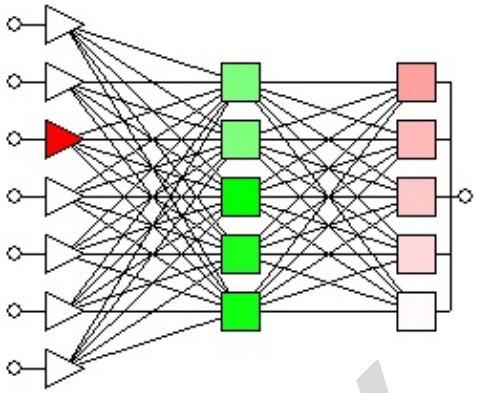


Рис. 2.21 – Архитектура построенной сети предсказанных значений

Уровни активации выделены цветом: черным – для положительной активации, серым – для отрицательной.

2.3 ПОСТРОЕНИЕ МОДЕЛИ ПОВЕДЕНЧЕСКОГО СКОРИНГА

Теоретические сведения

Рассмотрим схему построения нейросетевой модели для задачи поведенческого скоринга. Поведенческий скоринг (*behavior scoring*) используется для принятия решений по уже выданным кредитам.

Основные решения, принимаемые с использованием поведенческого скоринга, можно сформулировать следующим образом:

– предложение новых услуг и улучшение уже предоставляемых услуг;

– решение, выдавать ли кредитную карту заново после истечения срока действия или нет;

– меньший стартовый кредитный лимит или максимальное значение кредита на кредитной карточке;

– более строгий сбор платежей с нарушителей или отправка данных о них в агентства сбора платежей;

– повышение кредитного лимита;

– помещение под наблюдение ввиду потенциальных мошеннических действий и т.д.

В данном примере необходимо оценить кредитоспособность существующих заемщиков на основании данных о графике погашения кредитов и динамики средств на счетах клиента.

Практическое задание

Этапы выполнения упражнения

Структура данных

Каждого клиента будем характеризовать 22 признаками. К анкетным данным относятся 20 переменных, которые вписываются для получения кредита.

К этим переменным принадлежат:

1) текущий баланс счета;
2) продолжительность (в мес.);
3) выплаты по предыдущим кредитам;
4) назначение кредита, сумма кредита;
5) объем сбережений;

6) объем сбережений и акций;
7) время работы на данном рабочем месте;
8) частичный доход от доступного дохода (в %);
9) семейное положение/пол.;
10) поручители;
11) длительность проживания по текущему адресу;
12) наиболее ценные активы;
13) возраст в годах;
14) добавочные выплаты по кредитам;
15) тип жилья;
16) число предыдущих кредитов в банке;
17) должность;
18) число человек в подчинении;
19) наличие телефона;
20) иностранный работник.

График погашения кредита будем характеризовать двумя переменными: количество месяцев с момента выдачи кредита, общее количество невыплат или просрочек.

На основании перечисленных факторов все клиенты подразделяются на «хороших» и «плохих». Разбиение на эти группы записано в переменной *Кредитоспособность (Creditability).*

Всего имеются данные по тысяче клиентам. При этом 30 % относятся к «плохим», а остальные 70 % – к «хорошим». Процент невыплат по всей совокупности данных – около 3 % (данная величина относится к одному месяцу). Элемент таблицы данных показан на рис. 2.22.

Поскольку количество наблюдений, относящихся к разным группам («хороший» и «плохой»), существенно различается, то необходимо задать дополнительную переменную, содержащую веса наблюдений (рис. 2.23).

В противном случае группа «хороших» будет оказывать большее влияние на построение модели, чем группа «плохих».

Для группы «хороший» зададим вес равный 3, а для группы «плохой» – равный 7 (т.е. каждая группа будет оказывать одинаковое влияние на построение модели).

Переменную, содержащую веса, назовем *w.*

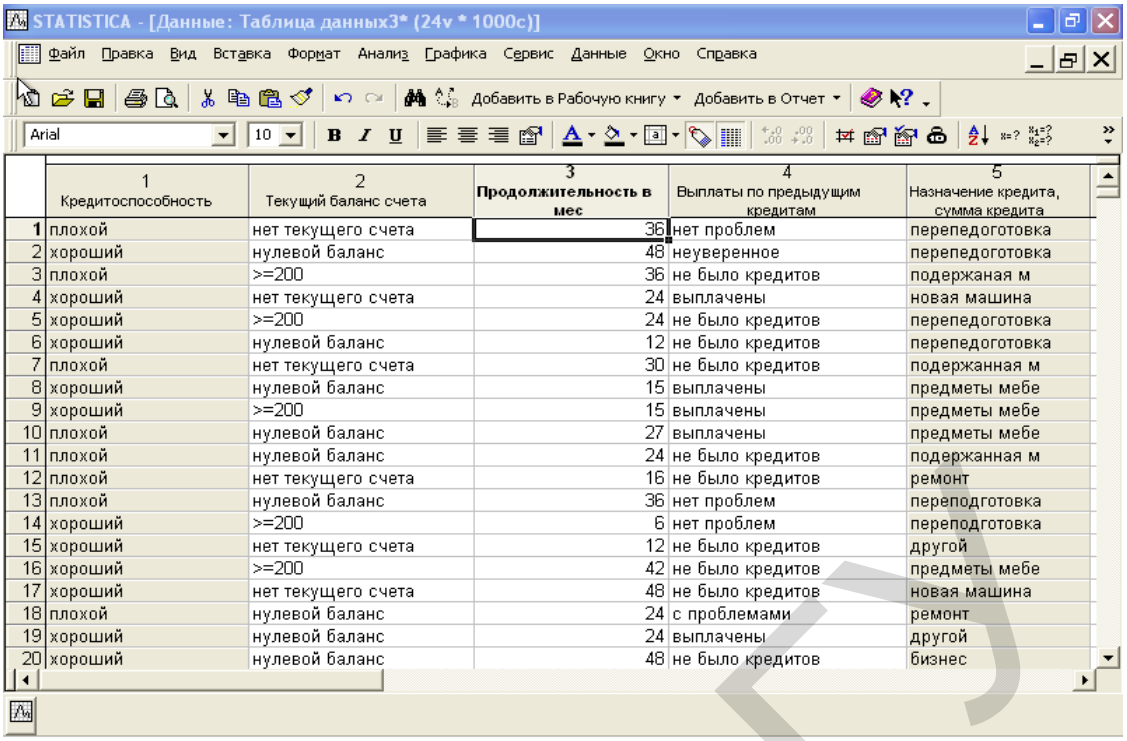
Построение модели

На первом шаге необходимо исключить из анализа переменные, которые не оказывают значимого влияния на принадлежность к тому или иному классу (на зависимую переменную).

Шаг 1. Задание анализа (рис. 2.24).

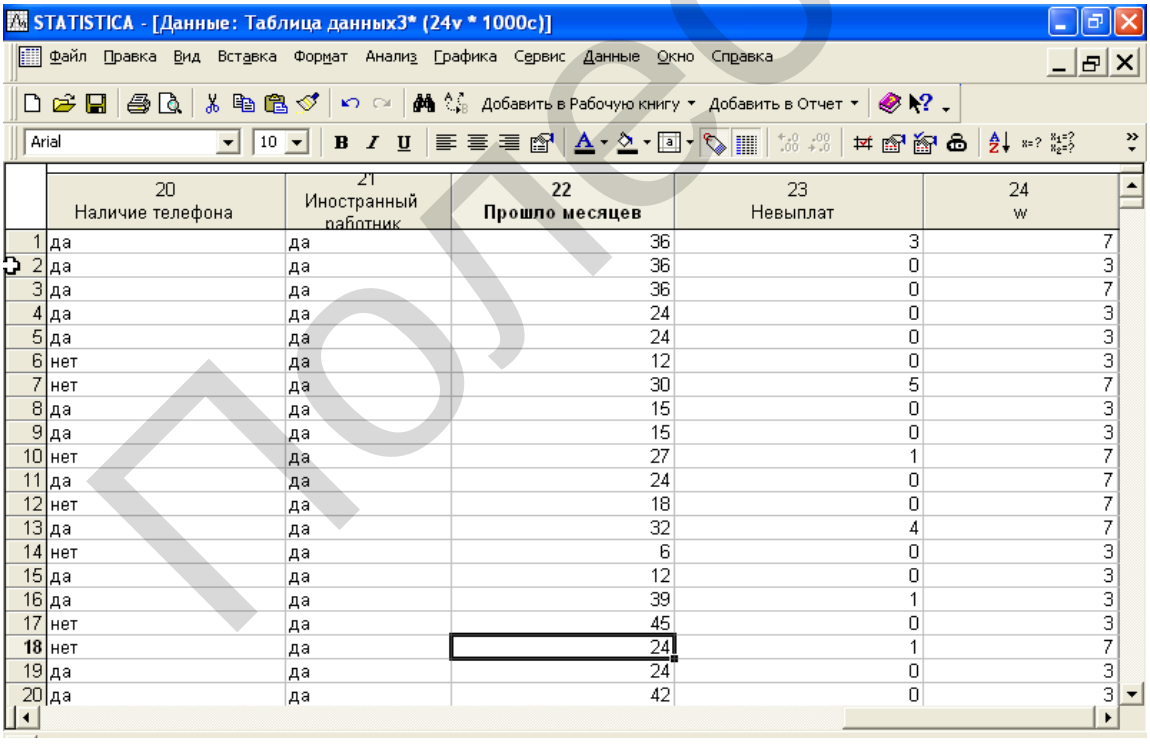
Заходим в меню *Анализ* и выбираем *Нейронные сети.*

В стартовом окне выбираем тип анализа: *Классификация* (тип задачи определяется типом зависимой переменной).



	1 Кредитоспособность	2 Текущий баланс счета	3 Продолжительность в мес	4 Выплаты по предыдущим кредитам	5 Назначение кредита, сумма кредита
1	плохой	нет текущего счета	36	нет проблем	переподготовка
2	хороший	нулевой баланс	48	неуверенное	переподготовка
3	плохой	>=200	36	не было кредитов	подержаная м
4	хороший	нет текущего счета	24	выплачены	новая машина
5	хороший	>=200	24	не было кредитов	переподготовка
6	хороший	нулевой баланс	12	не было кредитов	переподготовка
7	плохой	нет текущего счета	30	не было кредитов	подержанная м
8	хороший	нулевой баланс	15	выплачены	предметы мебе
9	хороший	>=200	15	выплачены	предметы мебе
10	плохой	нулевой баланс	27	выплачены	предметы мебе
11	плохой	нулевой баланс	24	не было кредитов	подержанная м
12	плохой	нет текущего счета	16	не было кредитов	ремонт
13	плохой	нулевой баланс	36	нет проблем	переподготовка
14	хороший	>=200	6	нет проблем	переподготовка
15	хороший	нет текущего счета	12	не было кредитов	другой
16	хороший	>=200	42	не было кредитов	предметы мебе
17	хороший	нет текущего счета	48	не было кредитов	новая машина
18	плохой	нулевой баланс	24	с проблемами	ремонт
19	хороший	нулевой баланс	24	выплачены	другой
20	хороший	нулевой баланс	48	не было кредитов	бизнес

Рис. 2.22 – Фрагмент исходной таблицы данных



	20 Наличие телефона	21 Иностранный работник	22 Прошло месяцев	23 Невыплат	24 w
1	да	да	36	3	7
2	да	да	36	0	3
3	да	да	36	0	7
4	да	да	24	0	3
5	да	да	24	0	3
6	нет	да	12	0	3
7	нет	да	30	5	7
8	да	да	15	0	3
9	да	да	15	0	3
10	нет	да	27	1	7
11	да	да	24	0	7
12	нет	да	18	0	7
13	да	да	32	4	7
14	нет	да	6	0	3
15	да	да	12	0	3
16	да	да	39	1	3
17	нет	да	45	0	3
18	нет	да	24	1	7
19	да	да	24	0	3
20	да	да	42	0	3

Рис. 2.23 – Фрагмент исходной таблицы данных весов переменной

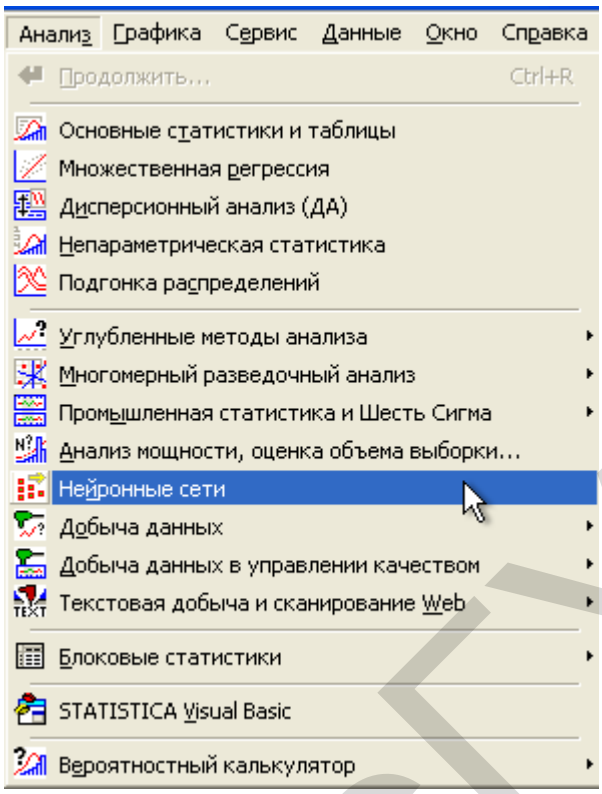


Рис. 2.24 – Меню *Анализ* системы *STATISTICA*

Для опытных пользователей имеется возможность (на вкладке *Дополнительно)* задать тип переменной, а также ее статус (входная/выходная), не ограничиваясь какой-либо одной постановкой исследования.

Среди всех имеющихся переменных только четыре непрерывные переменные (длительность в месяцах, сумма кредита, возраст в годах, число месяцев с момента выдачи кредита) (рис. 2.25).

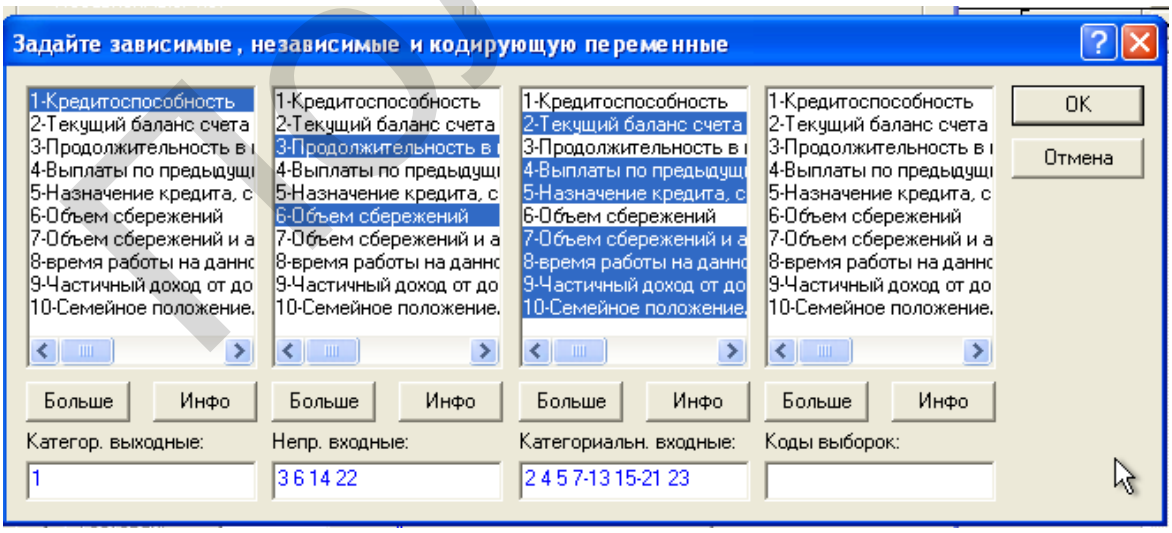


Рис. 2.25 – Окно выбора переменных

Нажимаем кнопку *ОК*. Стартовое окно принимает вид, показанный на рис. 2.26.

Переходим на вкладку *Дополнительно* (рис. 2.27) и выбираем инструмент – *Понижение размерности.*

Нажимаем *ОК*. В окне *Понижение размерности* переходим на вкладку *Быстрый* (рис. 2.27).

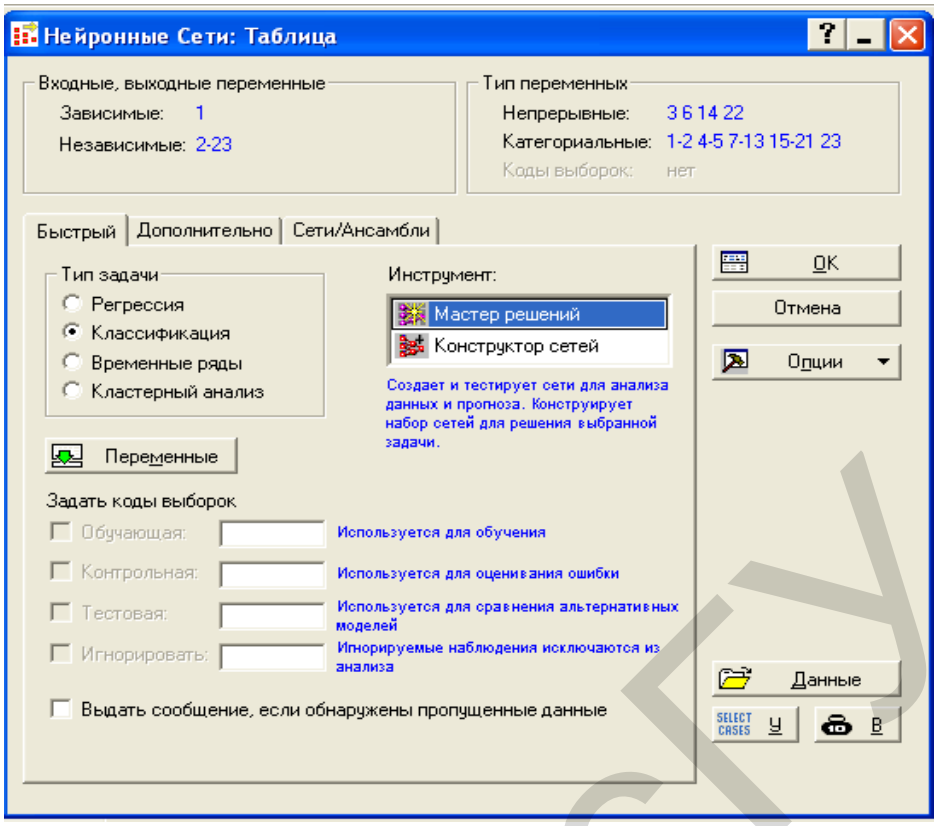


Рис. 2.26 – Стартовое окно модуля *Нейронные сети*

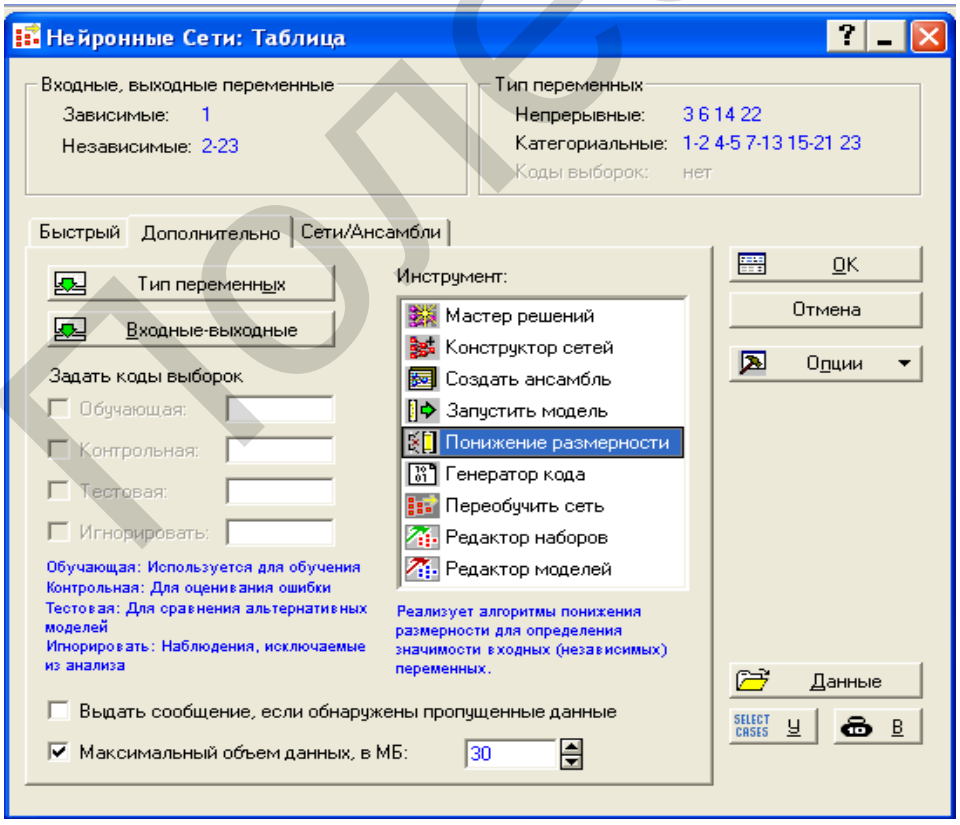


Рис. 2.27 – Стартовое окно модуля *Нейронные сети*, вкладка *Дополнительно*

Наиболее быстрым алгоритмом является *Метод последовательный с включением* (при 22 переменных и 1 000 наблюдений остальные работают существенно дольше).

Оставляя установки по умолчанию, нажимаем *ОК* (рис. 2.28). Результатом работы алгоритма является таблица, представленная на рис. 2.29.

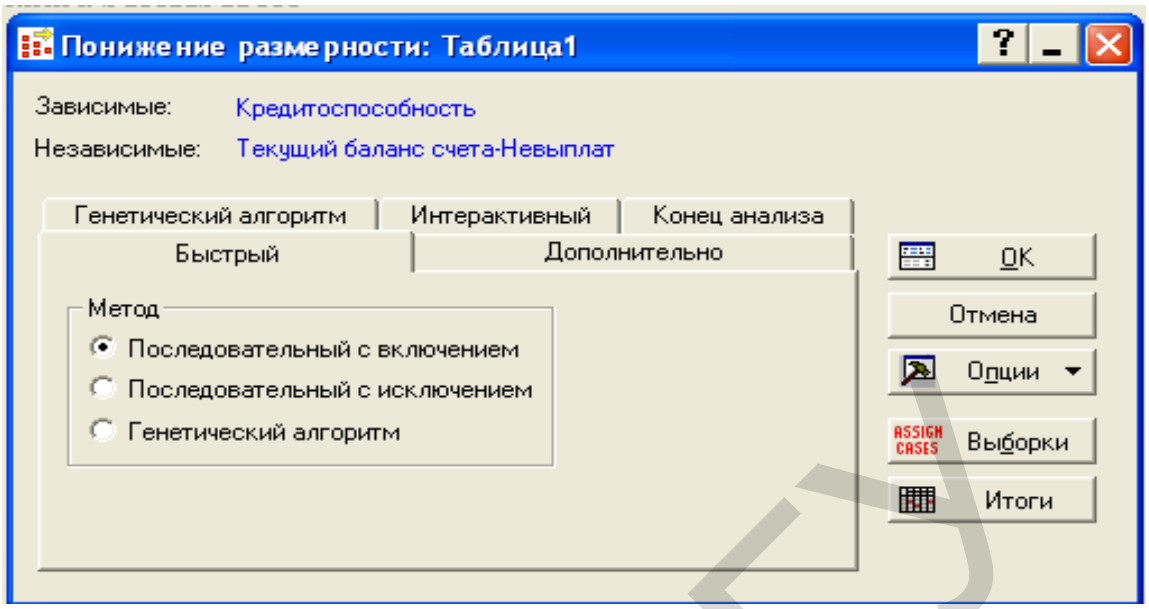


Рис. 2.28 – Окно *Понижение размерности*, вкладка *Быстрый*

	Последовательный выбор с включением (Таблица1)						
	Ошибка	Текущий баланс счета	Продолжительность в мес	Выплаты по предыдущим кредитам	Назначение кредита, сумма кедита	Объем сбережений	Сбер
Конечный	0,454479	-	Да	-	Да	-	

Рис. 2.29 – Таблица результатов

Значимые переменные отмечены словом «Да». Использование алгоритма снижения размерности уменьшает количество независимых переменных до 10.

В дальнейшем будем использовать эти 10 переменных в качестве независимых предикторов. В стартовом окне задаем новый набор независимых переменных (рис. 2.30).

В качестве инструмента построения сети будем использовать *Мастер решений*. После того, как установки проделаны, нажимаем *ОК*.

Шаг 2. Выбор модели.

Вначале проведем анализ для взвешенных переменных. На вкладке *Быстрый* окна *Мастер решений* (рис. 2.31) изменим величину длительности анализа и установим ее равной 100.

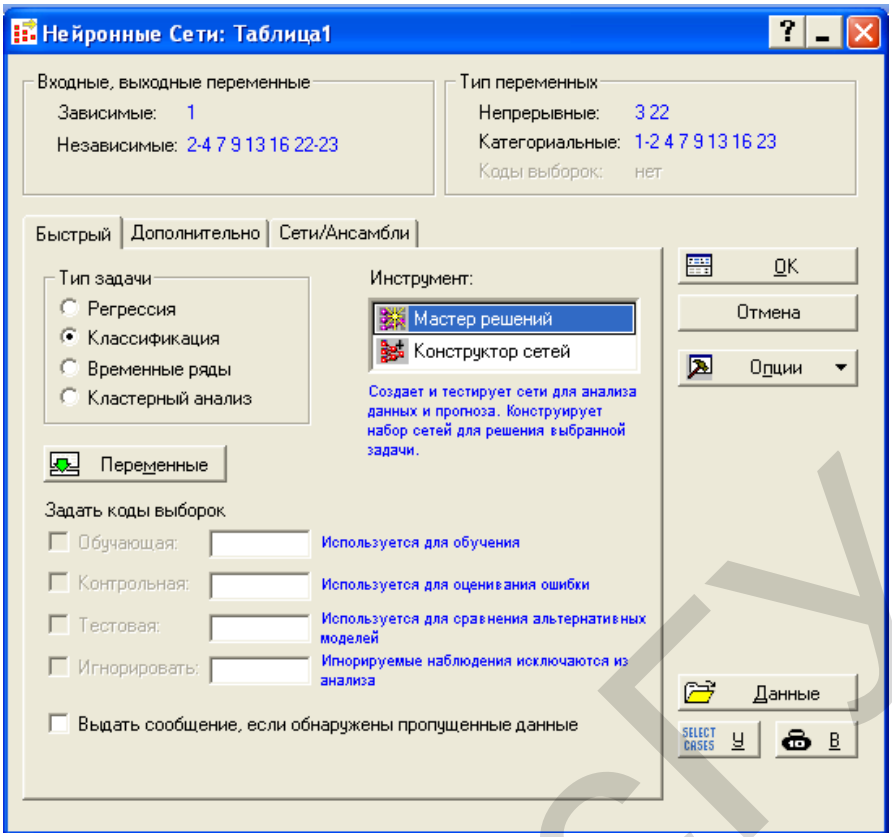


Рис. 2.30 – Стартовое окно модуля *Нейронные сети*

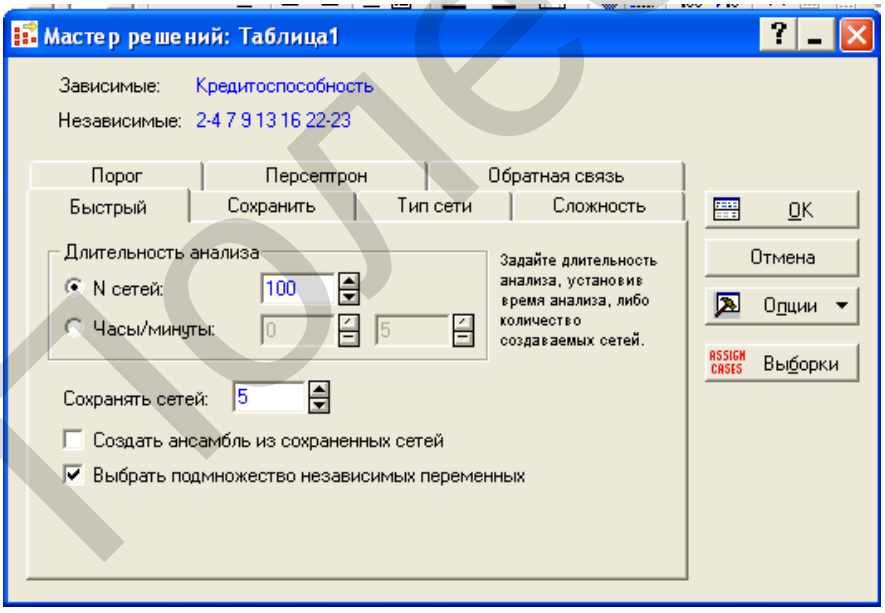


Рис. 2.31 – Окно *Мастера решений*, вкладка *Быстрый*

Для начала нам необходимо выяснить, какие модели будут работать эффективнее, поэтому, не изменяя остальных опций, нажимаем *ОК*.

Анализируя величину производительности на тестовом множестве, делаем вывод, что точность прогноза с помощью построенных моделей находится на уровне 80 % (рис. 2.32).

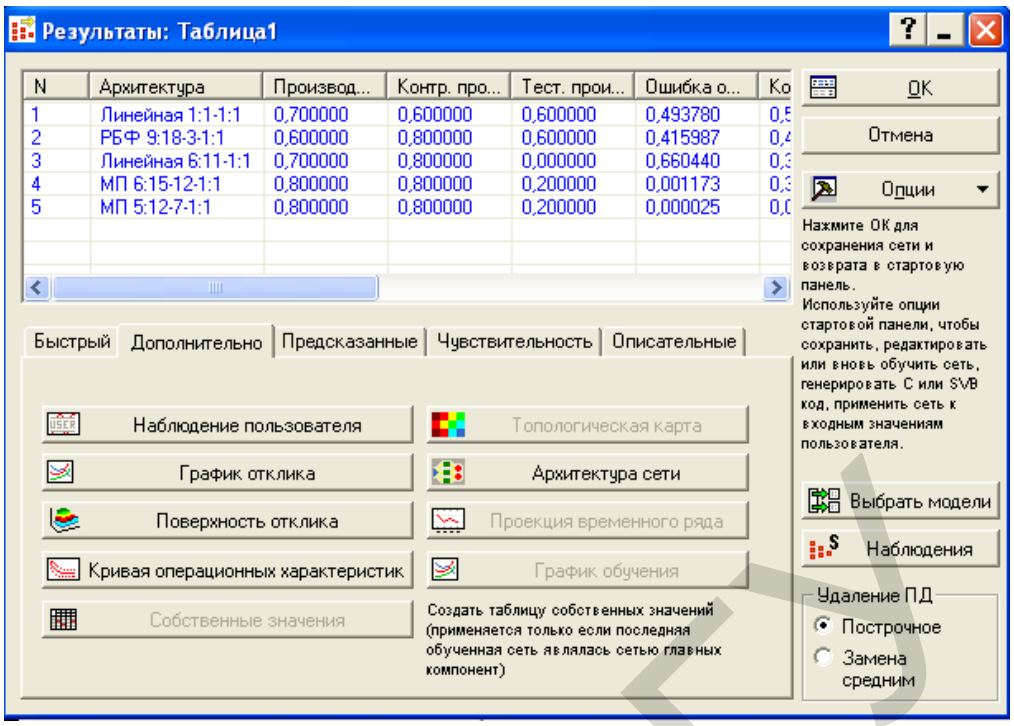


Рис. 2.32 – Окно результатов

Мы рассматриваем именно значение производительности (которая, в данном случае, интерпретируется как доля правильно классифицированных) на тестовом множестве, так как это множество не использовалось при обучении сети и предоставляет проведение анализа качества построенной модели.

Не углубляясь в дальнейшее изучение построенной модели, попробуем улучшить ее, включив в анализ переменную с весами.

Для этого на окне результатов нажимаем *ОК* и возвращаемся к стартовому окну, в котором нажимаем кнопку . В появившемся окне указываем в соответствующем поле название переменной, содержащей веса (рис. 2.33). Убедитесь, что в поле *Состояние* выбрано «ВКЛ». Нажимаем *ОК*.

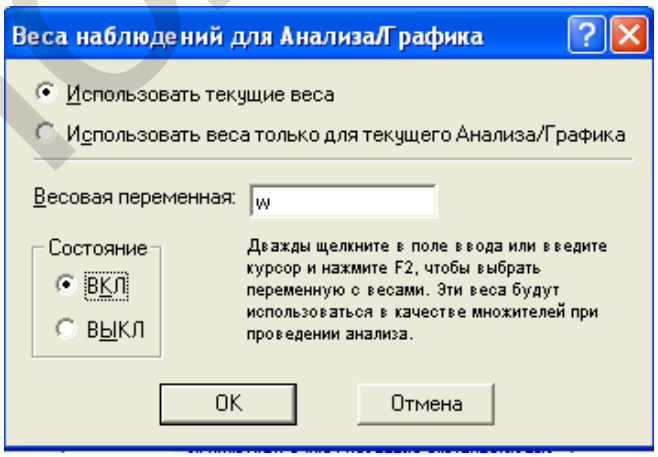


Рис. 2.33 – Задание весовой переменной

Вернувшись к стартовому окну, не изменяем сделанных ранее настроек и нажимаем *ОК* (при этом мы, как и прежде, работаем в *Мастере решений*). На следующем окне снова нажимаем *ОК* (рис. 2.34).

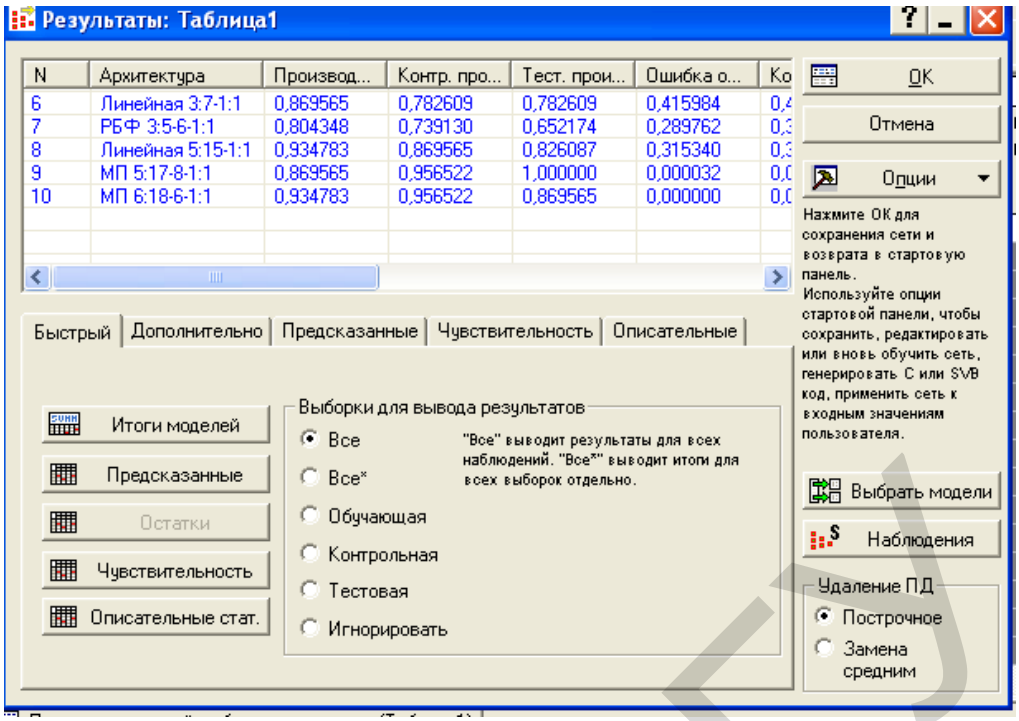


Рис. 2.34 – Окно результатов

Анализируя значения производительностей, делаем вывод о том, что сеть *Многослойного персептрона* наиболее удачно моделирует данные. Нажимаем кнопку *Выбрать модели.* На вкладке *Выбрать модели* выбираем первую сеть (рис. 2.35).

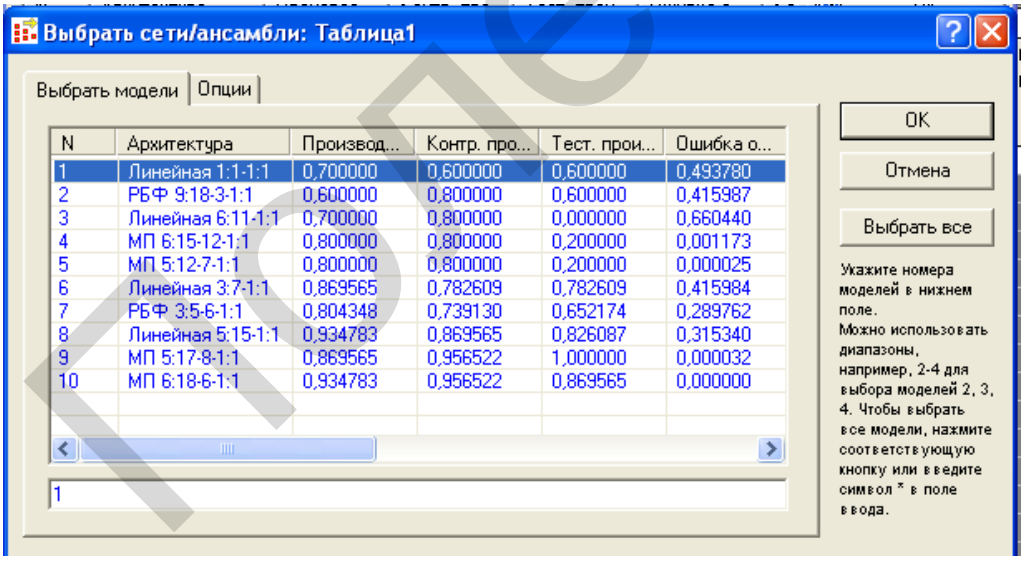


Рис. 2.35 – в Диалог выбора модели

Нажимам *ОК.* На вкладке *Быстрый* нажимаем *Описательные статистики.* Таблица статистик классификации приведена на рис. 2.36.

	Классификация (1) (Таблица1)	
	Кредитоспособность.плохой.1	Кредитоспособность.хороший.1
Всего	56,00000	36,00000
Правильно	35,00000	24,00000
Ошибочно	21,00000	12,00000
Неизвестно	0,00000	0,00000
% правильных	62,50000	66,66667
% ошибочных	37,50000	33,33333
% неизвестно	0,00000	0,00000

Рис. 2.36 – Таблица описательных статистик классификации

Процент правильно классифицированных в категорию «плохой» равен 98 %, а в категорию «хороший» – 96,7 %.

Шаг 3. Анализ результатов.

При выборе модели мы уже построили таблицу описательных статистик, поэтому проанализируем остальные результаты. На вкладке *Быстрый* пользователь может вывести таблицу *Предсказанных значений* (рис. 2.37).

	Предсказание (1) (Таблица1)	
	Кредитоспособность	Кредитоспособность.1
1	плохой	плохой
1	плохой	плохой
1	плохой	плохой
1	плохой	плохой
1	плохой	плохой
1	плохой	плохой
1	плохой	плохой
2	хороший	плохой
2	хороший	плохой
2	хороший	плохой
3	плохой	плохой
3	плохой	плохой
3	плохой	плохой
3	плохой	плохой
3	плохой	плохой
3	плохой	плохой
3	плохой	плохой
4	хороший	хороший
4	хороший	хороший

Рис. 2.37 – Таблица предсказанных значений

В случае необходимости в данной таблице помимо наблюдаемых и предсказанных значений зависимой переменной можно вывести независимые переменные, переменную, содержащую идентификатор выборки, а также переменные из исходной таблицы данных, которые не были включены в модель. Для задания перечисленных опций необходимо перейти на вкладку *Предсказанные* (рис. 2.38).

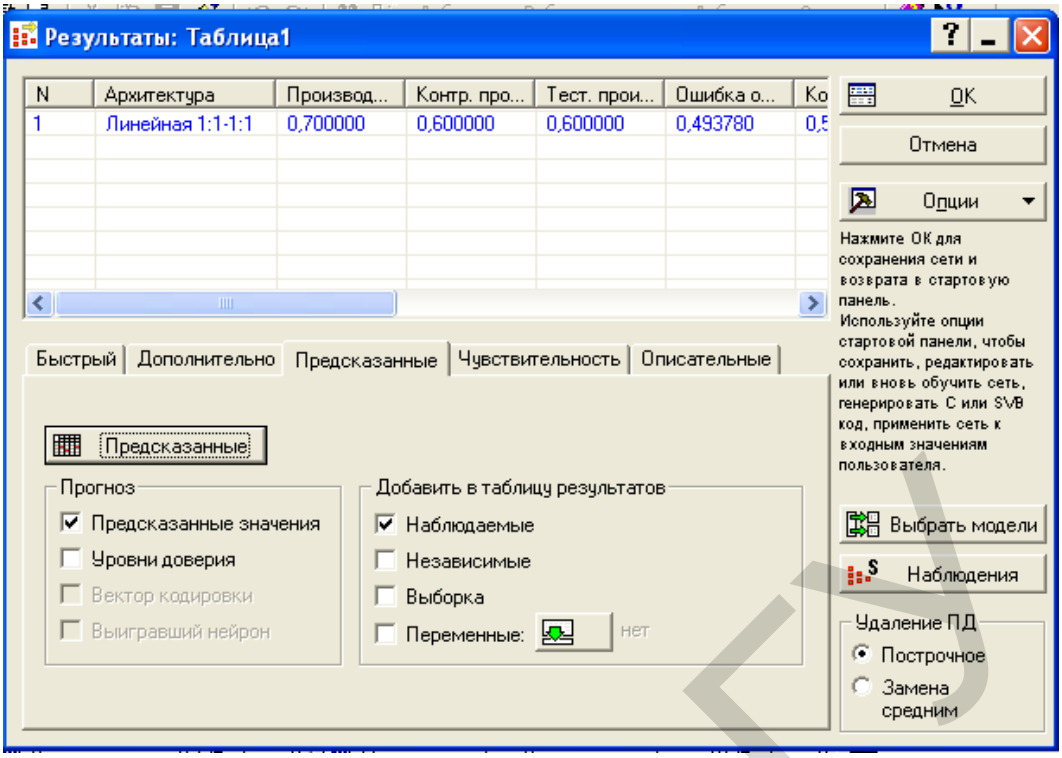


Рис. 2.38 – Окно результатов, вкладка *Предсказанные*

Для нейронных сетей с категориальными результирующими переменными доступна опция отображения уровней доверия. При помощи этой опции можно отобразить уровни доверия для нейронной сети в различных классах, которые представляются уровнями активации выходных нейронов, соответствующих итоговой переменной.

Уровни активации выходного нейрона предоставляют доверительные оценки для выходных классов. Иногда становится возможно интерпретировать подобные уровни в качестве вероятностей. Если в сочетании с правильно выбранной функцией ошибки используется правильная функция активации во время процедуры оптимизации, то можно выполнить подобную интерпретацию. Например, кросс-энтропийная функция ошибки комбинируется с логистической функцией активации для двухвариантных задач или с функцией максимума для многовариантных задач.

Энтропийный подход соответствует оптимизации максимального правдоподобия, предполагая, что данные взяты из экспоненциального семейства распределений. Важное свойство заключается в том, что выходы можно интерпретировать в качестве апостериорных оценок вероятностей членов классов.

Альтернативный подход заключается в использовании среднеквадратичной функции ошибки вместе с логистической функцией активации. Однако он имеет меньше статистической достоверности – нейронная сеть обучается с использованием дискриминантной функции, и хотя выходы можно интерпретировать как доверительные значения, они не являются оценками вероятностей (их сумма может быть не равна 1,0). С другой стороны, подобные сети обучаются намного быстрее и стабильнее (на их основе можно достигнуть больших долей правильных классификаций).

При отображении уровней доверия таблица *Предсказанных* принимает вид, показанный на рис. 2.39.

	Кредитоспособность	Кредитоспособность.1	Кредитоспособность.хороший.1
1	плохой	плохой	0,442308
1	плохой	плохой	0,442308
1	плохой	плохой	0,442308
1	плохой	плохой	0,442308
1	плохой	плохой	0,442308
1	плохой	плохой	0,442308
1	плохой	плохой	0,442308
2	хороший	плохой	0,442308
2	хороший	плохой	0,442308
2	хороший	плохой	0,442308
3	плохой	плохой	0,442308
3	плохой	плохой	0,442308
3	плохой	плохой	0,442308
3	плохой	плохой	0,442308
3	плохой	плохой	0,442308
3	плохой	плохой	0,442308
3	плохой	плохой	0,442308
4	хороший	хороший	0,524725
4	хороший	хороший	0,524725

Рис. 2.39 – Таблица *Предсказанных* с уровнями доверия

При масштабировании уровней доверия станет возможным интерпретировать их как «скор» данного клиента.

3 ЗАДАНИЯ ПО САМОСТОЯТЕЛЬНОМУ ОСВОЕНИЮ ПОСТРОЕНИЯНЕЙРОННЫХ СЕТЕЙ

1. Выбрать тему финального проекта согласовать ее с преподавателем.

Например, согласно перечню тем:

– Моделирование и проектирование нейронной сети учета деятельности системы «Терминал приема платежей» с помощью CASE-средств.

– Моделирование и проектирование нейронной сети учета деятельности гребной базы с помощью CASE-средств.

– Моделирование и проектирование нейронной сети «Разработка ПО для оптимизации складского учета» с помощью CASE-средств.

– Моделирование и проектирование нейронной сети учета деятельности строительной компании с помощью CASE-средств.

– Моделирование и проектирование нейронной сети учета валютных операций в банке с помощью CASE-средств.

– Моделирование и проектирование нейронной сети учёта деятельности интернет-сервиса по созданию сайтов с помощью CASE-средств.

– Моделирование и проектирование нейронной сети аналитического отдела банка с помощью CASE-средств.

– Моделирование и проектирование нейронной сети учета ценных бумаг с помощью CASE-средств.

– Моделирование и проектирование нейронной сети поиска маршрутов городского транспорта с помощью CASE-средств.

– Моделирование и проектирование нейронной сети учета деятельности грузоперевозок товаров с помощью CASE-средств.

– Моделирование и проектирование нейронной сети учёта деятельности отдела маркетинга предприятия с помощью CASE-средств.

– Моделирование и проектирование нейронной сети букмекерской конторы с помощью CASE-средств.

– Моделирование и проектирование нейронной сети «Банк Инноваций» с помощью CASE-средств.

– Моделирование и проектирование нейронной сети системы поиска вакансий с помощью CASE-средств.

– Моделирование и проектирование нейронной сети учета инновационных продуктов региона с помощью CASE-средств.

– Моделирование и проектирование нейронной сети учета грузоперевозок логистического центра с помощью CASE-средств.

2. ***Построить таблицу данных, с числовыми показателями характеризующими процесс*** (рис. 3.1).

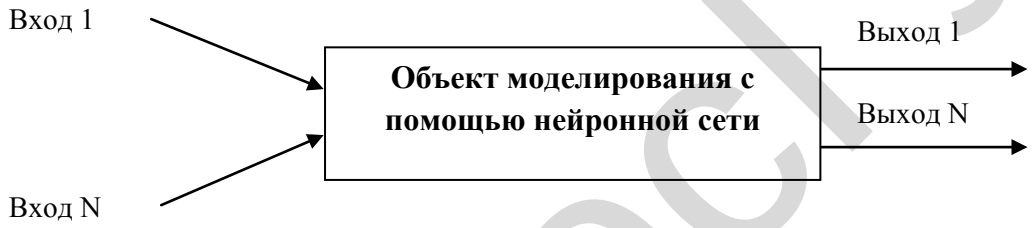


Рис. 3.1 – Структура данных для построения нейронной сети

3. Выполнить понижение размерности таблицы согласно примера пункта 2.1.

4. Выполнить распознавание образов согласно примера пункта 2.2.

5. Синтезировать систему поддержки принятия решений, исходя из примера скоринговой модели пункта 2.3.

6. Сделать выводы об адекватности созданной нейронной сети.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Мацяшек, Л.А. Анализ требований и проектирование систем: разработка информационных систем с использованием UML : научно-популярная литература / Л.А. Мацяшек; пер. с англ. – М. : ИД Вильямс, 2002. – 432 с.
2. Нейрокомпьютеры и интеллектуальные роботы / Амосов Н.М., Байдык Т.Н., Гольцев А.Д. и др. – М. : АН УССР Ин-т кибернетики. – К. : Наукова думка, 1991. – 272 с.
3. Малпас Д. Реляционный язык ПРОЛОГ и его применение / Д. Малпас. – М. : Наука, 1990. – 464 с.
4. Адаменко А.Н. Логическое программирование и Visual Prolog / А.Н. Адаменко, А. М. Кучуков. – СПб. : БХВ-Петербург, 2003. – 993 с.
5. Усков А.А. Интеллектуальные технологии управления. Искусственные нейронные сети и нечеткая логика / А.А. Усков, А. В. Кузьмин. – М. : Горячая линия – Телеком, 2004. – 143 с.
6. Рутковский Л. Методы и технологии искусственного интеллекта / Л. Рутковский. – М. : Горячая линия – Телеком, 2010. – 520 с.
7. Ясницкий Л.Н. Введение в искусственный интеллект / Л.Н. Ясницкий. – М. : Изд. центр «Академия», 2008. – 176 с.
8. Трофимов, С.А. CASE-технологии : практическая работа в Rational Rose : научно- техническое издание / С.А. Трофимов. – 2-е изд. – М. : Бином – Пресс, 2002. – 288 с.
9. Грабауров, В.А. Информационные технологии для менеджеров : производственно-практическое издание / В.А. Грабауров. – М. : Финансы и статистика, 2001. – 368 с.
10. Пилецкий, И.И. Проектирование, разработка и сопровождение баз данных с использованием CASE-средств: пособие по курсу «Методы и технологии программирования» для студентов специальности 1-31 03 04 «Информатика» всех форм обучения / И.И. Пилецкий ; Министерство образования Республики Беларусь, Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники». – Минск : БГУИР, 2009. – 116 с.
11. Нейроні мережі : навч. посіб. / М.О. Корчемний, В.П. Лисенко, М.В. Чапний, В.М. Штепа. – К. : «Аграр Медіа Груп», 2010. – 136 с.
12. Ручкин В.Н. Универсальный искусственный интеллект и экспертные системы / В.Н. Ручкин, В.А. Фулин. – СПб. : БХВ-Петербург, 2009. – 240 с.
13. Озерова, В.П. Менеджер и управленческая информация : учебное пособие по курсу «Компьютерные информационные технологии» : Для студентов заочной формы обучения / В.П. Озерова; Министерство образования Республики Беларусь, Белорусский государственный экономический университет. – Минск : БГЭУ, 2003. – 51 с.
14. Кратчен, Ф. Введение в Rational Unified Process. 2-е изд.; пер. с англ. – М. : Издательский дом «Вильямс», 2002. – 240 с.
15. Нейронные сети. STATISTICA Neural Networks : Методология и технологии современного анализа данных / Под редакцией В.П. Боровикова. – 2-е изд., перераб. и доп. – М. : Горячая линия – Телеком, 2008. – 288 с.
16. Круглов, В.В. Нечеткая логика и искусственные нейронные сети / В.В. Круглов, М.И. Дли, Р.Ю. Голунов. – М. : Физматлит, 2000. – 224 с.

Учебное издание

Штепа Владимир Николаевич
Кот Роман Евгеньевич

Распределенные информационные системы. Нейросетевые технологии

Методические рекомендации

Ответственный за выпуск *П.Б. Пигаль*

Редактор *Митянок Е.М.*

Подписано в печать 10.07.2017 г. Формат 60х84/16.
Бумага офсетная. Гарнитура «Таймс». Ризография.
Усл. печ. л. 3,60. Уч.-изд. л. 1,31.
Тираж 62 экз. Заказ № 565.

Отпечатано в редакционно-издательском отделе
Полесского государственного университета.
225710, г. Пинск, ул. Днепровской флотилии, 23.