ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ
РАБОТЫ СО ЗНАНИЯМИ
ОБНАРУЖЕНИЕ, ПОИСК, УПРАВЛЕНИЕ
(Вычислительные системы)

2008 год Выпуск 175
УДК 53.02+519.7+519.812.2

РЕАЛИЗАЦИЯ УНИВЕРСАЛЬНОЙ СИСТЕМЫ ИЗВЛЕЧЕНИЯ ЗНАНИЙ «Discovery» И ЕЕ ПРИМЕНЕНИЕ В ЗАДАЧАХ ФИНАНСОВОГО ПРОГНОЗИРОВАНИЯ[1]

А.В.Демин[2], Е.Е.Витяев[3]

Введение

В настоящее время разработано достаточно большое количество различных методов KDD&DM (Knowledge Discovery in Data Bases abd Data Mining) и реализующих их программных систем. Данное направление продолжает бурно развиваться и совершенствоваться. Однако ни один из используемых в данный момент KDD&DM–методов не способен извлечь из информации знания в полном объеме.

Анализ методов KDD&DM показывает [1–3], что для любого метода можно выделить его онтологию, включающую типы

[1]Работа выполнена при финансовой поддержке РФФИ (грант 08–07–00272–а), Интеграционными проектами СО РАН № 1,115, Совета по грантам Президента РФ и государственной поддержке ведущих научных школ (проект НШ–335.2008.1).

[2]Институт систем информатики им. А.П.Ершова СО РАН г. Новосибирск

[3]Институт математики СО РАН г.Новосибирск, e–mail: vityaev@math.nsc.ru

данных, с которыми работает метод и язык оперирования и интерпретации данных, также можно выделить класс гипотез, которые проверяет метод. Это накладывает на KDD&DM–методы ряд ограничений:

1) информация, содержащаяся в данных, определяется множеством отношений и операций, интерпретируемых в онтологии предметной области. Существующие методы KDD&DM могут работать только с конкретными типами данных и использовать только конкретные виды отношений и операций. Тем самым, они, во–первых, не могут использовать всю информацию, содержащуюся в данных, и, во–вторых, могут получать результаты, не интерпретируемые в онтологии предметной области;

2) методы обнаруживают в данных только вполне определенные типы гипотез.

Нами были разработаны реляционный подход (Relational Data Mining) к методам извлечения знаний и реализующая его программная система «Discovery» [1–4], снимающие практически все ограничения с методов KDD&DM за счет использования языка первого порядка, который практически неограниченно расширяет множество типов используемых данных, а также позволяет описывать любые виды гипотез. Проведенные практические сравнения системы «Discovery» с такими широко распространенными методами как нейронные сети, решающие деревья, ассоциативные правила, статистические методы, FOIL показывают, что система «Discovery» работает лучше и точнее других методов [1,2,5,6].

Существующие методы не в состоянии поддерживать режим исследования данных, когда обнаруживаемая закономерность заранее неизвестна. Каждый KDD&DM–метод обнаруживает свой специфический класс гипотез. Система «Discovery» способна поддерживать режим исследования данных. Кроме того, система «Discovery» может обнаружить и проверить на данных произвольный класс гипотез, который захочет проверить эксперт.

Система «Discovery» обнаруживает гипотезы, которые сформулированы в заданных экспертом (например, финансистом) терминах — множестве интерпретируемых отношений и операций, определенных на данных. Интерпретируемость получаемых закономерностей очень важна, например, для задач финансового

прогнозирования. К примеру, если речь идет о крупном вложении капитала и у нас есть два прогноза об ожидаемой прибыли, полученные нейронными системами и системой «Discovery», то доверие будет к тому прогнозу, который понятен и интерпретируем. Невозможно принимать ответственные решения, не понимая, как они получены. Нейронные сети воспринимаются как черный ящик и поэтому их прогнозу доверять трудно. Прогнозы, получаемые на основании интерпретируемых правил понятны, и по ним можно принимать решения.

Другой важной задачей, которую решает система «Discovery», является задача максимально полного извлечения знаний из данных. Полнота извлечения знаний системой «Discovery» обеспечивается двумя путями:

1) использованием теории измерений, позволяющей извлечь практически всю информацию из данных и представить ее множеством отношений и операций, определенных на многосортной эмпирической системе и интерпретируемых в онтологии предметной области;

2) обнаружением практически любого класса гипотез в терминах выявленных отношений и операций на этой эмпирической системе.

Все эти задачи показывают актуальность создания «универсальной» версии системы «Discovery». Однако ввиду чисто практической сложности такой задачи, ранее разрабатывались только ограниченные версии системы для решения конкретных задач. На данный момент разработана достаточно «универсальная» версия системы [7], основным отличием которой является то, что она позволяет пользователю самому описывать виды гипотез, при помощи которых будет осуществляться извлечение знаний.

1. Описание метода

1.1. Определение вида гипотез. Будем предполагать, что исходные данные представлены в виде реляционной таблицы D, строки которой соответствуют объектам, а колонки — признакам объектов. То есть, $D = \{D(1), \ldots, D(N)\}$, где $D(i)$ — i–я строка таблицы (объект с номером i), $D(i) = \{D(i,1), \ldots, D(i,m)\}$; $D(i,j)$ — значение таблицы на пересечении j–й колонки и i–й строки (значение j–го признака объекта $D(i)$).

Введем иерархию элементов конструирования гипотез, которые мы будем использовать для формализации способа задания видов гипотез.

Переменная по объектам пробегает множество строк (объектов) таблицы данных. В дальнейшем будем отождествлять значения переменных по объектам с номерами строк.

Будем обозначать кортеж переменных по объектам $\langle i_1, i_2, \ldots, i_n \rangle$ через $\langle i \rangle$.

Переменная-параметр (в дальнейшем будем называть этот тип переменных просто переменными) либо принимает значения фиксированной константы $x\langle i \rangle = const$, где $const$ — произвольное действительное число, либо принимает значения признаков объектов $x\langle i \rangle = D(g(\langle i \rangle), h(\langle i \rangle))$, где $g(\langle i \rangle)$ и $h(\langle i \rangle)$ — целочисленные функции, задающие номер строки и номер колонки таблицы. В простейшем случае, когда $g(\langle i \rangle) = i_j$, где $i_j \in \langle i \rangle$, $h(\langle i \rangle) = k$, переменная $x\langle i \rangle$ будет просто принимать значения k-го признака объекта i_j : $x\langle i \rangle = D(i_j, k)$. В более сложном случае $g(\langle i \rangle)$ может, к примеру, задавать смещение строки: $g(\langle i \rangle) = g(i_j) + b$, где b — фиксированное смещение относительно строки i_j, $i_j \in \langle i \rangle$, или осуществлять поиск объекта относительно текущих объектов.

Терм определяется следующим образом: 1) если $x(\langle i \rangle)$ — произвольная переменная, то $t(\langle i \rangle) = x(\langle i \rangle)$ — терм; 2) если f — n-местная веществennозначная функция, $t_1, \ldots, t_n$ — термы, то $t(\langle i \rangle) = f(t_1(\langle i \rangle), \ldots, t_n(\langle i \rangle))$ —терм. Терм может принимать любое вещественное значение.

Предикат определяет отношение на множестве данных. Предикат содержит термы, связанные этим отношением. Общий вид предиката: $P(\langle i \rangle) = P(t_1(\langle i \rangle), \ldots, t_n(\langle i \rangle))$, где $t_j(\langle i \rangle)$ — терм. Предикат может принимать значения «истина» или «ложь».

Правило служит для представления закономерности. Правило состоит из посылки и заключения. Посылка правила представляет собой конъюнкцию предикатов, заключение — некоторый целевой предикат. Общий вид правила: $\forall\langle i \rangle P_1^{\varepsilon}(\langle i \rangle)\&, \ldots, \&P_n^{\varepsilon}(\langle i \rangle) \rightarrow$ $\rightarrow P_0^{\varepsilon}(\langle i \rangle)$, где $P_1, \ldots P_n$ — предикаты посылки; P_0 — целевой предикат; $\varepsilon \in \{0, 1\}$ обозначает наличие отрицания, т.е. если $\varepsilon = 0$, то $P^{\varepsilon} = P$, если $\varepsilon = 1$, то $P^{\varepsilon} = \neg P$. Каждое правило R характери-

зуется условной вероятностью $p(R)$, с которой оно предсказывает истинность заключения при условии истинности посылки.

Введем понятие *интерпретации* объекта (переменной, терма или предиката) на множестве исходных данных. Будем обозначать интерпретацию объекта **A** через $\theta(\mathbf{A})$. Последовательно введем понятие интерпретации для каждого из вышеперечисленных объектов.

Под *интерпретацией переменной* будем понимать присвоение ей значения фиксируемой константы, либо присвоение значений признаков объектов. То есть, $\theta(x) = const$, где $const$ — произвольное действительное число, либо $\theta(x) = D(g(\langle i \rangle), h(\langle i \rangle))$, где $g(\langle i \rangle)$ и $h(\langle i \rangle)$ — целочисленные функции, задающие номер строки и номер столбца таблицы данных. Будем называть переменную *проинтерпретированной*, если она равна константе, либо ссылается на исходную таблицу данных.

Под *интерпретацией терма* будем понимать присвоение всем переменным, входящим в состав терма, значений фиксированных констант или признаков объектов. То есть, если $t = x$, где x — переменная, то $\theta(t) = \theta(x)$; если $t = f(t_1, \ldots, t_n)$, где $t_1, \ldots, t_n$ — термы, то $\theta(t) = f(\theta(t_1), \ldots, \theta(t_n))$.

По аналогии, под *интерпретацией предиката* будем понимать присвоение всем его термам значений проинтерпретированных функций, т.е. $\theta(P(t_1, \ldots, t_m)) = P(\theta(t_1), \ldots, \theta(t_m)) = P(f_1, \ldots$ $\ldots, f_m)$. Соответственно будем называть предикат *проинтерпретированным*, если все его термы проинтерпретированы.

Введем понятия *шаблона термов* и *шаблона предикатов*.

Шаблон термов задает терм и способы его интерпретации на исходных данных. Определим шаблон терма Tf для терма t следующим образом. Если $t = x$, где x — переменная, то $Tf\langle x, \Theta(x) \rangle$, где $\Theta(x)$ — множество интерпретаций переменной x на исходных данных: $\Theta(x) = \{\theta_1(x), \ldots, \theta_m(x)\}$. Если $t = f(t_1, \ldots$ $\ldots, t_n)$, где $t_1, \ldots, t_n$ — термы, то $Tf = \langle f(t_1, \ldots, t_n)$, $\Theta(t_1), \ldots, \Theta(t_n) \rangle$, где $\Theta(t_i)$ — множество интерпретаций терма t_i на исходных данных: $\Theta(t_i) = \{\theta_1(t_i), \ldots, \theta_m(t_i)\}$. Поскольку способы интерпретации термов задаются шаблонами термов, то мы можем определить множество интерпретаций терма $\Theta(t)$ через

другие шаблоны термов, т.е. $\Theta(t) = \{Tf_1, \ldots, Tf_k\}$, где Tf_i — некоторые шаблоны термов.

Приведем **пример**. Пусть k–я колонка таблицы данных содержит значения некоторого временного ряда (к примеру, цена закрытия акции). Рассмотрим шаблон термов

$$\langle x_1 - x_2, \Theta(x_1) = \{D(i,k)\}, \Theta(x_2) = \\ = \{D(i-1,k), D(i-2,k), D(i-3,k)\}\rangle.$$

Легко видеть, что данный шаблон определяет три проинтерпретированные функции, задающие временные лаги от 1 до 3:

1) $D(i,k) - D(i-1,k)$,
2) $D(i,k) - D(i-2,k)$,
3) $D(i,k) - D(i-3,k)$.

Шаблон предикатов задает предикат и способы его интерпретации на исходных данных. Обозначим через $\langle P(t_1, \ldots, t_n), \Theta(t_1), \ldots, \Theta(t_n)\rangle$ шаблон предикатов, где $\Theta(t_i)$ — множество интерпретаций терма t_i на исходных данных, которое задается шаблонами термов: $\Theta(t_i) = \{Tf_1, \ldots, Tf_k\}$, где Tf_i — некоторые шаблоны термов. Таким образом, шаблон предикатов, по сути, задает класс предикатов, определяющих один вид отношения, но по–разному, проинтерпретированных на исходных данных.

Приведем **пример задания шаблона предикатов**. Предположим, к примеру, что k–я колонка таблицы данных представляет временной ряд со значениями дневной цены закрытия какой–нибудь ценной бумаги. Предположим, что мы хотим определить множество предикатов, сравнивающих друг с другом цены закрытия последних пяти дней. Это можно сделать, определив следующий шаблон предикатов: $\langle t_1 < t_2, \Theta(t_1) = \{Tf\}, \Theta(t_2) = \\ = \{Tf\}\rangle$, где $Tf = \langle x, \Theta(x) = \{D(i,k), D(i-1,k), \ldots, D(i-4,k)\}\rangle$. В данном примере шаблон термов Tf задает пять тождественных функций: $x(i) = D(i,k), x(i) = D(i-1,k), \ldots, x(i) = D(i-4,k)$. Таким образом, данный шаблон предикатов задает множество предикатов вида: $D(i-b_1,k) < D(i-b_2,k)$, где $b_1, b_2 \in \{0, -1, \ldots, -4\}$.

Теперь, используя понятие шаблона предикатов, мы можем определить понятие *класса гипотез* как набор множества шаблонов предикатов и целевого предиката. Обозначим класс гипотез

через $\langle\{Tp_1,\ldots,Tp_m\}P_0^\varepsilon\rangle$, где Tp_i — шаблоны предикатов, P_0 — целевой предикат, $\varepsilon \in \{0,1\}$ обозначает наличие отрицания предиката.

Класс гипотез $\langle\{Tp_1,\ldots,Tp_m\},P_0^\varepsilon\rangle$ определяет класс правил вида $P_1^\varepsilon\&\ldots\&P_n^\varepsilon \to P_0^\varepsilon$, где все предикаты посылки $P_1,\ldots,P_n$ должны принадлежать множеству предикатов, определяемому шаблонами $Tp_1,\ldots,Tp_m$.

1.2. Алгоритм поиска вероятностных закономерностей. Не ограничивая общности, рассмотрим алгоритм поиска закономерностей только для случая, когда задан только один класс гипотез. Для случая, когда задано несколько классов гипотез, поиск закономерностей осуществляется независимо для каждого класса гипотез.

Пусть задан некоторый класс гипотез $Th = \langle\{Tp_1,\ldots,Tp_m\}, P_0^\varepsilon\rangle$.

Пусть $\{P_1,\ldots,P_n\}$ — множество всех проинтерпретированных предикатов, которые мы можем получить с помощью шаблонов $\{Tp_1,\ldots,Tp_m\}$.

Через $U(Th) = \{A_1,\ldots,A_m\}$ обозначим множество всех литер вида $A_i = P^\varepsilon$, $P \in U(Th)$, $\varepsilon \in \{0,1\}$ обозначает наличие отрицания, т.е. если $\varepsilon = 0$, то $P^\varepsilon = P$, если $\varepsilon = 1$, то $P^\varepsilon = \neg P$; $A_0 = P_0^\varepsilon$ — целевое высказывание.

Вероятностной закономерностью [1,8,9] будем называть правило $A_1\&\ldots\&A_m \to A_0$, удовлетворяющее следующим условиям:

а) условная вероятность $p(A_0|A_1\&\ldots\&A_m)$ правила определена, т.е. $p(A_1\&\ldots\&A_m) > 0$;

б) условная вероятность $p(A_0|A_1\&\ldots\&A_m)$ правила строго больше условных вероятностей каждого из его подправил, т.е. для любого правила $A_{i_1}\&\ldots\&A_{i_k} \to A_0$ такого, что $\{A_{i_1},\ldots,A_{i_k}\} \subset \{A_1,\ldots,A_m\}$, условная вероятность $p(A_0|A_{i_1}\&\ldots\&A_{i_k}) < p(A_0|A_1\&\ldots\&A_m)$.

Чтобы проверить при помощи обучающего множества D, является ли некоторое правило $A_1\&\ldots\&A_m \to A_0$ вероятностной закономерностью, необходимо проверить выполнимость вероятностных неравенств "а" и "б" и оценить его статистическую значимость.

Условная вероятность правила $A_1\&\ldots\&A_m \to A_0$ оценивается на обучающем множестве D следующим образом:

$$p(A_0|A_1\&\ldots\&A_m) = \frac{N(A_0\&A_1\&\ldots\&A_m)}{N(A_1\&\ldots\&A_m)},$$

где $N(A_0\&A_1\&\ldots\&A_m)$ — число событий $A_0\&A_1\&\ldots\&A_m$ на множестве D; $N(A_1\&\ldots\&A_M)$ — число событий $A_1\&\ldots\&A_m$ на D.

Для проверки статистической значимости правила используется статистический критерий Фишера (точный критерий независимости Фишера для таблиц сопряженности) [10]. Если правило удовлетворяет этому критерию с некоторым доверительным условием α а также удовлетворяет условиям "а" и "б", то оно будет являться вероятностной закономерностью.

Алгоритм поиска закономерностей основан на семантическом вероятностном выводе [1,8,9], который позволяет находить все статистически значимые вероятностные закономерности вида $A_1\&\ldots\&A_m \to A_0$.

Для дальнейшего описания введем несколько определений.

Длиной правила R будем называть величину $len(R)$, равную количеству литер, входящих в посылку правила.

Правило $A_1\&\ldots\&A_m\&A_{m+1} \to A_0$ является уточнением правила $A_1\&\ldots\&A_m \to A_0$, если оно получено добавлением в посылку правила $A_1\&\ldots\&A_m \to A_0$ произвольной литеры A_{m+1}.

Будем обозначать через $Spec(RUL)$ множество уточнений всех правил из RUL, где RUL — произвольное множество правил вида $A_1\&\ldots\&A_m \to A_0, A_i \in U(Th)$.

Опишем алгоритм поиска закономерностей, реализующий семантический вероятностный вывод.

Пусть d — заданная исследователем глубина начального перебора.

• На первом шаге генерируем множество RUL_1 всех правил единичной длины, имеющих вид $R = A_i \to A_0$, $A_i \in U(Th)$, $len(R) = 1$. Все правила из RUL_1 проходят проверку на выполнение условий для вероятностных закономерностей. Правила, прошедшие проверку, будут являться вероятностными закономерностями. Обозначим через REG_1 множество всех вероятностных закономерностей, обнаруженных на первом шаге.

То есть $REG_1 = \{R_i\}$, где $i \in I_1$, $R_i = A_j \to A_0$, $A_j \in U(Th)$, $len(R_i) = 1$, R_i — вероятностная закономерность.

• На шаге $k \leq d$ генерируется множество RUL_k всех уточнений правил, сгенерированных на предыдущем шаге, $RUL_k = Spec(RUL_{k-1})$. Все правила из RUL_k проходят проверку на выполнение условий для вероятностных закономерностей. Обозначим через REG_k множество всех вероятностных закономерностей, обнаруженных на данном шаге. То есть $REG_k = \{R_i\}$, где $i \in I_k$, $R_i = A_1 \& \ldots \& A_k \to A_0$, $A_j \in U(Th), len(R_i) = k$, R_i — вероятностная закономерность.

• На шаге $l > d$ генерируется множество RUL_l всех уточнений всех вероятностных закономерностей, обнаруженных на предыдущем шаге, $RUL_l = Spec(REG_{l-1})$. Все правила из RUL_l проходят проверку на выполнение условий для вероятностных закономерностей. Обозначим через REG_l множество всех вероятностных закономерностей, обнаруженных на данном шаге. То есть $REG_l = \{R_i\}$, где $i \in I_l$, $R_i = A_1 \& \ldots \& A_l \to A_0$, $A_j \in U(Th)$, $len(R_i) = l$, R_i — вероятностная закономерность.

• Алгоритм останавливается, когда невозможно далее установить ни одно правило, т.е. когда $RUL_l = Spec(REG_{l-1}) = REG_{l-1} = \emptyset$. Результирующее множество всех закономерностей REG будет равно объединению всех REG_l : $REG = \bigcup_l REG_l$.

Шаги алгоритма $k \leq d$ называются базовым перебором, а шаги $k > d$ — дополнительным перебором. Величина d называется глубиной базового перебора и является параметром алгоритма.

На рис.1 представлено дерево вывода, соответствующее описанному процессу.

Проверка правил на выполнение условий для вероятностных закономерностей осуществляется путем проверки описанных выше статистических критериев с некоторым доверительным уровнем α.

1.3. Формирование прогноза и принятие решения. Будем предполагать, что исходная задача может быть представлена как задача выбора одного варианта исхода из заранее известного набора исходов.

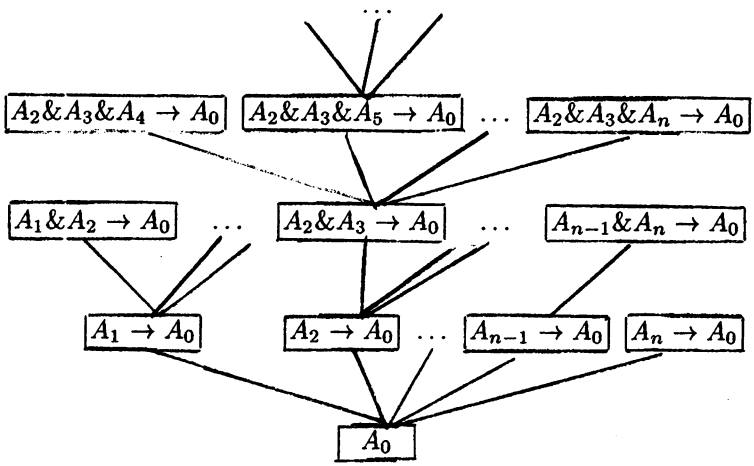


Рис.1. Дерево семантического вероятностного вывода

Под прогнозом будем понимать высказывание о варианте исхода с некоторой оценкой его точности, которую мы будем называть *оценкой точности прогноза.*

В качестве оценки точности прогноза наиболее естественно использовать оценку его вероятности, однако в некоторых задачах могут быть использованы и другие способы оценки точности прогноза, больше соответствующие специфики задачи. К примеру, в финансовых задачах вместо вероятности может быть использована величина, описывающая прибыльность.

Под принятием решения на основе прогноза будем понимать выбор одного варианта исхода, основываясь на прогнозах всех известных вариантов исхода.

Опишем способ получения прогноза и механизм принятия решения, основанный на множестве правил, предсказывающих различные варианты исходов.

Пусть нам известно множество вариантов исхода $\{исход_1, \ldots, исход_n\}$. Каждый вариант исхода $исход_i$ можно представить некоторым целевым предикатом TP_i. Таким образом, будем считать, что нам задан некоторый набор целевых предикатов $\{TP_1, \ldots, TP_n\}$ представляющих различные варианты исхода.

Пусть нам дано множество правил вида $P_1^{\varepsilon}\&\ldots\&P_n^{\varepsilon} \to TP_j$, предсказывающих целевые предикаты $\{TP_1,\ldots,TP_n\}$. Поскольку для одного и того же объекта могут одновременно сработать несколько правил с разной вероятностью предсказывающих различные варианты исхода, то необходимо определить: 1) способ формирования итогового прогноза каждого исхода на основе прогнозов отдельных правил; 2) механизм принятия решений на основе прогнозов всех вариантов исходов.

Пусть PR — множество правил, предсказывающих один и тот же целевой предикат (вариант исхода). Будем называть это множество правил *предиктором*. Таким образом, для каждого варианта исхода *исход*$_i$ мы будем иметь предиктор PR_i, состоящий из множества правил, предсказывающих данный вариант исхода (целевой предикат, соответствующий данному варианту исхода).

Определим способ формирования прогноза предиктора на основе множества прогнозов отдельных правил, входящих в его состав. Для этого необходимо опередилить способ формирования оценки точности прогноза предиктора.

Под оценкой точности прогноза предиктора PR для объекта с номером i будем понимать величину $pr_{PR}(i) \in [0,1]$, где pr_{PR} — отображение, определяющее способ формирования итогового прогноза. Отображение pr_{PR} ставит в соответствие множеству прогнозов отдельных правил значение из интервала [0,1], т.е. $pr_{PR}(i) : \{pr_R(i) : R \in PR\} \to [0,1]$, где pr_R — прогноз правила R для объекта с номером i : $pr_R(i) = p(R)$, если правило R применимо к объекту с номером i, и $pr_R(i) = 0$ в противном случае.

Наиболее естественным способом определения $pr_{PR}(i)$ является его задание равным оценки точности прогноза правила, имеющего максимальную оценку точности прогноза, т.е. $pr_{PR}(i) = \max\limits_R\{pr_R(i) : R \in PR\}$. Однако, в зависимости от выбранного способа оценки точности прогноза, возможны и другие способы задания $pr_{PR}(i)$.

Таким образом, прогнозом исхода *исход*$_j$ будем считать прогноз предиктора PR_j, содержащего множество правил, предсказывающих данный исход.

Для осуществления принятия решения необходимо определить решающее правило $DecRule(i)$, которое должно на основе множества прогнозов отдельных предикторов выбирать конкретный вариант исхода из множества возможных исходов, т.е.

$$DecRule(i): \{pr_{PR}(i)\} \to \{исход_1, \ldots, исход_n\},$$

где $\{исход_1, \ldots, исход_n\}$ — множество возможных исходов.

Для осуществления принятия решений на основе прогнозов предикторов предлагается использовать следующий механизм определения решающего правила. Пусть имеется набор предикторов $\{PR_j\}$, $j = 1, \ldots, n$. Каждый предиктор PR_j соответствует некоторому варианту исхода $исход_j$. Обозначим через $pr^j_{PR}(i)$ оценку точности прогноза j–го предиктора для объекта с номером i. Выбор варианта исхода $DecRule(i)$ для i–го объекта осуществляется следующим образом. Для каждого предиктора рассчитывается показатель согласованности его прогноза по формуле

$$Ctr_j(i) = pr^j_{PR}(i) - \max_{k \neq j}\{pr^k_{PR}(i)\},$$

т.е. как разность между оценкой точности прогноза данного предиктора и максимальной оценки точности прогнозов остальных предикторов. В качестве варианта исхода для i–го объекта выбирается исход, соответствующий предиктору, показатель согласованности которого строго больше заданного порога $\delta > 0$, т.е. $DecRule(i) = исход_k$, где $k = \arg\max_{j=1,\ldots,n}\{Ctr_j(i) : Ctr_j(i) > \delta\}$. Порог δ будем называть *порогом согласованности*. В случае, если не существует прогноза, показатель согласованности которого выше указанного порога, то решение о выборе варианта исхода не принимается. Величина порога δ зависит от специфики решаемой задачи и должна устанавиливаться исследователем.

2. Программная система «Discovery»

2.1. Описание системы. Программная система «Discovery» предназначена для 1) извлечения знаний из данных; 2) получения непротиворечивых прогнозов и принятия решений на основе

извлеченных знаний; 3) проверки гипотез на данных. Программа разработана на языке C++ и работает в среде операционных систем MS Windows 95/98/2000/XP/NT.

Система обладает следующими функциональными возможностями:

– ввод и редактирование данных;

– ввод собственных закономерностей при помощи специального формульного редактора;

– интерактивное задание достаточно произвольного класса обнаруживаемых закономерностей при помощи специального конструктора гипотез;

– автоматический поиск закономерностей заданного класса при помощи семантического вероятностного вывода;

– определение прогнозирующих систем на основе автоматически найденных и вручную введенных закономерностях;

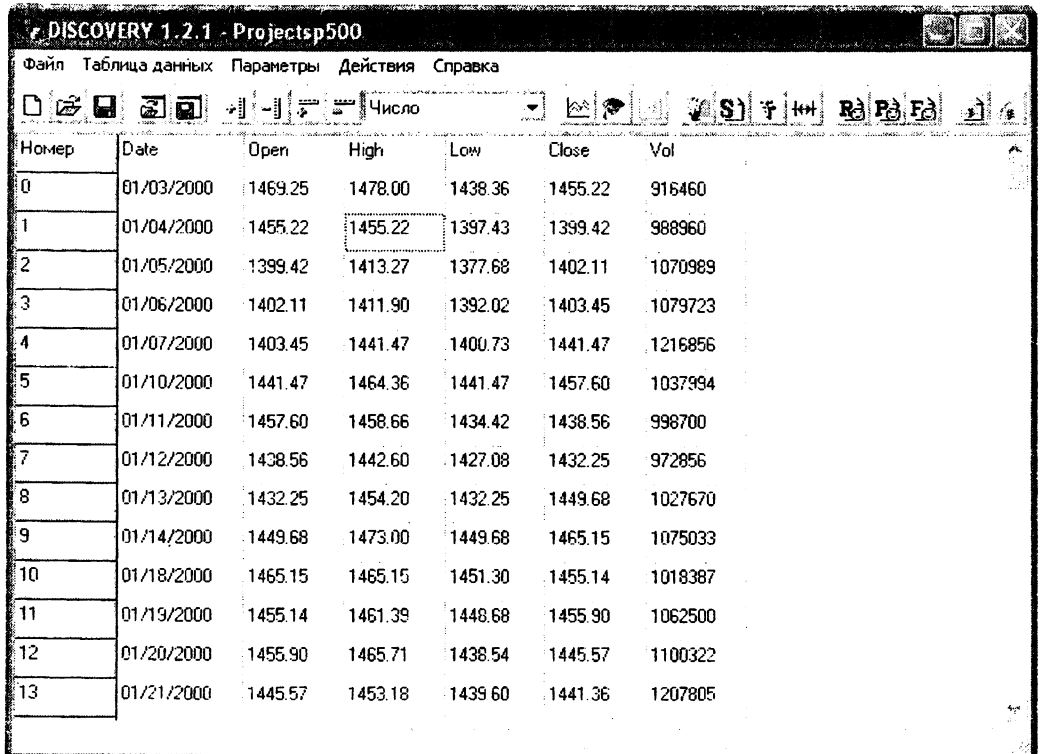
– осуществление прогноза;

– тестирование прогнозирующих систем.

Интерфейс программы построен по принципу наличия одного главного окна (см. рис.2), отображающего таблицу исходных данных, и множества подчиненных окон, несущих различные функциональные нагрузки. Подчиненные окна вызываются из главного окуа при помощи команд меню или кнопок панели инструментов.

Каждый сеанс работы с системой проходит в рамках некоторого проекта. Проект объединяет в себе анализируемые данные, набор предикторов, сформулированные классы гипотез, обнаруженные закономерности, введенные пользователем правила, параметры вычислений, а также все прочие настройки системы. То есть решение любой задачи в системе «Discovery», по сути, представляет собой работу с некоторым проектом. Пользователь может в любой момент сохранить текущий проект, чтобы в дальнейшем продолжить с ним работу.

В архитектуре системы можно выделить шесть основных функциональных блоков, на которые ложится основная нагрузка при работе пользователя: таблица данных, редактор формул, редактор предикторов, конструктор видов гипотез, редактор прогноза системы и блок тестирования прогнозов. Кроме того,



Номер	Date	Open	High	Low	Close	Vol
0	01/03/2000	1469.25	1478.00	1438.36	1455.22	916460
1	01/04/2000	1455.22	1455.22	1397.43	1399.42	988960
2	01/05/2000	1399.42	1413.27	1377.68	1402.11	1070989
3	01/06/2000	1402.11	1411.90	1392.02	1403.45	1079723
4	01/07/2000	1403.45	1441.47	1400.73	1441.47	1216856
5	01/10/2000	1441.47	1464.36	1441.47	1457.60	1037994
6	01/11/2000	1457.60	1458.66	1434.42	1438.56	998700
7	01/12/2000	1438.56	1442.60	1427.08	1432.25	972856
8	01/13/2000	1432.25	1454.20	1432.25	1449.68	1027670
9	01/14/2000	1449.68	1473.00	1449.68	1465.15	1075033
10	01/18/2000	1465.15	1465.15	1451.30	1455.14	1018387
11	01/19/2000	1455.14	1461.39	1448.68	1455.90	1062500
12	01/20/2000	1455.90	1465.71	1438.54	1445.57	1100322
13	01/21/2000	1445.57	1453.18	1439.60	1441.36	1207805

Рис.2. Главное окно программы

специально для решения финансовых задач, в системе предусмотрен блок тестирования торговых систем.

Таблица данных (рис.2) служит для отображения и редактирования данных, с которыми работает программа. Строки таблицы соответствуют объектам, а колонки — признакам объектов. Таблица данных поддерживает несколько типов данных: «число», «текст», «дата» и «сигнал».

Исходные данные для анализа могут быть загружены в таблицу данных из файла или введены вручную. Измененные в процессе работы данные могут быть в любой момент сохранены в виде файла.

Редактирование структуры таблицы данных осуществляется при помощи группы команд «Таблица данных» главного меню или нажатием соответствующих кнопок панели инструментов, кроме того, пользователь может воспользоваться всплывающим меню, которое появляется при щелчке правой кнопки мыши по таблице. Пользователь имеет возможность добавлять новые колонки и строки в таблицу данных, удалять выбранные колонки и строки из таблицы, изменять тип данных выбранной колонки.

Курсором в виде пунктирной рамки помечается выбранная в данный момент ячейка таблицы. Чтобы переместить курсор, достаточно щелкнуть левой кнопкой мыши на нужной ячейке таблицы. Двойной щелчок левой кнопкой мыши по ячейке таблицы переводит ее в режим редактирования. В режиме редактирования можно изменять содержимое ячейки, вводя нужное значение с клавиатуры.

Над любой колонкой таблицы данных может быть выполнено преобразование, при котором колонка **C** будет заполнена новыми значениями по формуле $C(i) = F(i)$, где $C(i)$ — значение колонки в строке с номером i, **F** — преобразование. В качестве преобразования может выступать произвольная функция, сигналы от срабатывания правила или предиката, прогнозы системы, прогнозы предиктора и др. К примеру, если в качестве преобразования выбрана функция, то $C(i) = f(i) = f(x_1(i), \ldots, x_n(i))$, где f — определенная пользователем функция. Если в качестве преобразования выбран прогноз системы, то $C(i) = \textit{Прогноз}(i)$, где *Прогноз*(i) — прогноз системы для объекта с номером i.

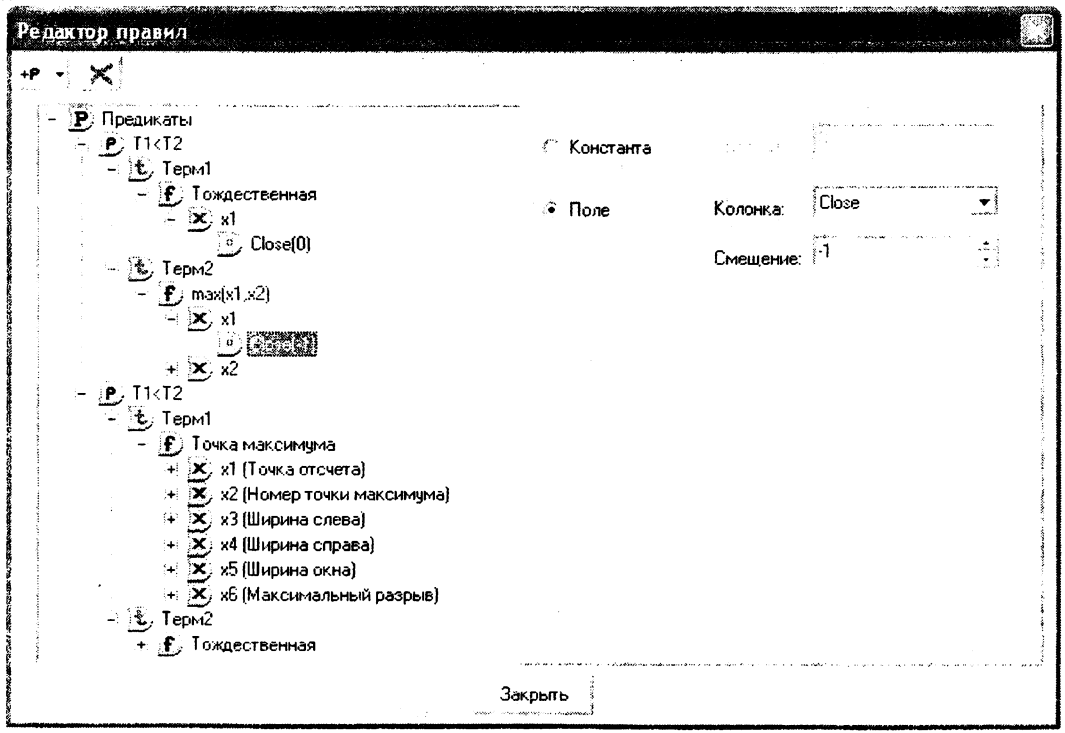


Рис.3. Редактор формул

Редактор формул вызывается каждый раз, когда пользователю необходимо вручную ввести или отредактировать правило, предикат или функцию. Элементы интерфейса окна редактора формул разбиты на группы: дерево формулы и панель свойств (см.рис.3).

Дерево формулы служит для визуального отображения формулы и навигации по ее элементам. Структура дерева полностью соответствует описанной выше иерархии элементов конструирования гипотез.

Панель свойств служит для отображения и редактирования параметров выбранного элемента формулы. Чтобы выбрать какой–либо элемент дерева достаточно щелкнуть по нему левой кнопкой мыши. При выборе элемента дерева на панели свойств будут отображены его параметры, которые могут быть отредактированы.

Редактор предикторов позволяет пользователю вводить предикторы и настраивать их параметы. В главном окне редактора (рис.4) в виде списка отображены все введенные пользователем предикторы. С помощью кнопок панели инструментов и команд всплывающего меню пользователь может добавлять новые предикторы в список, удалять выбранные предикторы из списка, вызывать окно настроек параметров или список правил выбранного предиктора.

Двойной щелчок левой кнопкой мыши по предиктору в списке либо выбор команды «Редактировать» всплывающего меню или нажатие соответствующей кнопки на панели инструментов вызывает окно редактирования данного предиктора. Окно редактирования предиктора позволяет установить целевой предикат и определить параметры семантического вероятностного вывода.

Выбрав команду «Правила» всплывающего меню или нажав соответствующую конпку на панели инструментов, пользователь может посмотреть список правил, входящих в состав выбранного предиктора. При этом будет вызвано окно, отображающее все правила, принадлежащие этому предиктору. Программа позволяет изменять любые правила в списке правил, а также добавлять новые правила в список, вводя их вручную, или удалять выбранные правила из списка. Данная особенность системы пре-

доставляет пользователю уникальную возможность объединять свои собственные знания о предметной области, выраженные в виде некоторых правил, со знаниями, автоматически обнаруженными на данных при помощи семантического вероятностного вывода.

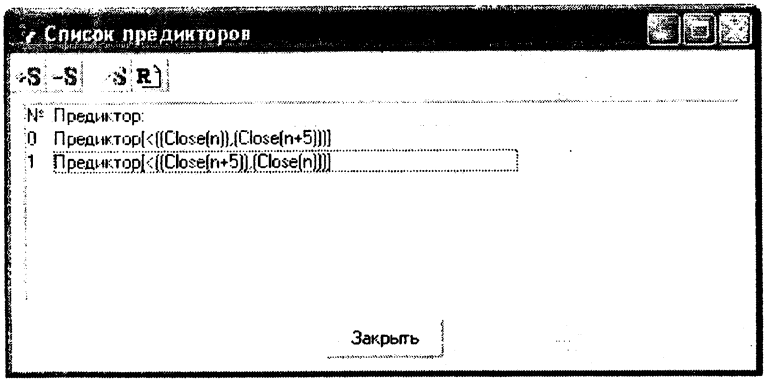


Рис.4. Редактор предикторов

Конструктор видов гипотез позволяет пользователю сформулировать общий вид гипотез относительно скрытых закономерностей в данных. Определенные таким образом классы гипотез будут в дальнейшем использовать при проведении семантического вероятностного вывод во время обучения предикторов.

Реализованная в программе и описанная выше иерархия элементов конструирования гипотез позволяет пользователю достаточно легко сформулировать классы гипотез, которые будут использованы для анализа исходных данных. Чтобы определить класс гипотез, пользователю достаточно указать виды предикатов, которые будут участвовать в формировании правил, виды функций, которые могут быть присвоены термам и множество значений, которые могут принимать переменные. В процессе обучения система, основываясь на заданном таким образом общем виде гипотез, будет генерировать уже проинтерпретированные на исходных данных предикаты и конструировать из них правила.

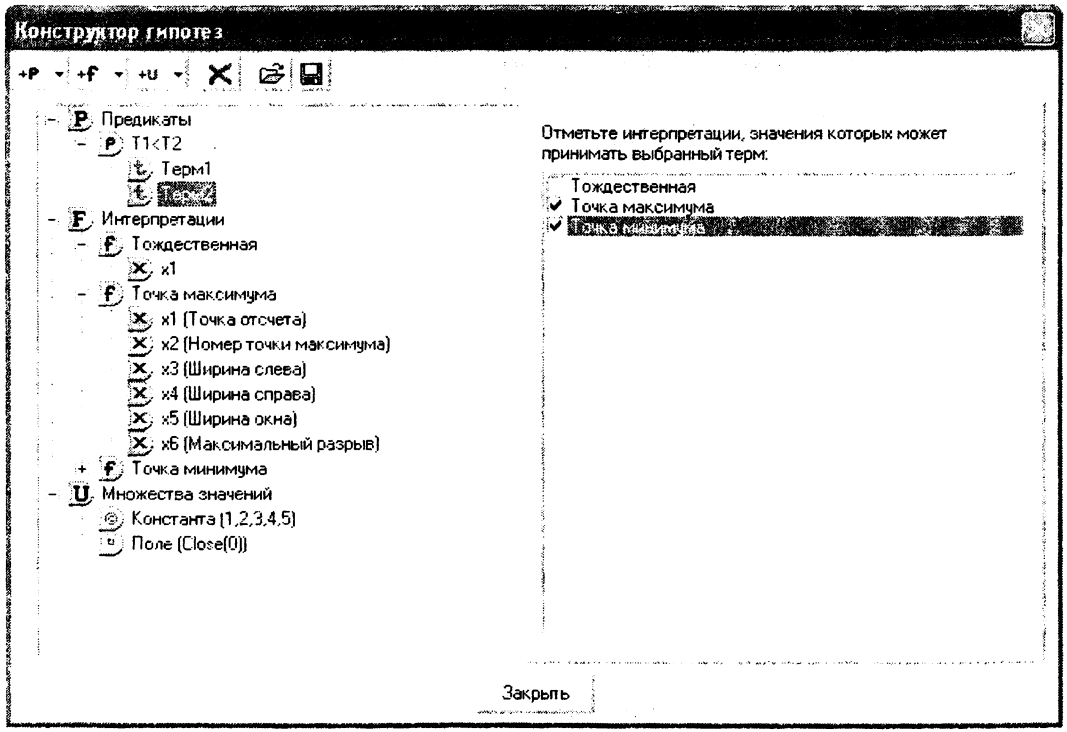


Рис.5. Коструктор видов гипотез

Элементы интерфейса окна конструктора видов гипотез разбиты на две группы: дерево элементов и панель свойств (см.рис.5). В дереве элементов в отдельные группы собраны виды предикатов, виды интерпретаций и множества значений, из которых будут конструироваться правила. Пользователь, при помощи команд всплывающего меню или кнопок панели инструментов, может добавлять в дерево новые элементы либо удалять выбранные При выборе элемента дерева на панели свойств будут отображены его свойства, которые могут быть отредактированы.

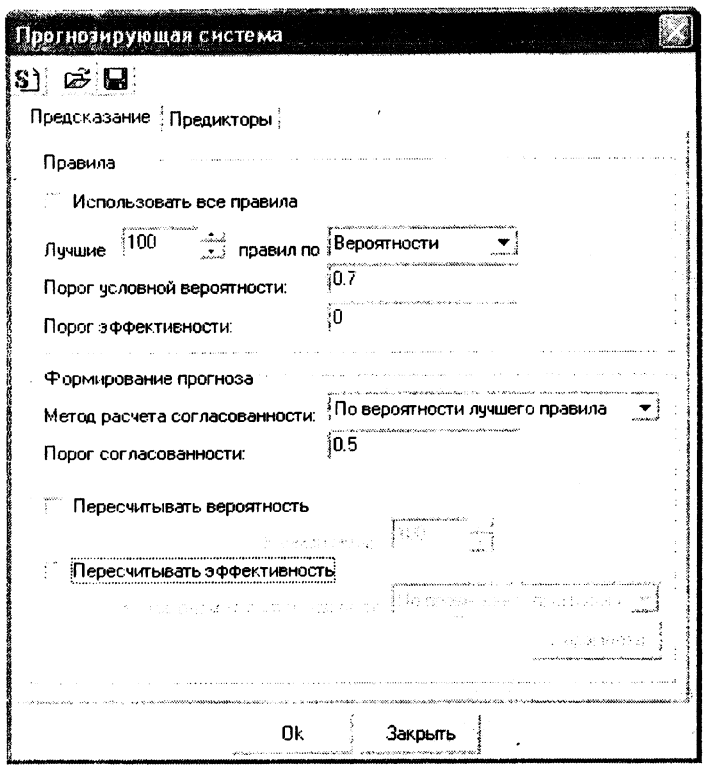


Рис.6. Редактор прогноза системы

Редактор прогноза системы позволяет определить каким образом будет формироваться итоговый прогноз системы на осно-

ве прогнозов отдельных правил. Окно редактора имеет достаточно простой вид (см.рис.6). Группа элементов «Формирование прогноза» позволяет выбрать способ формирования итоговых прогнозов предикторов и установить порог согласованности. При помощи группы элементов «Правила» можно установить критерии отбора для правил, чтобы ограничить количество правил предикторов, которые будут участвовать в прогнозировании.

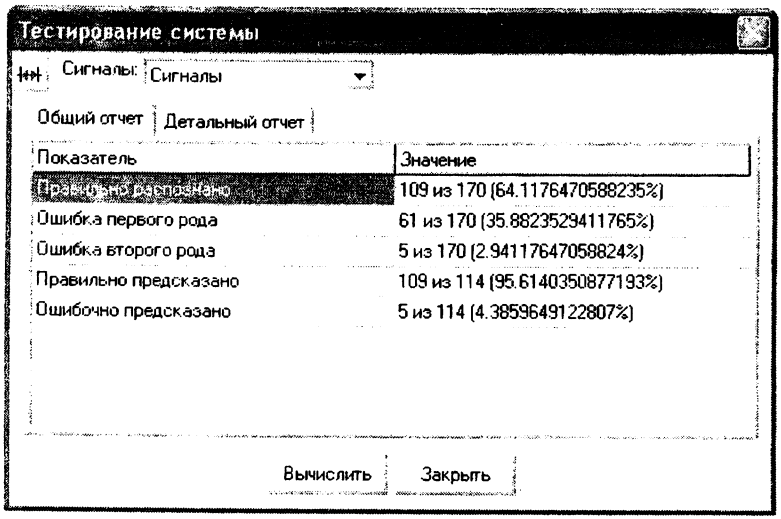


Рис. 7. Блок тестирования

Блок тестирования прогнозов (см.рис.7) позволяет оценить качество прогнозов системы. Чтобы провести тестирование, пользователю необходимо указать колонку таблицы данных, в которой содержится прогноз системы. В качестве такой колонки может выступать любая колонка таблицы, имеющая тип данных «сигнал». Таким образом, чтобы протестировать систему, пользователю необходимо сначала вывести ее прогнозы в какую–нибудь колонку таблицы данных. При этом появляется возможность протестировать не только систему, но и оценить прогнозы отдельного предиктора или даже правила. Для этого достаточно вывести прогнозы предиктора или сигналы о срабатывании правила в

какую–нибудь колонку таблицы данных и указать ее при тестировании. При нажатии кнопки «Вычислить» программа произведет необходимые расчеты и выведет результаты в виде отчета.

Блок тестирования торговых систем. Для облегчения тестирования торговых стратегий, построенных на основе прогнозов системы, в программе предусмотрен блок тестирования торговых систем (см.рис.8). Данный модуль позволяет рассчитать различные финансовые показатели эффективности торговой системы. Чтобы провести тестирование, пользователю необходимо указать колонку таблицы данных, в которой содержатся прогнозы системы. В качестве такой колонки может выступать любая колонка таблицы, имеющая тип данных «сигнал». Таким образом, чтобы протестировать торговую систему, пользователю необходимо сначала вывести ее сигналы в какую–нибудь колонку таблицы данных. При этом также существует возможность протестировать не только торговую систему, но и оценить прибыльность

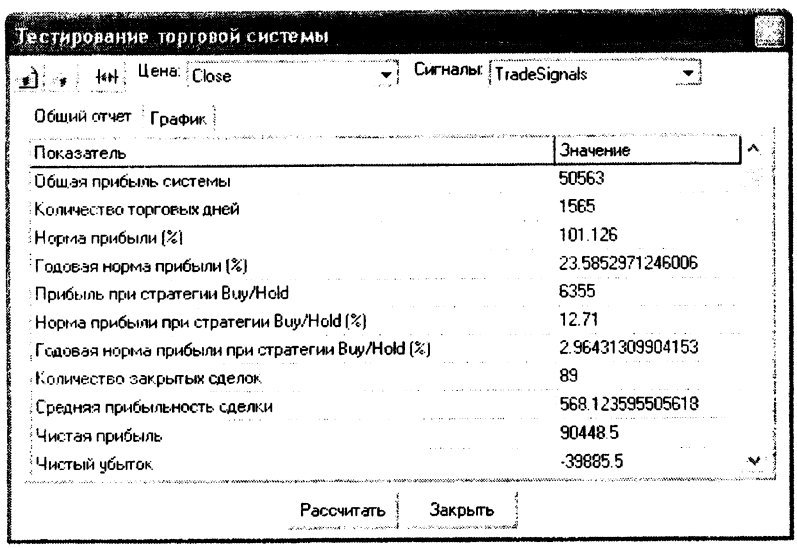


Рис. 8. Блок тестирования торговых систем — общий отчет

отдельного предиктора или даже правила. Для этого достаточно вывести прогнозы предиктора или сигналы о срабатывании правила в какую–нибудь колонку таблицы данных и указать ее при тестировании.

Кнопки, расположенные на панели инструментов, позволяют вызывать вспомогательные окна для редактирования типов торговых сигналов и настройки параметров торговли. Окно редактирования типов торговых сигналов позволяет создавать типы торговых сигналов и устанавливать их параметры, к которым относятся: вид сигнала (покупка или продажа), количество открываемых позиций, длительность удерживания открытой позиции и другие. Окно настройки параметров торговли позволяет устанавливать такие параметры тестирования как размер начального капитала, максимальное количество открытых позиций, уровень стоп–лосса, размер комиссионных и т.д.

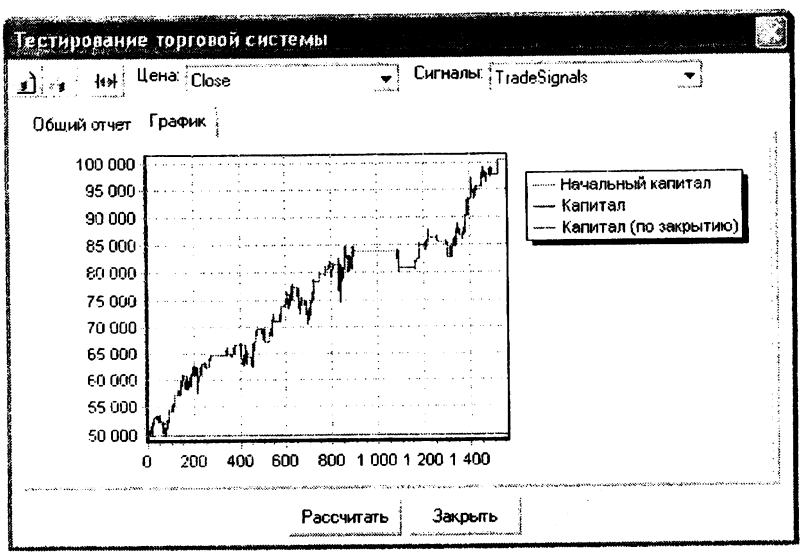


Рис. 9. Блок тестирования торговых систем — график

При нажатии кнопки «Рассчитать» программа произведет необходимые расчеты и выведет результаты в виде общего отчета (рис.8) и в виде графика (рис.9). В общем отчете будет выведена общая информация о проведенном тестировании и различные

показатели о работе системы, включая различные индексы эффективности и т.д. На графике будет показана динамика роста капитала во времени. Красной линией на графике обозначается размер начального капитала, синей — непрерывное изменение капитала во времени, зеленой – изменение капитала только в моменты закрытия позиций.

2.2. Технология решения задач в системе «Discovery». Рассмотрим основные этапы решения задачи в системе «Discovery».

Этап 1. Подготовка исходных данных для анализа. Решение задачи в системе «Discovery» начинается с заполнения таблицы данных для анализа. Исходные данные могут быть загружены в таблицу данных из файла при помощи команды меню «Файл» → «Открыть данные» или введены вручную. Программа позволяет редактировать таблицу данных и применять к ней различные преобразования.

Этап 2. Формирование предикторов. На данном этапе пользователь должен представить исходную задачу как задачу предсказания набора целевых предикатов. Для каждого целевого предиката необходимо создать соответствующий предиктор. Ввод и редактирование предикторов осуществляется при помощи специального редактора предикторов (см.рис.4), который можно вызвать командой меню «Параметры» → «Предикторы».

Этап 3. Конструирование видов гипотез. Данный этап работ в системе «Discovery» является наиболее ответственным для пользователя, поскольку именно здесь он формирует свое видение исследуемого объекта и представляет его в виде специальным образом сформулированных гипотез. Для этих целей он пользуется специальным конструктором видов гипотез (см.рис.5), который можно вызвать командой меню «Параметры» → «Конструктор гипотез». Пользователю необходимо указать виды предикатов, которые будуть использоваться для конструирования правил и способы их интерпретации на множестве исходных данных.

Этап 4. Определение прогноза системы. На данном этапе пользователь определяет способ формирования итоговых прогнозов системы. Выбор способа формирования прогноза не влияет на процесс обучения предикторов, однако, от него зависит, ка-

ким образом обнаруженные правила будут использоваться для формирования итогового прогноза системы. Таким образом, изменение способа формирования прогноза не требует переобучения системы, за исключением случая, когда используется скользящий контроль (см. ниже). Определить способ формирования прогноза можно при помощи специального редактора (см. рис.6), который вызывается командой меню «Параметры» → «Прогнозирующая система».

Этап 5. Обучение и тестирование системы. На данном этапе пользователь имеет две возможности: сначала обучить систему на каком–нибудь обучающем множестве, а затем провести тестирование, или провести скользящий контроль.

В первом случае пользователь командой меню «Действия» → «Обучить» должен вызвать окно обучения, с помощью которого он может выбрать обучающий интервал и провести обучение предикторов. Чтобы протестировать систему, пользователю необходимо сначала вывести ее прогнозы в какую–нибудь колонку таблицы данных, воспользовавшись командой меню «Таблица данных» → «Преобразования» → «Система». Выбранная колонка будет заполнена прогнозами системы по формуле $C(i) = \mathit{Прогноз}(i)$, где C — выбранная колонка, $\mathit{Прогноз}(i)$ — прогноз системы для объекта с номером i. Затем можно протестировать полученные прогнозы системы, вызвав окно тестирования (см.рис.7) командой меню «Действия» → «Тестировать систему». Для финансовых задач также можно протестировать торговую стратегию, основанную на прогнозах системы, вызвав командой меню «Действия» → «Тестировать торговую систему» окно тестирования торговой системы (см.рис.8).

Во втором случае, чтобы осуществить скользящий контроль, пользователь должен сначала вывести прогнозы системы, полученные при скользящем обучении, в какую–нибудь колонку таблицы данных. Данная процедура называется скользящим прогнозом системы. В программе предусмотрено два возможных варианта осуществления скользящего прогноза.

В первом варианте скользящего прогноза из общей выборки из M примеров циклически исключаются K примеров, система обучается на оставшихся $M - K$ примерах, а затем в выбранную

колонку выводятся прогнозы для исключенных примеров. После чего исключенные примеры возвращаются назад в общую выборку и из нее исключаются следующие K примеров и т.д. Данный процесс продолжается до тех пор, пока система не пройдет через всю выборку данных, в результате чего выбранная колонка будет заполнена скользящим прогнозом системы.

Во втором варианте скользящего прогноза система сначала обучается на интервале от 1 до M. Затем выбранная колонка таблицы данных заполняется сигналами системы на интервале от $M+1$ до $M+K$ по формуле $\mathrm{C}(i) = \textit{Прогноз}(i)$, $i = M+1, \ldots, M+K$. После этого система повторно обучается на интервале от $K+1$ до $K+M$. Затем выбранная колонка заполняется сигналами системы на интервале от $(K+V)+1$ до $(K+M)+K$: $\mathrm{C}(i) = \textit{Прогноз}(i)$, $i = (K+M)+1, \ldots, (K+M)+K$. Данный процесс продолжается до тех пор, пока система не пройдет через всю выборку данных, в результате чего выбранная колонка будет заполнена скользящим прогнозом системы.

Чтобы провести скользящее обучение системы и вывести ее сигналы в какую-нибудь колонку, необходимо воспользоваться командой меню «Таблица данных» → «Преобразования» → → «Скользящий выход системы». Затем можно протестировать полученные прогнозы, воспользовавшись командой меню «Действия» → «Тестировать систему». Для финансовых задач можно также протестировать торговую стратегию при помощи команды меню «Действия» → «Тестировать торговую систему».

3. Прогнозирование финансовых временных рядов

3.1. Применение системы «Discovery» для разработки торговой системы. Задача прогнозирования временных рядов была и остается актуальной, поскольку предсказание является необходимым элементом любой инвестиционной деятельности, ведь сама идея инвестирования — вложения денег с целью получения дохода в будущем — основывается на идее прогнозирования будущего. В последнее время, когда стали доступны мощные средства сбора и обработки информации, задача прогнозирования финансовых временных рядов также становится и одной из самых популярных задач для практического применения различных Data Mining-методов. Широкое применение Data Mining-методов в

данной области обусловлено наличием в большинстве временных рядов сложных закономерностей, не обнаруживаемых обычными линейными методами.

В данном исследовании рассмотрено применение системы «Discovery» для разработки торговой системы, основанной на предсказании дневных котировок индекса SP500 (The Standard and Poor's 500). Основная цель данного эксперимента прежде всего состоит в том, чтобы показать применимость системы «Discovery» и ее возможности как инструмента извлечения знаний из финансовых временных рядов.

Пусть $t_1, \ldots, t_n$ — некоторые точки временного ряда, соответствующие торговым дням, $c(t_1), \ldots, c(t_n)$ — значения временного ряда в соответствующих точках (цены закрытия SP500 — колонка Close).

Торговая система разрабатывалась на основе предсказаний увеличения или уменьшения цены через пять дней по отношению к текущей цене на момент закрытия биржи. С этой целью в программе было задано два предиктора. Первый предиктор предсказывал увеличение цены через пять дней, целевой предикат, соответствующий этому предиктору, имел вид $c(t_1) < c(t_1 + 5)$, где $c(t_1)$ — цена закрытия в текущий торговый день, $c(t_1 + 5)$ — цена закрытия пять дней вперед. Второй предиктор предсказывал уменьшение цены через пять дней, соответствующий целевой предикат имел вид $c(t_1) > c(t_1 + 5)$.

Наиболее ответственным шагом при проведении анализа данных является формулировка гипотез, которые будут проверяться на свойство быть вероятностными законами. В задаче предсказания финансовых временных рядов наиболее ярко проявляется проблема выбора адекватного вида гипотез и их формулировка.

Опыт классического технического анализа показывает, что зачастую, перед изменением направления своего движения, цены образуют на графике определенные модели, называемые фигурами технического анализа. Мы решили воспользоваться этим опытом и разработали специальный вид гипотез и предикатов, которые мы и использовали для обнаружения вероятностных закономерностей. Для формирования гипотез были использованы следующие предикаты.

Предикат $c(t_i) < c(t_j)$ сравнивает значение временного ряда (цену закрытия) в точках t_i и t_j.

Предикат $\mathrm{ext}(t_i) = \delta_i$, где δ_i принимает значение -1 или 1, определяет условие, проверяющее, является ли точка t_i локальным минимумом или локальным максимумом. То есть запись $\mathrm{ext}(t_i) = -1$ означает, что t_i — точка локального минимума, что эквивалентно выражению $(c(t_i) < c(t_i - 1))\&(c(t_i) < c(t_i + 1))$. А запись $\mathrm{ext}(t_i) = 1$ означает, что t_i — точка локального максимума, что эквивалентно выражению $(c(t_1 - 1) < c(t_1))\&(c(t_i + 1) < c(t_i))$. В случае, если t_i — текущая точка, относительно которой делается прогноз, то, поскольку значение временного ряда в следующей точке нам не известно, для нее проверяется условие локального максимума только слева. То есть, если t_i — текущая точка, то запись $\mathrm{ext}(t_i) = -1$ означает, что $c(t_i) < c(t_i - 1)$, а $\mathrm{ext}(t_i) = 1$ означает $c(t_i - 1) < c(t_i)$. В дальнейшем будем использовать запись $\mathrm{ext}(t_1, \ldots, t_n) = \langle \delta_1, \ldots, \delta_n \rangle$, которая эквивалентна выражению $(\mathrm{ext}(t_1) = \delta_1)\& \ldots \&(\mathrm{ext}(t_n) = \delta_n)$.

Определим общий вид гипотез, которые использовались для обнаружения вероятностных закономерностей:

$$\forall t_1 \exists t_2, \ldots, t_m : \ (\mathrm{ext}(t_1, \ldots, t_m) = \langle \delta_1, \ldots, \delta_m \rangle)\& \\ \&(c(t_i) < c(t_j))\&, \ldots, \&(c(t_k) < c(t_l)) \to T, \qquad (1)$$

где $i, j, k, l \in \{1, \ldots, m\}$, t_1 — текущая точка, относительно которой делается прогноз $t_1 > t_2 > \cdots > t_m$, $t_q - t_{q+1} \leq 7$, $q = 1, \ldots, m-1$ (т.е. соседние точки не должны отставать друг от друга более чем на семь дней); T — целевой предикат $T = (c(t_1) < c(t_1 + 5))$ или $T = (c(t_1) > c(t_1 + 5))$.

Данные гипотезы проверяют, сформировалась ли в прошлом временного ряда определенная комбинация точек локальных минимумом и максимумов, и, в случае если такая комбинация найдена, делают прогноз на пять дней вперед. Легко видеть, что эти комбинации локальных минимумов и максимумов, по сути, описывают некоторые фигуры, которые может образовывать график временного ряда.

Приведем пример правила:

$\forall t_1 \exists t_2, t_3, t_4, t_5, t_6$:

$$(ext(t_1, t_2, t_3, t_4, t_5, t_6) =< -1, 1, -1, 1, -1, 1 >)\&$$

$$\&(c(t_1) < c(t_2))\&(c(t_2) < c(t_4))\&(c(t_3) < c(t_2))\&$$
$$\&(c(t_5) < c(t_6))\&(c(t_6) < c(t_4)) \rightarrow (c(t_1) > c(t_1 + 5). \qquad (2)$$

Данное правило говорит о том, что если временной ряд сформировал фигуру, описываемую этим правилом, то значение ряда через пять дней станет меньше, чем значение ряда в текущий день. На рис.10 представлен общий вид фигуры, описываемой данным правилом. Лего заметить, что эта фигура напоминает известную фигуру «голова–плечи» из технического анализа

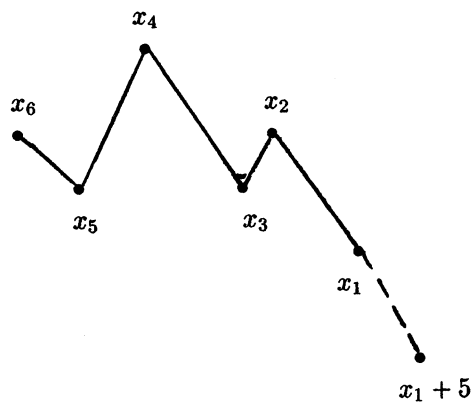


Рис. 10. Фигура, описываемая правилом (2)

Следующим шагом в разработке торговой системы является определение стратегии игры, т.е. определение того, каким образом полученные прогнозы о направлении движения цены будут использованы для принятия решения о вхождении в рынок. Поскольку в один и тот же торговый день может сработать несколько различных правил, то для одного и того же торгового дня мы можем иметь несколько прогнозов, с разной вероятностью предсказывающих направление движения цены. Более того, поскольку правила имеют вероятностный характер, эти прогнозы могут

противоречить друг другу. Поэтому стратегия игры должна учитывать различные, возможно, противоречивые прогнозы, и на их основе выдавать сигналы о вхождении в рынок. Таким образом, чтобы определить стратегию игры необходимо определить способ формирования итогового прогноза на основе прогнозов отдельных правил.

В данном эксперименте мы использовали наиболее простой способ формирования итогового прогноза, основанный на сравнении максимальных вероятностей прогнозов об увеличении или уменьшении цены через пять дней. Стратегия игры, основанная на итоговом прогнозе, заключается в том, что необходимо покупать, если максимальная вероятность того, что цена вырастет через пять дней, больше чем максимальная вероятность того, что она упадет, и продавать в противном случае. Дадим формальное описание данной стратегии.

Пусть PR_{Up} — предиктор, предсказывающий, что через пять дней цена увеличится, PR_{Down} — предиктор, предсказывающий, что цена упадет. Обозначим через $PR_{Up}(t)$ множество правил предиктора PR_{Up}, сработавших в день t, т.е. $PR_{Up}(t) = \{R : R \in PR_{Up},\ Cond(R) = true\}$, где $Cond(R)$ — условная часть правила R, а через $PR_{Down}(t)$ множество правил предиктора PR_{Down}, сработавших в день t, т.е. $PR_{Down}(t) = \{R : R \in PR_{Down}, Cond(R) = true\}$. Тогда стратегия игры может быть выражена следующей формулой:

$$
\begin{aligned}
&\mathit{Сигнал}(t) = \\
&= \begin{cases}
1, \text{если } \max\limits_R\{p(R) : R \in PR_{Up}(t)\} > \\
\qquad\qquad \max\limits_R\{p(R) : R \in PR_{Down}(t)\}, \\
-1, \text{если } \max\limits_R\{p(R) : R \in PR_{Up}(t)\} < \\
\qquad\qquad \max\limits_R\{p(R) : R \in PR_{Down}(t)\}, \\
0, \text{иначе},
\end{cases}
\end{aligned}
$$

где $\mathit{Сигнал}(t)$ — торговый сигнал на вход в рынок в день t. Если $\mathit{Сигнал}(t) = 1$, это означает, что надо открыть позицию на покупку на пять дней (купить и держать пять дней), если при этом уже открыта позиция на продажу, то ее надо закрыть. Если

Сигнал$(t) = -1$, то надо открыть позицию на продажу на пять дней (продать и держать пять дней), если при этом уже открыта позиция на покупку, то ее надо закрыть. Если *Сигнал* $= 0$, то входить в рынок в этот день не следует.

Следующим шагом является обучение и тестирование разработанной торговой системы. Чтобы наиболее объективно оценить торговую систему, мы использовали метод скользящего контроля. Скользящий контроль работает следующим образом. Система обучается на точках данных от 1 до M. Затем проводится тестирование на точках данных от $M+1$ до $M+K$. После этого система повторно обучается на точках от $K+1$ до $K+M$. Затем тестируется на точках от $(K+M)+1$ до $(K+M)+K$. Данный процесс продолжается до тех пор, пока система не пройдет через всю выборку данных. При тестировании методом скользящего контроля M — окно обучения (или исторического обзора), а K — интервал повторного обучения.

Хотя этот метод и требует большого количества вычислений, он позволяет наиболее объективно оценить торговую систему, поскольку при скользящем контроле система проходит несколько тактов обучения и тестирования на разных временных интервалах. Кроме того, скользящий контроль наиболее правдоподобно моделирует процесс, происходящий в реальной торговле — сначала ведется оптимизация, а затем система ведет торговлю на ранее неизвестных данных и время от времени повторно оптимизируется. Программа позволяет проводить скользящий контроль в автоматическом режиме — пользователю достаточно только задать параметры окна обучения и интервала повторного обучения.

Мы проводили тестирование разработанной нами торговой системы методом скользящего контроля на временном интервале с 3 января 1995 года до 20 марта 2003 года (см. рис.11). Данный интервал включает в себя 2065 торговых дней. Был использован размер окна обучения равный 500 торговым дням и интервал повторного обучения равный 100 дням. Таким образом, всего за весь период тестирования (2065 дней) система прошла 16 тактов обучения и тестирования, а суммарный интервал тестирования составил 1567 торговых дней (с 23 декабря 1996 года по 20 марта 2003 года).

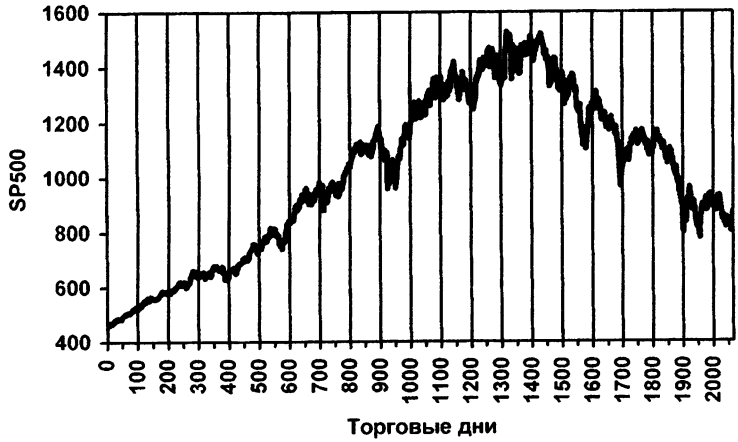


Рис. 11. SP500 за период с 03.01.95 по 20.03.03

Качество торговой системы во время тестирования оценивалось путем моделирования реальной торговли. Основной задачей данного тестирования являлась оценка потенциальной прибыльности системы, для чего необходимо убедиться, что система имеет стабильное положительное математическое ожидание, т.е. выражение

$$P_{Win}Trade_{Win} + P_{Loss}Trade_{Loss} > 0, \tag{3}$$

где P_{Win} — вероятность выигрыша, $Trade_{Win}$ — средний размер выигрыша, P_{Loss} — вероятность проигрыша, $Trade_{Loss}$ — средний размер проигрыша. Как показано в [11], при наличии пусть даже небольшого по размеру, но стабильного положительного математического ожидания, применение методов управления капиталом позволяет добиться экспоненциального роста капитала. С этой целью мы не использовали каких–либо специальных методов управления капиталом и ограничения рисков, т.е. система всегда входила в рынок только одной единицей контракта и не испоьзовала стоп–лосс приказы. Данный способ использования торговой

системы не является оптимальным, однако он позволяет достаточно объективно оценить математическое ожидание системы.

Результаты тестирования представлены на рис.12 и в табл.1. График на рис.12 показывает динамику роста капитала во времени на всем тестовом периоде (1565 дней). Торговые дни пронумерованы от от 1 до 1565. В табл.1 представлен ряд показателей, характеризующих торговую систему. Дадим пояснение к этим показателям.

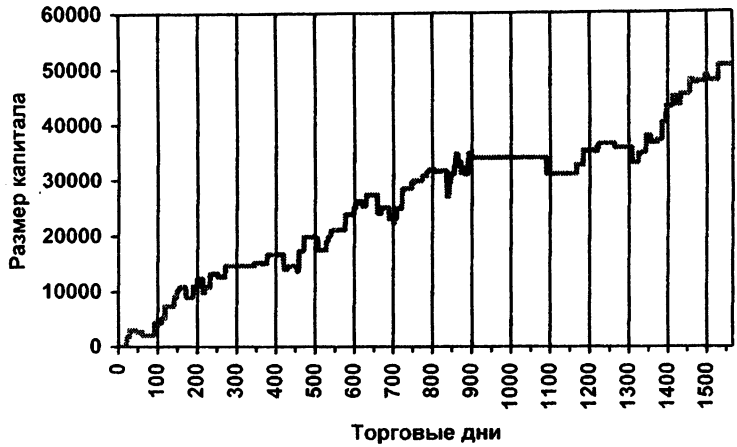


Рис.12. Результаты тестирования торговой системы

Таблица 1

Характеристики торговой системы

Показатель	Значение показателя
Прибыль	50563
Максимальная просадка счета	−4960
Математическое ожидание	568
Процент прибыльных сделок	69%
Годовая норма прибыли по отношению к максимальной просадке счета	238%

Прибыль— общая прибыль, произведенная системой на всем тестовом периоде.

Максимальная просадка счета — максимальная потеря капитала, рассчитанная как наибольшая разность между максимальным значением капитала и последующим его минимальным значением за весь тестовый период торговли. Данный показатель выступает в качестве меры риска потерь капитала торговой системой.

Математическое ожидание — математическое ожидание системы, рассчитываемое по формуле (3). Показывает среднюю прибыль от одной сделки.

Процент прибыльных сделок — процент прибыльных сделок в общем числе сделок.

Годовая норма прибыли по отношению к максимальной просадке счета характеризует норму прибыли на единицу риска, где в качестве меры риска выступает величина максимальной просадки счета. Вычисляется по формуле

$$\text{Норма прибыли} = \frac{\text{Прибыль}}{\text{Макс.просадка счета}} \cdot \frac{365}{\text{Кол–во торговых дней}}$$

Как показывают результаты тестирования, разработанная нами торговая система имеет достаточно стабильное положительное математическое ожидание. Этот вывод подтверждает график, на котором видно, что система обеспечивает устойчивый рост капитала на всем тестовом периоде. Применение методов управления капиталом и ограничения рисков позволило бы значительно улучшить данную торговую систему, однако, это является темой для отдельного исследования.

Приведем два примера правил, которые были найдены системой «Discovery» при первом такте обучения скользящего контроля. Первый такт обучения охватывает временной интервал с 3 января 1995 года по 20 декабря 1996 года (500 торговых дней). Остальные данные, охватывающие период с 23 декабря 1996 года по 20 марта 2003 года (1565 торговых дней), будем использовать в качестве тестового множества.

Первое правило имет вид:

$$\forall t_1 \exists t_2, t_3, t_4 : \ (ext(t_1, t_2, t_3, t_4) =< -1, -1, -1, -1 >)\& \\ \&(c(t_3) < c(t_2))\&(c(t_2) < c(t_1))\& \\ \&(c(t_1) < c(t_4)) \rightarrow (c(t_1) < c(t_1 + 5)). \qquad (4)$$

Данное правило предсказывает увеличение цены через пять дней в случае, если в прошлом временного ряда существовала определенная комбинация локальных минимумов, удовлетворяющая условию правила. На рис.13 представлен общий вид фигуры, описываемой этим правилом. Вероятность данного правила на тестовом множестве равна 0.61.

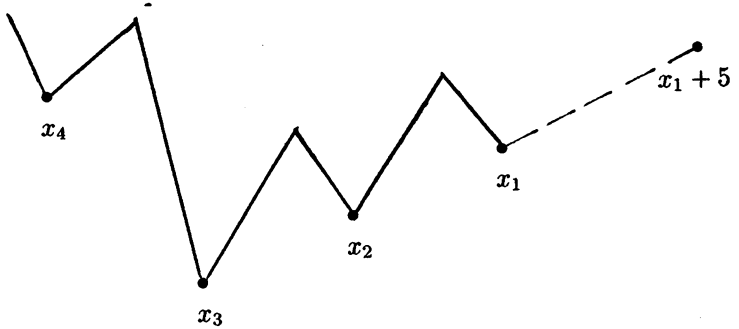


Рис.13. Фигура, описываемая правилом (4)

Чтобы понять торговую ценность данного правила, мы решили оценить, какую прибыль принесет торговля при помощи одного этого правила на тестовом множестве. Для этого мы смоделировали реальную торговлю с использованием очень простой стратегии игры, выдающей сигналы к покупке каждый раз, когда срабатывает данное правило. Как и в случае с описанной выше торговой системой, торговля велась только одной единицей контракта и не использовались стоп–лосс приказы.

Результаты тестирования первого правила представлены на рис.14 и в табл.2. График на рис.14 показывает динамику роста капитала во времени на всем тестовом периоде (1565 дней). Торговые дни пронумерованы от 1 до 1565. В табл.2 представлены описанные выше показатели, характеризующие данное правило.

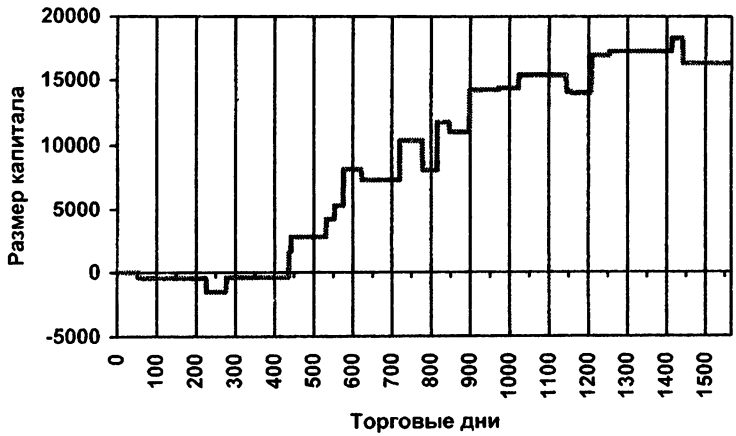


Рис.14. Результаты тестирования правила (4)

Таблица 2

Характеристики правила (4)

Показатель	Значение показателя
Прибыль	16307
Максимальная просадка счета	−2256
Математическое ожидание	709
Процент прибыльных сделок	61%
Годовая норма прибыли по отношению к максимальной просадке счета	169%

Второе правило имеет следующий вид:

$$\forall t_1 \exists t_2, t_3, t_4, t_5 : (ext(t_1, t_2, t_3, t_4, t_5) =< -1, 1, -1, -1, -1 >)\&$$

$$\&(c(t_5) < c(t_2))\&(c(t_4) < c(t_1))\&(c(t_4) < c(t_2))\&$$

$$\&(c(t_3) < c(t_4))\&(c(t_1) < c(t_5)) \rightarrow (c(t_1) > c(t_1 + 5)). \qquad (5)$$

Данное правило предсказывает уменьшение цены через пять дней. Общий вид фигуры, описываемой этим правилом, представлен на рис.15. Вероятность данного правила на тестовом множестве равна 0.71.

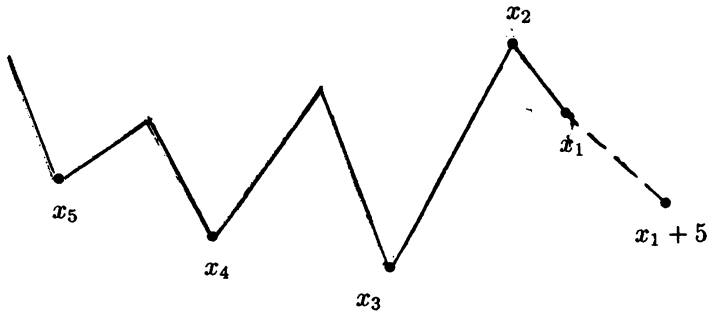


Рис.15. Фигура, описываемая правилом (5)

Для второго правила мы также смоделировали торговлю на тестовом множестве. Использовалась та же стратегия игры, что и для первого правила, с тем лишь отличием, что когда срабатывало правило, система выдавала сигналы на продажу. Результаты тестирования представлены на рис.16 и в табл.3.

3.2. Сравнение прогнозов системы «Discovery» с другими методами. Для того, чтобы оценить эффективность работы системы «Discovery», мы провели сравнение данной системы с методом скользящей линейной регрессии и нейронными сетями.

Чтобы сравнить качество различных методов, необходимо иметь унифицированный способ сравнения результатов работы различных методов. Применительно к финансовым задачам в качестве такого способа сравнения может выступать сравнение финансовых показателей эффективности торговых систем, построенных на основе этих методов. Очевидно, что метод, чей прогноз позволяет извлечь большую прибыль при меньшем риске, имеет преимущество. Таким образом, предсказание тестируется одновременно с торговой системой. Конечно, данный способ сравнения также имеет и некоторые недостатки, поэтому данное сравнение не может быть заключительным сравнением методов прогноза,

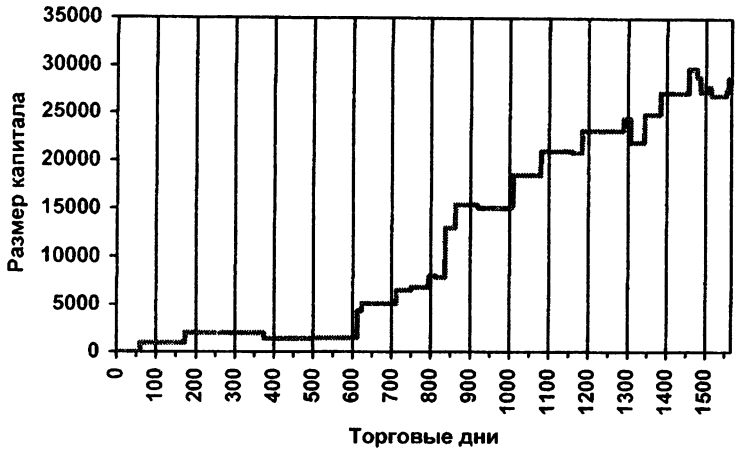


Рис.16. Результаты тестирования правила (5)

Таблица 3

Характеристики правила (5)

Показатель	Значение показателя
Прибыль	28786
Максимальная просадка счета	−2846
Математическое ожидание	993
Процент прибыльных сделок	72%
Годовая норма прибыли по отношению к максимальной просадке счета	236%

однако оно дает полезный результат о практическом значении различных методов прогноза.

Скользящая линейная регрессия. Для произвольной функции $c(t)$, представленой своими выборками $c(t_k)$, взятыми в дискретные равноотстоящие друг от друга моменты времени t_k, $k = 0, 1, 2, \ldots$, линейной регрессией называется линейная функция $y(t) = a_n t = b_n$, удовлетворяющая по отношению к функции $c(t)$ критерию наименьших квадратов. Коэффициенты a_n и b_n определяются по формулам:

$$a_n = \frac{n \sum\limits_{k=1,\ldots,n} t_k c(t_k) - \sum\limits_{k=1,\ldots,n} t_k \sum\limits_{k=1,\ldots,n} c(t_k)}{n \sum\limits_{k=1,\ldots,n} t_k^2 - (\sum\limits_{k=1,\ldots,n} t_k)^2},$$

$$b_n = \frac{\sum\limits_{k=1,\ldots,n} c(t_k) - a_n \sum\limits_{k=1,\ldots,n} t_k}{n}.$$

Параметр n называют длиной регрессии или размером окна, а полученную в результате вычислений линейную функцию называют линейной регрессией длины n.

Суть метода скользящей линейной регрессии заключается в том, что для каждого момента времени t строится линейная регрессия по n последним значениям временного ряда и на основании полученной линейной функции осуществляется прогноз следующего значения временного ряда по формуле $y'(t+1) = a_n(t+1) + b_n$, где $y'(t+1)$ — прогноз значения временного ряда в точке $(t+1)$.

Основным достоинством метода скользящей линейной регрессии является его простота. Она не требует никаких сложных вычислительных средств, а единственным параметром метода является размер окна n.

Стратегия игры, основанная на скользящей линейной регрессии, была определена следующим образом:

$$\textit{Сигнал}(t) = \begin{cases} 1, & \text{если } y(t) < y'(t+1), \\ -1, & \text{если } y'(t+1) < y(t), \\ 0, & \text{иначе,} \end{cases}$$

где $\textit{Сигнал}(t)$ — торговый сигнал на вход в рынок в день t. Если $\textit{Сигнал}(t) = 1$, то это означает, что надо открыть позицию на

покупку, если при этом уже открыта позиция на продажу, то ее надо закрыть. Если *Сигнал*$(t) = -1$, то надо открыть позицию на продажу, если при этом уже открыта позиция на покупку, то ее надо закрыть. Если *Сигнал*$(t) = 0$, то надо держать открытые позиции.

Точность прогноза скользящей линейной регрессии во многом зависит от выбора размера окна линейной регрессии. Для сравнения с системой «Discovery» мы использовали линейную регрессию, размер окна которой был оптимизирован с целью получения наилучших показателей финансовой эффективности торговой системы.

Нейронная сеть. Для сравнения с системой «Discovery» мы использовали многослойные нейронные сети с прямыми связями, обучавшиеся методом обратного распространения ошибки.

Большое влияние на качество прогнозов нейронных сетей оказывает способ представления входной информации. Поскольку изменения котировки гораздо меньше по амплитуде, чем сами котировки, между последовательными значениями курсов имеется большая корреляция — наиболее вероятное значение курса в следующий момент равно его предыдущему значению: $c(t) = c(t-1) + \Delta c(t) \approx c(t-1)$. В то же время для повышения качества обучения следует стремиться к статистической независимости входов, т.е. к отсутствию подобных корреляций. Поэтому наиболее значимыми для предсказания величинами являются не сами значения котировок, а их изменения $\Delta c(t)$ как наиболее статистически независимые величины. Исходя из этих соображений, мы преобразовали исходный ряд котировок $c(t)$ в ряд относительных приращений $d(t)$ по формуле $d(t) = \dfrac{\Delta c(t)}{c(t)}$. В качестве входных признаков нейронных сетей мы использовали k последних значений ряда: $d(t), d(t-1), \ldots, d(t-k+1)$. Выходным значением являлась величина $\dfrac{c(t+5) - c(t)}{c(t)}$, т.е. нейронные сети обучались предсказывать относительное изменение котировок через пять дней.

Обучение нейронных сетей всегда содержит элемент неопределенности, связанный со случайным выбором начальных значений весов синапсов. Этот недостаток приводит к отсутствию

стабильности в работе нейронных сетей и особенно ярко проявляется при работе с такими сильно зашумленными данными как финансовые временные ряды. Для повышения надежности предсказания рекомендуется использовать комитет нейронных сетей [12]. Для данного эксперимента мы использовали комитет из двенадцати нейронных сетей, имеющих различную архитектуру.

Стратегия игры, которую мы использовали для комитета нейронных сетей, определяется следующей формулой:

$$\textit{Сигнал}(t) = \begin{cases} 1, & \text{если } \textit{Out}(t) > 0, \\ -1, & \text{если } \textit{Out}(t) < 0, \\ 0, & \text{иначе,} \end{cases}$$

где $\textit{Out}(t)$ — выход комитета нейронных сетей для момента времени t, $\textit{Сигнал}(t)$ — торговый сигнал на вход в рынок в день t. Если $\textit{Сигнал}(t) = 1$, то это означает, что надо открыть позицию на покупку на пять дней, если при этом уже открыта позиция на продажу, то ее надо закрыть. Если $\textit{Сигнал}(t) = -1$, то надо открыть позицию на продажу на пять дней, если при этом уже открыта позиция на покупку, то ее надо закрыть. Если $\textit{Сигнал} = 0$, то входить в рынок в этот день не следует.

Тестирование нейросетевой торговой системы осуществлялось описанным выше методом скользящего контроля. Параметры скользящего контроля были выбраны такими же как в экспериментах с системой «Discovery»,т.е. размер окна обучения равнялся 500 торговым дням, а интервал повторного обучения — 100 дням.

Результаты сравнения.

Все методы тестировались на одном и том же временном интервале (с 23 декабря 1996 г. по 20 марта 2003 г.). Каждая торговая система всегда входила в рынок только одной единицей контракта и не использовала стоп–лосс приказы. Результаты сравнения системы «Discovery» с другими методами представлены в табл.4 и на рис.17. График на рис.17 показывает рост капитала во времени для всех трех систем. Из табл.4 видно, что система «Discovery» превосходит другие методы как по проценту правильных прогнозов, так и по показателям финансовой эффективности.

Рис.17. Результаты тестирования торговых систем

Таблица 4

Сравнение различных стратегий игры

Показатель	«Discovery»	Линейная регрессия	Нейронные сети
Прибыль	50563	27805	33117
Максимальная просадка счета	–4960	–19059	–16700
Математическое ожидание	568	66	106
Процент прибыльных сделок	69%	41%	57%
Годовая норма прибыли по отношению к максимальной просадке счета	238%	34%	46%

Если сравнить графики доходности торговых систем с графиком цены закрытия SP500, то будет видно, что торговая система на основе скользящей линейной регрессии хорошо работает на участках с явно выраженным трендом, однако в местах смены тренда и, особенно, на участках бокового тренда данная система показывает очень большие убытки. Это легко объясняется тем, что линейная регрессия в каждый момент времени пытается аппроксимировать временной ряд прямой линией, а это приемлемо только в случае наличия на рынке сильного тренда.

По сравнению с линейной регрессией нейросетевая торговая система несколько лучше работает на участках большого тренда, однако смена тренда по прежнему оказывает катастрофическое влияние на форму кривой доходности, вызывая большие провалы. Более того, если внимательно проанализировать графики, то становится заметно, что нейронные сети хорошо работают только на тех участках, на которых общая динамика цен совпадает с динамикой цен того участка, на котором она обучалась. Это объясняется тем, что нейронные сети пытаются аппроксимировать все обучающее множество, стремясь минимизировать среднюю ошибку, поэтому они, в первую очередь, находят наиболее общие закономерности, которые выполняются для большинства примеров из обучающего множества. Тем самым нейронные сети в основоном «улавливают» наиболее общую динамику временного ряда и при смене тенденции перестают работать.

Система «Discovery» в отличие от нейронных сетей способна находить высоковероятные статистически значимые закономерности, описывающие достаточно редкие события, предшествующие направленному движению цены. Большинство из этих закономерностей продолжают успешно работать и при изменениях тренда. Примерами таких закономерностей могут служить правила (4) и (5), которые, будучи найденными в самом начале временного ряда, продолжают хорошо предсказывать и на всем остальном периоде, Примечательно, что эти правила и вся торговая система в целом практически без потерь проходят глобальную смену тренда в 2000 году.

Заключение

Таким образом, проведенные эксперименты показывают, что логиковероятностные методы извлечения знаний в языке логики первого порядка в состоянии обнаружить закономерности в таких сильно зашумленных данных как финансовые временные ряды. Эти финансовые задачи уже на протяжении многих лет представляют серьезный вызов для всех методов Data Mining.

Предложенный нами метод извлечения знаний имеет практически неограниченные возможности в формулировании и проверки различных гипотез, которые не могут быть сформулированы другими методами. Класс гипотез (1), описанный в этой работе, уже показал преимущества перед гипотезами, используемыми другими методами.

Литература

1. ВИТЯЕВ Е.Е. Извлечение знаний из данных. Компьютерное познание. Модели когнитивных процессов. – Новосибирск: НГУ,2006. – 293 с.

2. KOVALERCHUK B., VITYAEV E. Data Mining in France: Advances in Relational and Hybrid methods. – Kluwer Academic Publishers, 2000. – 308 p.

3. VITYAEV E., KOVALERCHUK B. Empirical Theories Discovery based on the Measurement Theory// Mind and Machine. – 2004. – Vol.14, №4. – P.551–573.

4. ВИТЯЕВ Е.Е., МОСКВИТИН А.А. Введение в теорию открытий. Программная система Discovery// Логические методы в информатике. – Новосибирск. – 1993. – Вып.148: Вычислительные системы. – С.117–163.

5. KOVALERCHUK B., VITYAEV E., RUIZ J.F. Consistent and Complete Data abd "Expert" Mining in Medicine// Medical Data Mining and Knowlesge Discovery. – Springer, 2001. – P.238–280.

6. VITYAEV E., KOVALERCHUK B. Data Mining For Financial Applications// Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researchers/ Ed. by Maimon O., Rokach L. – Springer,2005. – P.1203–1224.

7. ДЕМИН А.В., ВИТЯЕВ Е.Е. Реализация универсальной версии системы «Discovery»// Тез.докл. конференции–конкурса «Технологии Microsoft в теории и практике программирования», Новосибирск, 24–26 февраля 2007 г. – С.106– 108.

8. ВИТЯЕВ Е.Е. Семантический подход к созданю баз знаний. Семантический вероятностный вывод наилучших для предсказания ПРОЛОГ–программ по вероятностной модели данных// Логика и семантическое программирование.– Новосибирск, 1992. – Вып.146: Вычислительные системы. – С.19–49.

9. ВИТЯЕВ Е.Е. Метод обнаружения закономерностей и метод предсказания// Эмпирическое предсказание и распознавание образов. – Новосибирск, 1976. – Вып.67: Вычислительные системы. – С.54–68.

10. КЕНДАЛ М., СТЮАРТ А. Статистические выводы и связи. – М.: Наука,1973. – 899 с.

11. VINCE R. The mathematics of money management. – New York: John Wiley & Sons, 1992. – 377 p.

12. BAXT W.G. Improving the accuracy of an artificial neural network using multiple differently trained networks// Neural Computation. – 1992. – Vol.4. – P.772-780

Поступила в редакцию
27 марта 2008 года