УДК 681.3:004.8

Сравнение системы «Discovery» с базовыми алгоритмами, встроенными в Microsoft SQL Server Analysis Services[1]

Фирсов Н.И. (Институт систем информатики СО РАН)

В работе проводится сравнение системы «Discovery» с алгоритмами Microsoft Association Rules, Decision Trees и Neural Network, встроенными в Microsoft SQL Server Analysis Services. Показывается, что система «Discovery», во-первых, обладает теоретическими преимуществами перед этими алгоритмами, во-вторых, практически работает лучше на данных, где эти преимущества проявляются явно и, в-третьих, хорошо себя показывает на данных, взятых из репозитория UCI. Эти результаты демонстрируют определенные преимущества системы Discovery перед методами, встроенными в Microsoft SQL Server Analysis Services.

Ключевые слова: *Интеллектуальный анализ данных, извлечение знаний, предсказание, обнаружение закономерностей.*

1. Введение

В последнее время получили широкое развитие и активно применяются на практике различные KDD&DM-методы (Knowledge Discovery in Data Bases and Data Mining). Однако, используемые сейчас KDD&DM-методы имеют серьезные ограничения [1, 7]: каждый метод может работать только с определенными типами данных, имеет свой язык оперирования и интерпретации данных, и обнаруживает только определенный класс гипотез. Таким образом, они не способны извлекать из данных знания в полном объеме, а также могут получать результаты, не интерпретируемые в терминах предметной области.

Система «Discovery» реализует реляционный подход к методам извлечения знаний [1, 7, 11], снимающий упомянутые ограничения, свойственные KDD&DM-методам.

Система «Discovery» обладает следующими важными теоретическими свойствами: может обнаруживать теорию предметной области, может обнаруживать все правила, имеющие мак-

[1] Работа поддержана грантом РФФИ № **15-07-03410**; интеграционными проектами СО РАН № 3, 87, 136, а также работа выполнена при финансовой поддержке Совета по грантам Президента РФ и государственной поддержке ведущих научных школ (проект НШ-3606.2010.1.)

симальные условные вероятности, может обнаруживать непротиворечивую вероятностную аппроксимацию теории предметной области [11], обнаруживает все максимально специфические правила, позволяющие предсказывать без противоречий [1, 12].

Наиболее близкими к системе «Discovery» методами можно считать поиск ассоциативных правил (Microsoft Association Rules) [10] и Decision Trees, в виду того, что закономерности в этих методах также представляются в форме логических правил. В данной работе ставится задача сравнения системы «Discovery» с Microsoft Association Rules, Decision Trees и Neural Network, встроенными в Microsoft SQL Server Analysis Services. Мы[2] покажем, что система «Discovery» обладает теоретическими преимуществами перед этими методами, практически работает лучше на данных, где эти преимущества явно проявляются и не хуже работает на реальных данных, взятых из DM-репозитория UC Irvine Machine Learning Repository (`http://archive.ics.uci.edu/ml/`).

Проведенное сравнение обосновывает необходимость реализации системы «Discovery» в виде плагина, подключаемого к службам Microsoft SQL Server 2005 Analysis Services (SSAS). Это позволяет использовать для сравнения алгоритмов единую среду разработки Business Intelligence Development Studio, единые средства визуализации Data Mining моделей, а также стандартные средства сравнения качества Data Mining моделей: диаграмму роста (Lift Chart) и классификационную матрицу (Classification Matrix).

2. Association Rules

Алгоритм Microsoft Association Rules состоит из двух шагов. Первый шаг – это ресурсоемкая фаза нахождения часто встречающихся наборов. Второй шаг – это генерация ассоциативных правил с использованием множества часто встречающихся наборов.

2.1. Нахождение часто встречающихся наборов

Под набором (itemset) мы понимаем набор истинностных значений предикатов. Например, $\{A(a) = 1; B(a) = 0; C(a) = 1\}$ – это набор длины 3. Запись a таблицы содержит некоторый набор, если на этой записи выполнены все предикаты данного набора. Поддержка набора (Support) – это количество записей таблицы, которые содержат данный набор.

Основным параметром, участвующим в нахождении часто встречающихся наборов, является параметр Minimum Support, который определяет, в каком минимальном количестве записей анализируемой таблицы должен содержаться некоторый набор, чтобы он являлся часто встречающимся.

На первой итерации находятся все часто встречающиеся наборы длиной 1. Алгоритм просто сканирует таблицу и подсчитывает поддержку каждого возможного предиката. Предика-

[2] Выражаю благодарность за помощь в работе и научное руководство, д.ф.-м.н. Витяеву Е.Е.

ты с поддержкой большей, чем Minimum Support, добавляются во множество часто встречающихся наборов длины 1. На второй итерации из часто встречающихся наборов, найденных на первой итерации, строятся всевозможные наборы длины 2, подсчитываются поддержки этих наборов, те наборы, которые проходят критерий Minimum Support, добавляются во множество часто встречающихся наборов длины 2. Далее из предикатов, входящих в часто встречающиеся наборы длины 2, строятся всевозможные наборы длины 3 и т.д. Алгоритм повторяется для наборов длины 3, 4, 5 и т.д., пока находятся наборы удовлетворяющие критерию Minimum Support.

Далее проверяется условие, что каждый поднабор часто встречающегося набора, также должен являться часто встречающимся набором.

2.2. Генерация ассоциативных правил

Следующая процедура генерирует ассоциативные правила:

1. Для любого часто встречающегося набора *f*, генерируем все поднаборы *x* и их дополнения $y = f - x$.
2. Если *Support* (*f*) / *Support* (*x*) > Minimum Probability, тогда $x \Rightarrow y$ является ассоциативным правилом с условной вероятностью *Prob* = *Support* (*f*) / *Support* (*x*).

Параметр Minimum Probability задается перед началом обучения модели.

2.3. Прогнозирование

Следующий алгоритм по набору предикатов, поданных на вход, предсказывает значение целевого признака, либо выдает множество (*n* штук) наиболее вероятных значений целевого признака:

1. На вход подается некоторый набор предикатов. Ищутся все правила, условная часть которых совпадает либо с данным набором, либо с некоторым поднабором данного набора, а целевая часть содержит целевой признак. Найденные правила (*k* штук) применяются: целевые части правил и соответствующие условные вероятности добавляются в список рекомендаций.
2. Если подходящих правил не найдено, или их слишком мало ($k < n$), находятся $n - k$ наиболее популярных значений целевого признака. То есть, среди всех правил вида $\Rightarrow \mathrm{P} = a_i$ (с пустой условной частью и целевым признаком в правой части) находятся $n - k$ правил с наибольшей условной вероятностью.
3. Предикаты, полученные на первых двух шагах, сортируются по вероятности.

3. Decision Trees

Основная идея алгоритма решающих деревьев состоит в рекурсивном разделении данных на подмножества, содержащие более или менее однородные состояния целевого (прогнозируемого) атрибута. При каждом разделении, все входные атрибуты оцениваются по их влиянию на целевой атрибут. Когда этот рекурсивный процесс заканчивается, решающее дерево сформировано.

3.1 Построение таблицы подсчета корреляций

Пусть для анализа с помощью алгоритма решающих деревьев дана некоторая таблица, с входными колонками F0, F1, F2, F3, и целевой колонкой P, размером 3000 записей, содержащую в своих ячейках только 0 или 1.

Тогда таблица подсчета корреляций на первом шаге алгоритма будет выглядеть следующим образом:

Таблица 1. Таблица подсчета корреляций.

		F0		F1		F2		F3	
		0	1	0	1	0	1	0	1
P	0	300	700	700	300	400	600	500	500
	1	200	1800	400	1600	400	1600	1100	900

Каждая колонка таблицы подсчета корреляций соответствует паре атрибут-значение одного из входных атрибутов. Каждая строка соответствует значению целевого атрибута.

Ячейки таблицы содержат количество корреляций соответствующих пар: входной атрибут-значение, целевой атрибут-значение. Например, пересечение первой строки и первого столбца таблицы подсчета корреляций содержит число 300, т.е. в анализируемой таблице есть 300 записей, у которых одновременно F0 = 0 и P = 0.

3.2 Нахождение наиболее подходящего для разбиения атрибута

Одним из широко известных критериев, с помощью которого можно найти необходимый атрибут, является *Энтропия*. Энтропия (H) определяется следующим образом:

$$H(p_1, p_2, \ldots, p_n) = -p_1 \cdot \log(p_1) - p_2 \cdot \log(p_2) - \ldots - p_n \cdot \log(p_n),\ p_1 + p_2 + \ldots + p_n = 1$$

Например, энтропия атрибута F1 равна: H(F1) = H(700, 400) + H(300, 1600) = 0.946 + 0.629 = 1.571

Атрибут с наименьшей энтропией является наиболее подходящим для разбиения. В данном примере первое разбиение будет по атрибуту F1. Решающее дерево после первого разбиения выглядит следующим образом:

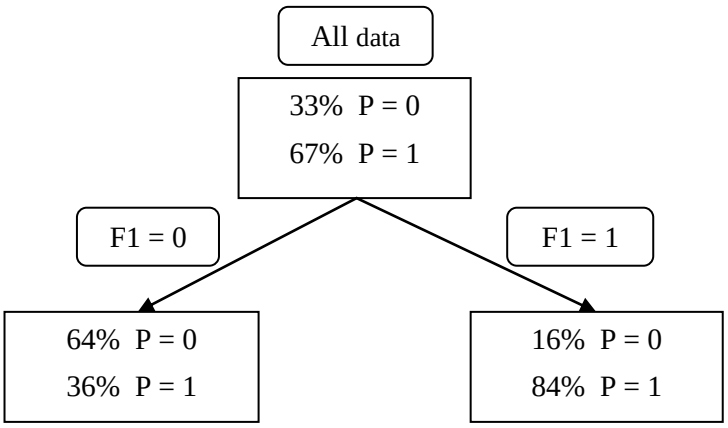

Рис. 1. Решающее дерево при разбиении по атрибуту F1.

Таким образом, разделив данные входной таблицы на два подмножества, далее рекурсивно применяем первый шаг алгоритма к новообразованным вершинам решающего дерева. Например, новая таблица подсчета корреляций для вершины решающего дерева F1 = 0 выглядит следующим образом:

Таблица 2. Таблица подсчета корреляций (после первого разбиения).

		F0		F1		F2		F3	
		0	1	0	1	0	1	0	1
P	0	300	700	700	0	400	600	500	500
	1	200	1800	400	0	400	1600	1100	900

3.3 Прогнозирование

Алгоритм предсказания решающих деревьев достаточно простой и эффективный. Поданный на вход предсказания набор предикатов, например (F1 = 1, F2 = 0, F3 = 0, F4 = 1), спускается по дереву от корня к листьям по соответствующему этому набору пути, вершина дерева, находящаяся в конце этого пути и определяет предсказанное значение целевого признака. Таким образом, количество шагов в процессе прогнозирования не превышает максимальной длины пути от корня к листьям решающего дерева.

4. Neural Network

4.1. Структура нейронной сети

Нейронная сеть состоит из множества узлов (нейронов) и соединяющих их ребер. Есть три вида узлов: входные, скрытые и выходные узлы. Каждое ребро соединяет два узла и также имеет некоторый вес.

Входные узлы формируют первый уровень сети. Каждый входной узел сети связан с входным атрибутом. Значения входных атрибутов отображаются на отрезок [-1, 1] и полученные числа подаются на соответствующие узлы первого уровня сети.

Скрытые узлы это узлы, составляющие промежуточные слои сети. Они получают на вход значения с первого (входного) слоя нейронной сети либо с предыдущего скрытого слоя. Каждый узел комбинирует значения, полученные с предыдущего слоя, с учетом весов соответствующих ребер, проводит некоторые вычисления и передает результирующее значение на следующий уровень сети.

Выходные узлы сети соответствуют выходным атрибутам модели. Результатом вычислений в таком узле обычно является число из отрезка [0, 1], которое затем отображается на множество значений соответствующего выходного атрибута.

Заметим, что в случае, когда скрытые узлы отсутствуют, нейронная сеть представляет собой логистическую регрессию (logistic regression). То есть, скрытые узлы, это важные элементы сети, которые позволяют ей обнаруживать нелинейные закономерности.

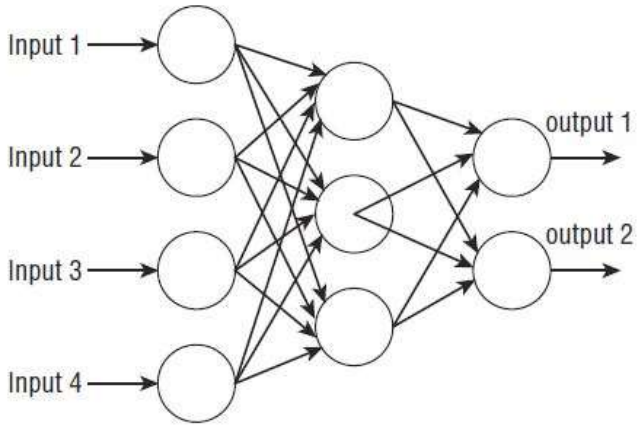

Рис. 2. Нейронная сеть со скрытыми слоями.

Каждый нейрон в нейронной сети является базовым вычислительным блоком. Нейрон имеет несколько входов и один выход. Он комбинирует все входные значения, производит вычисления и выдает на выход некоторое значение. Этот процесс похож на работу биологического нейрона.

Как показано на рисунке 3, нейрон использует две функции: одну для комбинирования входных значений и другую (так называемую функцию активации) для вычисления выходного значения. Существует несколько способов комбинировать входные значения, алгоритм Microsoft Neural Network использует для этого взвешенную сумму w1*input1+ w2*input2+ w3*input3+ w4*input4. Результат комбинирования входных значений подается затем в функцию активации. Алгоритмом Microsoft Neural Network используются две функции активации: tanh в нейронах промежуточных (скрытых) слоев сети и sigmoid в нейронах выходного слоя.

Где $\tanh = (e^a - e^{-a})/(e^a + e^{-a})$, $\text{sigmoid} = 1/(1 + e^a)$. На рисунке 4 показаны графики этих функций.

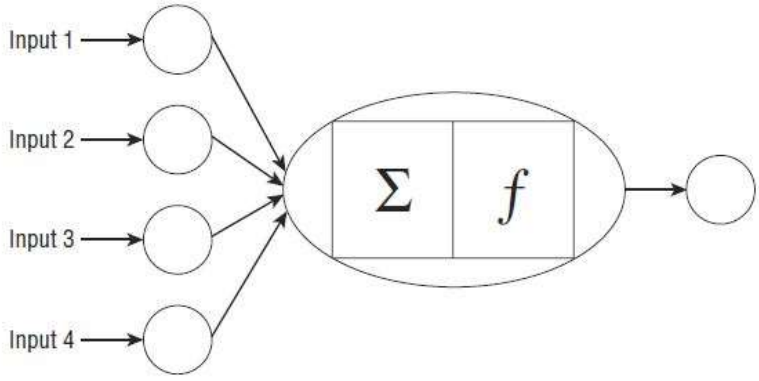

Рис. 3. Вычислительный блок нейронной сети (нейрон).

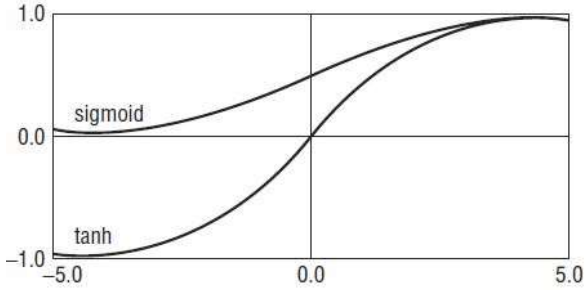

Рис. 4. График функций sigmoid и tanh.

4.2. Обучение.

Процесс обучения заключается в нахождении наилучшего набора весов для ребер сети, дающего минимальную ошибку предсказания.

1)Сначала все веса выставляются случайным образом (обычно в интервале от -1 до 1).

2)На каждом шаге обучения нейронная сеть, используя текущий набор весов, обрабатывает тренировочные записи, выполняя предсказание для каждой из них.

3)Далее, с помощью метода «обратного распространения ошибки» подсчитываются ошибки всех выходных и промежуточных (скрытых) нейронов, веса ребер сети корректируются.

4)Шаг 2 повторяется, пока не выполнится условие завершения обучения.

Ошибки нейронов выходного слоя считаются по формуле $Err_i = O_i(1 - O_i)(T_i - O_i)$, где O_i это выходное значение нейрона с номером i (т.е. предсказанное выходным нейроном значение), T_i это фактическое значение, которое нейрон должен был предсказать. Ошибки нейронов промежуточного (скрытого) слоя считаются по формуле $Err_i = O_i(1 - O_i)\sum_j Err_j w_{ij}$, где O_i это выходное значение нейрона с номером i, который имеет j ребер к

нейронам следующего слоя. Err_j – ошибка нейрона с номером j, w_{ij} – вес ребра между нейронами i и j.

После того, как ошибки всех нейронов были вычислены, веса ребер сети корректируются по формуле $w_{ij} = w_{ij} + l * Err_j * O_i$, где l величина от 0 до 1, называемая *скоростью обучения*. В процессе обучения величина l постепенно уменьшается для лучшей настройки весов сети.

Обучение останавливается, если достигнута достаточная точность на обучающем наборе, либо достигнут верхний лимит по количеству итераций обучения, либо веса после каждой итерации меняются меньше некоторой заданной границы.

4.3. Прогнозирование.

Хотя обучение нейронной сети это трудоемкий процесс, алгоритм предсказания достаточно эффективен. Поданные на вход значения атрибутов (например, Input1 = 2, Input2 = 4, Input3 = 5, Input4 = -1) нормализуются, и полученные значения записываются в нейроны входного уровня сети. Далее, нейроны скрытого слоя производят вычисления, как было описано выше, и полученные значения подаются на вход следующего скрытого слоя или выходного слоя. В конце, нейроны выходного слоя производят вычисления, и полученные числа отображаются (масштабируются) на множество значений соответствующих выходных атрибутов.

5. Система «Discovery»

Алгоритм поиска закономерностей системы «Discovery» реализует метод семантического вероятностного вывода, позволяющего находить все максимально специфические и максимально вероятные закономерности в данных [1]. Определим на высказываниях языка первого порядка вероятность, как описано в [6].

5.1 Семантический вероятностный вывод

Семантическим вероятностным выводом (СВВ) некоторого атома/литерала P является такая последовательность правил C_1, C_2 , ... , C_n , что:

1) $C_i = (A_1^i \& ... \& A_{k_i}^i \Rightarrow P),\ i = 1,\ldots,n$;

2) C_i является подправилом правила C_{i+1}, т.е. $\{A_1^i,...,A_{k_i}^i\} \subset \{A_1^{i+1},...,A_{k_{i+1}}^{i+1}\}$;

3) $\mathrm{Prob}(C_i) < \mathrm{Prob}(C_{i+1})$, $i = 1,2,...,n-1$, где $\mathrm{Prob}(C_i)$ – *условная Вероятность* (УВ) правила, $\mathrm{Prob}(C_i) = \mathrm{Prob}(P/A_1^i\&...\&A_{k_i}^i) = \mathrm{Prob}(P\&A_1^i\&...\&A_{k_i}^i) / \mathrm{Prob}(A_1^i\&...\&A_{k_i}^i)$;

4) C_i – *Вероятностные Законы* (ВЗ), т.е. для любого подправила $C' = (A_1\&...\&A_j \Rightarrow P)$ правила C_i, $\{A_1,...,A_j\} \subset \{A_1^i,...,A_{k_i}^i\}$ выполнено неравенство $\mathrm{Prob}(C') < \mathrm{Prob}(C_i)$;

5) C_n – *Сильнейший Вероятностный Закон* (СВЗ), т.е. правило C_n не является подправилом никакого другого вероятностного закона.

Вероятностные неравенства в пунктах 3-4 проверяются на данных с помощью точного критерия независимости Фишера и критерия Юла [2, 3].

Множество всех цепочек СВВ предиката P образуют дерево СВВ предиката P.

Реализовать семантический вероятностный вывод в чистом виде не представляется возможным ввиду требований к производительности алгоритма, т.к. пункты 2 и 4 определения СВВ подразумевает большое пространство перебора. Для уменьшения перебора применяется следующие упрощения.

Во-первых, положим, что при построении цепочки СВВ правило C_{i+1} получается из правила C_i добавлением к его условной части только одного предиката. Эксперименты показывают, что крайне редка ситуация, когда добавление в условную часть правила сразу двух предикатов дает ВЗ, а добавление любого из этих двух признаков по отдельности не дает ВЗ. Следовательно, мы можем значительно уменьшить пространство перебора, почти не снижая количество и качество извлеченных из данных закономерностей.

Во-вторых, для того чтобы уменьшить перебор при проверке условия в пункте 4, используется поуровневая схема генерация правил: сначала генерируются все ВЗ с одним предикатом в условной части и заключением P, затем с двумя предикатами, тремя и т.д. Таким образом, для проверки, является ли некоторое правило ВЗ, достаточно просмотреть все его подправила, находящиеся на предыдущем уровне дерева СВВ.

Перед началом обучения колонки входной таблицы помечаются атрибутами Input, PredictOnly и Predict, которые указывают, каким образом та или иная колонка участвует в обучении: в качестве входного признака, целевого признака, или в качестве обоих одновременно. Также в качестве параметров модели могут задаваться пороговые величины: условная частота правила, уровни значимости критериев Фишера и Юла, максимальное число интервалов значений для признака и др.

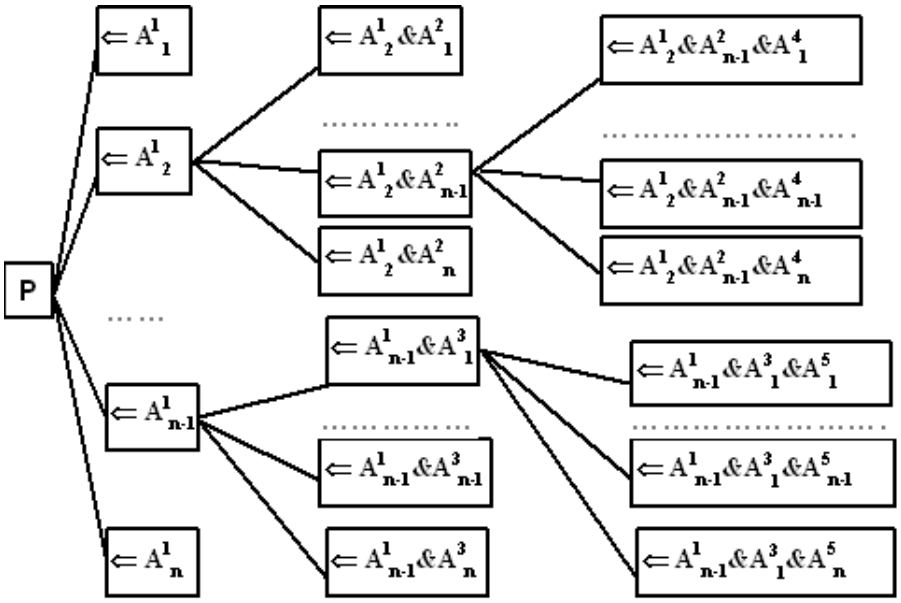

Рис. 5. дерево СВВ, включающее все СВВ, содержащие в заключении атом P.

Результатом работы алгоритма является:

1) дерево СВВ для каждого целевого предиката;
2) множество ВЗ и СВЗ этих деревьев;
3) *Максимально Специфический Закон* (МСЗ) для каждого целевого предиката, определяемый как СВЗ, обладающий наибольшей условной вероятностью среди других СВЗ дерева вывода этого предиката.

Множество всех МСЗ обладает таким важным свойством как потенциальная непротиворечивость [1].

5.2 Алгоритм поиска закономерностей

1. Generate_First_Tree_Level (Queue Q, Tree T)

   – Создаются все возможные правила, состоящие только из целевой части (они все по определению ВЗ). Для них сразу рассчитывается вся необходимая статистика на основе входных данных. Все элементы добавляются в корень дерева T; элементы, содержащие Predict предикаты, добавляются в очередь Q.

2. Generate_Subsequent_Tree_Level (Queue Q, Tree T)

   Обход дерева в ширину:

   – Берем элемент из начала очереди Q, для него генерируем потомков, т.е. уточняем правило путем добавления 1-го нового предиката в условную часть.

   – Проверяем, является ли новое правило ВЗ (см. Check_If_Probability_Low). Если является ВЗ, добавляем в дерево T, добавляем в очередь Q.

– Процесс повторяется, пока очередь не пуста, т.е. еще можно получить новые ВЗ, путем добавления 1-го предиката в условную часть правила. Правила, соответствующие элементы которых не имеют потомков, являются СВЗ.

3. Check_If_Probability_Low (Rule R)

– Проверяем, является ли статистически значимым правило R с помощью критериев Фишера и Юла [1: с. 117-120]. Если нет, то R не является ВЗ;

– просматриваем все подправила Sr длины Length(R) - 1

– Если Prob(Sr) > Prob(R), то R не является ВЗ;

– иначе R является ВЗ.

4. Extract_MSL (Tree T)

– Для каждого целевого предиката просматриваем его дерево СВВ. Сортируем множество СВЗ этого дерева по вероятности. Правила с наибольшей условной вероятностью являются Максимально Специфическими Законами (МСЗ) рассматриваемого целевого предиката.

5.3 Прогнозирование

Определим *меру правила* как неотрицательную величину, характеризующую качество и значимость правила в процессе предсказания.

Следующий алгоритм по набору предикатов, поданных на вход и множествам обнаруженных закономерностей ВЗ, СВВ и МСЗ, предсказывает значение целевого признака:

На вход подается некоторый набор предикатов. Ищутся все максимально специфичные закономерности, условная часть которых совпадает с данным набором либо с некоторым поднабором данного набора, а целевая часть содержит целевой признак. Максимально специфичная закономерность с наибольшей мерой определяет предсказанное значение целевого признака.

Если подходящих МСЗ не найдено, то рассматриваются все ВЗ с дерева СВВ целевого признака. Среди рассмотренных ВЗ ищутся те правила, условная часть которых совпадает с входным набором либо с некоторым поднабором входного набора. Правило с наибольшей мерой определяет предсказанное значение целевого признака.

В качестве меры правила можно использовать условную вероятность правила либо величину Юла. Величина Юла это число $\mathrm{U} = \mathrm{Q} - \mathrm{NormalCDF}^{-1}(p) \cdot \sqrt{D}$, где $\mathrm{Q} = \dfrac{f_{11} f_{22} - f_{12} f_{21}}{f_{11} f_{22} + f_{12} f_{21}}$ есть коэффициент связи Юла, $D = \dfrac{1}{4} \cdot \left(1 - Q^2\right)^2 \cdot \left(\dfrac{1}{f_{11}} + \dfrac{1}{f_{12}} + \dfrac{1}{f_{21}} + \dfrac{1}{f_{22}} \right)$, f_{ij}-частоты [3]. $\mathrm{NormalCDF}^{-1}(p)$ - функция, обратная к функции нормального распределения. p - граничная вероятность, подается в качестве параметра алгоритма, используется также как граничная вероятность в критерии Фишера.

На многих тестах величина Юла в качестве меры правила дает лучшие результаты чем условная вероятность правила.

6. Data Mining Plug-in «Discovery»

Data Mining алгоритм "Discovery" реализован в виде плагина (Plug-in), подключаемого к службам Microsoft SQL Server Analysis Services (SSAS) посредством COM интерфейсов (см. рис.6). Это позволяет использовать Discovery наряду с другими алгоритмами, реализованными фирмой Microsoft, а также сравнивать качество Data Mining моделей, получаемых с помощью этих алгоритмов.

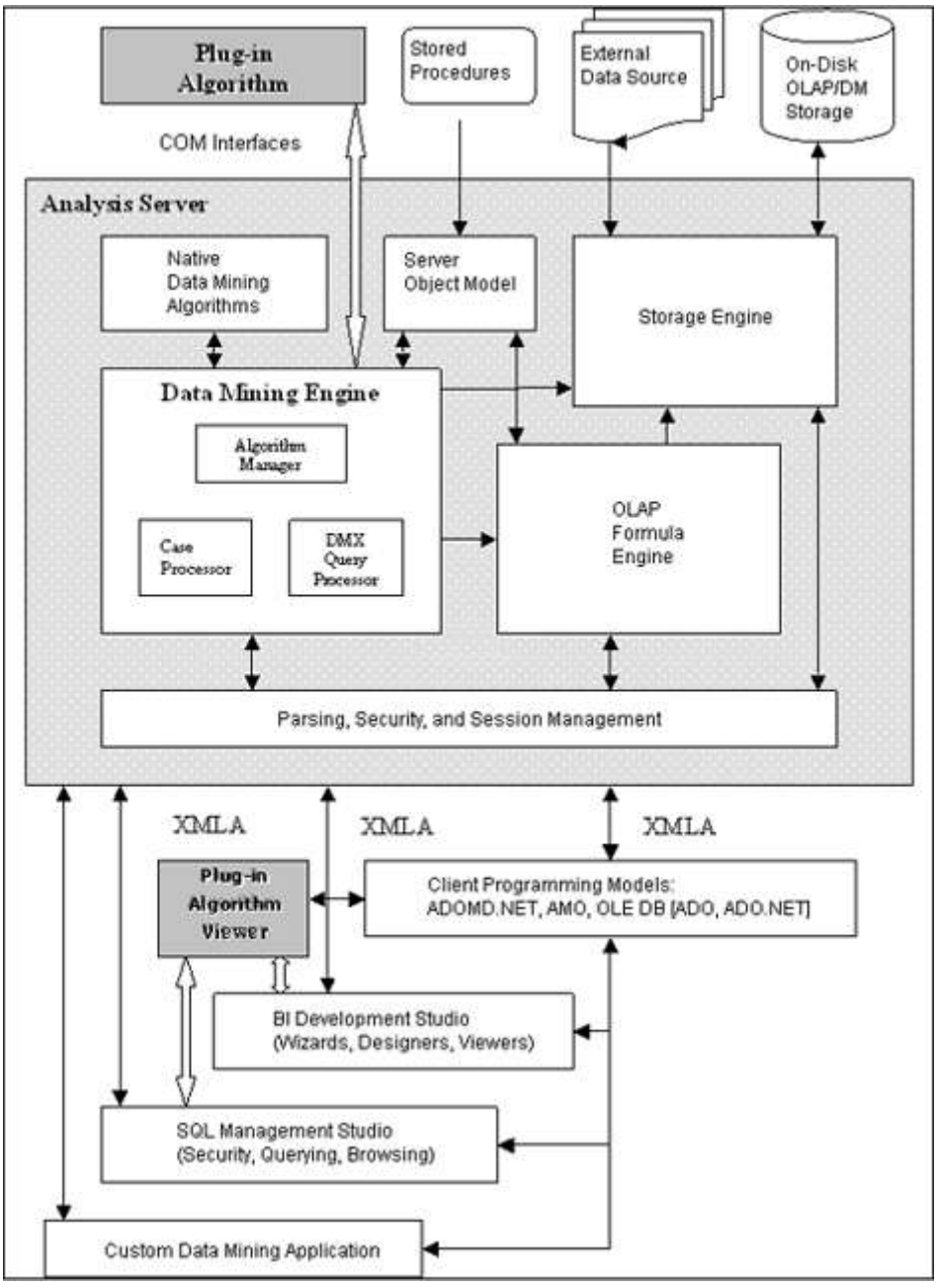

Рис.6. Архитектура Analysis Services.

Стандартным средством для работы с Data Mining плагинами SQL Server'a является среда разработки Business Intelligence Development Studio (BIDS), которая обладает инструментами для создания и обучения Data Mining моделей, визуализации и анализа качества моделей, предсказания.

Однако, возможностей BIDS для некоторых задач может быть недостаточно. Например, когда есть необходимость работать с Data Mining плагинами из стороннего приложения, для более глубокой автоматизации процесса обучения моделей и предсказания, для специфичной визуализации моделей и т.п.

Службы SQL Server Analysis Services поддерживают так называемую архитектуру с тонким клиентом. Среда разработки BIDS, среда SQL Management Studio, Excel и другие сторонние клиентские приложения обращаются к SSAS на языке XMLA (XML for Analysis).

XMLA-запросы могут содержать в себе DMX-запросы. Язык DMX (Data Mining Extensions to SQL) – SQL подобный язык запросов, позволяющий производить data mining-операции: создание и обучение модели, предсказание.

Функциональность XMLA базируется на двух основных функциях: Discover и Execute.

Метод Discover позволяет получать информацию о возможностях и состоянии провайдера данных (например DM-модели).

Метод Execute выполняет DMX запросы на сервере.

XMLA запрос использующий Execute выглядит следующим образом:

```
<Execute xmlns="urn:schemas-microsoft-com:xml-analysis">
 <Command>
  <Statement>
   CREATE MINING STRUCTURE [TestDMX]
   (
   [Key] DOUBLE KEY,
   [P] DOUBLE CONTINUOUS
   ) WITH HOLDOUT(30 PERCENT)
  </Statement>
 </Command>
 <Properties>
  <PropertyList>
   <Catalog>TestDS</Catalog>
   <Format>Tabular</Format>
   <Content>SchemaData</Content>
  </PropertyList>
 </Properties>
</Execute>
```

Клиентские DM-приложения на платформе .NET, работающие с Analysis Services, реализуются с помощью интерфейсов ADOMD.NET, которые инкапсулируют в себе создание и парсинг XMLA запросов. Среда SQL Server Management Studio позволяет выполнять XMLA и DMX запросы непосредственно на сервере.

6.1 Использование DMX для работы с Discovery Plugin

В данном разделе будет показано, как с помощью DMX и XMLA запросов можно создавать и обучать модель для алгорима Discovery, а также производить прогнозирование.

1)Сначала необходимо подключиться к Analysis Server'у с помощью SQL Server Management Studio и создать новую базу данных (с именем, например, Discovery DMX Test).

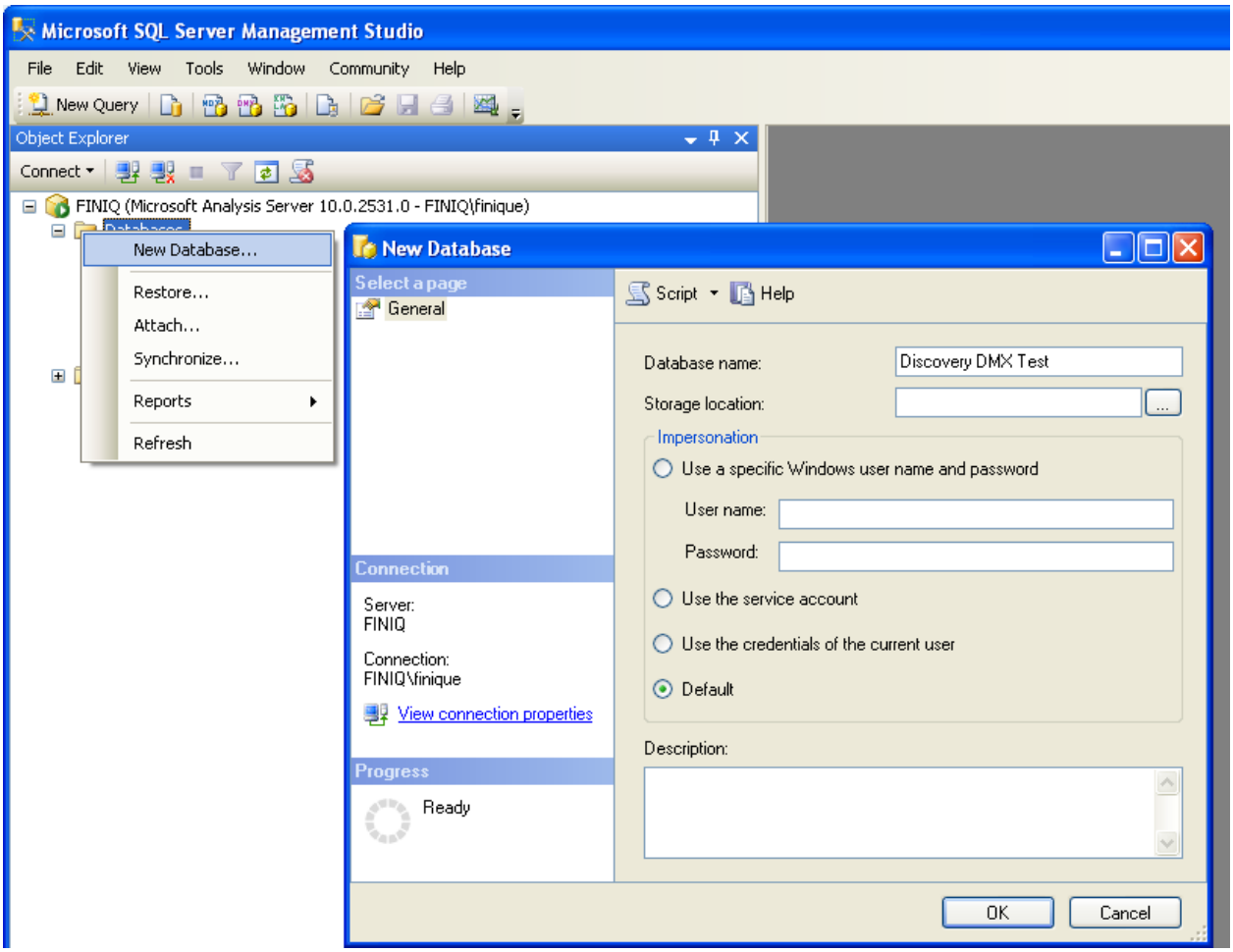

Рис. 7. Создание новой базы данных.

2)Затем с помощью DMX запроса создается Mining Structure, которая описывает задачу интеллектуального анализа данных (mining problem). Mining Structure определяет список используемых колонок, типы данных колонок и информацию о том, как с этими данными работать, т.е. являются ли они непрерывными, дискретными, цикличными и т.д.

```
CREATE MINING STRUCTURE [DiscoveryDMXTest]
(
[Key] DOUBLE KEY,
[F0] DOUBLE CONTINUOUS,
[F1] DOUBLE CONTINUOUS,
[F2] DOUBLE CONTINUOUS,
[F3] DOUBLE CONTINUOUS,
[F4] DOUBLE CONTINUOUS,
[P]  DOUBLE CONTINUOUS
)WITH HOLDOUT(0 PERCENT)
```

Выражение HOLDOUT(N PERCENT) делит данные на тренировочный и тестовый набор, для последнего случайным образом резервируется n процентов изначального набора данных.

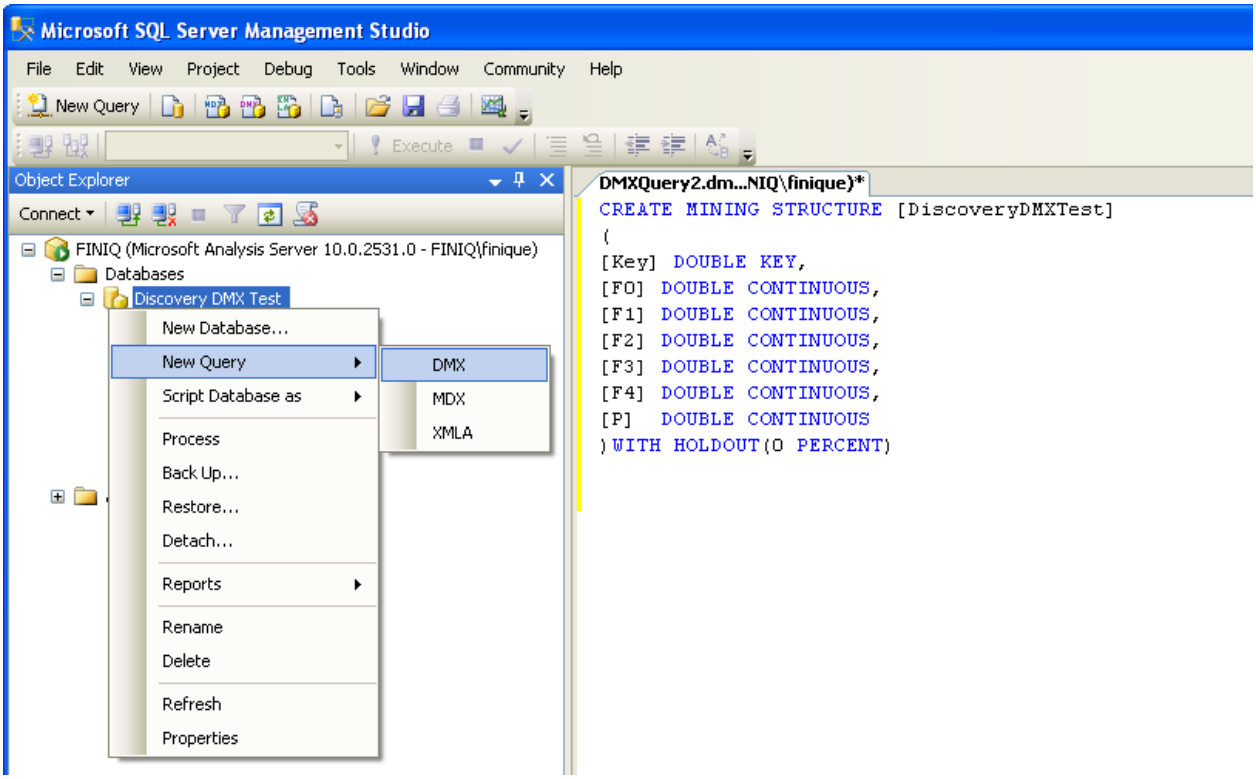

Рис. 8. Выполнение DMX запроса.

3)Следующий этап - создание Mining Model, которая определяет используемый DM-алгоритм, параметры алгоритма, а также как колонки используются в качестве DM-атрибутов (входной атрибут, предсказываемый атрибут, или оба значения одновременно).

```
ALTER MINING STRUCTURE [DiscoveryDMXTest]
ADD MINING MODEL [DiscoveryDMXTest]
(
```

```
[Key],
[F0],
[F1],
[F2],
[F3],
[F4],
[P] PREDICT_ONLY
)
USING Discovery_PlugIn([Condition probability]=0.6)
```

2)Далее необходимо предоставить модели данные для тренировки. Для этого в базу данных (Discovery DMX Test) добавляется источник данных (DataSource), привязанный к файлу .mdb, из которого и будет заполняться DM-модель.

К сожалению, в DMX нет конструкций позволяющих создавать источники данных, но это можно сделать несколькими другими способами, мы воспользуемся XMLA:

```
<Create xmlns="http://schemas.microsoft.com/analysisservices/2003/engine">
<ParentObject>
<DatabaseID>Discovery DMX Test</DatabaseID>  <!--Имя базы с п.2-->
</ParentObject>
<ObjectDefinition>
<DataSource xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ddl2="http://schemas.microsoft.com/analysisservices/2003/engine/2"
xmlns:ddl2_2="http://schemas.microsoft.com/analysisservices/2003/engine/2/2"
xmlns:ddl100_100=
"http://schemas.microsoft.com/analysisservices/2008/engine/100/100"
xsi:type="RelationalDataSource">
<ID>TestDS</ID>  <!--ID Датасурса-->
<Name>TestDS</Name>  <!--Имя Датасурса-->
<ConnectionString>Provider=Microsoft.Jet.OLEDB.4.0;DataSource=
\TestDS.mdb</ConnectionString>  <!--Путь к файлу mdb-->
<ImpersonationInfo>
<ImpersonationMode>Default</ImpersonationMode>
</ImpersonationInfo>
<Timeout>PT0S</Timeout>
</DataSource>
</ObjectDefinition>
</Create>
```

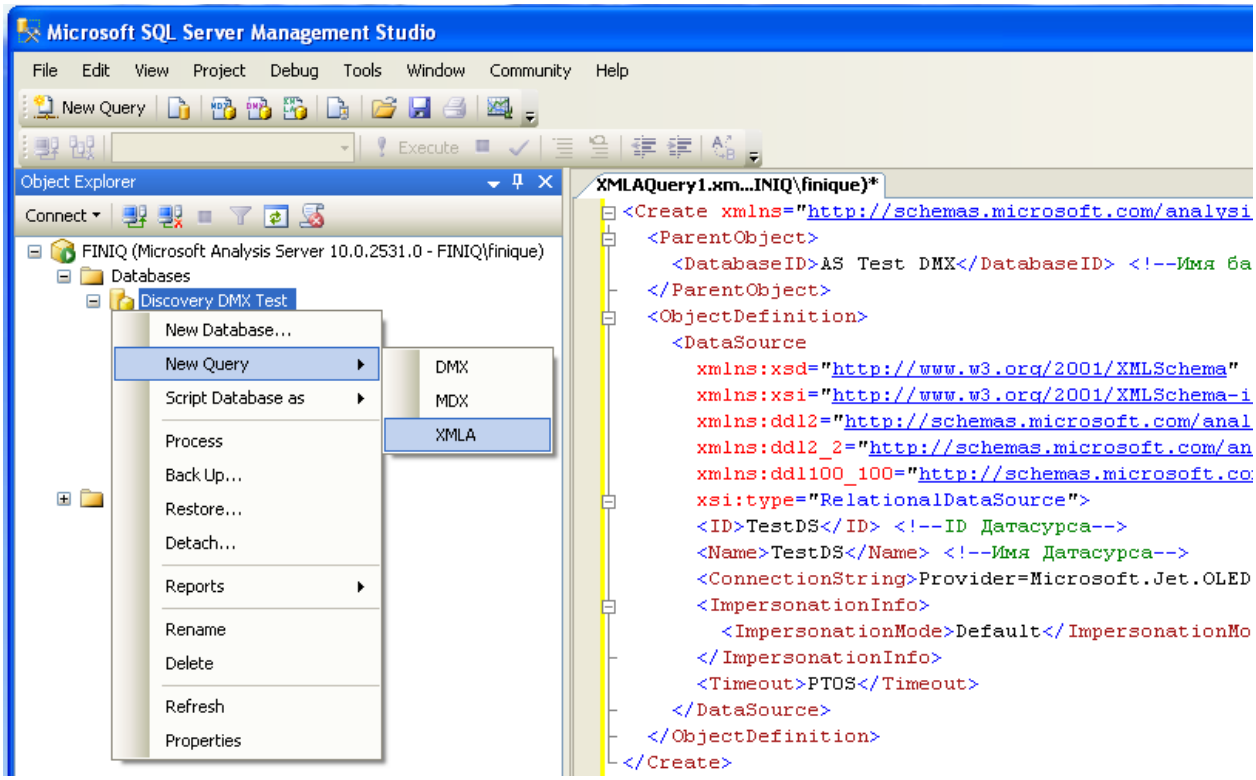

Рис. 9. Выполнение XMLA запроса.

Теперь заполняем модель из созданного источника данных:

```
INSERT INTO MINING STRUCTURE [DiscoveryDMXTest]
([Key], [F0], [F1], [F2], [F3], [F4], [P])
OPENQUERY(TestDS,
'SELECT [Key], [F0], [F1], [F2], [F3], [F4], [P]
FROM [TestDS_Table1]')
```

Данный DMX запрос одновременно производит обучение модели.

6.2 Прогнозирование

SQL Management Studio как и BIDS имеют удобный конструктор DMX запросов на предсказание, хотя, конечно, такие запросы можно писать вручную.

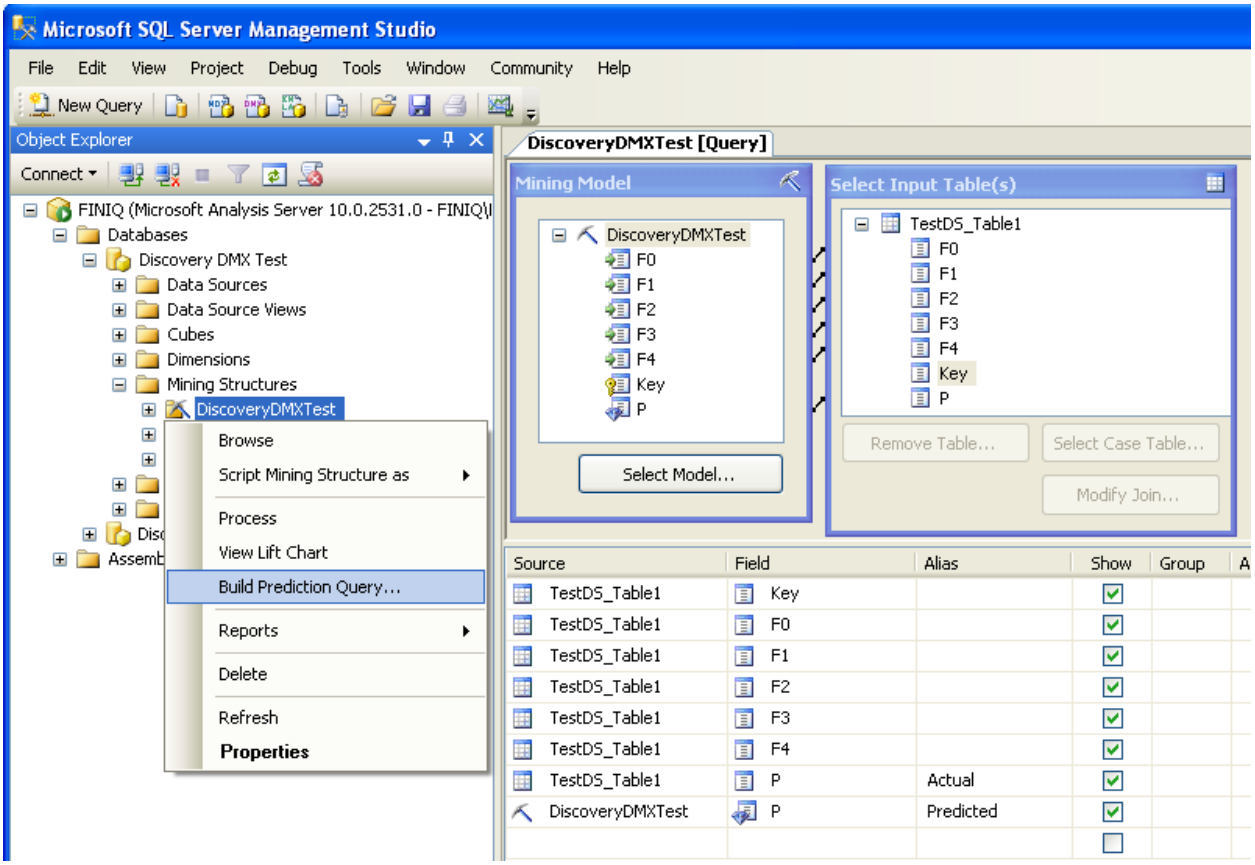

Рис. 10. Создание запроса на предсказание с помощью конструктора.

Следующий запрос выполняет функцию предсказания для всех записей таблицы TestDS_Table1 (на рис. 10. показано создание запроса). Запрос возвращает все колонки исходной таблицы, а также колонку предсказанных значений (Predicted). Далее можно сравнить действительные значения (колонка Actual) с предсказанными, подсчитать ошибку, сравнить с результатами других алгоритмов.

```
SELECT
  t.[Key], t.[F0], t.[F1], t.[F2], t.[F3], t.[F4],
  (t.[P]) as [Actual],
  ([DiscoveryDMXTest].[P]) as [Predicted]
From
  [DiscoveryDMXTest]
PREDICTION JOIN
  OPENQUERY([TestDS],
```

```
    'SELECT
      [Key], [F0], [F1], [F2], [F3], [F4], [P]
    FROM
      [TestDS_Table1]
    ') AS t
ON
  [DiscoveryDMXTest].[F0] = t.[F0] AND
  [DiscoveryDMXTest].[F1] = t.[F1] AND
  [DiscoveryDMXTest].[F2] = t.[F2] AND
  [DiscoveryDMXTest].[F3] = t.[F3] AND
  [DiscoveryDMXTest].[F4] = t.[F4] AND
  [DiscoveryDMXTest].[P] = t.[P]
```

Messages Results

Key	F0	F1	F2	F3	F4	Actual	Predicted
1	1	1	1	1	1	1	1
2	1	1	0	0	0	1	1
3	1	1	0	0	0	1	1
4	1	1	0	0	0	1	1
5	1	1	0	0	0	1	1
6	1	1	0	0	0	1	1
7	1	1	0	0	0	1	1
8	1	1	0	0	0	1	1
9	1	1	0	0	0	1	1
10	1	1	0	0	0	1	1
11	1	1	0	0	0	1	1
12	1	1	0	0	0	1	1
13	1	1	0	0	0	1	1
14	1	1	0	0	0	1	1
15	1	1	0	0	0	1	1

Рис. 11. Результат выполнения запроса предсказания.

6.3 Плагин Discovery для Excel

Как уже было упомянуто ранее, архитектура Analysis Services позволяет работать с DM-плагинами (в том числе и Discovery) с помощью Microsoft Excel [5]. Excel дает возможность использовать всю описанную выше функциональность с помощью хорошо знакомого многим пользовательского интерфейса.

Excel позволяет создавать, обучать, просматривать и управлять DM-моделями. В качестве данных можно использовать таблицу Excel, либо некоторый выбранный диапазон, либо внешний источник данных. На рисунке ниже показано создание DM-модели для алгоритма Discovery в Excel.

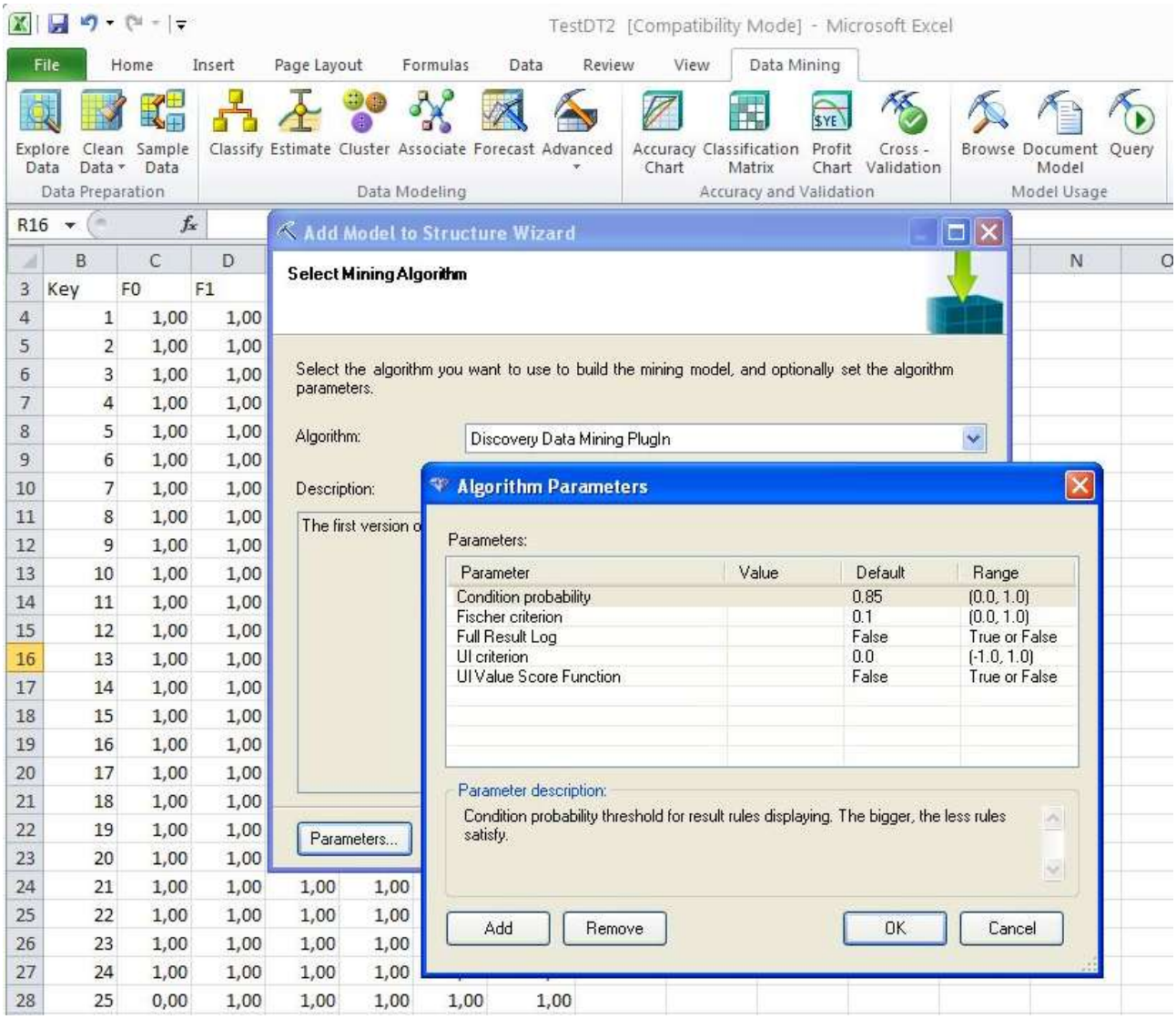

Рис. 12. Создание DM-модели для алгоритма Discovery в Excel.

Плагин «Discovery» имеет следующие параметры:

Condition probability – граничное значение условной вероятности. Правила с УВ меньшей данной границы не отображаются в списке правил, полученных в результате обучения модели.

Fischer criterion - граничная вероятность в критерии Фишера.

Ul criterion – граничное значение величины Юла.

Ul Value Score Function – определяет, использовать в качестве меры правила величину Юла или условная вероятность правила.

Для просмотра DM-моделей в Excel используется стандартное средство визуализации. Оно обладает очень ограниченной функциональностью, однако, есть возможность реализовать для плагина Discovery собственное средство просмотра (Plug-in Viewer), с необходимым набором функций. На рисунке ниже показано стандартное окно просмотра моделей с правилами, найденными в результате обучения модели с помощью алгоритма Discovery.

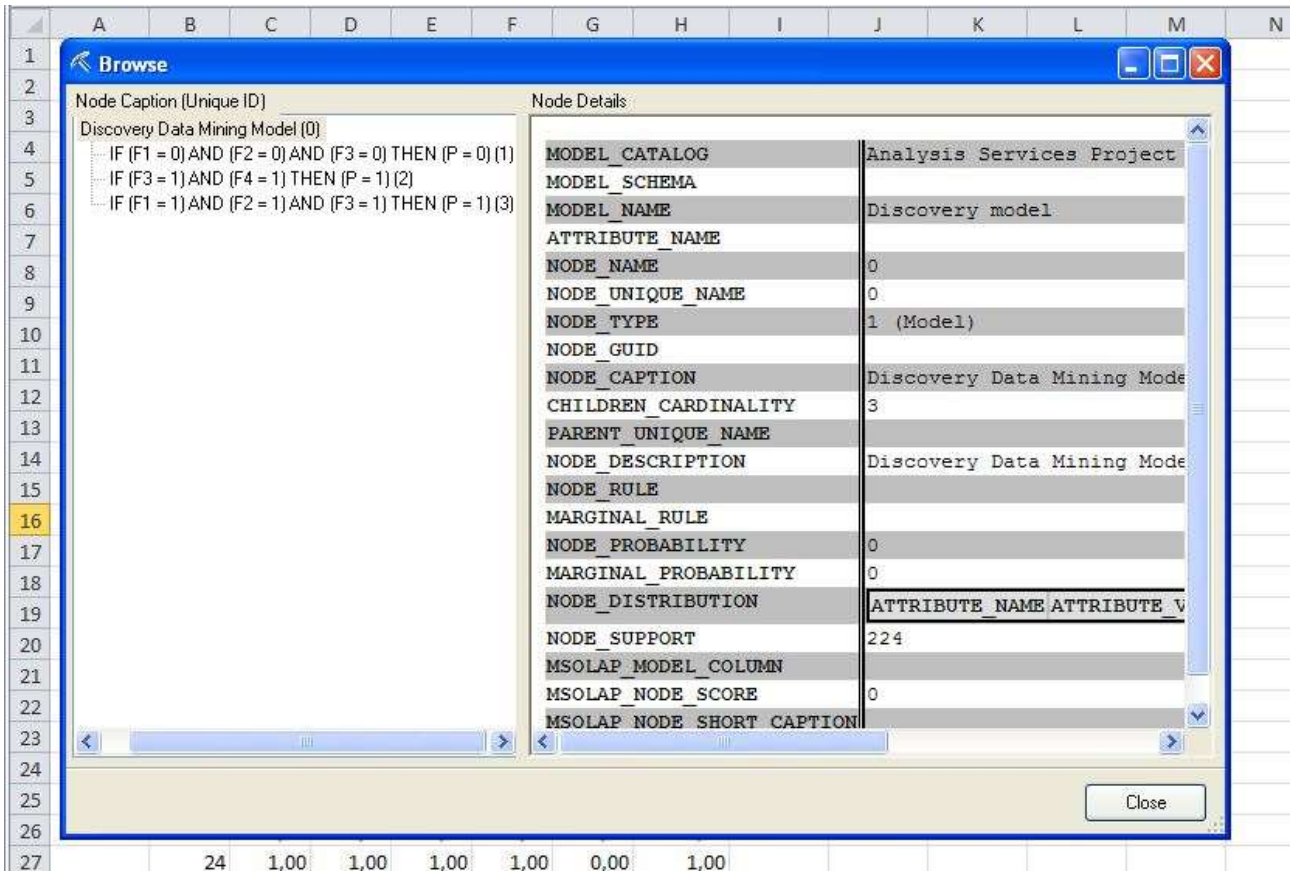

Рис. 13. Стандартное окно просмотра DM-моделей в Excel.

Excel позволяет оценивать точность предсказания DM-моделей с помощью таких известных инструментов, как Accuracy Chart, Classification Matrix, Profit Chart, Cross-Validation.

Также Excel обладает мощным средством для создания DMX запросов. С помощью мастера запросов можно создавать не только запросы на предсказание, но и множество других запросов для построения и управления DM-моделями, для чего в мастере содержится ряд дополнительных шаблонов. На следующем рисунке показан процесс создания запроса на предсказание с помощью мастера запросов.

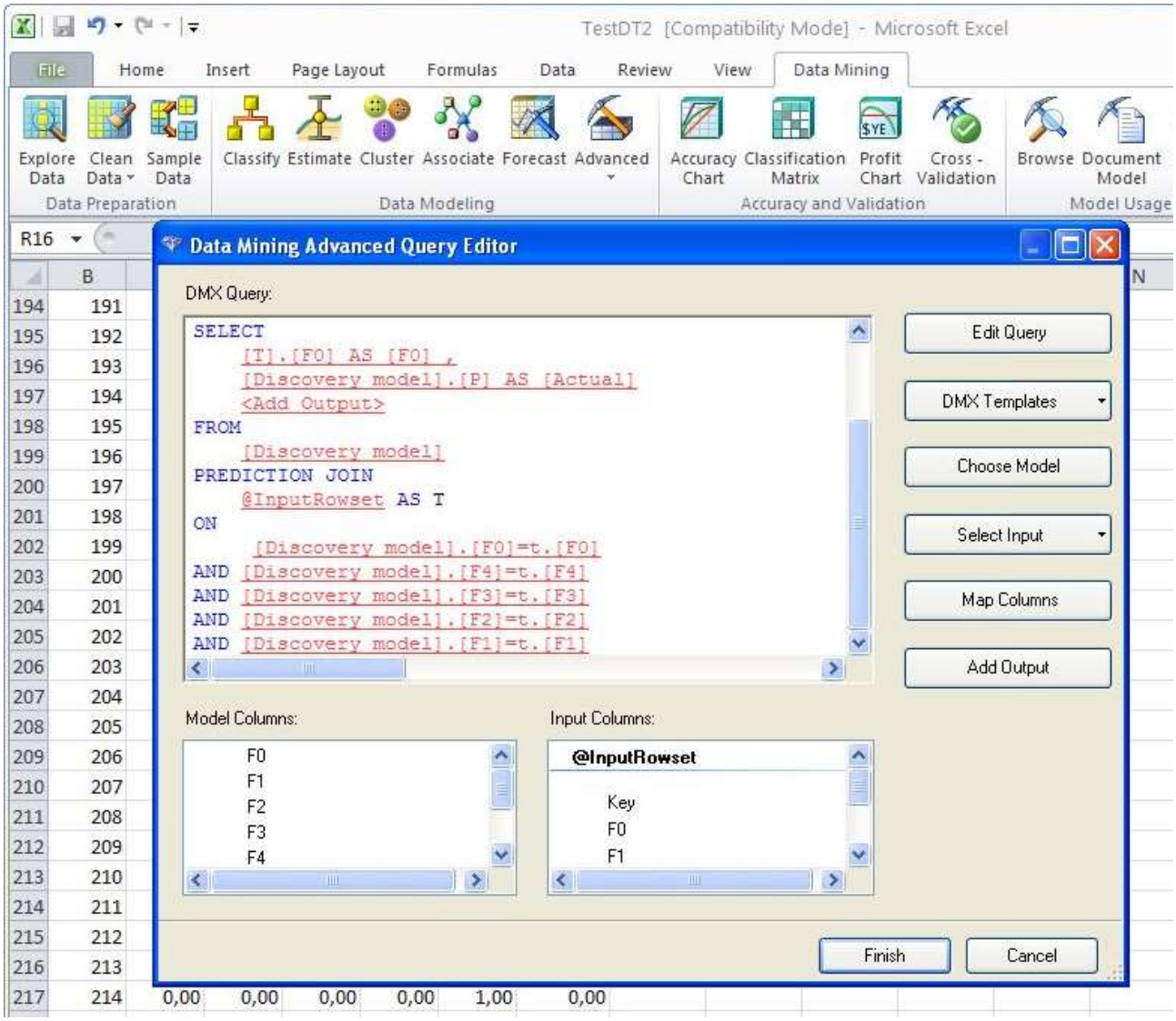

Рис. 14. Создание запроса на предсказание в Excel.

На данный момент Excel является самым удобным способом анализировать данные с помощью плагина Discovery.

7. Теоретическое сравнение

В силу определения семантического вероятностного вывода, который реализован системой «Discovery», в посылку правила всегда добавляются только такие предикаты, которые статистически значимо строго увеличивают условную вероятность правила. Такая проверка в других методах не проводится. В этом случае в правила могут включаться признаки, которые не имеют отношения к предсказанию и в том числе случайные. Опора на случайные признаки в предсказании может значительно ухудшить его точность.

Важность отсева случайных признаков критична, если нам надо не просто получить предсказание, а еще и проинтерпретировать полученные результаты. Специалист предметной области, интерпретируя правило, должен быть уверен, что признаки, входящие в правило действительно имеют отношение к предсказываемому признаку. Иначе интерпретация будет невозможна, либо специалист скажет, что правила бессмысленны и полученные предсказания – гадание на кофейной гуще и отчасти будет прав.

В Decision Trees для выбора разделяющих признаков используется энтропия, но на малых данных или в концах веток дерева могут включаться признаки, которые случайны и минимум энтропии признака получается чисто случайно. В Decision Trees нет статистического критерия проверки случайности добавляемых признаков. Поэтому правила, получаемые в Decision Trees, могут содержать случайные признаки, не имеющие отношение к предсказываемому признаку.

Хотя система «Discovery» и алгоритм Microsoft Association Rules достаточно похожи, в виду того, что в обоих подходах закономерности представляются в форме логических правил, тем не менее, между ними существуют принципиальные отличия.

1) В детерминированном случае, когда нет шума в данных, система «Discovery» обнаружит одно правило A&B $\Rightarrow$ C, истинное на данных. В то же время алгоритм, обнаруживающий ассоциативные правила, обнаружит все правила вида A&B&…&D $\Rightarrow$ C, которые получаются из правила A&B $\Rightarrow$ C добавлением дополнительных условий D,F,...,: A&B&D $\Rightarrow$ C, A&B&F $\Rightarrow$ C;

Например, при анализе тестовой таблицы 1 (описанной в приложении 1), где в качестве входных колонок использовались F_1, F_2, F_3, а также колонка R_1 со случайными данными, алгоритмом Association Rules было обнаружено правило $(F_1 = 1 \wedge F_2 = 1 \wedge F_3 = 1) \Rightarrow P = 1$ с УВ = 1, а также следующие 2 правила с УВ = 1:

$(F_1 = 1 \wedge F_2 = 1 \wedge F_3 = 1 \wedge R_1 = 1) \Rightarrow P = 1,$

$(F_1 = 1 \wedge F_2 = 1 \wedge F_3 = 1 \wedge R_1 = 0) \Rightarrow P = 1.$

Таким образом, в случае, когда цель анализа – найти закономерности в данных, эксперт, использующий алгоритм Association Rules, получит три противоречивых правила с УВ = 1. Доверия к таким результатам не будет.

Кроме того, алгоритмом Association Rules было обнаружено множество правил следующего вида:

$$(F_1 = 1 \wedge R_1 = 1) \Rightarrow P = 1, \quad (F_1 = 1 \wedge R_1 = 0) \Rightarrow P = 1,$$
$$(F_2 = 1 \wedge R_1 = 1) \Rightarrow P = 1, \quad (F_2 = 1 \wedge R_1 = 0) \Rightarrow P = 1,$$
$$(F_1 = 1 \wedge F_2 = 1 \wedge R_1 = 1) \Rightarrow P = 1, \quad (F_1 = 1 \wedge F_2 = 1 \wedge R_1 = 0) \Rightarrow P = 1.$$

Последние правила могут иметь приоритет над правилами с целевым предикатом $P = 0$, и, соответственно, ложно предсказывать 1, когда колонка P содержит 0. Например, в случае, когда на вход подаются колонки F_1, F_2, F_3 и только одна колонка со случайными данными R_1, процент правильно предсказанных алгоритмом Association Rules значений составит около 87%, когда на вход подаются F_1, F_2, F_3 плюс две колонки R_1 и R_2, точность падает до 70% (подробнее см. параграф 8.1, таб. 3).

Система «Discovery» в данном случае обнаружила только следующие правила:

$$(F_1 = 1 \wedge F_2 = 1 \wedge F_3 = 1) \Rightarrow P = 1, \quad (F_1 = 0) \Rightarrow P = 0,$$
$$(F_2 = 0) \Rightarrow P = 0, \quad (F_3 = 0) \Rightarrow P = 0,$$

т.к. они уже имеют УВ = 1, и добавление каких-либо предикатов в условную часть правила не может увеличить условную вероятность правила. Предсказание, основанное на этих правилах, будет 100% точным.

2) когда есть шум в данных, система «Discovery» может обнаружить одно правило A&B ⇒ C, представляющее собой вероятностный закон с определенным уровнем статистической значимости. В то же время алгоритм, обнаруживающий ассоциативные правила, должен обнаружить все детерминированные правила вида A&B&D ⇒ C, A&B&F ⇒ C, включающие случайные признаки, что приведёт к ухудшению предсказания.

Например, при анализе тестовой таблицы 1 с 3% шумом, и колонками F_1, F_2, F_3 и R_1 в качестве входных колонок, системой «Discovery» были обнаружены только 4 правила, являющиеся СВЗ:

$$(F_1 = 1 \wedge F_2 = 1 \wedge F_3 = 1) \Rightarrow P = 1, \quad (F_1 = 0) \Rightarrow P = 0,$$
$$(F_2 = 0) \Rightarrow P = 0, \quad (F_3 = 0) \Rightarrow P = 0.$$

Эти правила не содержат колонку со случайными данными, т.к. любое правило, имеющее в условной части колонку R_1 не пройдет проверку критерием Юла-Фишера и будет удалено.

Алгоритм Association Rules в данном примере обнаружил правило $(F_1 = 1 \wedge F_2 = 1 \wedge F_3 = 1) \Rightarrow P = 1$ с УВ = p, а также правила:

$$(F_1 = 1 \wedge F_2 = 1 \wedge F_3 = 1 \wedge R_1 = 1) \Rightarrow P = 1, (F_1 = 1 \wedge F_2 = 1 \wedge F_3 = 1 \wedge R_1 = 0) \Rightarrow P = 1,$$

причем одно из них имеет УВ > p. Таким образом, в случае, когда цель анализа – найти закономерности в данных, эксперт, использующий алгоритм Association Rules, получит три противоречивых правила. При этом правило с наибольшей УВ может содержать колонку со случайными данными.

Кроме того, из-за шума в данных алгоритм Association Rules обнаруживает множество правил следующего вида:

$$(F_1 = 1 \wedge F_2 = 1 \wedge F_3 = 0 \wedge R_1 = 1) \Rightarrow P = 1, \quad (F_1 = 1 \wedge F_2 = 1 \wedge F_3 = 0 \wedge R_1 = 0) \Rightarrow P = 1,$$
$$(F_1 = 1 \wedge F_2 = 0 \wedge F_3 = 1 \wedge R_1 = 1) \Rightarrow P = 1, \quad (F_1 = 1 \wedge F_2 = 0 \wedge F_3 = 1 \wedge R_1 = 0) \Rightarrow P = 1,$$
$$(F_1 = 0 \wedge F_2 = 1 \wedge F_3 = 1 \wedge R_1 = 1) \Rightarrow P = 1, \quad (F_1 = 0 \wedge F_2 = 1 \wedge F_3 = 1 \wedge R_1 = 0) \Rightarrow P = 1,$$

которые могут иметь приоритет над правилами с целевым предикатом P = 0 и ложно предсказывать 1, когда колонка P содержит 0. Это приводит к ухудшению предсказания (что показано на рис. 19 в приложении 5).

Сравнивая алгоритм «Discovery» с Neural Network, нужно еще раз отметить, что система «Discovery» имеет четкий статистический критерий, позволяющий исключать случайные признаки из правил. Алгоритм Neural Network такого критерия не имеет. Его метод остановки обучения связан с точностью предсказания на обучающем наборе, из-за чего этот алгоритм склонен к переобучению (overfitting). Если на обучающем наборе подбирать параметры алгоритма, например, увеличивать кол-во скрытых слоев, можно получить очень хорошие результаты предсказания на обучающем наборе, но на тестовом наборе в таком случае предсказание будет неудовлетворительным.

8. Экспериментальное сравнение

8.1. Сравнение Discovery с Association Rules на модельных данных

Key	F1	F2	F3	P	R1
1	1	1	1	1	0
2	1	1	1	1	1
3	1	1	0	0	0
4	1	1	0	0	0
5	1	0	1	0	1
6	1	0	1	0	0
7	1	0	0	0	1
8	1	0	0	0	1
9	0	1	1	0	1
10	0	1	1	0	0
11	0	1	0	0	1
12	0	1	0	0	0
13	0	0	1	0	0
14	0	0	1	0	1
15	0	0	0	0	0
16	0	0	0	0	1

Для обучения алгоритмов Discovery и Association Rules использовались: Тестовая таблица 1, полученная из приведенной слева таблицы, масштабированием в несколько раз, и добавлением колонок $R_1 - R_5$ со случайными данными; а также Тестовая таблица 2, которая построена по такому же принципу, только её колонки F_1, F_2, F_3 содержат значения 1,2,3, и, соответственно, её длина кратна 27 (формальное описание таблиц находится в приложении 1)

Заметим, что обе таблицы содержат две закономерности, достаточные для точного предсказания:

$$(F_1 = 1)\&(F_2 = 1)\&(F_3 = 1) \rightarrow (P = 1)$$

$$(F_1 \neq 1)|(F_2 \neq 1)|(F_3 \neq 1) \rightarrow (P = 0)$$

Т.к. оба сравниваемых алгоритма не могут содержать в своих правилах отрицание и логическое «или», вторая закономерность находится в виде нескольких правил, предсказывающих P = 0.

Для анализа данных таблиц применим систему «Discovery». В качестве входных колонок используем колонки $F_1, F_2, F_3, R_1 - R_5$, в качестве целевого признака – колонку P. В качестве критериев статистической значимости используемая реализация применяет точный критерий Фишера с пороговым значением 0,05 и критерий Юла с пороговым значением 0,1.

Система «Discovery» обнаруживает следующие правила с условной вероятностью (УВ) равной 1.

На тестовой таблице 1:

УВ 1: IF (F1 = 1) AND (F2 = 1) AND (F3 = 1) THEN (P = 1);

УВ 1: IF (F3 = 0) THEN (P = 0);

УВ 1: IF (F2 = 0) THEN (P = 0);

УВ 1: IF (F1 = 0) THEN (P = 0);

На тестовой таблице 2:

УВ 1: IF (F1 = 1) AND (F2 = 1) AND (F3 = 1) THEN (P = 1);

УВ 1: IF (F3 = 2) THEN (P = 0);

УВ 1: IF (F3 = 3) THEN (P = 0);

УВ 1: IF (F2 = 2) THEN (P = 0);

УВ 1: IF (F2 = 3) THEN (P = 0);

УВ 1: IF (F1 = 2) THEN (P = 0);

УВ 1: IF (F1 = 3) THEN (P = 0);

Теперь проанализируем тестовые таблицы с помощью Association Rules. В качестве входных колонок используем колонки F_1, F_2, F_3, в качестве целевого признака – колонку P.

Алгоритм Association Rules обнаруживает 20 правил с УВ, равной 1, на тестовой таблице 1 (57 на тестовой таблице 2), в том числе и все правила найденные системой «Discovery». При добавлении колонки R_1 ко входным колонкам Association Rules обнаруживает уже 60 правил с УВ, равной 1, на тестовой таблице 1 (228 на тестовой таблице 2). В Приложении 2 на рисунке 15 показано как растет количество правил с УВ, равной 1, обнаруженных алгоритмом Association Rules, при добавлении к F_1, F_2, F_3 колонок $R_1 - R_5$ в качестве входных колонок.

Далее посмотрим, как добавление в модель колонок со случайными данными ухудшает качество предсказания «Ассоциативных правил», а также покажем, что количество записей в таблице незначительно влияет на качество предсказания.

Таблица 3. Процент правильно предсказанных значений на тестовой таблице 1.

	0	1	2	3	4	5
Association Rules, n = 256	100%	87.5%	69.53%	45.70%	29.29%	19.53%
Association Rules, n = 1024	100%	87.59%	70.41%	45.31%	30.56%	23.92%
Association Rules, n = 4096	100%	88.15%	69.14%	47.07%	32.86%	24.43%

Таблица 4. Процент правильно предсказанных значений на тестовой таблице 2.

	0	1	2	3	4	5
Association Rules, n = 243	100%	91.81%	74.16%	58.46%	25.68%	16.26%
Association Rules, n = 2187	100%	91.95%	75.76%	55.05%	28.30%	17.92%
Association Rules, n = 19683	100%	91.06%	76.85%	55.19%	29.71%	18.46%

Отметим, что на тестовых таблицах 1 и 2 без шума система «Discovery» имеет 100% правильно предсказанных значений целевой колонки P при любом количестве случайных колонок $R_1 - R_5$, участвующих в обучении модели.

В Приложении 3 на рис. 16, 17 проводится сравнение качества предсказания алгоритма Association Rules и системы «Discovery».

Рассмотрим тестовые таблицы 1 и 2 с наложением 3% шума. В качестве входных колонок используем колонки $F_1, F_2, F_3, R_1 - R_5$, в качестве целевого признака – колонку P.

В результате система «Discovery» обнаруживает следующие правила, являющиеся СВЗ.

На тестовой таблице 1:

УВ 0,854: IF (F1 = 1) AND (F2 = 1) AND (F3 = 1) THEN (P = 1)

УВ 0,959: IF (F2 = 0) THEN (P = 0)

УВ 0,944: IF (F1 = 0) THEN (P = 0)

УВ 0,945: IF (F3 = 0) THEN (P = 0)

Первые два из них являются МСЗ.

На тестовой таблице 2:

УВ 0,910: IF (F1 = 1) AND (F2 = 1) AND (F3 = 1) THEN (P = 1)

УВ 0,988: IF (F1 = 2) AND (F2 = 3) AND (F3 = 2) THEN (P = 0)

УВ 0,962: IF (F2 = 2) AND (F3 = 3) THEN (P = 0)

УВ 0,967: IF (F1 = 3) AND (F2 = 2) THEN (P = 0)

УВ 0,971: IF (F1 = 3) AND (F3 = 3) THEN (P = 0)

УВ 0,977: IF (F1 = 3) AND (F2 = 3) THEN (P = 0)

Первые два из них также являются МСЗ.

Проанализируем тестовые таблицы 1 и 2 с наложением 3% шума с помощью Association Rules. В качестве входных колонок используем колонки F_1, F_2, F_3, в качестве целевого признака – колонку P. В результате, на тестовой таблице 1 Association Rules обнаруживает 29 правил, из них 20 правил с УВ > 0.85, в том числе и все правила, найденные системой «Discovery». На тестовой таблице 2 Association Rules обнаруживает 60 правил с УВ > 0.85, в том числе и все правила, найденные системой «Discovery». В Приложении 4 на рис. 18 показано, как растет количество правил с УВ > 0.85, обнаруженных алгоритмом Association Rules, при добавлении колонок $R_1 - R_5$ в качестве входных колонок.

В Приложении 5 проводится сравнение качества предсказания алгоритма Association Rules и системы Discovery на тестовой таблице 2 с шумом 0%, 2% и 3%.

8.2. Сравнение Discovery с Decision Trees и Neural Network на модельных данных.

В качестве анализируемых данных используются таблицы следующего вида:

F0	F1	F2	F3	F4	P	R1
1	1	0	0	0	1	0
1	1	1	0	0	1	1
1	1	1	1	0	1	0
0	1	1	1	1	1	0
0	0	1	1	1	1	1
0	0	0	1	1	1	0
1	0	1	0	0	0	1
1	0	1	0	0	0	1
1	0	1	0	0	0	1
1	0	1	0	0	0	0
1	0	0	1	0	0	1
1	0	0	1	0	0	0
0	1	0	1	0	0	0
0	1	0	1	0	0	1
0	1	0	1	0	0	0
0	1	0	1	0	0	0
0	1	0	0	1	0	1
0	1	0	0	1	0	0
0	0	1	0	1	0	1
0	0	1	0	1	0	1
0	0	1	0	1	0	1
0	0	1	0	1	0	0
0	0	0	1	0	0	1
0	0	0	1	0	0	0
0	0	0	0	1	0	1

Для обучения сравниваемых алгоритмов использовалась Тестовая таблица 3, полученная из таблицы, приведенной слева, масштабированием в 8 раз (и удалением нескольких записей в конце таблицы), так, что итоговая длина Тестовой таблицы 3 равна 197.

В качестве входных колонок использовались колонки $F_0 - F_4$ и $R_1 - R_5$, в качестве целевого признака – колонка P.

Данная тестовая таблица содержит следующие закономерности, достаточные для точного предсказания:

$$(F_0 = 1)\&(F_1 = 1) \rightarrow (P = 1);$$
$$(F_1 = 1)\&(F_2 = 1) \rightarrow (P = 1);$$
$$(F_2 = 1)\&(F_3 = 1) \rightarrow (P = 1);$$
$$(F_3 = 1)\&(F_4 = 1) \rightarrow (P = 1);$$

А также несколько закономерностей, предсказывающие P = 0.

Сначала для анализа таблицы применим систему «Discovery». В качестве входных колонок используем колонки $F_0 - F_4$, в качестве целевого признака – колонку P. В качестве критериев статистической значимости применим точный критерий Фишера с пороговым значением 0,13 и критерий Юла с пороговым значением минус 0,03.

Система «Discovery» обнаруживает следующие правила с условной вероятностью (УВ) равной 1, предсказывающие P = 1:

УВ 1: IF (F0 = 1) AND (F1 = 1) THEN (P = 1);

УВ 1: IF (F1 = 1) AND (F2 = 1) THEN (P = 1);

УВ 1: IF (F2 = 1) AND (F3 = 1) THEN (P = 1);

УВ 1: IF (F3 = 1) AND (F4 = 1) THEN (P = 1).

А также несколько правил, предсказывающие P = 0. Обнаруженные правила являются достаточными для точного предсказания.

Далее проанализируем Тестовую таблицу 3 с помощью Decision Trees. В качестве входных колонок используем колонки $F_0 - F_4$, в качестве целевой – колонку P. Параметр COMPLEXITY_PENALTY для Decision Trees установим равным 0.001. Алгоритм Decision Trees обнаружил следующие правила, предсказывающие P = 1:

УВ 0.965: IF (F0 = 1) AND (F1 = 1) THEN (P = 1);

УВ 0.982: IF (F1 = 0) AND (F3 = 1) AND (F4 = 1) THEN (P = 1);

А также несколько правил, предсказывающие P = 0. Нетрудно заметить, что найденных Decision Trees правил не достаточно для точного предсказания P = 1.

Алгоритм Neural Network не находит закономерности в виде логических правил, потому сложно сделать вывод о точности предсказания не проводя собственно теста.

Для сравнения качества предсказания алгоритмов будем использовать тестовую таблицу 4, полученную масштабированием тестовой таблицы 3 в 4 раза. Введем следующую меру ошибки: если алгоритм ошибочно предсказывает 1 вместо 0, то стоимость ошибки равна 1, если ошибочно предсказывает 0 вместо 1, то стоимость ошибки равна 3. Данную меру ошибки можно представить в виде таблицы:

Actual\Predicted	0	1
0	0	1
1	3	0

Максимальная ошибка, возможная на тестовой таблице 4 имеет стоимость 1172. В результате выполнения предсказания для записей тестовой таблицы 4, Discovery не допустила ни одной ошибки, т.е. предсказала абсолютно точно. Decision Trees на тестовой таблице 4 допустила ошибку общей стоимостью 96, или 8,2% от максимальной стоимости ошибки.

Проанализируем, как добавление шума в таблицы влияет на обнаруживаемые закономерности и ухудшает качество предсказания алгоритмов Discovery и Decision Trees.

Под таблицей с n-процентным шумом мы подразумеваем таблицу, в которой в n процентах ячеек, выбранных случайным образом, значение признака заменено на противоположное.

На тестовой таблице 3 с шумом 1% система «Discovery» обнаруживает следующие правила, предсказывающие P = 1:

УВ 0.958: IF (F0 = 1) AND (F1 = 1) THEN (P = 1);

УВ 0.962: IF (F1 = 1) AND (F2 = 1) THEN (P = 1);

УВ 0.960: IF (F2 = 1) AND (F3 = 1) THEN (P = 1);

УВ 1.000: IF (F3 = 1) AND (F4 = 1) THEN (P = 1);

А также правила, предсказывающие P = 0. На этой же таблице Decision Trees обнаруживает следующие правила, предсказывающие P = 1:

УВ 0.982: IF (F1 = 0) AND (F3 = 1) AND (F4 = 1) THEN (P = 1);

УВ 0.982: IF (F0 = 1) AND (F1 = 1) AND (F2 = 1) THEN (P = 1);

УВ 0.878: IF (F0 = 0) AND (F1 = 1) AND (F2 = 1) THEN (P = 1);

А также несколько правил, предсказывающие P = 0. В следующей таблице показаны результаты предсказаний, выполненных алгоритмами Discovery и Decision Trees. Алгоритмы обучались на таблице 3 с шумами в 2, 4 и 6 процента. Предсказание выполнялось для записей таблицы 4 с теми же шумами 2, 4, 6 процента.

Таблица 5. Абсолютная и относительная стоимость ошибок сравниваемых алгоритмов на тестовой таблице три с шумами в 2, 4 и 6 процентов.

	Discovery		Decision Trees		Neural Network	
	Стоимость	% от max	Стоимость	% от max	Стоимость	% от max
шум 0%	0	0	96	8.2	0	0
шум 2%	70	6	162	13.8	203	17.3
шум 4%	136	11.6	134	11.4	238	20.3
шум 6%	190	16.2	255	21.8	354	30.2

Во всех приведенных случаях система Discovery работала лучше Decision Trees и Neural Network.

8.3. Сравнение алгоритмов на реальных данных. Таблица "Statlog German Credit Data".

Для следующего теста возьмем данные с известного DM-репозитория UC Irvine Machine Learning Repository (http://archive.ics.uci.edu/ml/). Таблица "Statlog German Credit Data" содержит данные людей, претендующих на получение заемных средств в банке [9]. Цель интеллектуального анализа таблицы - определить кредитные риски претендента. Таблица размером 1000 записей, содержит 20 входных атрибутов, таких как: кредитная история, цель кредита, размер кредита, накопления на счетах, время занятости на текущем месте работы, пол/семейное положение, наличие недвижимости и т.д. Предсказываемый атрибут принимает два значения: 1 - низкие кредитные риски, 2 - высокие риски.

К данным прилагается матрица стоимостей ошибок, в которой совершенно логично указано, что ложное предсказание низких кредитных рисков наносит больший ущерб, чем ложное предсказание высоких рисков. Максимальная возможная ошибка имеет стоимость 2200.

Actual\Predicted	1	2
1	0	1
2	5	0

Для устранения искажений результатов теста вследствие эффекта переобучения (overfitting) тестируемых DM-алгоритмов, воспользуемся методом скользящего экзамена [4]: исходная таблица A случайным образом разбивается на n равных по размеру, не пересекающихся наборов A(i), i = 0…n-1. В нашем случае n = 20. Формируются наборы B(i) как дополнения к

A(i), т.е. B(i) = A – A(i). Затем проводится n тестов, в которых DM-модель обучается на B(i), а предсказание проводится для записей из A(i). Результаты предсказаний по всем A(i) соединяются в одну таблицу, для которой считается общая стоимость ошибки.

Проведем интеллектуальный анализ таблицы German Credit Data с помощью описанного выше метода алгоритмами Discovery, Decision Trees, Neural Network и Association Rules.

Для Discovery в качестве критериев статистической значимости применим точный критерий Фишера с пороговым значением 0,06 и критерий Юла с пороговым значением 0. В качестве меры правила возьмем величину Юла. Для Decision Trees установим параметр COMPLEXITY_PENALTY равным 0,01. Для Neural Network установим параметр HOLDOUT_PERCENTAGE равным 40 и параметр HIDDEN_NODE_RATIO равным 6. Данные параметры обеспечивают наилучшие результаты алгоритмов.

Таблица 6. Абсолютная и относительная стоимость ошибок сравниваемых алгоритмов на таблице German Credit Data.

	Discovery		Decision Trees		Neural Network	
	Стоимость	% от max	Стоимость	% от max	Стоимость	% от max
German Credit Data	700	31.8	724	32.9	936	42.5

Как видим, система Discovery в данном тесте показала несколько лучшие результаты, чем Decision Trees, и значительно лучшие, чем Neural Network. Association Rules на данном тесте показал неприемлемо плохие результаты: на всех записях тестовых таблиц алгоритм предсказал значение 1, и ни разу не предсказал значение 2.

8.4. Сравнение алгоритмов на реальных данных. Таблица "Magic".

Следующий тест также взят с DM-репозитория UC Irvine Machine Learning Repository. Таблица «Magic» содержит данные телескопа Черенкова, ведущего наблюдение за гамма-лучами высокой энергии [8]. Цель интеллектуального анализа таблицы – отделять истинные (сигнальные) высокоэнергетические фотоны от фоновых фотонов, инициированных космическим излучением в верхних слоях атмосферы. Таблица содержит 19020 записей, 10 атрибутов, содержащие параметры излучения, а также целевой (предсказываемый) атрибут, принимающий 2 значения: 1 – сигнальный фотон, 2 – фоновый фотон. Каждая запись таблицы соответствует одному фотону, зарегистрированному телескопом.

К данным не прилагается таблица стоимостей ошибок, потому определим стоимости ошибок обоих типов равными 1. Тогда максимальная возможная ошибка имеет стоимость 19020.

Actual\Predicted	1	2
1	0	1
2	1	0

Для устранения искажений результатов теста из-за эффекта переобучения (overfitting) тестируемых DM-алгоритмов, используем тот же прием, который был описан выше. А именно: исходная таблица «Magic» разбивается на 20 пар таблиц <M(i); M-M(i)>, где M-M(i) используется для обучения, а на M(i) производится предсказание. Т.е. обучение и последующее тестирование производятся на разных наборах данных.

Проведем интеллектуальный анализ таблицы «Magic» с помощью описанного выше метода алгоритмами Discovery, Decision Trees, Neural Network и Association Rules..

Для Discovery в качестве критериев статистической значимости применим точный критерий Фишера с пороговым значением 0.08 и критерий Юла с пороговым значением 0.1, условную вероятность правил ограничим числом 0.8. В качестве меры правила используем условную вероятность. Для Association Rules установим MINIMUM_IMPORTANCE равной 0.2 и MINIMUM_PROBABILITY равной 0.8. Для алгоритмов Decision Trees и Neural Network оставим параметры по умолчанию. Данные параметры обеспечивают наилучшие результаты алгоритмов на данной таблице.

Таблица 7. Абсолютная и относительная стоимость ошибок алгоритмов на таблице Magic.

	Стоимость	% от max
Discovery	3692	19.41
Decision Trees	3336	17.54
Neural Network	3946	20.75
Association Rules	3964	20.84

Как видно из таблицы, данный тест оказался особенно удобен для алгоритма Decision Trees, который показал здесь наилучшие результаты. Система Discovery показала лучшие результаты, чем Neural Network и Association Rules.

9. Заключение

В данной работе приведено сравнение системы «Discovery» с алгоритмами Microsoft Association Rules, Decision Trees и Neural Network, встроенными в Microsoft SQL Server Analysis Services (SSAS). Для проведения сравнения система «Discovery» была реализована в виде плагина SSAS, что позволило использовать единую среду разработки Business Intelligence Development Studio, единые средства визуализации Data Mining моделей, и стандартные средства сравнения качества Data Mining моделей.

Для сравнения DM-алгоритмов автором был также разработан ряд модельных тестов, на которых теоретически система «Discovery» должна работать лучше, чем другие рассматриваемые алгоритмы. Также были проведены эксперименты на этих модельных данных, показывающие, что и практически на этих данных система «Discovery» работает лучше. Эти эксперименты показывают, что система «Discovery» имеет свою область применимости. Для сравнения системы «Discovery» с другими методами на произвольных реальных данных были использованы данные из известного репозитория UCI Machine Learning Repository. На этих данных система «Discovery» работает не хуже других методов. В целом эти результаты показывают, что реализация системы «Discovery» в виде плагина к службам Microsoft SSAS вполне оправдана. Более того, подключение системы «Discovery» в качестве плагина к Microsoft Excel, как и другие возможности по созданию «тонких» клиентских приложений, использующих «Discovery», делают эту систему доступной максимально широкому кругу пользователей.

Приложение

Приложение 1

В качестве модельных данных используются следующие тестовые таблицы. Тестовая таблица 1 имеет три значимых колонки F_1, F_2, F_3 определяемые следующим выражением:

$$F_1(i) = \begin{cases} 1, i \in (1, 512k) \\ 0, i \in (512k+1, 1024k) \end{cases}$$, 1024 – количество записей в таблице.

$$F_2(i) = \begin{cases} 1, i \in (1, 256k) \cup (512k+1, 768k) \\ 0, i \in (256k+1, 512k) \cup (768k+1, 1024k) \end{cases}$$

$$F_3(i) = \begin{cases} 1, i \in (1, 128k) \cup (256k+1, 384k) \cup (512k+1, 640k) \cup (768k+1, 896k) \\ 0, i \in (128k+1, 256k) \cup (384k+1, 512k) \cup (640k+1, 768k) \cup (896k, 1024k) \end{cases}$$

и колонку P, используемую в качестве целевого признака:

$$P(i) = \begin{cases} 1, i \in (1, 128k) \\ 0, i \in (128k+1, 1024k) \end{cases},$$

а также 5 колонок $R_1 - R_5$ с независимыми Бернуллиевскими случайными значениями.

Тестовая таблица 2 также имеет три значимых колонки F_1, F_2, F_3 определяемые следующим выражением:

$$F_1(i) = \begin{cases} 1, i \in (1, 729k) \\ 2, i \in (729k+1, 1458k) \\ 3, i \in (1458k+1, 2187k) \end{cases}$$

$$F_2(i) = \begin{cases} 1, i \in (1, 243k) \cup (729k+1, 972k) \cup (1458k+1, 1701k) \\ 2, i \in (243k+1, 486k) \cup (972k+1, 1215k) \cup (1701k+1, 1944k) \\ 3, i \in (486k+1, 729k) \cup (1215k+1, 1458k) \cup (1944k+1, 2187k) \end{cases}$$

$$F_3(i) = \begin{cases} 1, i \in \bigcup_{j=0}^{2} \begin{pmatrix} (1+729kj, 81k+729kj) \cup (1+243k+729kj, 324k+729kj) \\ \cup (1+486k+729kj, 567k+729kj) \end{pmatrix} \\ 2, i \in \bigcup_{j=0}^{2} \begin{pmatrix} (1+81k+729kj, 162k+729kj) \cup (1+324k+729kj, 405k+729kj) \\ \cup (1+567k+729kj, 648k+729kj) \end{pmatrix} \\ 3, i \in \bigcup_{j=0}^{2} \begin{pmatrix} (1+162k+729kj, 243k+729kj) \cup (1+405k+729kj, 486k+729kj) \\ \cup (1+648k+729kj, 729k+729kj) \end{pmatrix} \end{cases}$$

(2187 – количество записей в таблице) и колонку P, используемую в качестве целевого признака:

$$P(i) = \begin{cases} 1, i \in (1, 81k) \\ 0, i \in (1+81k, 2187k) \end{cases},$$

а также 5 колонок $R_1 - R_5$ с независимыми Бернуллиевскими случайными значениями.

На тестовых таблицах можно увидеть две простые закономерности:

$$(F_1 = 1)\&(F_2 = 1)\&(F_3 = 1) \rightarrow (P = 1)$$

$$(F_1 \neq 1)|(F_2 \neq 1)|(F_3 \neq 1) \rightarrow (P = 0)$$

Под тестовой таблицей с n-процентным шумом мы подразумеваем тестовую таблицу, в которой в n процентах ячеек, выбранных случайным образом, значение заменено на противоположное.

Приложение 2

На следующем графике показано, как растет количество правил с УВ, равной 1, обнаруженных алгоритмом Association Rules, при добавлении к F_1, F_2, F_3 колонок $R_1 - R_5$ в качестве входных колонок. Здесь и далее на горизонтальной оси отмечено количество колонок $R_1 - R_5$ со случайными данными, используемых (в дополнение к F_1, F_2, F_3) в качестве входных данных.

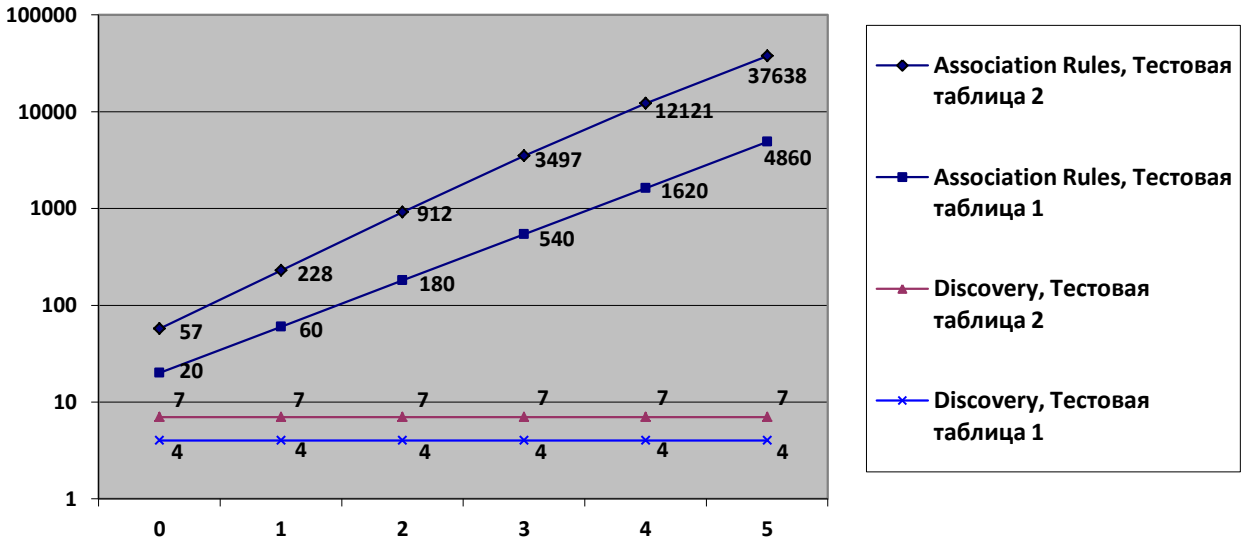

Рис. 15. Количество правил с УВ = 1, обнаруженных алгоритмом Association Rules.

Как видим, количество правил, найденных Association Rules, экспоненциально растет при добавлении новых колонок.

Приложение 3

На следующем графике показан процент правильно предсказанных значений алгоритма Association Rules и системы «Discovery» в зависимости от количества колонок $R_1 - R_5$ со случайными данными, используемых в качестве входных данных, а также размера тестовой таблицы.

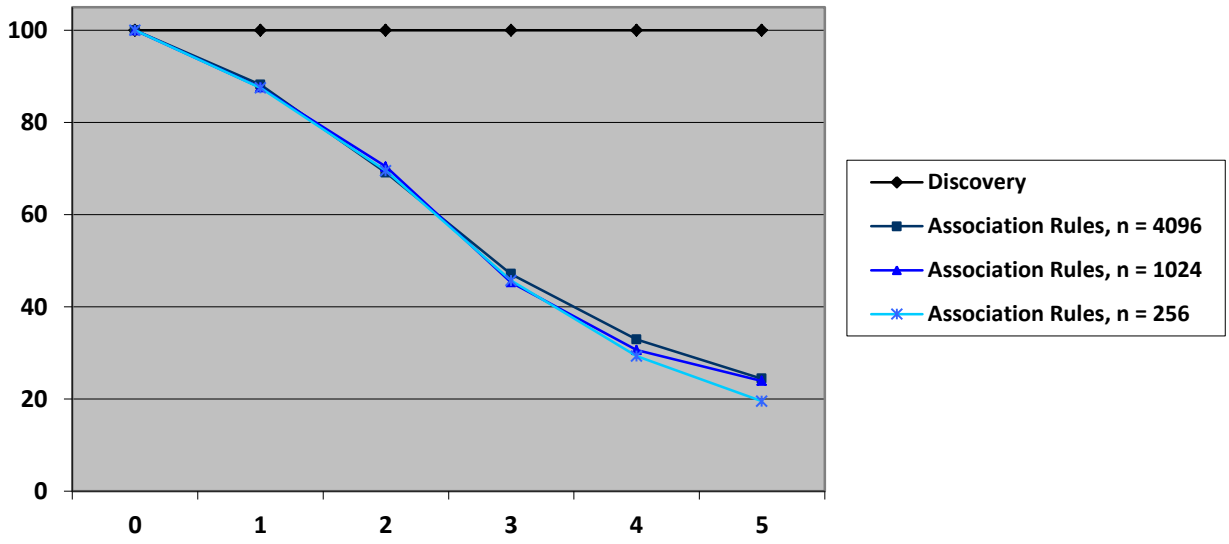

Рис. 16. Процент правильно предсказанных значений при анализе тестовой таблицы 1.

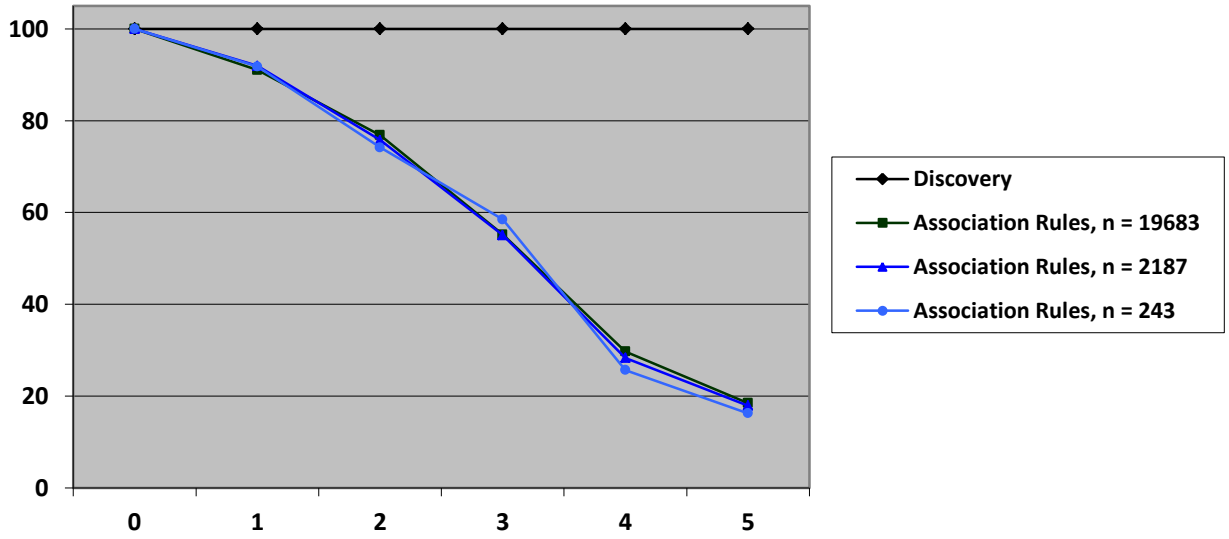

Рис. 17. Процент правильно предсказанных значений при анализе тестовой таблицы 2.

Как видим, при увеличении числа колонок $R_1 - R_5$ со случайными данными, используемых в качестве входных данных, качество предсказания алгоритма Association Rules значительно падает. Размер тестовой таблицы незначительно влияет на результат.

Приложение 4

На следующем графике показано, как растет количество правил с УВ > 0.85, обнаруженных алгоритмом Association Rules, при добавлении колонок $R_1 - R_5$ в качестве входных колонок. Анализируются тестовые таблицы с наложением 3% шума.

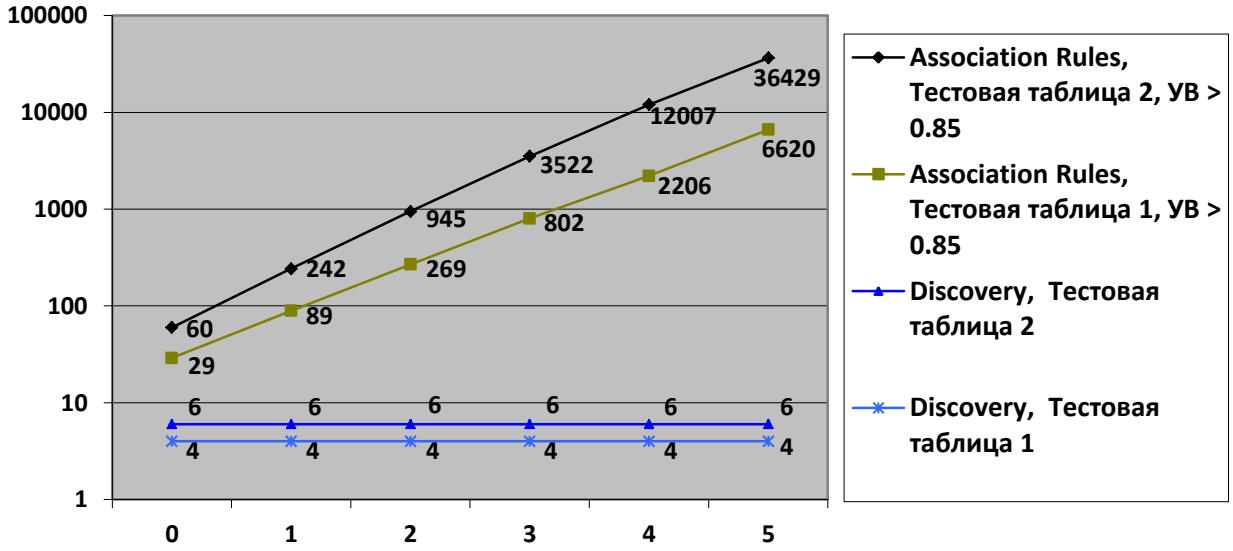

Рис. 18. Количество правил, обнаруженных алгоритмом Association Rules и системой Discovery.

Приложение 5

Сравним качество предсказания системы “Discovery” и алгоритма Association Rules на тестовой таблице 2 с шумом 0%, 2% и 3%. Как видим, система “Discovery” дает наиболее близкое к идеальному предсказание, причем качество прогнозирования не зависит от количества колонок со случайными данными, используемых в качестве входных данных, а зависит только от величины шума. Заметим, что модель, обученная с помощью алгоритма “Discovery” на данных с шумом, будет давать 100% верные предсказания на данных без шума. Алгоритм Association Rules дает аналогичное Discovery качество предсказания в случае, когда случайные колонки не участвуют в обучении модели. При добавлении в модель случайных колонок $R_1 - R_5$ качество прогнозирования Association Rules значительно падает.

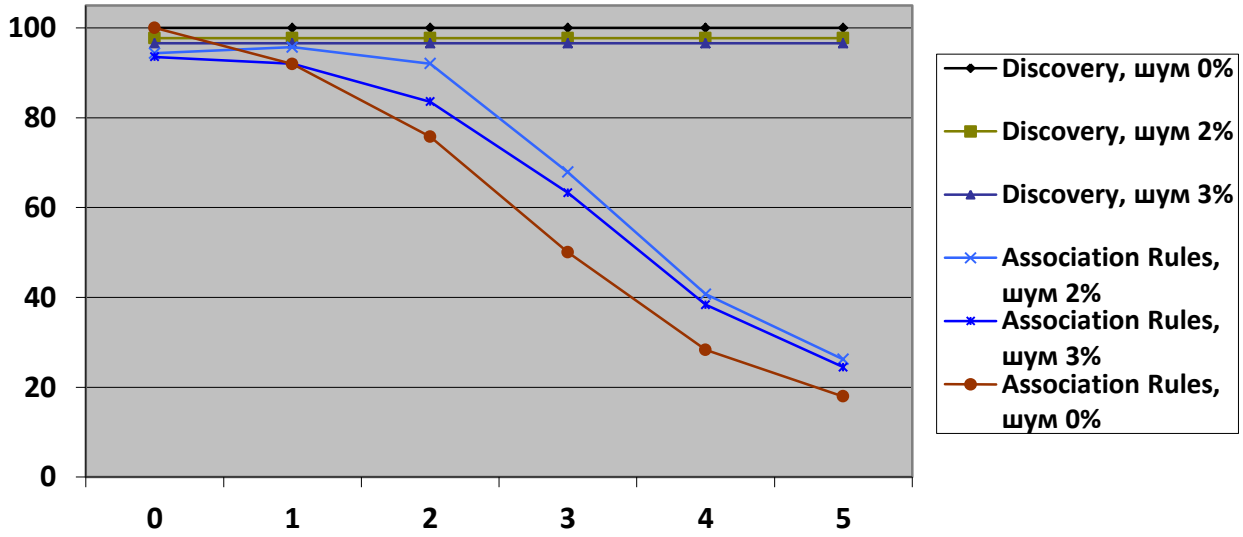

Рис. 19. Процент правильно предсказанных значений при анализе тестовой таблицы 2.

Заметим, что качество предсказания Association Rules на данных без шума при добавлении 2 и более колонок $R_1 - R_5$ заметно хуже, чем на данных с шумом 2% или 3%. Это объясняется тем, что на данных без шума Association Rules обнаруживает огромное количество «равноправных» правил с УВ = 1, выбрать верное из которых не представляется возможным.

Список литературы

1. Витяев Е.Е. Извлечение знаний из данных. Компьютерное познание. Модели когнитивных процессов. – НГУ, Новосибирск, 2006. – 293 с.
2. Закс Ш. Теория статистических выводов – М.: Мир, 1975. – 776 с.
3. Кендалл М. Дж., Стьюарт А. Статистические выводы и связи – М.: Наука, 1973. – 899 с.
4. Лбов Г.С., Бериков В.Б. Устойчивость решающих функций в задачах распознавания образов и анализа разнотипной информации. – Новосибирск: ИМ СО РАН, 2005. – С. 133–135.
5. Data Mining Add-ins // Microsoft Office documentation. [Электронный ресурс].

   URL: http://office.microsoft.com/en-us/excel-help/data-mining-add-ins-HA010342915.aspx (дата обращения: 31.08.2015).
6. Halpern J. Y. An analysis of first-order logic of probability. – Artificial Intelligence. 1990. – С. 311–350.
7. Kovalerchuk, B.Y., Vityaev E.E. Data mining in finance: Relational and hybrid methods. Kluwer, 2000. 308 p.
8. MAGIC Gamma Telescope Data Set // UCI Machine Learning Repository. [Электронный ресурс]. URL: http://archive.ics.uci.edu/ml/datasets/MAGIC+Gamma+Telescope (дата обращения: 31.08.2015).
9. Statlog (German Credit Data) Data Set // UCI Machine Learning Repository. [Электронный ресурс]. URL: http://archive.ics.uci.edu/ml/datasets/Statlog+(German+Credit+Data) (дата обращения: 31.08.2015).
10. Tang Z., MacLennan J. Data Mining with SQL Server 2005. Wiley Publishing, Inc., 2005. 483 p.
11. Vityaev E., Kovalerchuk B. Empirical Theories Discovery based on the Measurement Theory // Mind and Machine, 2006. V.14. N4, P. 551-573.
12. Vityaev E. The logic of prediction. In: Mathematical Logic in Asia // Proceedings of the 9th Asian Logic Conference (August 16-19, 2005, Novosibirsk, Russia). World Scientific, Singapore, 2006. P. 263-276.