14. Интеллектуальный анализ данных и СППР

План

1. Цели и задачи интеллектуального анализа данных
2. Этапы проведения интеллектуального анализа данных
3. Алгоритмы взаимосвязей и кластеризации последовательностей
4. Интеллектуальный анализ данных в СУБД MicrosoftSQLServer

1. Цели и задачи интеллектуального анализа данных

Целью интеллектуального анализа данных (англ. Data mining, другие варианты перевода - "добыча данных", "раскопка данных") является обнаружение неявных закономерностей в наборах данных. Как научное направление он стал активно развиваться в 90-х годах XX века, что было вызвано широким распространением технологий автоматизированной обработки информации и накоплением в компьютерных системах больших объемов данных. И хотя существующие технологии позволяли, например, быстро найти в базе данных нужную информацию, этого во многих случаях было уже недостаточно. Возникла потребность поиска взаимосвязей между отдельными событиями среди больших объемов данных, для чего понадобились методы математической статистики, теории баз данных, теории искусственного интеллекта и ряда других областей.

DataMining - исследование и обнаружение "машиной" (алгоритмами, средствами искусственного интеллекта) в сырых данных скрытых знаний, которые ранее не были известны, нетривиальны, практически полезны, доступны для интерпретации.

Учитывая разнообразие форм представления данных, используемых алгоритмов и сфер применения, интеллектуальный анализ данных может проводиться с помощью программных продуктов следующих классов:

- специализированных "коробочных" программных продуктов для интеллектуального анализа;
- математических пакетов;
- электронных таблиц (и различного рода надстроек над ними);
- средств интегрированных в системы управления базами данных (СУБД);
- других программных продуктов.

Задачи интеллектуального анализа данных.

В ходе проведения интеллектуального анализа данных проводится исследование *множества* объектов (или вариантов). В большинстве случаев его можно представить в виде таблицы, каждая строка которой соответствует одному из вариантов, а в столбцах содержатся значения параметров, его характеризующих. Зависимая *переменная* - *параметр*, *значение* которого рассматриваем как зависящее от других параметров (независимых переменных). Собственно эту зависимость и необходимо определить, используя методы интеллектуального анализа данных.

Задача классификации заключается в том, что для каждого варианта определяется категория или *класс*, которому он принадлежит. В качестве примера можно привести оценку кредитоспособности потенциального заемщика: назначаемые классы здесь могут быть "кредитоспособен" и "некредитоспособен". Необходимо отметить, что для решения задачи необходимо, чтобы множество классов было известно заранее и было бы конечным и счетным.

Задача регрессии во многом схожа с задачей классификации, но в ходе ее решения производится *поиск* шаблонов для определения числового значения. Иными словами, предсказываемый *параметр* здесь, как правило, число из непрерывного диапазона.

Отдельно выделяется **задача прогнозирования** новых значений на основании имеющихся значений числовой последовательности (или нескольких последовательностей, между значениями в которых наблюдается корреляция). При этом могут учитываться имеющиеся тенденции (тренды), сезонность, другие факторы. Классическим примером является прогнозирование цен акций на бирже.

По способу решения задачи интеллектуального анализа можно разделить на два класса: обучение с учителем (от англ. supervised learning) и обучение без учителя (от англ. unsupervise dlearning). В первом случае требуется обучающий набор данных, на котором создается и обучается модель интеллектуального анализа данных. Готовая модель тестируется и впоследствии используется для предсказания значений в новых наборах данных. Иногда в этом же случае говорят об управляемых алгоритмах интеллектуального анализа. Задачи классификации и регрессии относятся как раз к этому типу.

Во втором случае целью является выявление закономерностей имеющихся в существующем наборе данных. При этом обучающая *выборка* не требуется. В качестве примера можно привести задачу анализа потребительской корзины, когда в ходе исследования выявляются товары, чаще всего покупаемые вместе. К этому же классу относится задача кластеризации.

Также можно говорить о классификации задач интеллектуального анализа данных *по* назначению, в соответствии с которой, они делятся на описательные (descriptive) и предсказательные (predictive). Цель решения *описательных задач* - лучше понять исследуемые данные, выявить имеющиеся в них закономерности, даже если в других наборах данных они встречаться не будут. Для предсказательных задач характерно то, что в ходе их решения на основании набора данных с известными результатами строится модель для предсказания новых значений.

Задача кластеризации - заключается в делении *множества* объектов на группы (кластеры) схожих *по* параметрам. При этом, в отличие от классификации, число кластеров и их характеристики могут быть заранее неизвестны и определяться в ходе построения кластеров исходя из степени близости объединяемых объектов *по* совокупности параметров.

Другое название этой задачи - *сегментация*. Например, интернет-магазин может быть заинтересован в проведении подобного анализа базы своих клиентов, для того, чтобы потом сформировать специальные предложения для выделенных групп, учитывая их особенности.

Кластеризация относится к задачам обучения без учителя (или "неуправляемым" задачам).

Задача определения взаимосвязей, также называемая **задачей поиска ассоциативных правил**, заключается в определении часто встречающихся наборов объектов среди *множества* подобных наборов. Классическим примером является *анализ* потребительской корзины, который позволяет определить наборы товаров, чаще всего встречающиеся в одном заказе (или в одном чеке). Эта *информация* может потом использоваться при *размещении товаров* в торговом зале или при формировании специальных предложений для группы связанных товаров. Данная задача также относится к классу "обучение без учителя".

Анализ последовательностей или сиквенциальный *анализ* одними авторами рассматривается как вариант предыдущей задачи, другими - выделяется отдельно. Целью, в данном случае, является обнаружение закономерностей в последовательностях событий. Подобная *информация* позволяет, например, предупредить сбой в работе информационной системы, получив сигнал о наступлении события, часто предшествующего сбою подобного типа. Другой пример применения - *анализ* последовательности переходов *по* страницам пользователей web-сайтов.

Анализ отклонений позволяет отыскать среди *множества* событий те, которые существенно отличаются от нормы. Отклонение может сигнализировать о каком-то необычном событии (неожиданный результат эксперимента, мошенническая операция *по* банковской карте …) или, например, об ошибке ввода данных оператором.
В таблице 1 приведены примеры задач интеллектуального анализа данных из различных областей.

Таблица 1. Примеры применения интеллектуального анализа данных

	Информационные технологии	**Торговля**	**Финансовая сфера**

Классификация			Оценка кредитоспособности
Регрессия			Оценка допустимого кредитного лимита
Прогнозирование		Прогнозирование продаж	Прогнозирование цен акции
Кластеризации		Сегментация клиентов	Сегментация клиентов
Определения взаимосвязей		Анализ потребительской корзины	
Анализ последовательностей	Анализ переходов по страницам web-сайта		
Анализ отклонений	Обнаружение вторжений в информационные системы		Выявление мошенничества с банковскими картами

2. Этапы проведения интеллектуального анализа данных

Этапы проведения интеллектуального анализа данных и декомпозиции задачи:

1. постановка задачи;
2. подготовка данных;
3. изучение данных;
4. построение моделей;
5. исследование и проверка моделей;
6. развертывание и обновление моделей.

На рисунке 1 схематично представлены перечисленные этапы и указаны на примере средства MS SQLServer, с помощью которых они выполняются. Указанные этапы не обязательно будут пройдены один за другим. Например, на одном из промежуточных этапов может выясниться, что в текущей постановке для решения задачи не хватает данных и понадобится снова вернуться к первому этапу.

Рис. 1. Этапы интеллектуального анализа данных

На этапе **постановки задачи** нужно определить, что является целью анализа. В частности, требуется ответить на ряд вопросов, главный из которых - что именно необходимо определить в результате анализа. Также в этом списке:

- Нужно ли будет делать прогнозы на основании модели интеллектуального анализа данных или просто найти содержательные закономерности и взаимосвязи?

- Если требуется прогноз, какой атрибут набора данных необходимо спрогнозировать?
- Как связаны столбцы? Если существует несколько таблиц, как они связаны?
- Каким образом распределяются данные? Являются ли данные сезонными? Дают ли данные точное представление о предметной области?

Как правило, в процессе постановки задачи *аналитик* работает совместно со специалистами в *предметной области*.

Этап **подготовки данных** включает *определение* источников данных для анализа, *объединение* данных и их очистку. Используемые данные могут находиться в различных базах и на разных серверах. Более того, какие-то данные могут быть представлены в виде текстовых файлов, электронных таблиц, находиться в других форматах. В процессе объединения и преобразования данных часто используются возможности служб SQL Server Integration Services (рис.1). Это позволяет существенно автоматизировать процесс подготовки.

Собранные таким образом данные, как правило, нуждаются в дополнительной обработке, называемой очисткой. В процессе очистки при необходимости может производиться удаление "выбросов" (нехарактерных и ошибочных значений), обработка отсутствующих значений параметров, численное преобразование (например, *нормализация*) и т.д.

Следующим этапом является **изучение данных**, которое позволит понять, насколько адекватно подготовленный набор представляет исследуемую предметную область. Здесь может проводиться *поиск* минимальных и максимальных значений параметров, *анализ* распределений значений и других статистических характеристик, сравнение полученных результатов с представлениями о *предметной области*.

Четвертый этап - **построение моделей**. Как уже разбиралось в предыдущей лекции, сначала создается *структура данных*, а потом для структуры создается одна или несколько моделей. Модель включает указание на *алгоритм* интеллектуального анализа данных и его параметры, а также анализируемые данные. При определении модели могут использоваться различные фильтры. Таким образом, не все имеющиеся в описании структуры данные будут использоваться каждой созданной для нее моделью. На рисунке 2 показан пример, в котором для одной структуры создается несколько моделей, использующие различные наборы столбцов и фильтров.

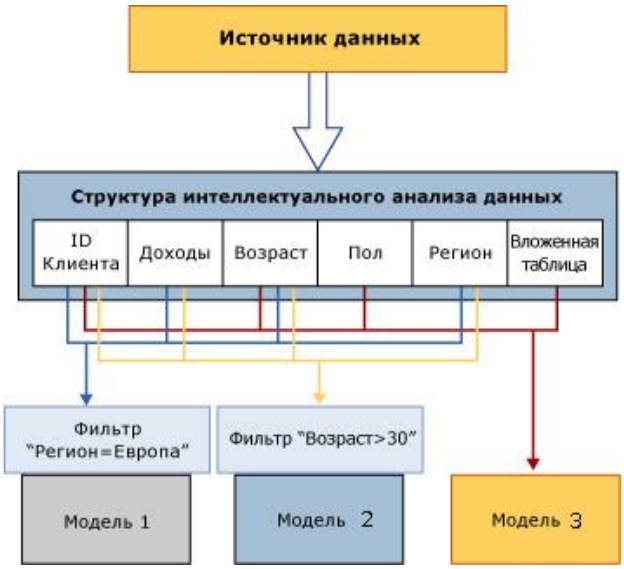

Рис. 2. Несколько моделей, созданных в рамках одной структуры

Модель может проходить обучение, заключающееся в применении выбранного алгоритма к обучающему набору данных. После этого в ней сохраняются выявленные закономерности.

Новую модель можно определить с помощью мастера интеллектуального анализа данных в среде BI DevStudio или с помощью языка *DMX*. Нередко для решения задачи

создается несколько моделей, основанных на разных алгоритмах, чтобы была возможность сравнить результаты и выбрать наилучшую.

Пятый этап - **проверка модели**. Здесь целью является оценка качества работы созданной модели перед началом ее использования в "производственной среде". Если создавалось несколько моделей, то на этом этапе делается выбор в пользу той, что даст наилучший результат.

При решении предсказательных задач интеллектуального анализа качество выдаваемого моделью прогноза можно оценить на проверочном наборе данных, для которого известно *значение* прогнозируемого параметра. В MS SQLServer службы Analysis Services предоставляют средства, упрощающие разделение данных на обучающий и проверочный наборы. Такое *секционирование* можно выполнить автоматически во время построения модели интеллектуального анализа данных. *Точность* прогнозов, создаваемых моделями, можно проверить при помощи таких средств, как *диаграмма* точности прогнозов и *матрица* классификации.

Другой подход, называемый перекрестной проверкой, заключается в том, что создаются подмножества данных и сравниваются результаты работы модели на каждом подмножестве. Такой подход может использоваться как при решении предсказательных, так и описательных задач. *Средства автоматизации* перекрестной проверки доступны при использовании MS SQLServer версии *Enterprise* или *Developer*.

Наиболее эффективные модели **развертываются** в производственной среде. При этом, возможны сценарии интеграции средств интеллектуального анализа данных и пользовательских приложений. И *конечный пользователь*, в ответ на сформированный *запрос*, будет получать результаты анализа в виде отчета. При формировании отчетов о результатах проведенного анализа могут использоваться возможности службы SQLServer Reporting Services.

Со временем характеристики *предметной области* могут меняться, что потребует и изменения шаблонов интеллектуального анализа данных. Может потребоваться *переобучение* существующих моделей или создание новых. В ряде случаев SQLServer позволяет автоматизировать процесс обновления моделей за счет использования служб Integration Services.

3. Алгоритмы взаимосвязей и кластеризации последовательностей

Алгоритм взаимосвязей или ассоциативных правил (AssociationRules) позволяет выявить часто встречающиеся *сочетания* элементов данных и использовать обнаруженные закономерности для построения прогноза. Классический пример - это *анализ* покупательской корзины, когда проводится *поиск* товаров, наиболее часто встречающихся в одном заказе (чеке, транзакции), после чего, на основе выявленных закономерностей становится возможной выдача рекомендаций..

Пример набора правил, формируемых подобным, алгоритмом приведен на рис.3. Предметная область - *торговля* велосипедами и связанными спортивными товарами. Для примера, первое правило говорит о том, что если в заказе присутствует держатель для велосипедной фляги и кепка, с высокой вероятностью будет приобретена и сама фляга.

Правило
Road Bottle Cage = Existing, Cycling Cap = Existing -> Water Bottle = Existing
Mountain-200 = Existing, Mountain Tire Tube = Existing -> HL Mountain Tire = Existing
Mountain-200 = Existing, Water Bottle = Existing -> Mountain Bottle Cage = Existing
Touring-1000 = Existing, Water Bottle = Existing -> Road Bottle Cage = Existing
Road-750 = Existing, Water Bottle = Existing -> Road Bottle Cage = Existing
Touring Tire = Existing, Sport-100 = Existing -> Touring Tire Tube = Existing

Рис. 3. Пример набора правил, формируемых алгоритмом Association Rules

Когда подобный набор правил сформирован, его можно использовать, например, для формирования рекомендаций. Например, покупателю держателя фляги и кепки,

предложат обратить внимание и на имеющиеся фляги. Вопрос заключается в том, как сформировать подобные правила.

Для выявления часто встречающихся наборов объектов может использоваться *алгоритм* Apriori, реализация которого лежит в основе и Microsoft Association Rules, использующегося в SQL Server. *Алгоритм* Apriori последовательно выделяет часто встречающиеся одно-, двух- и т.д., n-элементные наборы. На i-м этапе выделяются i-элементные наборы. Причем сначала выполняется формирование наборов-кандидатов, после чего для них рассчитывается *поддержка*.

Поддержка (англ. *support*) используется для измерения популярности набора элементов. Например, *поддержка* набора элементов {A,B} - общее количество транзакций, которые содержат как A, так и B. Чтобы сократить *запись*, здесь и далее указывается просто A и B, а не A=Existing, B=Existing, как на рис. 3. (existing с англ. "присутствует").

Чтобы количественно охарактеризовать правило, используется *вероятность* (англ. *probability*). Этот же показатель иногда называется достоверностью.

$$Probability(A => B) = Probability(B|A) = Support(A,B)/Support(A)$$

Вероятность для набора $\{A,B\}$ рассчитывается как *отношение* числа транзакций, содержащих этот набор, к общему числу транзакций.

Чтобы оценить взаимную зависимость элементов используется важность (англ. importance) или показатель интереса.

$$Importance(\{A,B\}) = Probability(\{A,B\})/(Probability(A)*Probability(B))$$

Если $Importance(\{A,) = 1$, то A и B - независимые элементы.

$Importance(\{A,B\}) > 1$ означает, что A и B имеют положительную корреляцию (клиент купивший *товар* A вероятно купит и B).

$Importance(\{A,B\}) < 1$ указывает на отрицательную корреляцию.

Для правил важность рассчитывается как логарифм отношения вероятностей:

$$Importance(A => B) = log(Probability(B|A)/(Probability(B|notA))$$

В данном случае равная 0 важность означается, что между A и B нет взаимосвязи. Положительная важность означает, что *вероятность* B повышается, когда справедливо A; отрицательная - *вероятность* B понижается, когда справедливо A.

Настройками пороговых значений можно регулировать максимальное число элементов в рассматриваемых наборах, минимальную *вероятность* при которой правило будет рассматриваться, минимальную поддержку для рассматриваемых наборов и т.д.

Кластеризация последовательностей

Как было рассмотрено выше, алгоритмами ассоциативных правил (взаимосвязей) выявляются часто встречающиеся наборы элементов. Задача кластеризации последовательностей в чем-то схожая - выявить часто встречающиеся последовательности событий. Важное различие заключается в том, что в данном случае учитывается, в какой очередности события происходят (или элементы добавляются в набор). Схожие последовательности объединяются в кластеры. Кроме анализа характеристик кластеров, возможно решение задачи прогнозирования наступления событий на основании уже произошедших ранее.

Примеры применения подобных алгоритмов - *анализ* переходов *по* страницам web-сайтов, *анализ* событий, предшествовавших сбоям в работе информационной системы, и т.д.

Используемый аналитическими службами SQLServer *алгоритм* Micorosoft *Sequence* Clustering - это гибридный *алгоритм*, сочетающий методы кластеризации с анализом марковских цепей. Анализируемое множество вариантов формируется с использованием вложенных таблиц. В таблице 2 представлен условный пример подобного варианта интеллектуального анализа. Важно, чтобы *вложенная таблица* содержала собственный *идентификатор*, который позволил бы определить последовательность элементов.

Таблица 2. Пример формирования варианта для анализа обращений к сайту

Идент. Пользователя	Расположение	Идент. послед.	Тематика
1	Санкт-Петербург	1	Главная страница
		2	Велосипеды
		3	Запчасти
		4	Велосипеды

С помощью марковских моделей анализируется *направленный граф*, хранящий переходы между различными состояниями. *Алгоритм* Microsoft Sequence Clustering использует марковские цепи n-го порядка. Число n говорит о том, сколько состояний использовалось для определения вероятности текущих состояний. В модели первого порядка *вероятность* текущего состояния зависит только от предыдущего состояния. В марковской цепи второго порядка *вероятность* текущего состояния зависит от двух предыдущих состояний, и так далее. *Вероятности перехода* между состояниями хранятся в *матрице переходов*. *По* мере удлинения марковской цепи размер матрицы растет экспоненциально, соответственно растет и время обработки, что надо учитывать при решении практических задач.

Далее *алгоритм* изучает различия между всеми возможными последовательностями, чтобы определить, какие последовательности лучше всего использовать в качестве входных данных для кластеризации. Созданный алгоритмом *список* вероятных последовательностей используется в качестве входных данных для применяемого *по* умолчанию EM-метода кластеризации (англ. *Expectation* Maximization, максимизации ожидания). Целями кластеризации являются как связанные, так и не связанные с последовательностями атрибуты. У каждого кластера есть марковская цепь, представляющая полный набор путей, и *матрица*, содержащая переходы и вероятности последовательности состояний. На основе начального распределения используется правило Байеса для вычисления вероятности любого атрибута, в том числе последовательности, в конкретном кластере.

Некоторые задачи интеллектуального анализа данных, в частности, задача классификации, могут решаться с помощью различных алгоритмов. В случае наличия в данных сложных зависимостей между атрибутами, "быстрые" алгоритмы интеллектуального анализа, такие как упрощённый *алгоритм* Байеса, могут давать недостаточно точный результат. Улучшить ситуацию может применение нейросетевых алгоритмов.

Нейронные сети - это *класс* моделей, построенных по аналогии с работой человеческого мозга. Существуют различные типы сетей, в частности, в SQLServer *алгоритм* нейронной сети использует *сеть* в виде многослойного перцептрона, в состав которой может входить до трех слоев нейронов, или перцептронов. Такими слоями являются *входной* слой, необязательный скрытый слой и *выходной слой* (рис. 4).

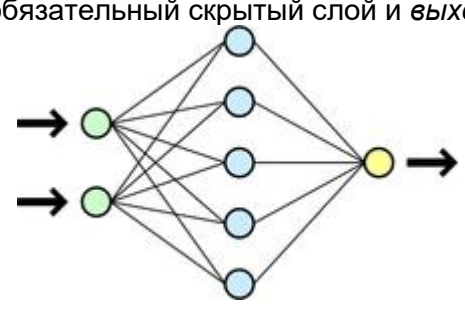

Рис. 4. Пример схемы нейронной сети: входной слой - зеленые узлы, скрытый слой - голубые, выходной - желтый

Каждый *нейрон* получает одно или несколько входных значений (входов)и создает выходное *значение* (один или несколько одинаковых выходов). Каждый *выход* является простой нелинейной функцией суммы входов, полученных нейроном. Входы передаются в прямом направлении от узлов во входном слое к узлам в скрытом слое, а оттуда передаются на *выходной слой*. Нейроны в составе слоя не соединены друг с другом. Скрытый слой может отсутствовать (в частности, это используется алгоритмом логистической регрессии).

Имеющий более двух состояний дискретный *входной атрибут* модели интеллектуального анализа приводит к созданию одного входного нейрона для каждого состояния и одного входного нейрона для отсутствующего состояния (если обучающие данные содержат какие-либо значения NULL). Непрерывный *входной атрибут* "создает" два входных нейрона: один *нейрон* для отсутствующего состояния и один *нейрон* для значения самого непрерывного атрибута. Входные нейроны обеспечивают входы для одного или нескольких скрытых нейронов.

Выходные нейроны представляют значения прогнозируемых атрибутов для модели интеллектуального анализа данных. Дискретный прогнозируемый *атрибут*, имеющий более двух состояний, "создает" один выходной *нейрон* для каждого состояния и один выходной *нейрон* для отсутствующего или существующего состояния. Непрерывные прогнозируемые столбцы "создают" два выходных нейрона: один *нейрон* для отсутствующего или существующего состояния и один *нейрон* для значения самого непрерывного столбца.

Нейрон получает входы от других нейронов или из других данных, в зависимости от того, в каком слое сети он находится. *Входной нейрон* получает входы от исходных данных. Скрытые нейроны и выходные нейроны получают входы из выхода других нейронов нейронной сети. Входы устанавливают связи между нейронами, и эти связи являются путем, по которому производится *анализ* для конкретного набора вариантов.

Каждому входу присвоено *значение*, именуемое весом, которое описывает *релевантность* или важность конкретного входа для скрытого или выходного нейрона. Чем больше *вес*, присвоенный входу, тем более релевантным или важным является *значение* этого входа. Значения веса могут быть отрицательными; это означает, что вход может подавлять, а не активировать конкретный *нейрон*. Чтобы выделить важность входа для конкретного нейрона, *значение* входа умножается на *вес*. В случае отрицательных весов *умножение* значения на *вес* служит для уменьшения важности входа. Схематично это представлено на рис. 5, где x_i - вход, w_i - соответствующий ему *вес*. Каждому нейрону сопоставлена простая нелинейная *функция*, называемая функцией активации, которая описывает *релевантность* или важность определенного нейрона для этого слоя нейронной сети. В качестве функции активации в алгоритме Microsoft Neural Network скрытые нейроны используют функцию гиперболического тангенса (tanh), а выходные нейроны - сигмоидальную (логистическую) функцию. Обе функции являются не линейнымии непрерывными, позволяющими нейронной сети моделировать нелинейные связи между входными и выходными нейронами.

$$sigmoid(x) = \frac{1}{1 + e^{-tx}} \qquad tanh(Ax) = \frac{e^{Ax} - e^{-Ax}}{e^{Ax} + e^{-Ax}}$$

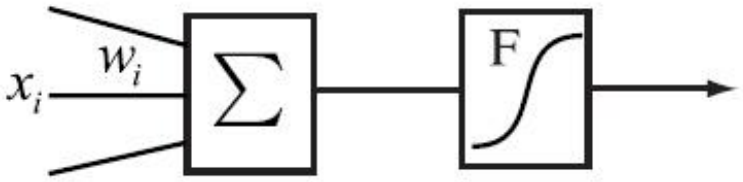

Рис. 5. Формальное представление нейрона

Обучение модели интеллектуального анализа данных производится по следующей схеме. *Алгоритм* сначала оценивает обучающие данные и резервирует определенный *процент* из них для использования при определении точности сети.

Затем *алгоритм* определяет количество и сложность сетей, включаемых в модель интеллектуального анализа данных. Определяется число нейронов в каждом слое. Процесс обучения строится по следующей схеме:

1. На начальной стадии случайным образом присваиваются значения всем весам всех входов в сети. Значения обычно берутся из интервала (-1,1).
2. Для каждого обучающего варианта вычисляются выходы.

3. Вычисляются ошибки выходов. В качестве функции ошибки может использоваться квадрат остатка (квадрат разности между спрогнозированым и фактическим значением).
4. Шаги 2,3 повторяются для всех вариантов, используемых в качестве образцов. После этого веса в сети обновляются таким образом, чтобы минимизировать ошибки.

В процессе обучения может выполняться несколько итераций. После прекращения роста точности модели обучение завершается.

Теперь перейдем к алгоритму логистической регрессии.

Логистическая регрессия является известным статистическим методом для определения влияния нескольких факторов на логическую пару результатов. Например, задача может быть следующей. Предположим, что прогнозируемый столбец содержит только два состояния, и необходимо провести *регрессионный анализ*, сопоставляя входные столбцы с вероятностью того, что прогнозируемый столбец будет содержать конкретное состояние. Результаты, полученные методами линейной и логистической регрессии представлены на рис. 6-a и 6-b соответственно. Линейная регрессия не ограничивает значения функции диапазоном от 0 до 1, несмотря на то, что они должны являться минимальным и максимальным значениями этого столбца. Кривая, формируемая алгоритмом логистической регрессии, в этом случае более точно описывает исследуемую характеристику.

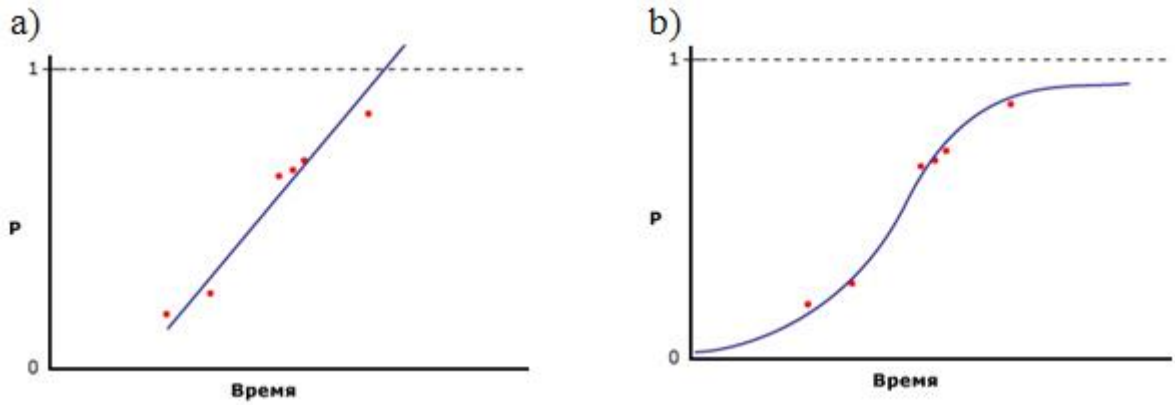

Рис. 6. Сравнение результатов, полученных методами линейной (a) и логистической регрессии (b)

В реализации Майкрософт, для моделирования связей между входными и выходными атрибутами применяется видоизмененная *нейронная сеть*, в которой отсутствует скрытый слой. Измеряется вклад каждого входного атрибута, и в законченной модели различные входы снабжаются весовыми коэффициентами. Название "логистическая регрессия" отражает тот факт, что кривая данных сжимается путем применения логистического преобразования, чтобы снизить эффект экстремальных значений.

4. Интеллектуальный анализ данных в СУБД MicrosoftSQLServer

Рассмотрим реализацию средств интеллектуального анализа данных в СУБД Microsoft SQL Server. На рисунке 7 схематично представлены компоненты СУБД MS SQLServer и выделена подсистема интеллектуального анализа данных.

Рис. 7. Службы и компоненты СУБД Microsoft SQLServer

Службы Analysis Services предоставляют следующие функции и средства для создания решений по интеллектуальному анализу данных:

1. набор стандартных алгоритмов интеллектуального анализа данных;
2. конструктор интеллектуального анализа данных, предназначенный для создания и просмотра моделей интеллектуального анализа данных, управления ими и построения прогнозов;
3. язык расширений интеллектуального анализа данных (Data Mining EXtensionsto SQL,DMX).

Для работы с предоставляемыми средствами интеллектуального анализа используется среда Business Intelligence Development Studio, сокращенно BI Dev Studio (рис. 8, 9).

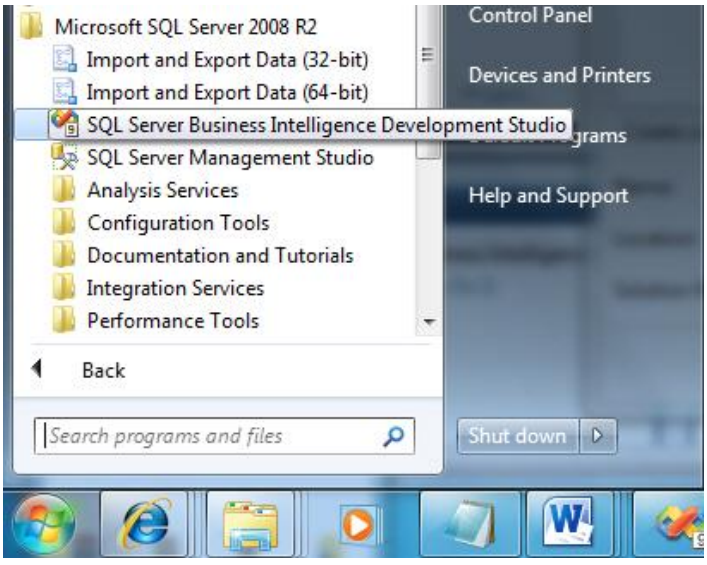

Рис. 8. Запуск SQL Server Business Intelligence Development Studio

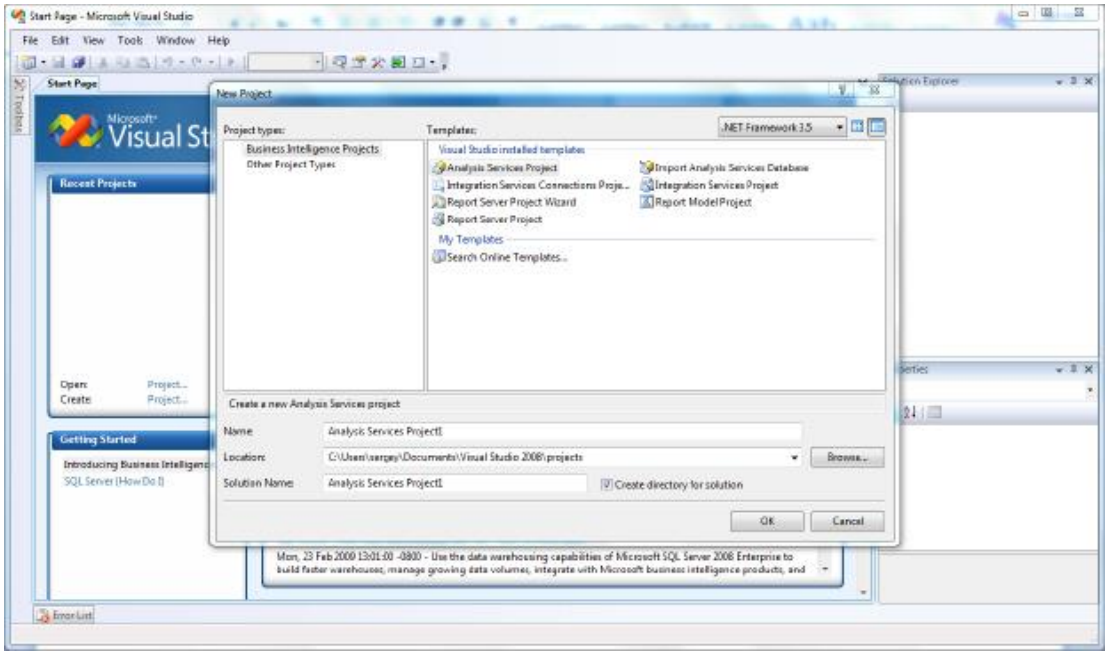

Рис. 9 Создание нового проекта в Business Intelligence Development Studio

Также SQL Server 2008 и 2008 R2 поддерживают создание, управление и использование моделей интеллектуального анализа данных из Microsoft Excel с помощью Надстроек интеллектуального анализа данных SQL Server 2008 для Office 2007. Надстройки свободно доступны для скачивания на сайте Microsoft по адресу (ссылка приводится для локализованной версии, возможно, выпущены более свежие версии): http://www.microsoft.com/downloads/ru-ru/details.aspx?FamilyID=a42c6fa1-2ee8-43b5-a0e2-cd30d0323ca3&displayLang=ru

Выполнению интеллектуального анализа данных с помощью надстроек посвящена первая часть лабораторных работ данного курса.

Структура интеллектуального анализа данных может быть представлена как совокупность исходных данных и описания способов их обработки. Структура содержит модели, которые используются для анализа ее данных. В частности, одна структура может поддерживать несколько моделей. В структуре интеллектуального анализа данных можно выделить обучающий и проверочный набор данных, задав процентное отношение или объем данных.

Модель интеллектуального анализа данных представляет собой сочетание самих данных, алгоритма интеллектуального анализа данных и коллекции значений параметров и фильтров, управляющих использованием и обработкой данных. Модель интеллектуального анализа данных определяется на языке расширений интеллектуального анализа данных или с помощью мастера интеллектуального анализа данных в среде BI DevStudio.

Алгоритм интеллектуального анализа данных представляет собой механизм, создающий модель интеллектуального анализа данных. Чтобы создать модель, алгоритм сначала анализирует набор данных, осуществляя поиск определенных закономерностей и трендов. Алгоритм использует результаты этого анализа для определения параметров модели интеллектуального анализа данных. Затем эти параметры применяются ко всему набору данных, чтобы выявить пригодные к использованию закономерности и получить подробную статистику.

Ниже перечислены алгоритмы интеллектуального анализа данных, реализованные в Microsoft SQL Server 2008 R2 (указание на Майкрософт говорит о том, что это ее реализации алгоритмов, а приводимые английские названия понадобятся нам в дальнейшем):

1. упрощенный алгоритм Байеса (Майкрософт) - MicrosoftNaiveBayes;
2. алгоритм дерева принятия решений (Майкрософт) - MicrosoftDecisionTrees;

3. алгоритм временных рядов (Майкрософт) - MicrosoftTimeSeries;
4. алгоритм кластеризации (Майкрософт) - MicrosoftClustering;
5. алгоритм кластеризации последовательностей (Майкрософт) - MicrosoftSequenceClustering;
6. алгоритм взаимосвязей Майкрософт - MicrosoftAssociationRules;
7. алгоритм нейронной сети (Майкрософт) - MicrosoftNeuralNetwork;
8. алгоритм линейной регрессии (Майкрософт) - MicrosoftLinearRegression;
9. алгоритм логистической регрессии (Майкрософт) - MicrosoftLogisticRegression.

Подробно с перечисленными алгоритмами мы познакомимся в следующих разделах, сейчас же приведем некоторые примеры использования интеллектуального анализа данных и соответствующие им алгоритмы.

Таблица 3. Примеры использования алгоритмов интеллектуального анализа

Задача и пример	Подходящие алгоритмы
Прогнозирование дискретного атрибута. Например, купит ли получатель целевой рассылки определенный продукт.	Алгоритм дерева принятия решений. Упрощенный алгоритм Байеса. Алгоритм кластеризации. Алгоритм нейронной сети.
Прогнозирование непрерывного атрибута. Например, прогноз продаж на следующий год.	Алгоритм дерева принятия решений. Алгоритм временных рядов.
Прогнозирование последовательности. Например, анализ маршрута перемещения по веб-узлу компании.	Алгоритм кластеризации последовательностей
Нахождение групп общих элементов в транзакциях. Например, использование анализа покупательского поведения для предложения дополнительных продуктов заказчику.	Алгоритм взаимосвязей Алгоритм дерева принятия решений
Нахождение групп схожих элементов. Например, разбиение демографических данных на группы для лучшего понимания связей между атрибутами.	Алгоритм кластеризации. Алгоритм кластеризации последовательностей.

Два слова надо сказать и о различиях в версиях СУБД Microsoft SQLServer 2008 и 2008 R2, в которых доступны средства интеллектуального анализа данных. Это версии Standard, Enterprise, а также версия для разработчиков Developer, функционально аналогичная Enterprise, но отличающаяся от нее лицензионными условиями использования. В таблице 4 приведены результаты сравнения возможностей (в соответствии со статьей MSDN "Возможности, поддерживаемые различными выпусками SQL Server 2008", http://msdn.microsoft.com/ru-ru/library/cc645993(v=SQL.100).aspx). По ходу изложения материала, тема различия версий СУБД еще будет неоднократно затрагиватьсяи некоторые непонятные пока возможности будут подробно рассмотрены.

Таблица 4. Различия версий СУБД Microsoft SQLServer 2008 в области интеллектуального анализа данных

Возможность	Enterprise/ Developer	Standard
Стандартные алгоритмы	Да	Да
Средства интеллектуального анализа данных: мастера, редакторы, построители запросов	Да	Да
Перекрестная проверка	Да	
Модели на фильтрованных подмножествах структур интеллектуального анализа данных	Да	
Временные ряды: пользовательское объединение моделей ARTXP и ARIMA	Да	
Временные ряды: прогноз новых данных	Да	
Неограниченные параллельные запросы интеллектуального анализа данных	Да	
Дополнительная настройка алгоритмов	Да	
API-интерфейс для подключаемых модулей алгоритмов	Да	
Параллельная обработка модели	Да	
Временные ряды: прогноз перекрестных рядов	Да	
Неограниченные атрибуты для правил взаимосвязи	Да	
Прогнозирование последовательности	Да	
Множественные цели прогнозирования для упрощенного алгоритма Байеса, нейронной сети и логистической регрессии	Да	

Рассмотрим схему взаимодействия аналитических служб SQLServer с внешними приложениями (рис. 10).

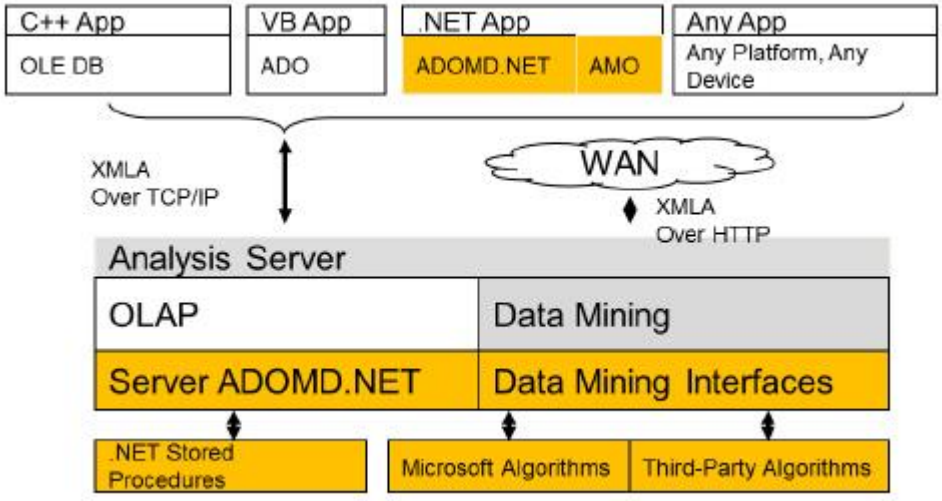

Рис. 10. Схема взаимодействия аналитических служб SQLServer с внешними приложениями

На рисунке 10 видно, что внешние приложения, используя разнообразные средства (ADO.Net и др.) и протокол XMLA (XML for Analysis) "поверх" протокола TCP или HTTP(характерно для web-решений) могут взаимодействовать с аналитическими службами. При этом, в зависимости от типа запроса, задействуется или подсистема OLAP, или подсистема интеллектуального анализа данных. Обрабатываться запросы интеллектуального анализа могут с помощью как стандартных алгоритмов Майкрософт, так и алгоритмов разработки третьих фирм. Результат посредством XMLA передается обратно приложению