5. ЛЕКЦИЯ. Интеллектуальный анализ данных: байесовский классификатор.

Простой байесовский классификатор

Байесовский подход представляет собой группу алгоритмов классификации, основанных на принципе максимума апостериорной вероятности. Сначала определяется апостериорная вероятность (условная вероятность случайного события при условии того, что известны апостериорные данные, т.е. полученные после опыта.) отношения объекта к каждому из классов, а затем выбирается тот класс, для которого она максимальна. Алгоритм предсказывает вероятность того, что объект или наблюдение относится к определенному классу.

Этот подход к классификации является одним из старейших и до сих пор сохраняет прочные позиции в технологиях анализа данных. Кроме того, он лежит в основе многих удачных алгоритмов классификации. Рассмотрим один из них так называемый простой, или наивный, байесовский классификатор. Это специальный случай байесовского классификатора, в котором используется предположение о статистической независимости признаков описывающих классифицируемые объекты. Такое предположение существенно упрощает задачу, поскольку вместо одной многомерной плотности вероятности по всем признакам достаточно оценить несколько одномерных плотностей. К сожалению, на практике предположение о независимости признаков редко выполняется, является «наивным», что и дало название методу.

Основные преимущества наивного байесовского классификатора: легкость программной реализации и низкие вычислительные затраты при обучении и классификации. В тех редких случаях, когда признаки действительно независимы (или близки к этому), он почти оптимален. Главный его недостаток относительно низкое качество классификации в большинстве реальных задач. Поэтому чаще всего его используют либо как примитивный эталон для сравнения различных моделей, либо как блок для построения более сложных алгоритмов.

Рассмотрим базовые принципы работы простого байесовского классификатора.

Основные теоретические сведения

Пусть имеется объект или наблюдение X, класс которого неизвестен. Пусть также имеется гипотеза H, согласно которой X относится к некоторому классу C. Для за дачи классификации можно определить вероятность $P(H|X)$, то есть вероятность того, что гипотеза H для X справедлива. $P(H|X)$ называется условной вероятностью того, что гипотеза H верна при условии, что классифицируется объект X, или апостериорной вероятностью.

(| - множество элементов, удовлетворяющих условию, множество всех… таких, что верно…). Апостериорное распределение – условное распределение вероятностей какой-либо случайной величины при некотором условии, рассматриваемое в противоположность ее безусловному или априорному распределению)

Предположим, что объектами классификации являются фрукты, которые описываются их цветом и размером, Определим объект X как красный и круглый и выдвинем гипотезу H, что это яблоко. Тогда условная вероятность $P(H|X)$ отражает меру уверенности в том, что объект X является яблоком при условии, что он красный и круглый, Кроме условной (апостериорной) вероятности, рассмотрим так называемую априорную вероятность $P(H)$. В нашем примере это вероятность того, что любой наблюдаемый объект является яблоком, безотносительно к тому, как он выглядит. Таким образом, апостериорная вероятность основана на большей информации, чем априорная, не предполагающая зависимость от свойств объекта X.

Аналогично $P(H|X)$ апостериорная вероятность X при условии H, или вероятность того, что X является красным и круглым, если известно, что это яблоко, $P(X)$априорная вероятность X. В нашем примере это просто вероятность того, что объект является красным и круглым. Вероятности $P(X)$, $P(H)$ и $P(X|H)$ могут быть оценены на основе наблюдаемых данных.

Для вычисления апостериорной вероятности на основе $P(X)$, $P(H)$ и $P(X|H)$ используется формула Байеса:

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)} \quad (1)$$

Алгоритм работы простого байесовского классификатора содержит следующие шаги.

1. Пусть исходное множество данных S содержит атрибуты A_1, A_2, ..., A_n Тогда каждый объект или наблюдение $X \in S$ будет представлено своим набором значений этих атрибутов $x_1, x_2, \ldots, x_n$ где x_i значение, которое принимает атрибут A_i в данном наблюдении.

2. Предположим, что задано m классов $C = \{C_1, C_2, \ldots C_m\}$ и наблюдение X, для которого класс неизвестен. Классификатор должен определить, что X относится к классу, который имеет наибольшую апостериорную вероятность $P(H|X)$. Простой байесовский классификатор относит наблюдение X к классу $C_k(k = 1, \ldots, m)$ тогда и только тогда, когда выполняется условие $P(C_k|X) > P(C_j|X)$ для любых $1 \le j \le m$:.т.к, $k \ne j$.

По формуле Байеса:

$$P(C_k|X) = \frac{P(X|C_k)P(C_k)}{P(X)} \quad (2)$$

3. Поскольку вероятность $P(X)$ для всех классов одинакова, максимизировать требуется только числитель формулы (2), Если априорная вероятность класса $P(C_k)$ неизвестна, то можно предположить, что классы равновероятны, $P(C_1) = P(C_2) = \ldots = P(C_m)$, и, следовательно, мы должны выбрать максимальную вероятность $P(C_k|X)$.

Заметим, что априорные вероятности классов могут быть оценены как $P(C_k) = s_k/s$, где s_k – число наблюдений обучающей выборки, которые относятся к классу C_k, а s – общее число обучающих примеров.

4. Если исходное множество данных содержит большое количество атрибутов, то определение $P(X|C_k)$ может потребовать значительных вычислительных затрат. Чтобы их уменьшить, используется «наивное» предположение о независимости признаков. То есть для набора атрибутов $X = (x_1, x_2, \ldots, x_n)$ можно записать:

$$P(X|C_k) = P(x_1|C_k) \times P(x_2|C_k) \times \ldots \times P(x_n|C_k) \quad (3)$$

Вероятности, стоящие в правой части формулы (3), могут быть определены из обучающего набора данных для следующих случаев.

Атрибут A является категориальным, тогда $P(X|C_k) = {s_{ik}}/{s_k}$, где s_{ik} – общее число наблюдений класса C_i, в которых A_i принимает значение x_i, а s_k – общее число наблюдении, относящихся к классу C_k.

Атрибут A является непрерывным, тогда предполагается, что его значения подчиняются закону распределения Гаусса:

$$P(x_i|C_k) = \frac{1}{\sqrt{2\pi\sigma}} exp\left(\frac{(x-m)^2}{2\sigma^2}\right)$$

где m и σ^2 – математическое ожидание и дисперсия значений атрибута A_i для наблюдений, относящихся к классу C_k.

При классификации неизвестного наблюдения объект X будет относиться к классу, для которого $P(X|C_i) \times P(C_i)$ принимает наибольшее значение.

Пример работы простого байесовского классификатора

Рассмотрим пример работы наивного байесовского классификатора на основе задачи предсказания текучести абонентской базы, в качестве набора данных 14 записей с квантованными входными полями (табл. 1).

Таблица 1. Набор данных «Текучесть абонентской базы»

№	Количество звонков	SMS-активность	Internet-активность	Международные звонки	Уход

1	< 100	Высокая	Средняя	Нет	Да
2	< 100	Высокая	Высокая	Нет	Да
3	$100 - 150$	Высокая	Средняя	Нет	Нет
4	> 150	Средняя	Средняя	Нет	Да
5	> 150	Низкая	Средняя	Да	Нет
6	> 150	Низкая	Высокая	Да	Нет
7	$100 - 150$	Низкая	Высокая	Да	Да
8	< 100	Средняя	Средняя	Нет	Нет
9	< 100	Низкая	Средняя	Да	Да
10	> 150	Средняя	Средняя	Да	Да
11	< 100	Средняя	Высокая	Да	Да
12	$100 - 150$	Средняя	Высокая	Нет	Да
13	$100 - 150$	Высокая	Средняя	Да	Да
14	> 150	Средняя	Высокая	Нет	Нет

Обозначим как C_1 класс клиентов, для которых Уход = Да. К классу C_2 будем относить клиентов, которые сохраняют лояльность компании. Для них метка класса будет Нет. Пусть новый клиент, вероятность ухода которого мы хотим оценить, обладает следующими параметрами:

$$(\text{X} = (\text{Количество звонков} =< 100, SMS - \text{активность} = \text{Средняя}, Internet - \text{активность} = \text{Средняя}, \text{Международные звонки} = \text{Да}).$$

Согласно простому алгоритму Байеса нужно максимизировать $P(X|C_k) \times P(C_k)$ для k = 1,2 (поскольку классов всего два). Априорная вероятность появления каждого $P(C_k)$ может быть вычислена с помощью обучающего множества из таблицы как отношение числа примеров -го класса к общему числу примеров. Поскольку всего примеров 14, наблюдений с меткой класса Да - 9, а с меткой Нет - 5, то $P(C_2) = 5/14 = 0{,}3\,57.$

Для вычисления $P(X|C_k)$, $k = 1{,}2$ определим следующие условные вероятности (табл.2.).

Таблица 2. - Расчет условных вероятностей

$\text{P}(\text{Число звонков} =< 100|\text{Уход} = \text{Да}) = 4/9 = 0{,}444$
$\text{P}(\text{Число звонков} =< 100|\text{Уход} = \text{Нет}) = 1/5 = 0{,}2$
$\text{P}(SMS - \text{активность} = \text{Средняя}|\text{Уход} = \text{Да}) = 4/9 = 0{,}444$
$\text{P}(SMS - \text{активность} = \text{Средняя}|\text{Уход} = \text{Нет}) = 2/5 = 0{,}4$
$\text{P}(Internet - \text{активность} = \text{Средняя}|\text{Уход} = \text{Да}) = 5/9 == 0{,}555$
$\text{P}(Internet - \text{активность} = \text{Средняя}|\text{Уход} = \text{Нет}) = 3/5 == 0{,}6$

$\text{P(Международные звонки} = \text{Да}|\text{Уход} = \text{Да}) = 5/9 = 0{,}555$
$\text{P(Международные звонки} = \text{Да}|\text{Уход} = \text{Нет}) = 2/5 = 0{,}4$

Используя рассчитанные условные вероятности, получим (табл. 3).

Таблица 3. Расчет условных вероятностей

$\text{P(X}|\text{Уход} = \text{Да}) = 0{,}444 \times 0{,}444 \times 0{,}555 \times 0{,}555 = 0{,}061$
$\text{P(X}|\text{Уход} = \text{Нет}) = 0{,}6 \times 0{,}4 \times 0{,}2 \times 0{,}4 = 0{,}019$
$\text{P(X}|\text{Уход} = \text{Да}) \times \text{P(Уход} = \text{Да}) = 0{,}061 \times 0{,}643 = 0{,}039$
$\text{P(X}|\text{Уход} = \text{Нет}) \times \text{P(Уход} = \text{Нет}) = 0{,}019 \times 0{,}357 = 0{,}007$

Таким образом, для наблюдения X произведение вероятностей для класса C_1, то есть $P(X|C_1) \times P(C_1) = 0{,}039$, а для класса C_2 $P(X|C_1) \times P(C_1) = 0{,}007$, Поэтому выбирается класс C_1, для которого оно больше. Можно нормализовать эти вероятности, тогда получим: $P/(X|C_1) \times P(C_1) = 0{,}039/(0{,}039 + 0{,}007) = 0{,}848$ и $P/(X|C_2) \times P(C_2) = 0{,}007/(0{,}039 + 0{,}007) = 0{,}152$

Следовательно, для наблюдения X метка класса будет Да, а клиент, характеризуемый соответствующими признаками, должен рассматриваться как склонный к уходу.

Байесовские сети

Проблема заключается в том, что распределения, которые нас интересуют, обычно слишком сложные, чтобы их можно было максимизировать напрямую, аналитически. В них слишком много переменных, между переменными слишком сложные связи. Но, с другой стороны, часто в них есть дополнительная структура, которую можно использовать, структура в виде независимостей $(p(x, y) = p(x)p(y))$ и условных независимостей $(p(x, y|\,\text{z}) = p(x|\text{z})p(y|z))$ некоторых переменных

В результате сложное апостериорное распределение $(p(a_1, \dots, a_n|\, x = v)p(x = v)$ удалось переписать как $p(x = v)p(a_1|x = v)p(a_2|x = v) \dots p(a_n|x = v)$ и в этой форме с ним гораздо проще справиться (обучить параметры каждого маленького распределения по отдельности, а затем выбрать v, дающее максимум произведения. Это один из простейших примеров разложения (факторизации) сложного распределения в произведение простых.

Итак, давайте рассмотрим один из самых удобных способов представлять большие и сложные распределения вероятностей – байесовские сети доверия, которые в последнее время чаще называются просто направленными графическими моделями.

Байесовская сеть – это направленный граф без направленных циклов (*это очень важное условие!*), в котором вершины соответствуют переменным в

распределении, а рёбра соединяют «связанные» переменные. В каждом узле задано условное распределение узла при условии своих родителей $p(x|parents(x))$ (родители), и граф байесовской сети означает, что большое совместное распределение раскладывается в произведение этих условных распределений. Вот, например, граф, соответствующий наивному Байесу (рис.1):

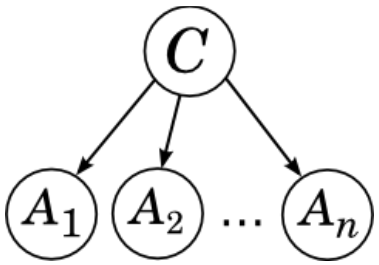


Рис.1. Граф, соответствующий наивному Байесу

Он соответствует разложению $p(A_1, \dots, A_n, C) = p(C)p(A_1|C)p(A_2|C) \dots p(A_n|C))$: у C нет предков, так что мы берём его безусловное распределение, а каждый из A_i «растёт» непосредственно из C и больше ни с кем не связан.

Мы уже знаем, что в этом случае все атрибуты A_i условно независимы при условии категории C:

$$p(A_i, A_j|C) = p(A_i|C)p(A_j|C)$$

Давайте теперь рассмотрим все простейшие варианты байесовской сети и посмотрим, каким условиям независимости между переменными они соответствуют.

Байесовские сети с двумя и тремя переменными: тривиальные (упрощенные) случаи

Начнём с сети из двух переменных, x и y. Здесь всего два варианта: либо между x и y нет ребра, либо есть. Если ребра нет, это просто значит, что x и y независимы (рис.2), ведь такой граф соответствует разложению $p(x, y) = p(x)p(y)$

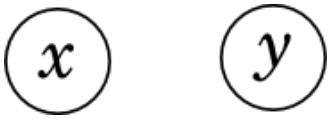


Рис.2. Переменные x и y независимы

А если ребро есть (пусть оно идёт из x в y, это не важно), мы получаем разложение $p(x, y) = p(x)p(y|x)$, которое буквально по определению условной вероятности тупо верно всегда, для любого распределения $p(x, y)$. Таким образом, граф из двух вершин с ребром не даёт нам новой информации (рис.3).

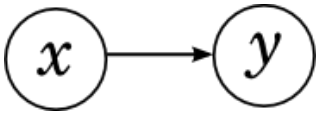


Рис.2. Переменные x и y зависимы

Теперь переходим к сетям из трёх переменных x, y и z. Самый простой случай – когда рёбер совсем нет.

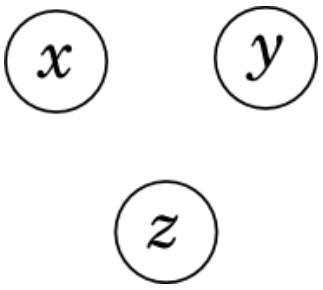


Рис.3. Переменные x, y и z независимы

Как и с двумя переменными, это значит, что x, y и z просто независимы: $p(x, y, z) = p(x)p(y)p(z)$.Другой простой случай – когда между переменными проведены все рёбра (рис.4).

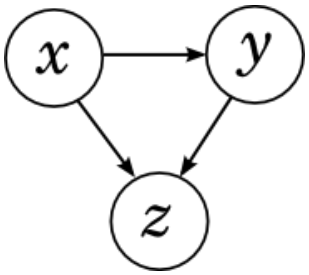


Рис.4. Переменные x, y и z независимы

Этот случай тоже аналогичен рассмотренному выше; пусть, например, рёбра идут из x в y и z, а также из y в z. Получаем разложение $p(x, y, z) = p(x)p(y|x)p(z|\mathrm{x}, \mathrm{y})$ которое, опять же, верно всегда, для любого совместного распределения $p(x, y, z)$. В этой формуле можно было бы выбирать переменные в любом порядке, ничего не изменилось бы. Обратите внимание, что направленные циклы в байесовских сетях запрещены, и в результате вариантов, как можно провести все рёбра, всего шесть, а не восемь.

Рассмотрим три более интересных случая – это и будут те «кирпичики», из которых можно составить любую байесовскую сеть. К счастью, для этого достаточно рассмотреть графы на трёх переменных – всё остальное будет обобщаться из них.

В примерах ниже будут интуитивно интерпретировать ребро, стрелочку между двумя переменными, как «x влияет на y», т.е. по сути, как причинно-следственную связь. На самом деле это, конечно, не совсем так.

Последовательная связь

Начнём с последовательной связи между переменными: x «влияет на» y, а y, в свою очередь, «влияет на» z (рис.5).

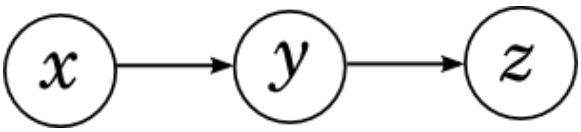


Рис.5. Последовательная связь переменных x, y и z

Такой граф изображает разложение $p(x, y, z) = p(x)p(y|x)p(z|y)$

Интуитивно это соответствует последовательной причинно-следственной связи: если вы будете бегать зимой без шапки, вы простудитесь, а если простудитесь, у вас поднимется температура. Очевидно, что x и y, а также y и z друг с другом связаны, между ними даже непосредственно стрелочки проведены. Связаны ли между собой в такой сети x и z, зависимы ли эти переменные? Конечно! Если вы бегаете зимой без шапки, вероятность получить высокую температуру повышается. Однако в такой сети x и z связаны только через y, и если мы уже знаем значение y, x и z становятся независимыми: если вы уже знаете, что простудились, совершенно не важно, чем это было вызвано, температура теперь повысится (или не повысится) именно от простуды.

Формально это соответствует условной независимости x и z при условии y; давайте это проверим:

$$p(x, z|y) = \frac{p(x, y, z)}{p(y)} = \frac{p(x)p(y|x)p(z|y)}{p(y)} = p(x|y)p(z|y)$$

где первое равенство – это определение условной вероятности, второе – наше разложение, а третье – применение теоремы Байеса.

Итак, последовательная связь между тремя переменными говорит нам о том, что крайние переменные условно независимы при условии средней. Всё очень логично и достаточно прямолинейно.

Расходящаяся связь

Следующий возможный вариант – расходящаяся связь: x «влияет» и на y, и на z. (рис.6)

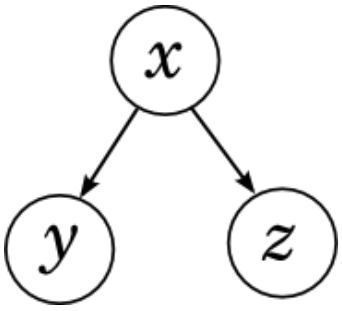


Рис.6. Расходящаяся связь переменных x, y и z

Такой граф изображает разложение

$$p(x, y, z) = p(x)p(y|x)p(z|x)$$

Интуитивно это соответствует двум следствиям из одной и той же причины: если вы простудитесь, у вас может подняться температура, а также может начаться насморк. Как и в предыдущем случае, очевидно, что x и y, а также x и z зависимы, и вопрос заключается в зависимости между y и z. Опять же, очевидно, что эти переменные зависимы: если у вас насморк, это повышает вероятность того, что вы простудились, а значит, вероятность высокой температуры тоже повышается.

Однако в такой сети, подобно предыдущему случаю, y и z связаны только через x, и если мы уже знаем значение общей причины x, y и z становятся независимыми: если вы уже знаете, что простудились, насморк и температура становятся независимы.

Формально это соответствует условной независимости y и z при условии x; проверить это ещё проще, чем для последовательной связи:

$$p(y,z|x) = \frac{p(x,y,z)}{p(y)} = \frac{p(x)p(y|x)p(z|\mathrm{x})}{p(y)} = p(y|x)p(z|x)$$

Итак, расходящаяся связь между тремя переменными говорит нам о том, что «следствия» условно независимы при условии своей «общей причины». Если причина известна, то следствия становятся независимы; пока причина неизвестна, следствия через неё связаны.

Сходящаяся связь

У нас остался только один возможный вариант связи между тремя переменными: сходящаяся связь, когда x и y вместе «влияют на» z.

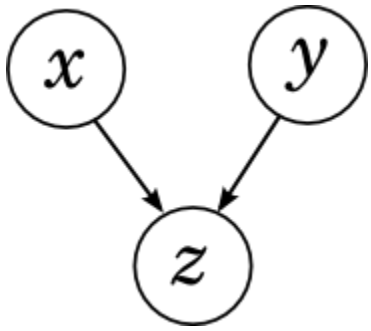


Рис.6. Сходящаяся связь переменных x,y и z

Разложение здесь получается такое:

$$p(x,y,z) = p(x)p(y)p(z|x,y)$$

Это ситуация, в которой у одного и того же следствия могут быть две разные причины: например, температура может быть следствием простуды, а может – отравления. Зависимы простуда и отравление? Нет! В этой ситуации, пока общее следствие неизвестно, две причины никак не связаны друг с другом, и это очень легко проверить формально:

$$p(x,y) = \sum_z p(x,y,z) = \sum_z p(x), p(y), p(z|x,y) = p(x)p(y)$$

Однако если «общее следствие» z становится известным, ситуация меняется. Теперь общие причины известного следствия начинают влиять друг на друга. Предположим, что вы знаете, что у вас температура. Это сильно повышает вероятность, как простуды, так и отравления. Однако если вы теперь узнаете, что отравились, вероятность простуды уменьшится – симптом «уже объяснён» одной из возможных причин, и вторая становится менее вероятной.

Таким образом, при сходящейся связи две «причины» независимы, но только до тех пор, пока значение их «общего следствия» неизвестно; если же общее следствие получает означивание, причины становятся зависимыми.

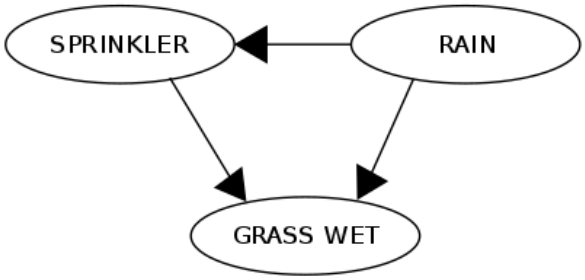
Но есть один нюанс: когда на пути встречается сходящаяся связь, недостаточно посмотреть только на её «общее следствие», чтобы определить независимость. На самом деле, если даже у z означивания нету, но оно есть у одного из её потомков (возможно, достаточно далёких), две причины всё равно станут зависимы. Интуитивно это тоже легко понять: например, пусть мы не наблюдаем собственно температуру, а наблюдаем её потомка – показания градусника. На свете, наверное, бывают неисправные градусники, так что это тоже некая вероятностная связь. Однако наблюдение показаний градусника точно так же делает простуду и отравление зависимыми.

Последовательная, сходящаяся и расходящаяся связи – это те три кирпичика, из которых состоит любой ациклический направленный граф. И наших рассуждений вполне достаточно для того, чтобы обобщить результаты об условной зависимости и независимости на все такие графы. Конечно, здесь не время и не место для того, чтобы формально доказывать общие теоремы, но результат достаточно предсказуем – предположим, что вам нужно в большом графе проверить, независимы ли две вершины (или даже два множества вершин). Для этого вы смотрите на все пути в графе (без учёта стрелочек), соединяющие эти два множества вершин. Каждый из этих путей можно «разорвать» одной из вышеописанных конструкций: например, последовательная связь разорвётся, если в середине у неё есть означивание (значение переменной известно), а сходящаяся связь разорвётся, если, наоборот, означивания нет (причём нет ни в самой вершине из пути, ни в её потомках). Если в результате все пути окажутся разорваны, значит, множества переменных действительно независимы; если нет – нет.

Пример

Предположим, что может быть две причины, по которым трава может стать мокрой (GRASS WET): сработала дождевальная установка (SPRINKLER), либо прошёл дождь (RAIN). Также предположим, что дождь влияет на работу дождевальной машины (во время дождя установка не включается). Тогда ситуация может быть смоделирована проиллюстрированной Байесовской сетью. Каждая из трёх переменных может принимать лишь одно из двух возможных значений: T (правда — true) и F (ложь — false), с вероятностями, указанными в таблицах на иллюстрации.

RAIN	SPRINKLER T	SPRINKLER F
F	0.4	0.6
T	0.01	0.99



RAIN T	RAIN F
0.2	0.8

SPRINKLER	RAIN	GRASS WET T	GRASS WET F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

Совместная вероятность функции:

$$\mathrm{P}(G, S, R) = \mathrm{P}(G|S, R)\mathrm{P}(S|R)\mathrm{P}(R)$$

где имена трёх переменных означают G = Трава мокрая (Grass wet), S = Дождевальная установка (Sprinkler), и R = Дождь (Rain).

Модель может ответить на такие вопросы как «Какова вероятность того, что прошел дождь, если трава мокрая?» используя формулу условной вероятности и суммируя переменные:

$$\mathrm{P}(R = T \mid G = T) = \frac{\mathrm{P}(G = T, R = T)}{\mathrm{P}(G = T)} = \frac{\sum_{S \in \{T,F\}} \mathrm{P}(G = T, S, R = T)}{\sum_{S,R \in \{T,F\}} \mathrm{P}(G = T, S, R)}$$

$$= \frac{(0.99 \times 0.01 \times 0.2 = 0.00198_{TTT}) + (0.8 \times 0.99 \times 0.2 = 0.1584_{TFT})}{0.00198_{TTT} + 0.288_{TTF} + 0.1584_{TFT} + 0_{TFF}} \approx 35.77\%.$$

Трансформация данных

Введение в трансформацию данных

Помимо предобработки, целью которой является приведение данных в соответствие с определенными критериями качества, процесс подготовки данных к анализу обычно включает в себя еще один этап трансформацию.

Каждая выборка исходных данных, загружаемая в аналитическое приложение, характеризуется набором свойств, которые могут повлиять на эффективность работы модели и снизить достоверность результатов анализа. Даже если данные очищены от таких факторов, ухудшающих их качество, как дубликаты, противоречия, шумы, аномальные значения, про пуски и др., они все еще могут не соответствовать методике и целям анализа. Это связано не с содержанием данных, а с их представлением и внутренней организацией. Может возникнуть парадокс: данные совершенно корректны с точки зрения качества, а их информационного содержания вполне достаточно для решения аналитической задачи, но представление и организация данных затрудняют анализ или вообще делают его невозможным. Данные могут быть разобщены, не упорядочены, представлены в форматах, с которыми не работает тот или иной алгоритм. Трансформация данных, то есть их преобразование к определенному представлению, формату или виду, оптимальному с точки зрения решаемой задачи, и призвана решить эту проблему.

В то время как очистка данных занимается только их содержанием, трансформация – это процесс оптимизации их представления и организации с точки зрения определённого метода анализа. Трансформация данных зависит от задач, алгоритмов и целей анализа. То есть для одной задачи могут потребоваться одни виды трансформации, а для другой другие.

Несоответствие представления и организации данных методике и целям анализа может проявляться как на уровне моделей, так и на уровне отдельных задач анализа. Несоответствие на уровне моделей означает, что даже если данные в их нынешнем преставлении пригодны для использования в нейронной сети, то вполне возможно, что их использование с другими моделями Data Mining, например, с деревьями решений, окажется затруднено либо вообще невозможно. Что касается уровня задач, то, скажем, для прогнозирования достаточно иметь одномерный ряд наблюдений за прогнозируемым параметром исследуемого процесса или объекта, тогда как для решения задачи классификации требуется выборка, содержащая несколько признаков классифицируемых объектов. Если для решения задачи регрессии выходные данные должны быть числовыми, то при классификации – категориальными, В то же время все три задачи Data Mining:

классификация, регрессия и прогнозирование могут быть решены в рамках одной модели (например, нейросетевой).

Трансформация данных очень широкое понятие, не имеющее четко очерченных границ. В различных направлениях обработки данных термин «трансформация» иногда распространяют на любые манипуляции с данными независимо от их целей и методов. Однако в контексте бизнес аналитики трансформация данных имеет вполне конкретные цели и задачи, а также использует достаточно стабильный набор методов.

Определение

Трансформация данных комплекс методов и алгоритмов, направленных на оптимизацию представления и форматов данных с точки зрения решаемых задач и целей анализа. Трансформация данных не ставит целью изменить информационное содержание данных. Ее задача представить эту информацию в таком виде, чтобы она могла быть использована наиболее эффективно.

Трансформация данных не обязательно связана только с аналитическим приложением в том или ином виде она выполняется на всех этапах аналитического процесса: в регистрирующих системах (OLTP-системах), в процессе переноса и загрузки данных в ХД (ETL-процесс) и непосредственно при подготовке данных к анализу в аналитическом приложении. Такая распределенность процесса трансформации обусловлена тем, что на каждом этапе он преследует различные цели. В системах первичной регистрации данных (OLTP-системах) трансформация может про изводиться для обеспечения технической и логической совместимости данных, их подготовки к извлечению, переносу в хранилище данных и т.д. Например, адреса часто вводят одной строкой. В то же время для анализа могут представлять интерес отдельные компоненты адреса, которые имеют как текстовый формат (улица, город), так и числовой (номер дома, офиса). С помощью трансформации можно распределить соответствующие элементы по отдельным полям и преобразовать их в нужный формат. Приведем еще один пример. Различные системы регистрации (сканер штрихкодов, контрольно-кассовый аппарат и т. п,) могут выдавать однотипные данные в разных форматах. Так, в одном случае в качестве разделителя целой и дробной частей чисел может использоваться точка, а в другом запятая.

Попытка вставить такие данные в одно поле базы данных может привести к серьезным техническим проблемам, а трансформация, то есть преобразование к единому формату, поможет решить этот вопрос.

Основными целями трансформации данных на этапе процесса ETL являются приведение их в соответствие с моделью данных, используемой в

хранилище, осуществление корректной консолидации и собственно загрузка в хранилище.

Возникает вопрос: если трансформации данных так миого внимания уделяется на этапе сбора и организации хранения данных, то зачем включать средства трансформации в аналитическое приложение (что, несомненно, усложняет и делает более дорогим его разработку)? Ответ прост. Не все данные поступают в аналитическое приложение из систем, где они прошли предварительную подготовку, таких как OLTP-системы и хранилища данных. Но главная причина заключается в том, что трансформация данных в этих системах в большей степени носит технический характер и слабо связана с возможными методами, алгоритмами и целями анализа.

Основные отличия применения трансформации данных на разных этапах аналитического процесса представлены в табл. 1.

Таблица 1. Трансформация данных на разных этапах аналитического процесса

OLTP-системы	ETL-процесс	Аналитическое приложение
Обеспечение поддержки корректности форматов н типов данных. Оптимизация процессов доступа и выгрузки данных.	Обеспечение поддержки корректности форматов н типов данных. Преобразование данных с целью приведения в соответствие с моделью данных, которая используется в хранилище. Обеспечение процесса консолидации данных и их загрузки в хранилище	Непосредственная подготовка данных к анализу, объединение и выделение наиболее ценной информации, обеспечение корректной работы аналитических алгоритмов, методов и моделей

Хотя и OLTP-системы, и хранилища данных во многих случаях разрабатываются с учетом перспективных направлений анализа, предусмотреть все возможные ситуации очень трудно. В условиях динамики современного бизнеса нельзя предсказать, какой набор аналитических решений понадобится в ближайшем будущем. Именно процесс подготовки данных в аналитическом приложении непосредственно перед выполнением анализа, когда его цели и методы ясны, позволяет аналитику наилучшим образом подготовить данные,

выделить и подчеркнуть в них необходимую информацию для наиболее эффективного ее использования. Эту задачу и решает трансформация данных.

Необходимость использования методов трансформации данных в аналитическом приложении обусловлена еще и их доступностью аналитику. Действительно, в процессе подготовки данных пользователь аналитического приложения имеет возможность по своему усмотрению применять те или иные методы трансформации данных, настраивать их параметры, проводить эксперименты, позволяющие определить влияние трансформации на результаты анализа. Что касается хранилищ данных и ETL-процессов, то, как правило, выполняемые в них преобразования данных заданы жестко, их адаптация в зависимости от характера данных и решаемых задач анализа не предусмотрена и аналитик не может вмешиваться в эти системы оперативной регистрации данных (OLTP) обычно вообще недоступны аналитику.

Основные методы трансформации данных

I1режде чем приступить к подробному рассмотрению методов, алгоритмов и целей отдельных направлений трансформации данных, проведем краткий обзор типичных средств трансформации, которыми оснащается большинство аналитических платформ.

Преобразование упорядоченных данных. Позволяет оптимизировать представление таких данных с целью обеспечения дальнейшего анализа, например решения задачи прогнозирования временного ряда или группировки по временному периоду.

Квантование. Позволяет разбить диапазон возможных значений числового признака на заданное количество интервалов и присвоить номера интервалов или иные метки попавшим в них значениям.

Сортировка. Позволяет изменить порядок следования записей исходной выборки данных в соответствии с алгоритмом, определенным пользователем. В некоторых случаях сортировка дает возможность упростить визуальный анализ выборки, оперативно определить наибольшие и наименьшие значения признаков и т. д.

Слияние. Позволяет объединить две таблицы по одноименным полям или дополнить одну таблицу записями из другой, которые отсутствуют в дополняемой. Слияние применяется в тех случаях, когда информацию в анализируемой выборке данных необходимо дополнить информацией из другой выборки. При объединении к записям исходной выборки добавляются все записи другой. В случае дополнения к исходной выборке добавляются только те данные, которые отсутствовали в исходной. Операция слияния является одним из

способов обогащения данных: если выборка содержит недостаточно данных для анализа, то ее можно дополнить недостающей информацией из другой выборки.

Группировка и разгруппировка. Очень часто информация, интересующая аналитика, в таблице оказывается «разбавлена» посторонними данными, разобщена, разбросана по отдельным полям и записям. Используя группировку, можно обобщить нужную информацию, объединить ее в минимально необходимое количество полей и значений. Обычно предусматривают возможность выполнения и обратной операции разгруппировки.

Настройка набора данных. Позволяет изменять имена, типы, метки и назначения полей исходной выборки данных. Например, если поле, содержащее числовую информацию, в источнике данных по какой либо причине имеет строковый тип, значения этого поля не могут обрабатываться как числа. Чтобы работа с числовыми данными этого поля стала возможной, их следует преобразовать к числовому типу.

Табличная подстановка значений. Позволяет про изводить замену значений в исходной выборке данных на основе так называемой таблицы подстановки. Таблица подстановки содержит пары «исходное значение новое значение». Каждое значение выборки данных проверяется на соответствие исходному значению таблицы подстановки, и если такое соответствие найдено, то значение выбор изменяется на соответствующее новое значение из таблицы подстановки. Это очень удобный способ для автоматической корректировки значений.

Вычисляемые значения. Иногда для анализа требуется информация, которая отсутствует в явном виде в исходных данных, но может быть получена на основе вычислений над имеющимися значениями. Например, если известны цена и количество товара, то сумма может быть рассчитана как их про изведение. Для этих целей в аналитическое приложение включается своего рода калькулятор, который позволяет выполнять над данными исходной выборки различные вычисления. поскольку анализируемые данные могут быть различных типов (строковый, числовой, дата/время, логический), то механизм расчетов должен поддерживать работу Не только с числовыми данными, но и с данными других типов, например выделять подстроку, выполнять логические операции и т. д.

Нормализация. Нормализация позволяет преобразовать диапазон изменения значений числового признака в другой диапазон, более удобный для применения к данным тех или иных аналитических алгоритмов, а также согласовать диапазоны изменений различных признаков. Часто используется приведение к единице, когда весь имеющийся диапазон данных «сжимается» в интервал [0; 1] или [1; 1]. Особенно важно произвести правильную нормализацию данных в

алгоритмах Data Mining, которые основаны на измерении расстояния между векторами объектов в многомерном пространстве признаков (например, в кластеризации).

Трансформация упорядоченных данных

Многие аналитические задачи, например прогнозирование, анализ продаж, динамики спроса, состояния бизнес-объектов и других протяженных во времени процессов, связаны с обработкой данных, зависящих от времени. Такие данные называют упорядоченными, или временными рядами. В процессе обработки временных рядов требуется специальная подготовка данных, чтобы оптимизировать их представление для решения определенных аналитических задач: прогнозирования, классификации состояний объектов, выявления закономерностей, объясняющих динамику бизнес процессов, и т. д. – для всех возможных интервалов даты и времени.

Временной ряд состоит из последовательности наблюдений за состоянием параметров (признаков) исследуемых объектов или процессов. Если наблюдения содержат один признак, то ряд является одномерным, а если два или более многомерным. Поскольку значения временного ряда определены только в фиксированные моменты времени, так называемые отсчеты, последовательность его значений может быть представлена в следующем виде:

$$X = \{x(1), x(2) \dots x(n)\},$$

где $x(n)$ – последнее значение рассматриваемой временной последовательности.

При этом отсчеты времени полагаются равноотстоящими друг от друга.

Дадим замечание.

Временной ряд всегда представляет собой упорядоченные во времени данные. Однако обратное не всегда верно: не все упорядоченные данные могут образовывать временной ряд. Скажем, при снятии показаний из скважины измерения характеристик производятся с некоторым шагом глубины. Это тоже упорядоченные данные, но их нельзя назвать временным рядом, так как фиксируется не время, а глубина. Тем не менее многие рассуждения о временных рядах распространяются на любые упорядоченные данные.

Целью трансформации временных рядов является не изменение их содержания, а представление информации таким образом, чтобы обеспечивалась максимальная эффективность решения определенной задачи анализа. Можно выделить два основных типа преобразования, которые наиболее часто используются при подготовке временных рядов к анализу.

- *Скользящее окно* применяется при решении задач прогнозирования и квалификации состояний бизнес объектов, чтобы преобразовывать последовательность значений ряда в таблицу, которую можно использовать для построения моделей или какой-либо другой обработки.

Преобразование даты и времени заключается в приведении даты и времени к виду, наиболее удобному для визуального анализа и обработки временного ряда. При этом результаты преобразования даты уже не являются значениями типа Дата/Время и могут обрабатываться как обычные числа и строки.

Скользящее окно

Широко при меняется при обработке временных рядов, например, чтобы построить модель прогноза временного ряда. Целью прогнозирования значений временного ряда является предсказание значения $x(n+1)$ на основе предыдущих значений признака. Решение задачи прогнозирования возможно только в том случае, если значения временного ряда связаны между собой.

Пример

При регистрации заявок, поступивших от клиентов, среди прочего фиксируются дата подачи заявки и адрес клиента. Из базы данных системы регистрации заявок можно извлечь временной ряд с последовательностью номеров квартир, указанных в адресах. Очевидно, что пытаться предсказать номер квартиры следующего клиента на основе знания номеров квартир клиентов, чьи заявки были зарегистрированы ранее, бессмысленно.

Для построения прогностической модели необходимо знать три параметра.

Интервал прогноза – временной интервал, на котором будет осуществляться прогнозирование: день, неделя, месяц, квартал или год.

Горизонт прогноза – на какое количество интервалов (дней, недель и т. д.) мы хотим получить прогноз.

Глубина погружения – количество значений интервалов прогноза в прошлом, которое мы будем использовать для предсказания значений интервалов в будущем.

При выборе интервала прогноза, возможно, понадобится агрегирование данных внутри интервала. Например, если в базе данных информация о продажах представлена по дням, то для построения прогноза по неделям придется объединить информацию за отдельные дни, вычислив сумму, среднее значение или используя другую функцию агрегации.

Если в качестве интервала прогноза выбрана неделя, то глубина погружения – количество недель в прошлом, значения продаж за которые мы будем использовать в качестве исходных данных для предсказания.

Горизонт прогноза в этом случае количество недель, значения продаж За которые мы хотим предсказать. При выборе глубины погружения и горизонта прогноза следует руководствоваться следующими соображениями.

- Модель прогноза функционирует по принципу обобщения, то есть вывод о возможном значении в будущем делается на основе анализа большого числа значений из прошлого. Следовательно, глубина погружения должна в несколько раз превышать горизонт прогноза.

Чем большее число значений из прошлого используется для прогнозирования, тем выше степень обобщения и тем достовернее результаты предсказания. Однако при этом есть два ограничения. Во-первых, глубина погружения должна быть такой, чтобы значения из прошлого, используемые для прогнозирования, оставались актуальными, то есть правильно отражали текущее поведение исследуемого процесса. Использование для построения модели прогноза слишком старых данных, утративших актуальность, приведет к снижению достоверности или к получению некорректных результатов. Во-вторых, слишком большая глубина погружения увеличит размерность выборки данных, полученной в результате обработки ряда скользящим окном, что повлечет за собой усложнение модели. Кроме того, возрастут временные и вычислительные затраты на анализ данных.

Чем дальше простирается горизонт прогнозе, тем ниже достоверность результатов.

Обобщенная модель прогноза имеет следующий вид (рис. 1).

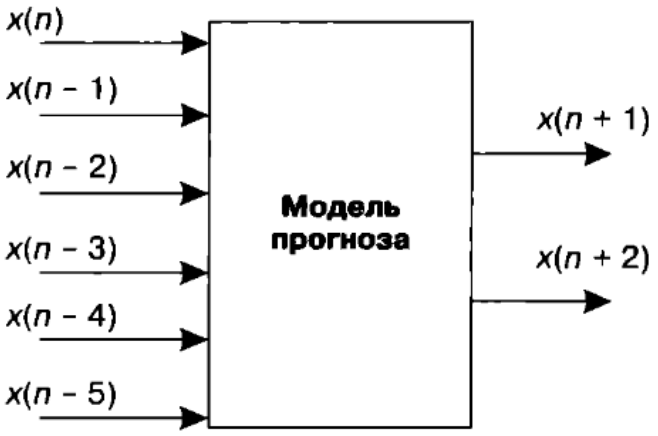


Рис. 1. Обобщенная модель прогноза

В данной модели глубина погружения составляет $d = 5$, а горизонт прогноза $h = 2$. Сама модель может представлять собой скользящее среднее, нейронную сеть, регрессию и т. д. Значение $x(n)$ текущее состояние исследуемого параметра, которое также включается в набор исходных данных для прогноза.

Выборка данных для такой модели должна содержать входных полей, соответствующих входным переменным модели, и 2 выходных поля, соответствующих выходным переменным. Преобразование ряда последовательно идущих значений в набор записей для модели прогноза выполняется с помощью скользящего окна.

Пусть имеется ряд данных, содержащий 10 наблюдений:

$X = \{x(0), x(1), x(2), x(3), x(4), x(5), x(6), x(7), x(8), x(9), x(10)\}$.

Если глубину погружения задать равной 5, то с помощью скользящего окна можно преобразовать исходный ряд данных в табличную форму, состоящую из 6 записей, которая может быть использована для работы с моделью, представленной на рис. 1. Преобразованный ряд будет выглядеть следующим образом (табл. 2).

Таблица 2. Преобразованный ряд данных

№	$x(n-5)$	$x(n-4)$	$x(n-3)$	$x(n-2)$	$x(n-1)$	$x(n)$	$x(n+1)$	$x(n+2)$
1	$x(0)$	$x(1)$	$x(2)$	$x(3)$	$x(4)$	$x(5)$	$x(6)$	$x(7)$
2	$x(1)$	$x(2)$	$x(3)$	$x(4)$	$x(5)$	$x(6)$	$x(7)$	$x(8)$
3	$x(2)$	$x(3)$	$x(4)$	$x(5)$	$x(6)$	$x(7)$	$x(8)$	$x(9)$
4	$x(3)$	$x(4)$	$x(5)$	$x(6)$	$x(7)$	$x(8)$	$x(9)$	$x(10)$
5	$x(4)$	$x(5)$	$x(6)$	$x(7)$	$x(8)$	$x(9)$	$x(10)$	$x(11)$
6	$x(5)$	$x(6)$	$x(7)$	$x(8)$	$x(9)$	$x(10)$	$x(11)$	$x(12)$

Рассмотрим еще пример. Пусть имеется ряд наблюдений, отражающий количество клиентов, обслуженных за месяц (табл. 3). Задача состоит в том, чтобы построить модель прогноза числа клиентов на будущий месяц. При этом глубина погружения задается равной 2, горизонт прогнозирования – 1, то есть на основе двух предыдущих месяцев прогнозируется следующий.

Таблица 3. Ряд наблюдений

Дата начала месяца	Число клиентов
01.01.2006	1540
01.02.2006	960
01.03.2006	1150
01.04.2006	1230
01.05.2006	1056
01.06.2006	995

Выборка, полученная в результате обработки данных скользящим окном, будет содержать в начале и в конце неполные записи, количество которых равно глубине погружения (табл. 4). Неполные записи можно исключить из

рассмотрения. Если это сделать, то результирующая выборка будет иметь следующий вид (табл. 5).

Таблица 4. Скользящее окно с неполными записями для ряда из табл. 3.

Дата начала месяца	$x(n-2)$	$x(n-1)$	$x(n)$	$x(n+1)$
				1540
01.01.2006			1540	960
01.02.2006		1540	960	1150
01.03.2006	1540	960	1150	1230
01.04.2006	960	1150	1230	1056
01.05.2006	1150	1230	1056	995
01.06.2006	1230	1056	995	
	1056	995		
	995			

Таблица 5. Скользящее окно с полными записями для ряда из табл. 3.3

Дата начала месяца	$x(n-2)$	$x(n-1)$	$x(n)$	$x(n+1)$
01.03.2006	1540	960	1150	1230
01.04.2006	960	1150	1230	1056
01.05.2006	1150	1230	1056	995

На рис. 2 поясняется принцип формирования последней таблицы. Первое положение окна соответствует строке таблицы с датой 01.03.2006, когда в него попадают три значения 1540, 960 и 1150. Значение, соответствующее горизонту прогноза для данного положения окна, выделено темно-серым цветом и равно 1230. Затем окно смещается, формируя таким же образом вторую строку для даты 01.04.2006, а затем и третью.

Положения окна на рис. 2 соответствуют набору полных записей. Нетрудно заметить, что если начальное положение окна будет левее, то одно или два места в окне окажутся пустыми, что и порождает неполные записи в начале табл. 4. Смещение окна до крайнего правого положения, когда в него попадет только последнее значение ряда, приведет к появлению неполных строк в конце таблицы.

Если временной ряд образован не одним, а двумя параметрами и более, то он является многомерным. Например, для двумерного ряда можно записать:

$$X = \{a(i), b(i)\},$$

где i – номер временного отсчета, в который было зафиксировано наблюдение, то есть $i = 1,2 \ldots n$.

Процедура преобразования многомерного ряда в табличную форму выполняется по тому же алгоритму, что и преобразование одномерного (табл. 6, 7).

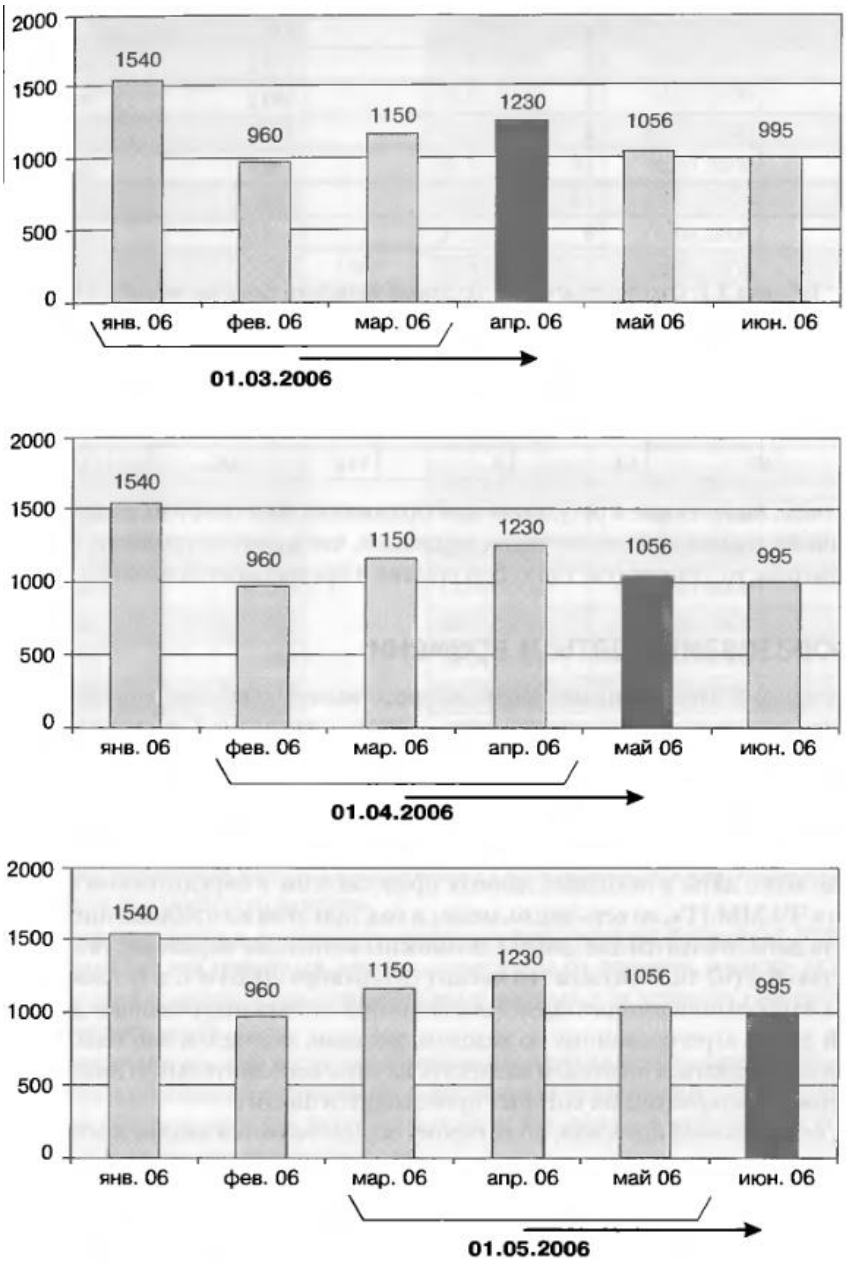


Рис. 2. Принцип работы скользящего окна

Таблица 6. Ряд наблюдений

i	a	b
1	8	111
2	5	112
3	4	106
4	10	114
5	12	96
6	9	116

Таблица 7. Скользящее окно с полными записями для ряда из табл. 6

№ записи	$a(n-2)$	$a(n-1)$	$a(n)$	$b(n-2)$	$b(n-1)$	$b(n)$
	8	5	4	111	112	106
	5	4	10	112	106	114
	4	10	12	106	114	96
	10	12	9	114	96	116

Таблицы, полученные в результате преобразования многомерных рядов данных, как правило, содержат большое число признаков, что может потребовать сокращения размерности данных.