Лекция 15. Модели прогнозирования

На современном этапе главным инструментом прогнозирования являются прогностические модели. От того, насколько модель прогноза адекватна условиям, в которых работает компания, насколько полно в ней учитываются внешние и внутренние факторы, воздействующие на те или иные бизнес процессы, зависит точность и достоверность прогноза.

Обобщенная модель прогноза

Структура прогностической модели похожа на структуры моделей, используемых для решения других задач анализа, например распознавания, идентификации и т. д. Модель прогноза отличается только характером используемых данных и алгоритмами их обработки. Обобщенная структура прогностической модели представлена на рис. 1.

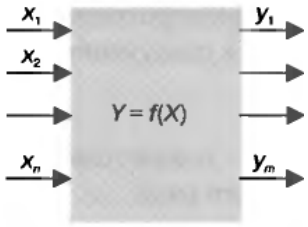

Рис. 1. Обобщенная модель прогноза

Здесь набор входных переменных $(i = 1 \ldots n)$, образующих вектор X, исходные данные для прогноза. Набор выходных переменных y_j $(j = 1 \ldots m)$, образующих вектор результата Y, есть набор прогнозируемых величин. Когда решается задача прогнозирования значений временного ряда, описывающего динамику изменения некоторого бизнес-процесса, входные значения – наблюдения за развитием процесса в прошлом, а выходные прогнозируемые значения процесса в будущем. При этом временные интервалы прошлых наблюдений и временные интервалы, по которым требуется получить прогноз, должны соответствовать друг другу. Например, требуется получить прогноз по продажам на будущую неделю. Наблюдения, на основе которых будет строиться прогноз, также должны быть взяты за неделю. Если в базе данных история продаж представлена по дням, то получить данные по неделям можно с помощью соответствующего агрегирования. Обучающая выборка строится путем преобразования временного ряда с помощью скользящего окна.

Метод скользящего окна - алгоритм трансформации, позволяющий сформировать из членов временного ряда набор данных, который может служить обучающим множеством для построения модели прогнозирования. Под окном в данном случае понимается временной интервал, содержащий набор значений, которые используются для формирования обучающего примера. В

процессе работы алгоритма окно смещается по временной последовательности на единицу наблюдения, и каждое положение окна образует один пример.

Кроме того, количество наблюдений за историей развития процесса в прошлом, на основе которого строится прогноз, должно быть больше, чем число прогнозируемых интервалов, то есть $n > m$. Иначе говоря, если мы хотим получить прогноз на неделю, то для этого должны взять наблюдения за несколько прошедших недель. Рассмотрим основные модели прогнозирования.

«Наивная» модель прогнозирования

«Наивная» модель предполагает, что последний период прогнозируемого временного ряда лучше всего описывает будущее этого ряда. В таких моделях прогноз, как правило, является довольно простой функцией от наблюдений прогнозируемой величины в недалеком прошлом.

Простейшая модель описывается выражением:

$$y(t+1) = y(t)$$

где $y(t)$ – последнее наблюдаемое значение;

$\mathrm{y}(t+1)$ – прогноз.

В основу этой модели заложен принцип: *«Завтра будет как сегодня»*.

Ждать от такой примитивной модели точного прогноза не стоит. Она не только не учитывает закономерности прогнозируемого процесса (что в той или иной степени свойственно многим статистическим методам прогнозирования), но и не защищена от случайных изменений в данных, а также не отражает сезонного колебания и тренды.

Чтобы модель учитывала наличие возможных трендов, ее можно несколько усложнить, например преобразовав к виду $y(t+1) = y(t) + [y(t) - y(t-1)]$ или $y(t+1) = y(t) \times [y(t)/y(t-1)]$.

При необходимости учета сезонных колебаний «наивная», модель модифицируется следующим образом:

$$y(t+1) = y(t-s),$$

где s - показатель, учитывающий сезонные изменения прогнозируемого временного ряда.

Экстраполяция

Экстраполяция представляет собой попытку распространить закономерность поведения некоторой функции из интервала, в котором известны ее значения, за его пределы. Иными словами, если значения функции $f(x)$ известны в некотором интервале $[x_0, x_n]$, то целью экстраполяции является определение наиболее вероятного значения в точке x_{n-1}.

Экстраполяция применима только в тех случаях, когда функция $f(x)$ (а

соответственно, и описываемый с ее помощью временной ряд) достаточно стабильна и не подвержена резким изменениям. Если это требование не выполняется, скорее всего, поведение функции в различных интервалах будет подчиняться разным закономерностям.

К существенным факторам, определяющим эффективность применения метода экстраполяции, относится надежность данных, лежащих в основе анализа.

Экстраполяция тенденций получила широкое применение в нормативном прогнозировании. В частности, с помощью этого метода устанавливается, можно ли, используя существующие технологии, достичь заданных производственных или других бизнес показателей.

Наиболее популярным методом экстраполяции в настоящее время является экспоненциальное сглаживание. Основной его принцип заключается в том, чтобы учесть в прогнозе все наблюдения, но с экспоненциально убывающими весами.

Метод позволяет принять во внимание сезонные колебания ряда и предсказать поведение трендовой составляющей.

Прогнозирование методом среднего и скользящего среднего.

Наиболее простая модель этой группы обычное усреднение набора наблюдений прогнозируемого ряда:

$$y(t+1) = \frac{(y(t) + y(t-1) + y(t-2) + \cdots + y(1))}{t}$$

Принцип модели простого среднего: «Завтра будет как в среднем за последнее время». Преимущество такого подхода по сравнению с «наивной» моделью очевидно: при усреднении ссглаживаются резкие изменения и выбросы данных, что делает результаты прогноза более устойчивыми к изменчивости ряда. Но в целом эта модель прогноза столь же примитивна, как и «наивная», и ей присущи те же недостатки. В формуле прогноза на основе среднего предполагается, что ряд усредняется по достаточно длительному интервалу времени (в пределе по всем наблюдениям).

С точки зрения прогноза это не вполне корректно, так как старые значения временного ряда могли формироваться на основе иных закономерностей и утратить актуальность. Поэтому свежие наблюдения из недалекого прошлого лучше описывают прогноз, чем более старые значения того же ряда. Чтобы повысить точность прогноза, можно использовать скользящее среднее:

$$y(t+1) = \frac{(y(t) + y(t-1) + y(t-2) + \cdots + y(t-T))}{T+1}$$

Смысл данного метода заключается в том, что модель «видит» только ближайшее прошлое на T отсчетов по времени и прогноз строится только на этих

наблюдениях.

Например, при скользящем среднем за три месяца прогнозом на май будет среднее значение показателей за февраль, март и апрель, а при скользящем среднем за четыре месяца майский показатель можно оценить, как среднее значение

Метод скользящего среднего весьма прост, и его результаты довольно точно отражают изменения основных показателей предыдущего периода. Иногда он оказывается даже эффективнее, чем методы, основанные на долговременных наблюдениях.

Допустим, требуется составить прогноз объема продаж продукции предприятием, если средний показатель объема продаж за последние несколько лет – 1000 ед. При этом прогноз должен учитывать планируемое сокращение дилерской сети, что, очевидно, повлечет за собой временное снижение объемов продаж.

Если для прогнозирования объема продаж в следующем месяце воспользоваться средним значением этого показателя за четыре последних месяца, то, скорее всего, полученный результат окажется несколько завышенным по сравнению с фактическим. Но если прогноз будет составлен на основании данных за два последних месяца, то он более точно отразит последствия сокращения штата торговых агентов. В этом случае прогноз будет отставать по времени от фактических результатов всего на один-два месяца.

И так, чем меньшее количество наблюдений используется для вычисления скользящего среднего, тем точнее будут отражены изменения показателей, на основе которых строится прогноз. Однако, если для прогнозируемого скользящего среднего используется только одно или два наблюдения, такой прогноз может быть слишком упрощенным. Чтобы определить, сколько наблюдений желательно включить в скользящее среднее, нужно исходить из предыдущего опыта и имеющейся информации о наборе данных. Необходимо соблюдать равновесие между повышенным откликом скользящего среднего на несколько самых поздних наблюдений и большой изменчивостью скользящего среднего. Одно отклонение в наборе данных для трёхкомпонентного среднего может исказить весь прогноз.

Более хороших результатов удается добиться при использовании метода *экспоненциальных средних.* Соответствующая модель записывается с помощью формулы:

$$y(t+1) = \alpha y(t) + (1-\alpha)y'(t)$$

где $y(t+1)$ – прогнозируемое значение;

$y(t)$ – текущее наблюдаемое значение;

$y/(t)$ – прошлый прогноз текущего значения;

α – параметр сглаживания ($0 \leq \propto \leq 1$).

Параметр α позволяет определять степень участия прошлых значений ряда в формировании прогноза. Видно, что с удалением значений ряда от текущего момента в прошлое их вклад в прогноз экспоненциально убывает, а скорость этого убывания регулируется параметром α.

В пределе. когда $\propto = 1$, получим обычный «наивный» прогноз, а при $\propto = 0$ прогнозируемая величина всегда будет равна предыдущему прогнозу. При больших $\propto$ увеличивается вклад самых свежих значений ряда, а с удалением в прошлое вклад значений резко уменьшается. При малых значениях $\propto$ вклад значений ряда в результат прогнозирования распределяется более равномерно.

Обычно при использовании модели экспоненциального сглаживания строятся прогнозы на некотором тестовом наборе при $\propto = \{0{,}01;\ 0{,}02 \ldots 0{,}98;\ 0{,}99\}$. Затем определяется, при каком $\propto$ обеспечивается наиболее высокая точность прогнозирования, и это значение применяется в дальнейшем.

Рассмотренные модели («наивный» метод, методы, основанные на средних, скользящих средних и метод экспоненциального сглаживания) используются при решении несложных задач прогнозирования. Например, при прогнозировании продаж на устоявшихся рынках. Но ввиду примитивности этих моделей следует очень осторожно подходить к их использованию на практике.

Регрессионные модели

К числу наиболее мощных, развитых и универсальных моделей прогнозирования относятся регрессионные модели. Регрессия – это технология статистического анализа, целью которой является определение лучшей модели, устанавливающей взаимосвязь между выходной (зависимой) переменной и набором входных (независимых) переменных.

Применение регрессионных моделей оказывается особенно полезным в следующих случаях:

- Входные переменные задачи известны или легко поддаются измерению, а выходные нет.
- Значения входных переменных известны из начально, и на их основе требуется предсказать значения выходных переменных.
- Требуется установить причинно-следственные связи между входными и выходными переменными, а также силу этих связей.

В технологиях прогнозирования наиболее широко используется такой вид регрессионной модели, как обобщенная линейная модель. Ее популярность вызвана тем, что многие процессы в управлении, экономике и бизнесе линейны по

своей природе. Кроме того, существуют методы, которые позволяют привести нелинейную модель к линейной с минимальными потерями точности.

Обобщенная линейная модель регрессии описывается уравнением:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

где x_i – значение i-го наблюдения;

i – коэффициент регрессии.

Метод декомпозиции временного ряда

Одним из методов прогнозирования временных рядов является определение факторов, которые влияют на каждое значение временного ряда. Для этого выделяется каждая компонента временного ряда, вычисляется ее вклад в общую составляющую, а затем на его основе прогнозируется будущие значения временного ряда. Данный метод получил название декомпозиции временного ряда.

Термин «декомпозиция» означает, что исходный временной ряд представляется как композиция компонент – тренда, сезонной и циклической. Для построения прогноза выполняется выделение этих компонент из ряда, то есть декомпозиция или разложение ряда по компонентам. Методы декомпозиции могут использоваться для построения как краткосрочных, так и долгосрочных прогнозов.

Фактически декомпозиция – это выделение компонент временного ряда и их проекция на будущее с последующей комбинацией для получения прогноза. Метод был разработан довольно давно, однако сейчас его использует все реже и реже из-за присущих ему ограничений. Проблема заключается в том, что обеспечить достаточно высокую точность прогноза для отдельных компонент очень затруднительно.

Рассмотрим прогнозирование методом декомпозиции с помощью тренда (рис. 2).

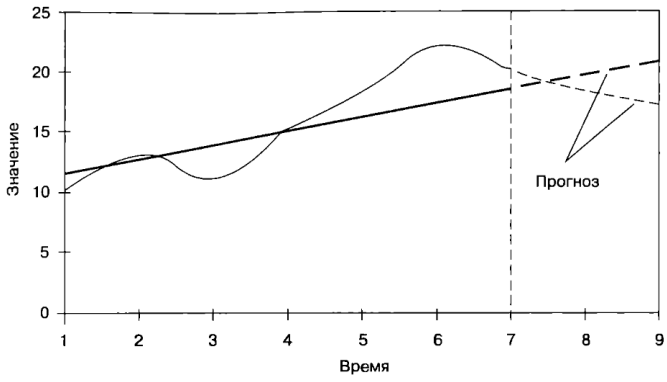

Рис. 2. Прогнозирование с помощью тренда

Если тренд линейный, что типично для многих реальных временных рядов, то он представляет собой прямую линию, описываемую уравнением:

$$y = a + b \times t$$

где y – значение ряда;

a, b – коэффициенты, определяющие расположение и наклон линии тренда;

t – время.

Если уравнение линии тренда известно, с его помощью можно рассчитать значения тренда в любой, в том числе будущий, момент времени. Достаточно воспользоваться уравнением:

$$y_{i+k} = a + b \times (t' + k)$$

де t' – начало прогноза;

k – горизонт прогноза.

При использовании сезонности для прогнозирования методом декомпозиции сначала из временного ряда убирается тренд и слаживается возможная циклическая компонента. Тогда можно считать, что оставшиеся данные будут обусловлены в основном сезонными колебаниями. На основе этих данных вычисляются так называемые сезонные индексы, которые характеризуют изменения временного ряда во времени. Например, временной ряд содержит наблюдения по месяцам в течении года. Сезонный индекс, равный 1, будет установлен для месяца, ожидаемое значение в котором составляет 1/12 от общей суммы по всем месяцам. Если для некоторого месяца устанавливается индекс 1,2, то ожидаемое значение для этого месяца составляет 1/12 + 20 %, а если 0,8 – то 1/12 - 20 % и т. д. Ясно, что сумма месячных сезонных индексов за год должна равняться 12.

Диаграмма сезонных индексов позволяет оценить вероятный относительный вклад месяцев будущего года в общегодовой показатель на основе сезонной компоненты. Использовать сезонность для прогнозирования можно тогда, когда сезонные колебания имеют хорошую повторяемость.

Ансамбли моделей. Комбинирование решений.

Если обученной модели хорошо удается разделить классы и она допускает мало ошибок классификации, то такая модель может рассматриваться как сильная. Слабая модель, напротив, не позволяет надежно разделять классы или давать точные предсказания, допускает в работе большое количество ошибок. Если в результате обучения мы получили слабую модель, то ее необходимо усовершенствовать, подбирая тип классификатора, алгоритмы и параметры обучения. Но часто встречаются ситуации, когда все возможности совершенствования единственной модели исчерпаны, а качество ее работы по-прежнему неудовлетворительно. Это может быть связано со сложностью решаемой задачи и искомых закономерностей, с низким качеством обучающих данных и

другими факторами.

Неизбежно возникает вопрос: как усилить слабую модель, что сделать, для повышения эффективности классификации? Вполне логичным выходом из ситуации является попытка применить к неудачным результатам работы первой модели еще одну модель, задача которой классифицировать те примеры, что остались нераспознанными.

Если и после этого результаты неудовлетворительны, можно применить третью модель и так далее до тех пор, пока не будет получено достаточно точное решение. Таким образом, для решения одной задачи классификации или регрессии мы применили несколько моделей, при этом нас интересует не результат работы каждой отдельной модели, а результат, который дает весь набор моделей. Такие совокупности моделей называются *ансамблями моделей*.

Набор моделей, применяемых совместно для решения единственной задачи, называется ансамблем (комитетом) моделей.

Цель объединения моделей очевидна улучшить (усилить) решение, которое дает отдельная модель. При этом предполагается, что единственная модель никогда не сможет достичь той эффективности, которую обеспечит ансамбль. Использование ансамблей вместо отдельной модели в большинстве случаев позволяет повысить качество решений, однако такой подход связан с рядом проблем, основными из которых являются:

- увеличение временных и вычислительных затрат на обучение нескольких моделей;
- сложность интерпретации результатов;
- неоднозначный выбор методов комбинирования результатов, выдаваемых отдельными моделями.

Перечисленные проблемы аналогичны тем, что возникают при работе нескольких людей экспертов. Действительно, приходится собирать группу экспертов, предоставлять им необходимую информацию, обсуждать задачу и т. д. Все это отнимает намного больше времени, чем принятие решения одним человеком. Сложность интерпретации результатов экспертных оценок также имеет место, ведь каждый эксперт оперирует терминами своей предметной области, формулирует выводы на уровне своего понимания проблемы. И наконец, выбор метода обобщения отдельных заключений экспертов, позволяющего получить наилучшие результаты, не является однозначным.

Виды ансамблей

В последнее десятилетие ансамбли моделей стали областью очень активных исследований в машинном обучении, что привело к разработке большого числа

разнообразных методов формирования ансамблей.

Первым вопросом при формировании ансамбля является выбор базовой модели (base model). Ансамбль в целом может рассматриваться как сложная, составная модель (multiple model), состоящая из отдельных (базовых) моделей. Здесь возможны два случая.

1. Ансамбль состоит из базовых моделей одного типа, например, только из деревьев решений, только из нейронных сетей и т. д. (рис. 3).

Рис. 3. Однородный ансамбль

2. Ансамбль состоит из моделей различного типа нейронных сетей, деревьев решений, регрессионных моделей и т. д. (рис. 4).

Рис. 4. Ансамбль, состоящий из моделей различного типа

Каждый подход имеет свои преимущества и недостатки. Использование моделей различных типов дает классификатору дополнительную гибкость. Но, поскольку выход одной модели применяется для формирования обучающего множества для другой, возможно, потребуются дополнительные преобразования, чтобы согласовать входы и выходы моделей.

Второй вопрос: как использовать обучающее множество при построении ансамбля? Здесь также существуют два подхода.

1. *Перевыборка (resampling)*. Из исходного обучающего множества извлекается несколько подвыборок, каждая из которых используется для обучения одной из моделей ансамбля. Данный подход иллюстрируется с помощью рис. 5. Если ансамбль строится на основе моделей различных типов, то для каждого типа будет свой алгоритм обучения.

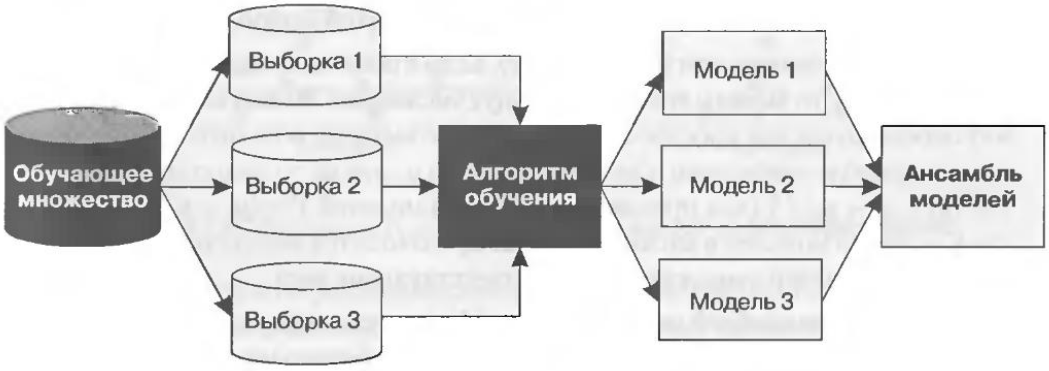

Рис. 5. Перевыборка

2. Использование одною обучающею множества для обучения всех моделей ансамбля (рис. 6).

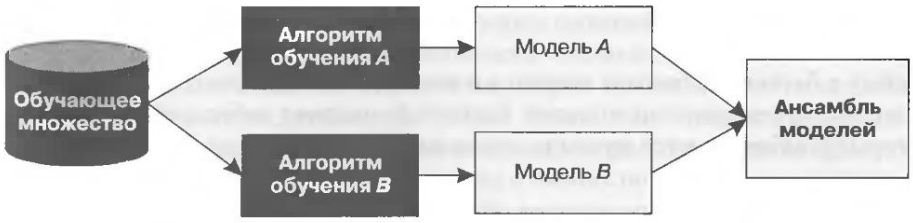

Рис. 6. Использование одного обучающего множества для всех моделей ансамбля

Третий вопрос касается метода комбинирования результатов, выданных отдельными моделями: что будет считаться выходом ансамбля при определенных состояниях выходов моделей? Обычно используются следующие способы комбинирования:

1. *Голосование.* Применяется в задачах классификации, то есть для категориальной целевой переменной. Выбирается тот класс, который был выдан простым большинством моделей ансамбля. Пусть, например, решается задача бинарной классификации с целевыми переменными, *Да* и *Нет*, для чего используется ансамбль, состоящий из трех моделей. Если две модели выдали выход *Нет* и только одна *Да*, то общий выход ансамбля будет *Нет*.

2. *Взвешенное голосование.* В ансамбле одни модели могут работать лучше, а другие хуже. Соответственно, к результатам одних моделей доверия больше, а к результатам других меньше. Чтобы учесть уровень достоверности результатов, для моделей ансамбля могут быть назначены веса (баллы). Например, в случае, рассмотренном в предыдущем пункте, для моделей, выдавших результат *Нет*, установлены веса 30 и 40, указывающие на невысокую достоверность этих результатов, В то же время единственная модель, которая выдала *Да*, имеет вес 90. Тогда голосование будет производиться с учетом весов моделей: 30 (Нет) + 40 (Нет) == 70 (Нет) < 90 (Да). Таким образом, модель с выходом *Да* перевесила обе модели с выходом *Нет* и общий выход ансамбля будет *Да*.

3. *Усреднение* (взвешенное или невзвешенное). Если с помощью ансамбля решается задача регрессии, то выходы его моделей будут числовыми. Выход всего ансамбля может определяться как простое среднее значение выходов всех моделей. Например, если в ансамбле три модели и их выходы равны y_1, y_2 и y_3, то выход ансамбля будет $Y = (y_1 + y_2 + y_3)/3$ (для произвольного числа моделей $Y = (y_1 + y_2 + \cdots + y_k)/K$ где K – число моделей в ансамбле)/ Если производится взвешенное усреднение, то выходы моделей умножаются на соответствующие

веса.

Исследования ансамблей моделей в Data Mining стали проводиться относительно недавно. Тем не менее к настоящему времени разработано множество различных методов и алгоритмов формирования ансамблей. Среди них наибольшее распространение получили такие методы, как бэггинг и бустинг.

Бэггинг - основная идея

Сначала на основе исходного множества данных путем случайного отбора формируется несколько выборок. Они содержат такое же количество примеров, что и исходное множество. Но, поскольку отбор производится случайно, набор примеров в этих выборках будет различным: одни примеры могут быть отобраны по нескольку раз, а другие ни разу. Затем на основе каждой выборки строится классификатор и выходы всех классификаторов комбинируются (агрегируются) путем голосования или простого усреднения. Ожидается, что полученный результат будет намного точнее любой одиночной модели, построенной на основе исходного набора данных. Обобщенная схема процедуры бэггинга представлена на рис.7 (на примере дерева решений).

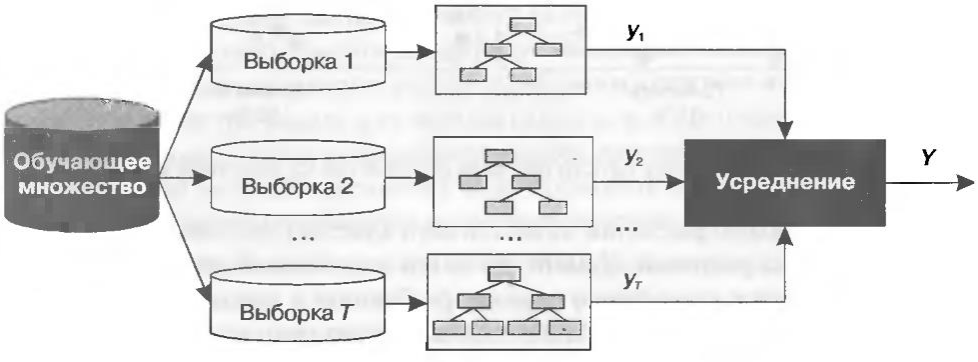

Рис. 7. Схема процедуры бэггинга

Таким образом, бэггинг включает следующие шаги.

1. Из обучающего множества извлекается заданное количество выборок одинакового размера.

2. На основе каждой выборки строится модель.

3. Определяется общий результат путем голосования или усреднения выходов моделей.

Остановка процедуры бэггинга производится на основе следующих критериев:

- На некоторой итерации t ошибка ε^t классификатора C^t становится равной 0 или больше либо равной 0,5. В этом случае процедура бэггинга останавливается, а последний классификатор удаляется: $\mathrm{T} = t - 1$. Таким образом, бэггинг не пускает в ансамбль «плохие» классификаторы.

- Число итераций достигло заданного пользователем предела T. Как и для большинства других итеративных алгоритмов Data Mining, однозначного ответа на вопрос, каково достаточное число итераций, не существует/ Оно подбирается эмпирическим путем.
- Бэггинг, хотя и в меньшей степени, чем отдельные модели, склонен к переобучению. Поэтому критерием для остановки процедуры может служить возрастание ошибки на тестовом множестве.

Случайный лес

Случайный лес – алгоритм машинного обучения заключающийся в использовании ансамбля решающих деревьев каждое из которых само по себе даёт очень невысокое качество классификации, но за счёт их большого количества получается необходимый результат. Точно так же, как инвестиции с низкими корреляциями (например, акции и облигации) объединяются, чтобы сформировать портфель больший, чем сумма его частей.

Алгоритм сочетает в себе две основные идеи: метод бэггинга Бреймана, и метод случайных подпространств, предложенный Тин Кам Хо. Достаточно, чтобы алгоритмы были в некоторой степени не похожи друг на друга. Алгоритм применяется для задач классификации, регрессии и кластеризации.

Построим ансамбль алгоритмов, где базовый алгоритм — это решающее дерево. Будем строить по следующей схеме:

1. Для построения i-го дерева:

а. Сначала, как в обычном бэггинге, из обучающей выборки X выбирается с возвращением случайная подвыборка X^i того же размера, что и X/

б. В процессе обучения каждого дерева в каждой вершине случайно выбираются $n < N$ признаков, где $N -$ полное число признаков (метод случайных подпространств), и среди них ищется оптимальный сплит (разделение). Такой приём как раз позволяет управлять степенью скоррелированности базовых алгоритмов.

2. Чтобы получить предсказание ансамбля на тестовом объекте, усредняем отдельные ответы деревьев (для регрессии) или берём самый популярный класс (для классификации).

3. В результате построили Random Forest (случайный лес) — комбинацию бэггинга и метода случайных подпространств над решающими деревьями

Достоинства:

- Способность эффективно обрабатывать данные с большим числом признаков и классов;
- Нечувствительность к любым монотонным преобразованиям значений

признаков;

- Одинаково хорошо обрабатываются как непрерывные, так и дискретные признаки;
- Существуют методы оценивания значимости отдельных признаков;
- Внутренняя оценка способности модели к обобщению;
- Высокая параллелизуемость и масштабируемость;
- Случайные леса очень гибки и обладают очень высокой точностью.

Недостатки:

- Большой размер получающихся моделей;
- Построение леса сложнее и отнимает больше времени;
- Чем больше объем, тем меньше интуитивное понимание.

Бустинг – основная идея.

По сравнению с бэггингом бустинг (boosting) несколько более сложная процедура, но во многих случаях работает эффективнее. Как и бэггинг, бустинг использует неустойчивость алгоритмов обучения и начинает создание ансамбля на основе единственного исходного множества. Но если в бэггинге модели строятся параллельно и независимо друг от друга, то в бустинге каждая новая модель строится на основе результатов ранее построенных моделей, то есть модели создаются последовательно.

Бустинг создает новые модели таким образом, чтобы они дополняли ранее построенные, выполняли ту работу, которую другие модели сделать не смогли на предыдущих шагах. И наконец, последнее отличие бустинга от бэггинга заключается в том, что всем построенным моделям в зависимости от их точности присваиваются веса (бэггинг, напомним, использует взвешенное голосование или усреднение).

В настоящее время разработано большое количество различных модификаций бустинга. Рассмотрим один из наиболее популярных алгоритмов - AdaBoost.M1, который предназначен для решения задач классификации.

Вместо извлечения выборок из исходного множества данных бустинг в качестве возмущающего фактора применяет взвешивание примеров. Вес каждого примера устанавливается в соответствии с его влиянием на обучение классификатора.

На каждой итерации вектор весов подстраивается таким образом, чтобы отражать эффективность данного классификатора. В результате вес неправильно классифицированных примеров увеличивается. Итоговый классификатор также агрегирует обученные классификаторы путем голосования, но теперь голос классификатора является функцией его точности.

Обозначим вес примера x на итерации t как ω_x^t, при этом вес на первой итерации задается как $\omega_x^t = \frac{1}{N}$ для каждого x ($N -$ число примеров). На каждой итерации $t = 1,2\ldots T$ классификатор C^t конструируется из данных примеров в Соответствии с распределением их весов ω^t (то есть как будто вес ω_x^t отражает вероятность появления примера x). Ошибка ε^t классификатора t также измеряется относительно весов и представляет собой сумму весов примеров, которые были классифицированы неправильно. Когда ε^t становится больше 0,5, итерации прекращаются последний классификатор удаляется и T изменяется на $t-1$. Наоборот, если $\varepsilon^t = 0$ (классификатор C^t правильно классифицировал все примеры), итерации останавливаются и $T = t$. В остальных случаях вектор весов ω^{t+1} для следующей итерации генерируется путем умножения текущих весов примеров, которые были правильно распознаны классификатором C^t, на коэффициент $\beta^t = \varepsilon^t/(1-\varepsilon^t)$, а затем нормируется так, чтобы $\sum_x \omega^{t+1} = 1$. Тогда:

$$\omega^{t+1} = \omega^t \frac{\varepsilon^t}{(1-\varepsilon^t)} \quad (1)$$

Итоговый классификатор C^* получается путем суммирования голосов классификаторов C^1, C^2 ... C^T , где голос классификатора C^t определяется как $log(1/\beta^t)$ единиц.

Если ошибка отдельного классификатора ε^t всегда меньше 0,5. То значение ошибки итогового классификатора C^* экспоненциально стремится к 0 с увеличением числа итераций t. Последовательность слабых классификаторов C^1, C^2 ... C^T может быть усилена до классификатора C^*, который обычно получается более точным, чем отдельные классификаторы, Конечно, при этом нельзя гарантировать высокую обобщающую способность C^*.

Процедура бустинга включает следующие шаги:

1. Для всех примеров исходного множества данных устанавливаются равные начальные веса ω_0.

2. На основе взвешенного набора примеров строится классификатор C^t , вычисляется и запоминается выходная ошибка данного классификатора ε^t.

3. Рассчитывается коррекция весов примеров обучающего множества, и веса корректируются по формуле (1),

4, Если ошибка $\varepsilon^t = 0$ или $\varepsilon^t \geq 0{,}5$ то классификатор C^t удаляется и процедура бустинга останавливается.

5. В противном случае осуществляется переход на шаг 2 и начинается следующая итерация.

Таким образом, параметрами, настраиваемыми на каждой итерации, являются веса примеров. При этом чем больше раз пример был неправильно распознан предыдущими моделями, тем выше его вес. Вес можно рассматривать как вероятность попадания примера на следующую итерацию.

Стекинг

Стекинг (stacking – укладка) — алгоритм ансамблирования, основные отличия которого от предыдущих состоят в следующем:

1. он может использовать алгоритмы разного типа, а не только из какого-то фиксированного семейства. Например, в качестве базовых алгоритмов могут выступать метод ближайших соседей и линейная регрессия
2. результаты базовых алгоритмов объединяются в один с помощью обучаемой мета-модели, а не с помощью какого-либо обычного способа агрегации (суммирования или усреднения)

Простейшая схема стекинга — блендинг (Blending): обучающую выборку делят на две части. На первой обучают базовые алгоритмы. Затем получают их ответы на второй части и на тестовой выборке. Понятно, что ответ каждого алгоритма можно рассматривать как новый признак (т.н. «метапризнак»). На метапризнаках второй части обучения настраивают метаалгоритм. Затем запускают его на метапризнаках теста и получают ответ.

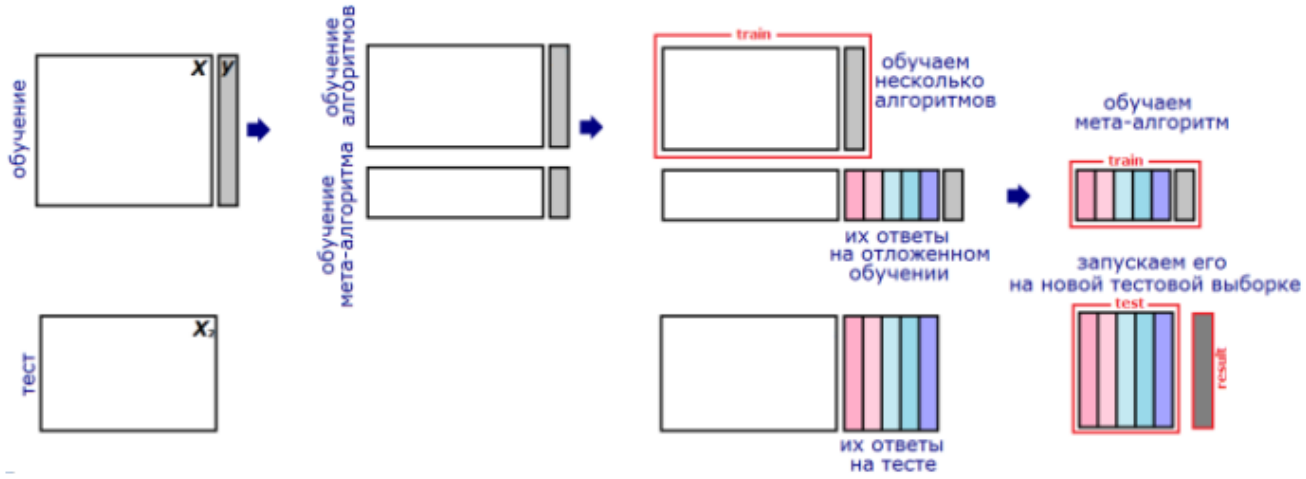

Рис.8. Схема классического блендинга.

Самый большой недостаток блендинга (в описанной реализации) – деление обучающей выборки. Получается, что ни базовые алгоритмы, ни метаалгоритм не используют всего объёма обучения (каждый — только свой кусочек). Понятно, что для повышения качества надо усреднить несколько блендингов с разными разбиениями обучения. Вместо усреднения иногда конкатенируют обучающие (и тестовые) таблицы для метаалгоритма, полученные при разных разбиениях (см. рис. 9): здесь мы получаем несколько ответов для каждого объекта тестовой

выборки – их усредняют. На практике такая схема блендинга сложнее в реализации и более медленная, а по качеству может не превосходить обычного усреднения.

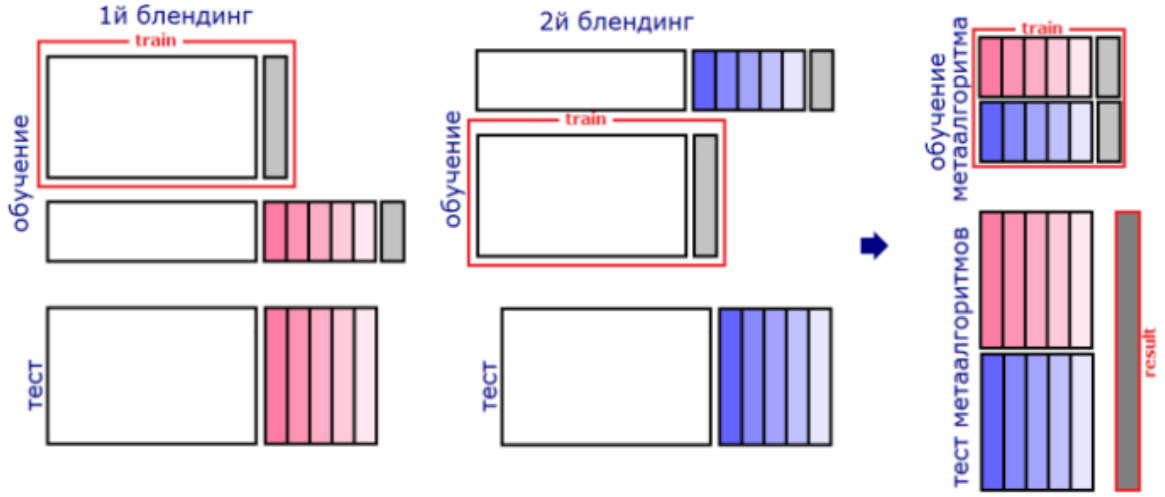

Рис.9. Возможная модификация блендинга.

Второй способ борьбы за использование всей обучающей выборки — реализация классического стекинга. Ясно, что совсем не делить обучение на подвыборки (т.е. обучить базовые алгоритмы на всей обучающей выборке и потом для всей выборки построить метапризнаки) нельзя: будет переобучение, поскольку в каждом метапризнаке будет «зашита» информация о значении целевого вектора (чтобы понять, представьте, что один из базовых алгоритмов — ближайший сосед). Поэтому выборку разбивают на части (фолды), затем последовательно перебирая фолды обучают базовые алгоритмы на всех фолдах, кроме одного, а на оставшемся получают ответы базовых алгоритмов и трактуют их как значения соответствующих признаков на этом фолде. Для получения метапризнаков объектов тестовой выборки базовые алгоритмы обучают на всей обучающей выборке и берут их ответы на тестовой.

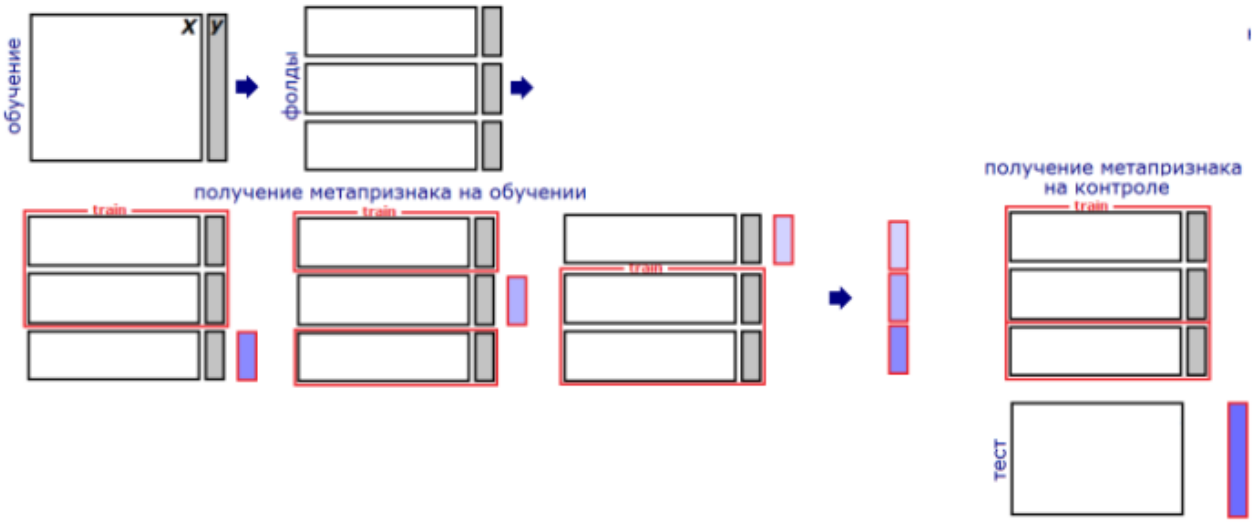

Рис.10. Получение метапризнака в классическом стекинге.

Здесь тоже желательно реализовывать несколько разных разбиений на фолды и затем усреднить соответствующие метапризнаки. Но самый главный недостаток (классического) стекинга в том, что метапризнаки на обучении (пусть и полноценном — не урезанном) и на тесте разные. Для объяснения возьмём какой-нибудь базовый алгоритм, например, гребневую регрессию. Мета-признак на обучающей выборке — это не ответы какого-то конкретного регрессора, он состоит из кусочков, которые являются ответами разных регрессий (с разными коэффициентами). А метапризнак на контрольной выборке вообще является ответом совсем другой регрессии, настроенной на всём обучениии. В классическом стекинге могут возникать весьма забавные ситуации, когда какой-то метапризнак содержит мало уникальных значений, но множества этих значений на обучении и тесте не пересекаются!