5. Случайный лес. Продвинутые техники ансамблирования. Дилемма смещения-дисперсии

Цель занятия

В результате обучения на этой неделе вы:

- разберете различные техники ансамблирования и теоретические предпосылки к их применению

а именно:

- познакомитесь с процедурой bootstrap, на основе которого строится метод ансамблирования bagging
- рассмотрите принципы построения случайного леса
- научитесь использовать технику blending, которая позволяет строить ансамбли не параллельно друг другу, а последовательно
- разберете один из самых популярных методов ансамблирования - стэкинг

а также:

- узнаете, в чем состоит дилемма смещения-дисперсии (bias-variance tradeoff)
-

План занятия

1. Техника ансамблирования
2. RSM
3. Дилемма смещения
4. Смешивание
5. Стекинг

Конспект занятия

1. Техника ансамблирования

Процедура Bootstrap

Рассмотрим некоторую выборку X. Мы можем выбирать из нее объекты с повторениями. То есть берем объект, изучаем его признаковое описание, а затем возвращаем объект обратно. На втором шаге мы опять берем случайный объект.

Из-за того, что предыдущий мы вернули обратно, есть небольшая вероятность, что мы возьмем тот же.

Bootstrap – техника порождений нескольких выборок из одной исходной с помощью выбора с повторениями.

Логично, что бутстрапирование выборки похожи друг на друга, потому что есть совпадающие объекты.

Такие выборки обычно используются в прикладной статистике для оценки различных параметров распределения, доверительных интервалов.

Мы будем использовать Bootstrap для обучения сразу нескольких моделей.

Пусть у нас есть несколько моделей, и каждая обладает некоторой ошибкой:

$\varepsilon_j(x) = b_j(x) - y(x), \ j = 1, \ldots, N$

$b_j(x)$ – предсказание j-й модели,

$y(x)$ – истинная зависимость, которую мы пытаемся описать.

N – количество бутстрапированных выборок.

На каждой из выборок обучено по одной модели, то есть всего обучено N моделей.

Тогда средняя квадратичная ошибка – мат. ожидание по x:

$$\mathrm{E}_x\Big(b_j(x) - y(x)\Big)^2 = \mathrm{E}_x\varepsilon_j^2(x).$$

Средняя ошибка для N моделей:

$$E_1 = \frac{1}{N}\sum_{j=1}^{N} \mathrm{E}_x\varepsilon_j^2(x).$$

На практике вместо мат. ожидания будем использовать выборочное среднее, поскольку нам не всегда будет доступно все распределение.

Добавим некоторые предположения:

- Ошибки не смещенные

  $\mathrm{E}_x\varepsilon_j(x) = 0$

- Ошибки некоррелированы между собой

  $\mathrm{E}_x\varepsilon_i(x)\varepsilon_j(x) = 0, \ i \neq j.$

  То есть наши модели ошибаются случайным образом.

Мы предполагаем, что ошибка случайна.

Такие предположения приводят к тому, что итоговая оценка на предсказание модели – теперь усредненная модель из всех:

$$a(x) = \frac{1}{N}\sum_{j=1}^{N} b_j(x).$$

То есть у нас теперь есть некоторая модель, которая агрегирует данные всех моделей, по сути строит их ансамбль, и выдает усредненное предсказание.

Тогда ошибка модели $a(x)$ – ошибка ансамбля из N моделей:

$$E_N = \mathrm{E}_x\left(\frac{1}{N}\sum_{j=1}^{n} b_j(x) - y(x)\right)^2 = \mathrm{E}_x\left(\frac{1}{N}\sum_{j=1}^{N} \varepsilon_j(x)\right)^2 = \frac{1}{N^2}\mathrm{E}_x\left(\sum_{j=1}^{N} \varepsilon_j^2(x) + \sum_{i\neq j} \varepsilon_i(x)\varepsilon_j(x)\right) = \frac{1}{N}E_1. \quad (1)$$

Ошибка становится меньше в N раз.

Если ошибка так сильно уменьшается, почему бы тогда не использовать ансамблирование везде? Но мы предположили, что ошибки не зависимы друг от друга и не скоррелированы, что на практике встречается довольно редко.

Но даже если ошибки имеют корреляцию меньше, чем 1, сумма (1) все равно будет меньше, чем ошибка одной модели. Ошибку мы все равно понизим, но конечно не в N раз.

> **Пример**. “Мудрость толпы”. В некотором городе собралась на площади толпа крестьян. Крестьянам предложили задачу: привели быка и сказали, что тот, кто сможет оценить вес быка с точностью до фунта, получит этого быка. Показания людей записывали. Все оценки усреднили, и получили довольно точный вес быка.
>
> Идея “мудрости толпы” в том, что множество крестьян не советовались друг с другом. Каждый из них делал оценку веса на глаз, не слушая других. Оценки конечно должны иметь смысл. Крестьяне имели опыт оценки на глаз в сельском хозяйстве.

Bagging

На основе процедуры Bootstrap можно построить Bagging:

Bagging = Bootstrap aggregating

То есть порождаем с помощью бутстрапирования несколько выборок, затем обучаем на этих выборках и агрегируем предсказание, усредняя между собой. Это позволяет снизить дисперсию предсказания.

Bagging на практике используется часто с решающими деревьями. Решающие деревья очень хорошо подходят под Bagging, потому что они за счет переобучения становятся сильно непохожими друг на друга. Из-за этого их ошибки не будут полностью скоррелированы, и усредняя несколько таких переобученных деревьев, мы повышаем точность предсказания.

2. RSM

Случайный лес

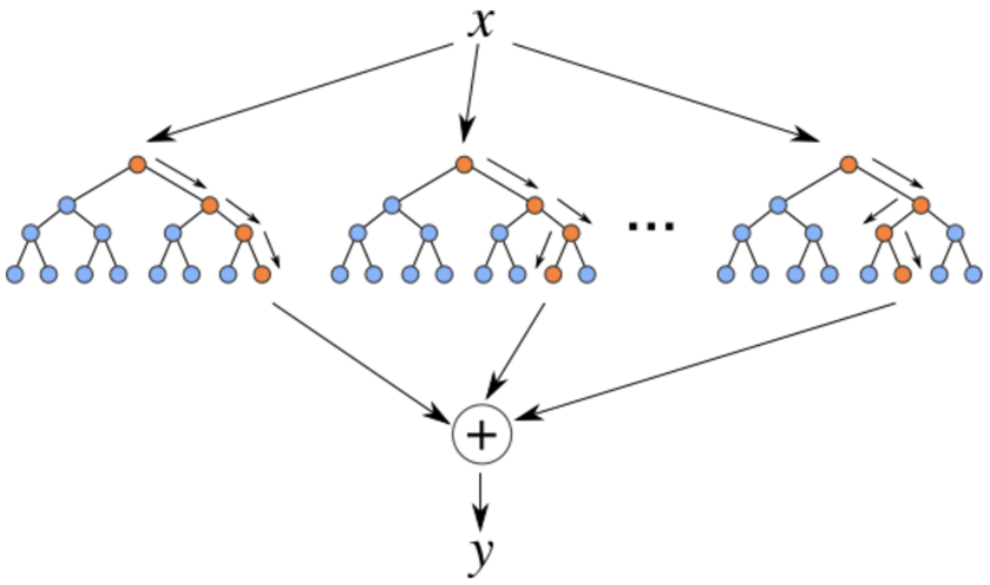
Как сделать деревья еще более непохожими друг на друга?

Можно воспользоваться **методом случайных подпространств** (Random Subspace Method, RSM). Мы будем использовать почти такой же подход, как в Bagging, но случайным образом выбирать не объекты, а признаки.

То есть мы взяли подвыборку, и эта выборка описывается только, допустим, пятью признаками из доступных восьми. Признаки выбрали случайным образом.

Или на каждом шаге построения дерева мы можем случайным образом выбирать признаковое подмножество. То есть мы теперь не позволяем на каждом шаге выбирать дереву наилучший признак из всех с точки зрения разбиения. Мы позволяем выбирать наилучший только из случайно выбранных. Это позволяет сделать дерево более “случайным”, но при этом дерево продолжит работать.

За счет этого получаем много деревьев, которые сильно не похожи друг на друга:



Деревья рандомизированы и на уровне выборки, и на уровне признаков.

Теперь несколько непохожих обученных деревьев можно усреднить, и тем самым, значительно уменьшить ошибку.

Такой метод называется **случайным лесом (Random Forest)**:

Bagging + RSM = Random Forest.

Это один из наиболее сильных, эффективных методов в машинном обучении. Является классической моделью машинного обучения.

Метод случайного леса, поскольку это решающие деревья, позволяет работать и с задачей классификации, и с задачей регрессии.

Random Forest – это техника, которая использует слабость деревьев в качестве их силы.

В случайном лесе все деревья строятся независимо друг от друга. Именно за счет этого достигается результат снижения ошибки.

Свойства случайного леса

Метод случайного леса обладает несколькими свойствами:

- Он универсален. Подходит задачам регрессии, классификации и ряду других.
- Довольно быстр, эффективно обучается.
- Есть некоторые модификации:
  - Extremely Randomized Trees – позволяет работать с еще более зашумленной выборкой
  - Isolation Forest – метод для детекции аномалий
- OOB (Out of Bag estimation) – оценка на объектах, не попавших в выборку. Каждое дерево учится на бутстрапированной выборке, которая содержит в себе некоторые объекты из исходной выборки. Для каждой бутстрапированной выборки есть множество объектов, которые в неё не попали. Соответственно, эти объекты не попали в обучающую выборку для конкретного дерева. По сути для этого дерева данные объекты являются валидационными. Если ансамбль достаточно большой, то для каждой точки из обучающей выборки найдется несколько деревьев, которые в себя эту точку не включают для обучения. Значит с помощью этих деревьев можно сделать предсказание на тех точках, которые не попали в обучающую выборку.

  $$OOB = \sum_{i=1}^{l} L\left(y_i, \frac{1}{\sum_{n=1}^{N}\left[x_i \notin X_n\right]} \sum_{n=1}^{N}\left[x_i \notin X_n\right] b_n\left(x_i\right)\right)$$

  OOB по сути позволяет в некотором роде избавиться от деления выборки на train и validation. Теперь в процедуре Bootstrap мы уже неявным образом это делаем.

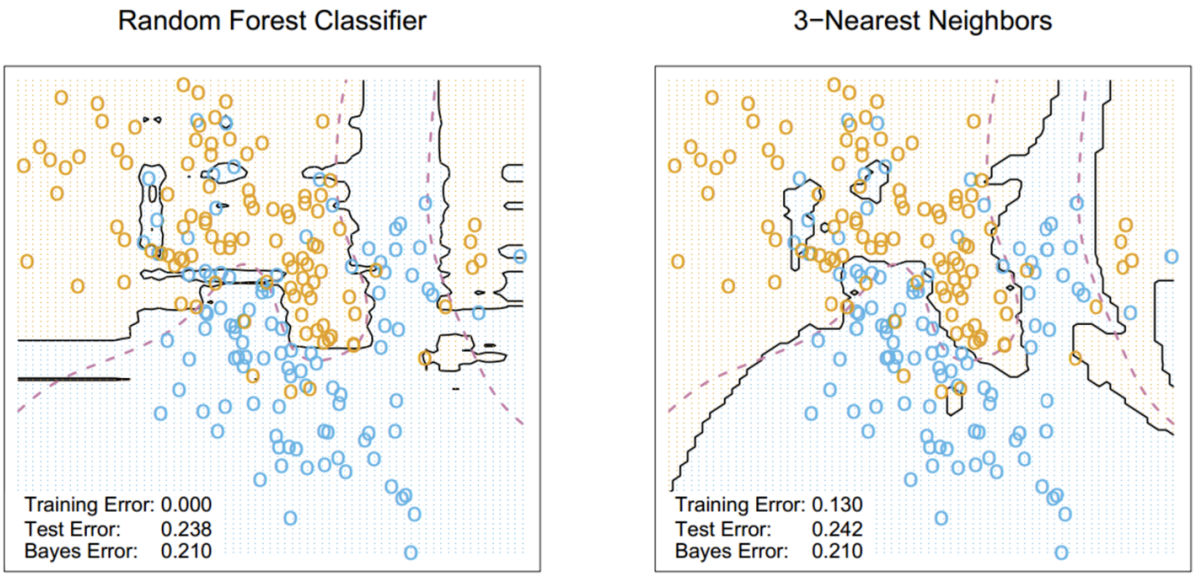
Домашнее задание. Посчитать, с какой вероятностью объект не попадет в бутстрапированную выборку.

Поскольку случайный лес – алгоритм ансамблирования деревьев, то он неплохо работает, если у нас сильно скоррелированы признаки.

Также случайный лес хорошо работает с пропущенными значениями (как и решающие деревья по отдельности).

Случайный лес и KNN

Пример.



В важных местах разделения границы похожи.

Случайный лес некоторым образом аппроксимирует метод kNN. Решающее дерево делит признаковое пространство на подобласти, каждой из которых приписывает некоторое значение. kNN для каждой точки в признаковом пространстве находит значение целевой функции, как некоторое усреднение из ближайших соседей. Если достаточно много соседей, и если достаточно большая глубина в дереве, картинки в примере практически совпадут.

Но в отличие от kNN случайный лес не является квадратичной моделью. У нас нет необходимости считать расстояние до всех точек на каждом объекте.

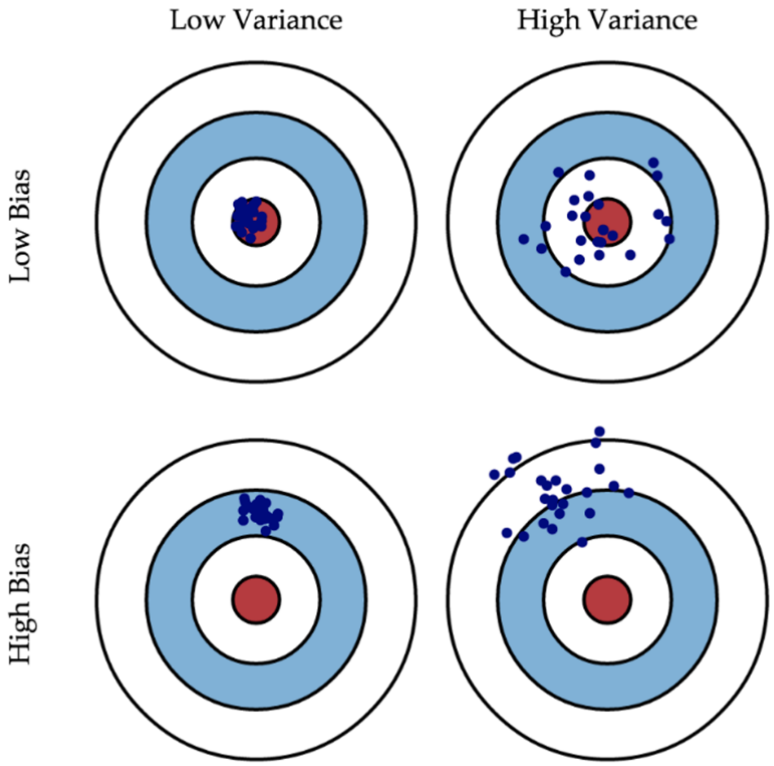
Мы можем в случайном лесе не какую-нибудь энтропию снижать, а на каждом шаге пытаться отделить одну точку от всех остальных (так работает Isolation Forest). Это сделать проще для тех точек, которые находятся далеко от всех остальных. Если мы построим много деревьев и посчитаем их среднюю глубину, чтобы отделить некоторую точку, то можно увидеть, что для большинства точек глубина достаточно большая. Для аномальных точек средний путь будет гораздо короче. Это способ детекции аномалий.

3. Дилемма смещения

Дилемма смещения (Bias-variance tradeoff) или дилемма смещения разброса – это попытка свести к минимуму источники ошибок, которые не позволяют алгоритмам

обучения с учителем делать правильные предсказания за пределами своего обучающего набора.

Рассмотрим классическую картинку дилеммы смещения – попадание дротиком по цели:

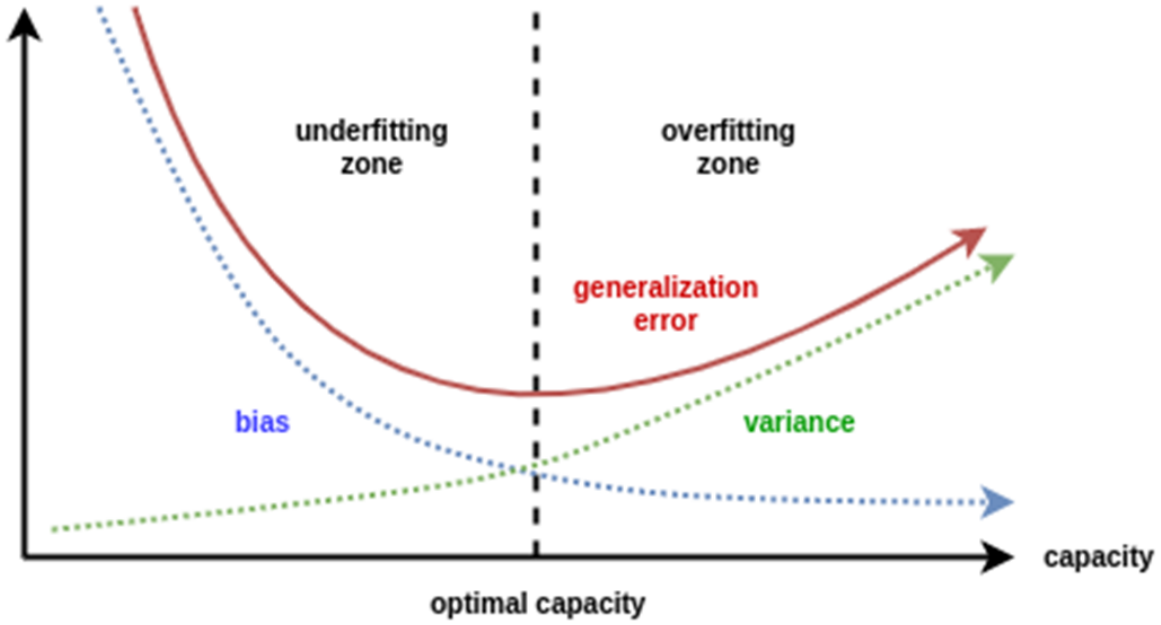


В данной задаче человек – это модель. Он как-то научился кидать дротики. Может кидать дротики достаточно кучно (Low variance, предсказания достаточно стабильны) и достаточно близко к цели (Low Bias, небольшое смещение от центра). На практике в дилемме смещения получается минимизировать либо Bias, либо Variance.

Линейные модели достаточно простые, и не всегда могут точно предсказать целевую переменную, то есть могут часто ошибаться на какую-то величину. Если зависимость нелинейная, то оценка линейной модели будет смещенной, то есть Bias будет высокий. При этом линейные модели довольно стабильные, то есть Variance низкий.

Если взять решающее дерево, особенно переобученное решающее дерево, у него Bias будет маленький. Но если мы чуть-чуть поменяем наш датасет, то предсказания модели “поедут”. Дерево является неустойчивой моделью, у него высокий Variance.

Общая картина выглядит следующим образом:



Простые модели достаточно устойчивы, но не могут точно предсказывать целевую переменную на сложных задачах. Сложные модели – наоборот.

Суммарная ошибка – сумма Bias и Variance. Рассмотрим пример для средней квадратичной ошибки, получим мат. ожидания по всем переменным:

$$L(\mu) = \mathrm{E}_{x,y}\left[(y - \mathrm{E}[y|x])^2\right] + \mathrm{E}_x\left[\left(\mathrm{E}_X[\mu(X)] - \mathrm{E}[y|x]\right)^2\right] + \mathrm{E}_x\left[\mathrm{E}_X\left[\left(\mu(X) - \mathrm{E}_X[\mu(X)]\right)^2\right]\right].$$

Здесь первое слагаемое – шум, который вообще есть в данных,

второе слагаемое – Bias,

третье слагаемое – Variance.

Модели могут либо точно описывать данные, но при этом быть неустойчивыми, либо быть устойчивыми, но сильнее ошибаться. Есть оптимальная точка – optimal capacity, модель оптимальной сложности, которая уже достаточно хорошо описывает данные и не сильно переобучается.

4. Смешивание

Смешивание (Blending) – еще одна техника ансамблирования. Достаточно широко применяется на практике и достаточно интуитивна. В ней нет явных теоретических обоснований. Скорее это уже попытка построить ансамбль моделей не параллельно друг другу (когда все модели не знали о существовании друг друга), а последовательно. Каждая следующая модель будет некоторым образом помогать предыдущей.

Глядя на случайный лес или Bagging, почему мы усредняем модели с одинаковыми весами. Возможно, некоторые модели чуть более полезны, чем другие.

Каждая модель умеет делать предсказание. При помощи наших моделей попробуем улучшить итоговое предсказание, но не просто методом обычного усреднения.

Пример. Пусть у нас есть несколько моделей. Возьмем обучающую выборку и поделим на две части:

Name	Age	Statistics (mark)	Python (mark)	Eye color	Native language	Target (mark)
John	22	5	4	Brown	English	5
Aahna	17	4	5	Brown	Hindi	4
Emily	25	5	5	Blue	Chinese	5
Michael	27	3	4	Green	French	5
Some student	23	3	3	NA	Esperanto	2

На желтой части выборки обучим M базовых алгоритмов. Базовый алгоритм – несколько различных моделей (kNN, Наивный Байес, SVM, логистическая регрессия, решающее дерево, блуждающий лес). После обучения эти модели могут делать предсказания.

Возьмем синюю часть обучающей выборки и для каждого объекта сделаем предсказание каждой из наших моделей. Для синей выборки получим M предсказаний для каждого объекта:

f1	f2	f3	f4	f5	Target
2.2	8	17	5	44	5
7.2	21	33	4	9	2

По сути у нас теперь есть новая обучающая выборка, которая находится в новом признаковом пространстве. Теперь каждый из новых признаков – это предсказание одной из M моделей, которые были обучены на желтой части выборки.

Теперь мы можем обучить какую-нибудь другую модель, например, взвешенное среднее:

$$\hat{f}(x) = \sum_{i=1}^{M} \rho_i f_i(x)$$

Здесь $f_i(x)$ – предсказания моделей, ρ_i – веса.

При этом $\sum_{i=1}^{M} \rho_i = 1, \rho_i \in [0, 1] \; \forall i$.

То есть получаем выпуклую комбинацию всех наших предсказаний.

Это и есть техника Blending, и она позволяет усреднить несколько моделей между собой. На практике неплохо работает, если есть несколько хороших моделей. Например, в соревновательном машинном обучении.

В формуле для взвешенного среднего можно сделать ρ зависимым от x и получить **смесь экспертов**, но эта техника сложна в обучении.

Blending также работает как в задаче классификации, так и в задаче регрессии.

Замечание:

1. В смешивании мы усредняем достаточно сложные модели, которые не похожи друг на друга. Если модели будут простыми, то усреднение не даст преимущества.
2. Плюсы и минусы метода смешивания:
   + Каждая из моделей в смешивании достаточно сильная и может сама по себе работать. Поэтому интуитивно понятно, почему при усреднении нескольких сильных моделей мы получаем хороший результат.
   + Усреднение приводит к тому, что если одна модель сильно ошибалась в каком-то определенном куске признакового пространства, после усреднения эта ошибка уже будет не так существенно влиять на результат.
   - Не всегда линейной композиции моделей может быть достаточно.
   - Приходится разбивать данные на части, а это бывает достаточно трудоемко. Кроме того, если оставить мало на обучение линейной композиции моделей, то можно переобучиться под неё. Если очень много, то мало данных для базовых алгоритмов.

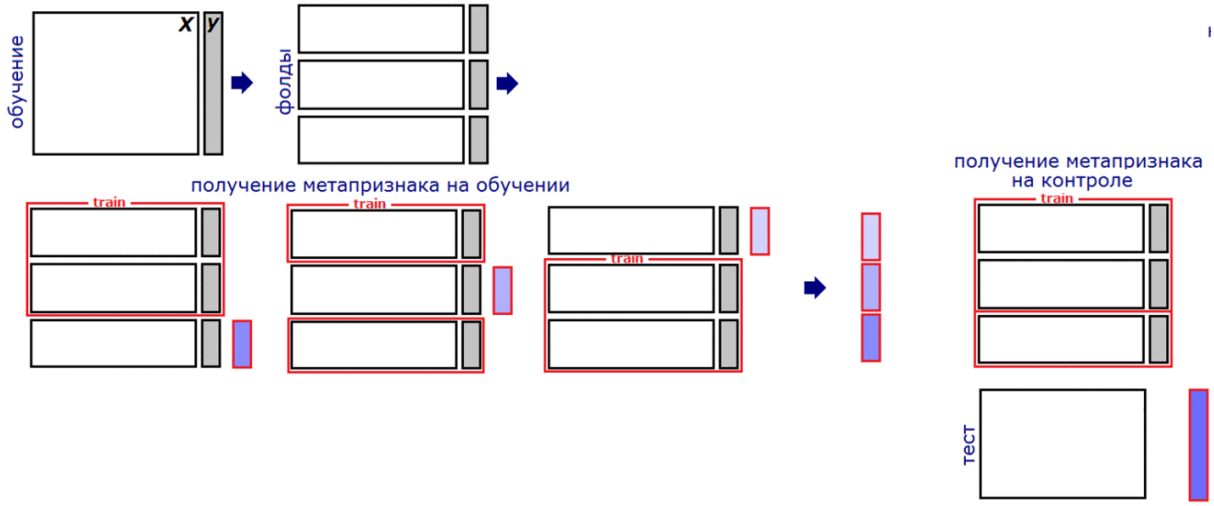
     Чтобы это починить, нужно развить идею смешивания до стекинга.

5. Стекинг

Для смешивания одной линейной композиции моделей может быть недостаточно для решения сложной задачи.

Возьмем некоторую выборку. Мы бы не хотели отделять от нее кусок, потому что разделять данные – довольно трудоемкий процесс. И мы бы хотели использовать не только линейный ансамбль из моделей.

Можно вспомнить, как работает кросс-валидация. Можно взять данные и разбить их на фолды:



При этом мы можем обучать каждый из базовых алгоритмов на первых двух фолдах и сделать предсказание на оставшемся (нижнем), затем обучить на верхнем и нижнем и сделать предсказание для среднего, затем сделать обучение на двух нижних и сделать предсказание для верхнего. Получится, что для всей обучающей выборки есть предсказание от 10 базовых алгоритмов, и при этом мы не теряем данные. У нас есть предсказания для всей обучающей выборки.

Далее мы можем взять наши модели, обучить теперь на всех данных полностью и предсказать с помощью них какие-то метапризнаки для тестовой части выборки.

Теперь каждый объект в выборке описан в терминах 10 предсказаний от 10 моделей, и у нас все еще есть истинный таргет. А после этого мы можем сделать еще одну итерацию стекинга, поскольку мы получили новую обучающую выборку для построения метамодели.

Теперь у нас метаалгоритм – любой алгоритм машинного обучения, какой был.

Преимущества и недостатки стекинга:

+ Это очень сильный метод. На практике особенно для табличных данных хорошо работает. Стекинг некоторым образом предвосхищает нейронные сети.
+ Очень популярен в различных соревнованиях по машинному обучению. С помощью стекинга можно очень сильно повысить качество предсказания модели.
+ Ничто не мешает использовать стекинг несколько раз подряд, то есть делать стекинг над стекингом. Как правило, они бывают глубиной 3 или 4. С каждым новым шагом улучшение предсказания будет все ниже, все выше будет вероятность переобучиться.
- На каждом фолде метапризнаки предсказываются различными моделями. Их обучали на разных данных. И они будут различны. А значит эти метапризнаки обладают различными свойствами, приходят из различных распределений. Мы ломаем гипотезу IID. Если сильно маленькие фолды, модель может не заработать. Но регуляризация, которая ограничивает модели, и не позволяет им быть сильно различными, может помочь.
- Очень сложно объяснить, что “натворила” модель. Понять ее работу трудно. В некоторых задачах, например в кредитных рисках, понятность работы модели может быть решающим фактором.

Более подробно можно почитать про стекинг по ссылке на блог Александра Дьяконова.

NB. Базовые алгоритмы должны обладать хоть каким-то представлением о решаемой задаче. При ансамблировании моделей нужно помнить, что модели должны быть адекватными.

Дополнительные материалы для самостоятельного изучения

1. https://dyakonov.org/2017/03/10/стекинг-stacking-и-блендинг-blending/