

ЦИФРОВАЯ  
КУЛЬТУРА  
УНИВЕРСИТЕТ ИТМО

Ансамбли моделей

Высшая Школа Цифровой Культуры  
Университет ИТМО

[dc@itmo.ru](mailto:dc@itmo.ru)

# Содержание

<b>1 Наводящие размышления</b>	<b>2</b>
<b>2 Повторные выборки (Ресемплинг)</b>	<b>3</b>
2.1 Введение . . . . .	3
2.2 Джекнайф . . . . .	5
2.2.1 Немного о самом методе . . . . .	5
2.2.2 Джекнайф-оценка смещения . . . . .	10
2.2.3 Джекнайф-оценка дисперсии . . . . .	15
2.3 Бутстрэп . . . . .	16
2.3.1 Общая идея оценок методом бутстрэп . . . . .	16
2.3.2 Метод бутстрэп . . . . .	17
2.3.3 Построение доверительных интервалов . . . . .	19
2.3.4 Некоторые финальные замечания . . . . .	20
<b>3 Ансамбли моделей</b>	<b>21</b>
3.1 Общее понятие ансамбля . . . . .	21
3.2 Бэггинг . . . . .	25
3.2.1 Пример решения задачи классификации . . . . .	26
3.2.2 Пример задачи регрессии . . . . .	29
3.3 Адаптивный бустинг для двухклассовой классификации (AdaBoost) . . . . .	30
3.3.1 Пример . . . . .	34
3.4 Стекинг (Stacking) . . . . .	38
3.4.1 Пример на пальцах . . . . .	39
3.5 Случайный лес . . . . .	42
<b>4 Многоклассовая классификация</b>	<b>44</b>
4.1 Один против всех (one-vs-all) . . . . .	44
4.1.1 Пример . . . . .	45
4.2 Все против всех (all-vs-all) . . . . .	48
4.2.1 Пример . . . . .	49

## 1 Наводящие размышления

Здравствуйте, уважаемые слушатели. К настоящему моменту мы уже изучили некоторые приемы работы с данными: мы научились определять наиболее важные признаки, сокращать размерность данных, строить различные виды классификаторов, проводить регрессию, оценивать точность построенной модели и многое-многое другое. В то же время, при решении той или иной задачи, мы всегда исходили из следующей парадигмы: возьмем конкретный метод (или алгоритм), настроим его параметры на исходных данных, оценим точность и, если она неплоха, вуаля – задача решена. Но насколько описанный подход является правильным и эффективным?

К чему это мы ведем, спросите вы? А к тому, что ни один из методов машинного обучения, конечно же, не является полностью универсальным. Каждый из них хорошо работает на конкретных данных, в конкретной ситуации, и надеяться, что он сработает, причем хорошо, на любых данных, достаточно наивно. Да и большинство алгоритмов разработано в предположении, что данные удовлетворяют достаточно большому набору весьма синтетических условий, что на практике встречается, увы, редко.

В этой лекции мы будем говорить об ансамблях, или композициях моделей. Идея ансамблей совершенно не нова и знакома каждому из нас, а на бытовом уровне может быть описана следующим образом: «собери как можно больше мнений об одном и том же от разных специалистов, а затем, основываясь на новых данных, согласно какому-то правилу прими окончательное решение». В этой фразе, применительно к машинному обучению, специалисты – это различные алгоритмы, обученные на одних и тех же данных, а правило – это окончательное решение, принятое по набору предсказанных ответов. Подробнее обо всем этом мы поговорим чуть позже.

Еще один из способов понять идею ансамблей – басня о слепцах и слоне<sup>1</sup>, в которой у каждого из слепцов имеется свое описание слона, причем верное, но очень однобокое. Куда лучше было бы собраться вместе и обсудить их расхождения до того, как прийти к окончательному выводу.

Итак, мотивации к изучению вроде бы достаточно, но есть одно но: для обучения разных моделей, нам желательно иметь разные данные, что бывает редко, ведь обычно мы имеем дело лишь с одной выборкой. Но может быть из этой выборки мы можем получать какие-то другие, так называемые псевдовыборки? С обсуждения этого вопроса, относящегося чисто к статистике, мы и начнем.

---

<sup>1</sup>Elephant and blind sages by Blanca Martí for Equilibre.  
<https://wildequus.org/2014/05/07/sufi-story-blind-men-elephant/>

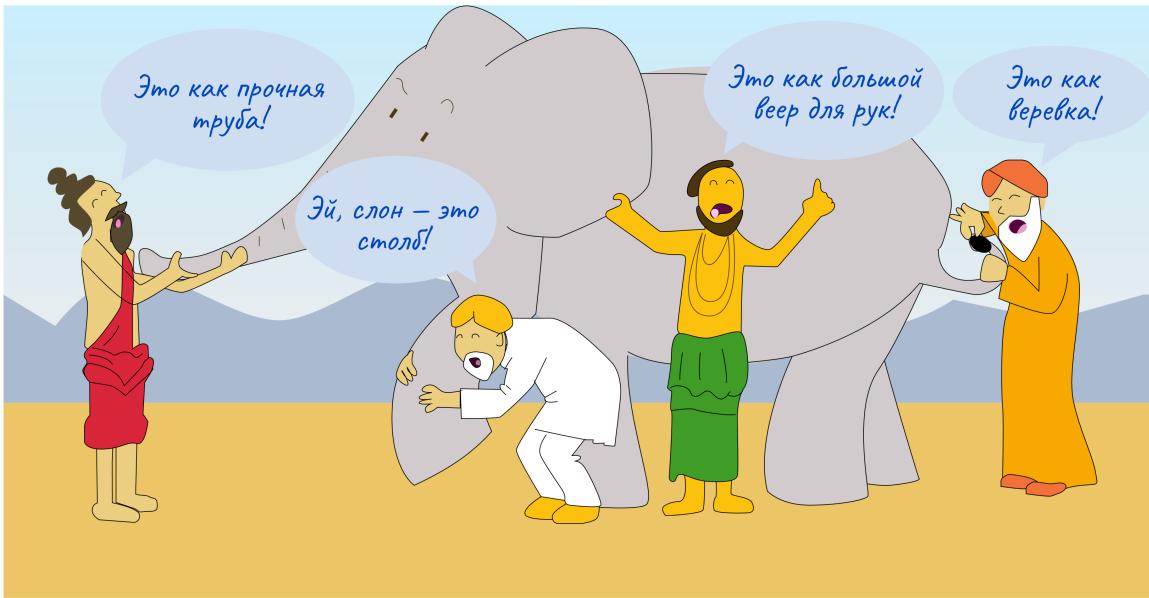


Рис. 1: The Blind Men and the Elephant.

## 2 Повторные выборки (Ресемплинг)

### 2.1 Введение

Итак, как мы уже поняли, многие методы машинного обучения основаны на аппарате математической статистики. Но чем же занимается статистика? Статистика, как мы знаем, занимается оценкой различных характеристик генеральной совокупности. Естественно, чтобы что-то делать с этими характеристиками на практике, получать их конкретные значения, изучать, сравнивать и применять в конкретных прикладных задачах, статистический аппарат нуждается в хлебе – в «проявлениях» генеральной совокупности – в выборке. Методы ресемплинга, о которых пойдет речь далее, как раз-таки помогают (что, наверное, понятно и из названия) получать какие-то выборки. Но какие?

Перед тем как сказать какие (и привести примеры конкретных методов), давайте все-таки начнем сначала и ответим себе на вопрос: а что такое методы ресемплинга, и зачем они нужны? Методы ресемплинга, или методы получения повторных выборок, – это статистические методы, позволяющие из данной нам выборки сгенерировать некоторым образом множество новых выборок. Зачем? Это хороший вопрос, который мы подробно осветим чуть позже, а сейчас ответим чисто практически – ради более высокой точности оценки. Ведь дальнейший план таков: на каждой полученной выборке мы вычисляем интересующую нас статистику, полученные значения как-то усредняем и в итоге получаем величину, которая оказывается не хуже, чем та, что просто вычислена на всей исходной выборке.

По сути дела, методы ресемплинга – это методы моделирования поведения генеральной совокупности на основе какой-то выборки из нее. Еще раз подчеркнем: в отличие от методов, обсуждаемых ранее, новые выборки берутся не из генеральной совокупности, а формируются из исходной выборки, взятой из генеральной совокупности. Кстати, этот прием для нас не нов. Кросс-валидация, о которой велись разговоры ранее – это ровно-таки один из методов ресемплинга.

Теперь поподробнее остановимся на вопросе: зачем? Чем плоха исходная выборка? В принципе, ничем не плоха, но на практике то и дело сталкиваешься со следующими проблемами (перечислим только некоторые из них):

1. Ошибочность предположения о распределении генеральной совокупности. Многие методы статистики основаны на каких-то предположениях о распределении генеральной совокупности. Но если предположение неверно или размер выборки мал (что может случаться по совершенно разным причинам), то параметрические методы, основанные на теоретических хорошо изученных распределениях, вряд ли сильно нам помогут. В такой ситуации может быть вообще лучше рассмотреть так называемую «непараметрическую» модель.
2. Неслучайные выборки. Классическое, «стерильное» предположение многих методов статистики – это случайность выборки. Однако бывают ситуации, когда выборка вовсе не случайна, ведь на практике их получать намного быстрее и даже дешевле (как, например, «self-selected samples»). Ну и как тут применять изученные математические догмы?
3. Маленький объем выборки. Для получения адекватных результатов во многих статистических методах требуется, чтобы объем выборки был достаточно велик (ведь результаты обычно асимптотические, когда  $n \rightarrow +\infty$ ). Если же выборка имеет маленьких объем, то методы либо не работают вовсе, либо очень неточны.
4. Невозможность в явном виде вычислить интересующие характеристики той или иной статистики.

Чтобы осветить последний пункт более подробно, остановимся вот на каком чисто теоретическом примере. Пусть  $X_1, X_2, \dots, X_n$  – выборка объема  $n$  из генеральной совокупности, имеющей равномерное распределение  $U_{0,\theta}$  на отрезке  $[0, \theta]$ , где  $\theta > 0$ . Как оценить параметр  $\theta$ ?

Как известно, математическое ожидание случайной величины  $\xi$ , имеющей равномерное распределение  $U_{a,b}$  на отрезке  $[a, b]$  – это

$$E\xi = \frac{a+b}{2}.$$

Значит, для случайной величины  $\xi$ , имеющей распределение  $U_{0,\theta}$ ,

$$E\xi = \frac{\theta}{2} \Rightarrow \theta = 2E\xi.$$

В то же время прекрасно известно, что хорошей оценкой математического ожидания случайной величины, имеющей какое угодно распределение, является выборочное среднее  $\bar{X}$ . Тогда, в качестве оценки параметра  $\theta$  можно предложить следующую статистику:

$$\hat{\theta} = 2\bar{X}.$$

Для написанной статистики можно вычислить все основные характеристики: математическое ожидание (или среднее), дисперсию, среднеквадратическое отклонение, можно построить асимптотический доверительный интервал и многое-многое другое.

**Замечание 2.1.1** *На самом деле более хорошей оценкой (оценкой максимального правдоподобия) параметра  $\theta$  является  $n$ -ый член вариационного ряда  $X_{(n)}$ , то есть*

$$\hat{\theta} = X_{(n)},$$

*для которой также можно вычислить все перечисленные выше характеристики, но это технически сложнее и не пригодится нам в канве дальнейшего изложения.*

С другой стороны, если использовать достаточно известный метод моментов, то для параметра  $\theta$  можно предложить и другие оценки, например такую:

$$\hat{\theta} = \sqrt{3 \cdot \bar{X}^2} = \sqrt{3 \cdot \frac{X_1^2 + \dots + X_n^2}{n}}.$$

Для написанной статистики аналитически получить математическое ожидание или дисперсию – задача совершенно не простая. Тут и пригождаются методы ресемплинга. Методы, которые мы будем рассматривать – это джекнайф («складной нож», англ. Jackknife) и бутстрэп (от англ. bootstrap).

## 2.2 Джекнайф

### 2.2.1 Немного о самом методе

Метод Джекнайф – это один из методов ресемплинга. Изначально его разработал Морис Анри Кенуилль (Maurice Henri Quenouille) для корректировки смещения статистики  $\hat{\theta}$  для малого числа  $n$ . Лишь позже Джон Тьюки

(John Wilder Tukey) обнаружил, что метод может применяться для построения достаточно точных характеристик статистики  $\hat{\theta}$ , даже для построения доверительных интервалов. Именно поэтому метод носит название (которое и предложил Тьюки) того самого швейцарского ножа — универсального инструмента, заменяющего собой целый набор приспособлений. В своей сути метод складного ножа призван заменить различные частные методы и свести их к одному.

Итак, перейдем непосредственно к методу. Предположим, что в результате эксперимента мы наблюдаем выборку  $X_1, X_2, \dots, X_n$  объема  $n$  из генеральной совокупности  $\xi$ .

**Замечание 2.2.1** *Под выборкой  $X_1, X_2, \dots, X_n$  из генеральной совокупности  $\xi$  мы классически понимаем набор независимых (в совокупности) случайных величин, одинаково распределенных с  $\xi$ .*

Теперь опишем, как множатся выборки при использовании метода складного ножа. Начнем с основного определения.

**Определение 2.2.1** *Джекнайф-выборками  $X_{[1]}, X_{[2]}, \dots, X_{[n]}$  называются наборы, формируемые из исходной выборки следующим образом:*

$$X_{[i]} = (X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n).$$

Иными словами,  $i$ -ая джекнайф-выборка получается из исходной удалением  $i$ -го элемента. Обратите внимание: вместо одной выборки мы получили сразу  $n$  псевдовыборок.

Пусть теперь  $\theta$  — это некоторая характеристика генеральной совокупности, а

$$\hat{\theta} = \hat{\theta}_n(X_1, X_2, \dots, X_n)$$

— ее оценка по выборке  $X_1, X_2, \dots, X_n$ . Введем напрашивающееся определение.

**Определение 2.2.2** *Частичной оценкой  $\hat{\theta}_{(-i)}$ ,  $i \in \{1, 2, \dots, n\}$  параметра  $\theta$  на джекнайф-выборке  $X_{[i]}$  называется статистика*

$$\hat{\theta}_{(-i)} = \hat{\theta}_{n-1}(X_{[i]}) = \hat{\theta}_{n-1}(X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n).$$

Итак, вместо одной оценки интересующей характеристики по исходной выборке, мы получили  $n$  частичных оценок. Поясним сказанное выше на примере.

**Пример 2.2.1** *Пусть параметр  $\theta$  — математическое ожидание генеральной совокупности. Как обычно, не имея никаких дополнительных сведений*

о распределении, в качестве его оценки резонно рассмотреть выборочное среднее

$$\hat{\theta} = \overline{X_n} = \frac{1}{n} \sum_{i=1}^n X_i.$$

Тогда частичная оценка  $\hat{\theta}_{(-i)}$ ,  $i \in \{1, 2, \dots, n\}$  параметра  $\theta$  на джекнайф-выборке  $X_{[i]}$  – это статистика

$$\hat{\theta}_{(-i)} = \overline{(X_{[i]})_{n-1}} = \frac{1}{n-1} \sum_{j=1, j \neq i}^n X_j.$$

Полученная статистика есть не что иное, как выборочное среднее  $i$ -ой джекнайф-выборки.

Для удобства дальнейших выкладок, введем следующее обозначение для среднего арифметического частичных оценок:

$$\hat{\theta}_{(\bullet)} = \frac{1}{n} \sum_{i=1}^n \hat{\theta}_{(-i)}.$$

Для того чтобы дать определение джекнайф-оценки, введем понятие псевдозначения.

**Определение 2.2.3** Величина

$$\tilde{\theta}_i = n\hat{\theta}_n - (n-1)\hat{\theta}_{(-i)}, \quad i \in \{1, 2, \dots, n\}$$

называется  $i$ -ым псевдо значением параметра  $\theta$ , построенным по джекнайф-выборке  $X_{[i]}$ .

Теперь должно быть понятно, как определяется джекнайф-оценка.

**Определение 2.2.4** Джалкнайф-оценкой  $\hat{\theta}_{jack}$  параметра  $\theta$  называется величина

$$\hat{\theta}_{jack} = \frac{1}{n} \sum_{i=1}^n \tilde{\theta}_i.$$

**Замечание 2.2.2** Перепишем последнее определение в несколько другой форме. Так как  $\tilde{\theta}_i = n\hat{\theta}_n - (n-1)\hat{\theta}_{(-i)}$ , то

$$\hat{\theta}_{jack} = \frac{1}{n} \sum_{i=1}^n \tilde{\theta}_i = \frac{1}{n} \sum_{i=1}^n \left( n\hat{\theta}_n - (n-1)\hat{\theta}_{(-i)} \right) =$$

$$= n\hat{\theta}_n - \frac{n-1}{n} \sum_{i=1}^n \hat{\theta}_{(-i)} = n\hat{\theta}_n - (n-1)\hat{\theta}_{(\bullet)}.$$

В итоге, джекнайф-оценка параметра  $\theta$  – это  $n$  исходных оценок на выборке  $X_1, X_2, \dots, X_n$  минус  $(n-1)$  среднее арифметическое частичных оценок.

**Пример 2.2.2** Рассмотрим пример вычисления оценки методом джекнайф на конкретной выборке. Пусть  $X = (1, 4, 7, 9)$  – выборка объема 4,  $\theta$  – неизвестное математическое ожидание генеральной совокупности. Оценим его, используя выборочное среднее  $\bar{X}$ , то есть  $\hat{\theta} = \bar{X}$ . Тогда получим, что

$$\hat{\theta} = \frac{1+4+7+9}{4} = 5.25.$$

Джекнайф-выборка  $X_{[1]}$  состоит из следующих элементов:

$$X_{[1]} = (4, 7, 9),$$

а частичная оценка, посчитанная по нему, равна:

$$\hat{\theta}_{(-1)} = \hat{\theta}(X_{[1]}) = \frac{4+7+9}{3} = \frac{20}{3}.$$

Тогда первое псевдозначение может быть найдено, как

$$\tilde{\theta}_1 = 4 \cdot 5.25 - 3 \cdot \frac{20}{3} = 1.$$

Аналогично определяются оставшиеся джекнайф-выборки:

$$X_{[2]} = (1, 7, 9), \quad X_{[3]} = (1, 4, 9), \quad X_{[4]} = (1, 4, 7),$$

затем по ним вычисляются частичные оценки

$$\hat{\theta}_{(-2)} = \frac{17}{3}, \quad \hat{\theta}_{(-3)} = \frac{14}{3}, \quad \hat{\theta}_{(-4)} = 4,$$

и, в конце концов, по ним вычисляются и псевдозначения:

$$\tilde{\theta}_2 = 4, \quad \tilde{\theta}_3 = 7, \quad \tilde{\theta}_4 = 9,$$

Как мы видим, полученные псевдозначения совпадают с элементами исходной выборки, поэтому оценка  $\hat{\theta}_{jack}$  будет совпадать с исходной оценкой  $\hat{\theta} = \bar{X}$ :

$$\hat{\theta}_{jack} = \frac{1}{4} \sum_{i=1}^n \tilde{\theta}_i = \frac{1+4+7+9}{4} = 5.25 = \bar{X}.$$

Оказывается, справедлив и более общий факт: выборочное среднее инвариантно относительно метода джекнайф.

**Пример 2.2.3** Пусть  $\theta$  – неизвестное математическое ожидание генеральной совокупности. Для его оценки, как уже было сказано много раз, удобно использовать выборочное среднее  $\bar{X}$ . Тогда

$$\begin{aligned}\hat{\theta}_n &= \bar{X}, \\ \hat{\theta}_{(\bullet)} &= \frac{1}{n} \sum_{i=1}^n \hat{\theta}_{(-i)} = \frac{1}{n} \sum_{i=1}^n \frac{1}{n-1} \sum_{j=1, j \neq i}^n X_j = \\ &= \frac{1}{n(n-1)} ((n-1)X_1 + (n-1)X_2 + \dots + (n-1)X_n) = \frac{1}{n} \sum_{i=1}^n X_i = \bar{X},\end{aligned}$$

откуда

$$\hat{\theta}_{jack} = n\hat{\theta}_n - (n-1)\hat{\theta}_{(\bullet)} = n\bar{X} - (n-1)\bar{X} = \bar{X}.$$

и джекнайф-оценка совпадает с исходной.

На самом деле легко понять, что справедливо и более общее утверждение.

**Лемма 2.2.1** Пусть дана выборка  $X_1, X_2, \dots, X_n$  и функция  $f : \mathbb{R} \rightarrow \mathbb{R}$ . Если оценка параметра  $\theta$  имеет вид

$$\hat{\theta} = \hat{\theta}_n(X_1, X_2, \dots, X_n) = \frac{1}{n} \sum_{i=1}^n f(X_i),$$

то

$$\hat{\theta}_{jack} = \hat{\theta}.$$

**Доказательство.** Доказательство проводится напрямую. Проведем его, чтобы еще раз повторить введенные обозначения.

$$\begin{aligned}\hat{\theta}_{(\bullet)} &= \frac{1}{n} \sum_{i=1}^n \hat{\theta}_{(-i)} = \frac{1}{n} \sum_{i=1}^n \frac{1}{n-1} \sum_{j=1, j \neq i}^n f(X_j) = \\ &= \frac{1}{n(n-1)} ((n-1)f(X_1) + (n-1)f(X_2) + \dots + (n-1)f(X_n)) = \\ &= \frac{1}{n} \sum_{i=1}^n f(X_i) = \hat{\theta}_n.\end{aligned}$$

Но тогда

$$\hat{\theta}_{jack} = n\hat{\theta}_n - (n-1)\hat{\theta}_{(\bullet)} = n\hat{\theta}_n - (n-1)\hat{\theta}_n = \hat{\theta}_n.$$

□

Из этой теоремы следует, что для оценок, скажем, моментов случайной величины, метод складного ножа не дает ничего нового, ведь оценка  $k$ -ого момента имеет вид

$$\hat{\theta} = \hat{\theta}_n(X_1, X_2, \dots, X_n) = \frac{1}{n} \sum_{i=1}^n X_i^k$$

и попадает под условие теоремы, если взять  $f(x) = x^k$ .

В то же время, если рассматриваемая статистика  $\hat{\theta}$  имеет более сложную структуру, метод джекнайф позволяет уменьшить ее смещение, если оно есть. Поговорим об этом подробнее.

### 2.2.2 Джекнайф-оценка смещения

Итак, приступим к изучению свойств джекнайф-оценки. Так как

$$\hat{\theta}_{jack} = n\hat{\theta}_n - (n-1)\hat{\theta}_{(\bullet)},$$

то резонно рассмотреть величину

$$\hat{\theta}_{jack} - \hat{\theta}_n = (n-1)(\hat{\theta}_n - \hat{\theta}_{(\bullet)}),$$

которая показывает отклонение выбранной оценки от джекнайф-оценки.

**Определение 2.2.5** Величина

$$\widehat{\text{bias}}_{jack} = (n-1)(\hat{\theta}_n - \hat{\theta}_{(\bullet)})$$

называется джекнайф-оценкой смещения статистики  $\hat{\theta}_n$ .

Логично полагать, что если изначальная оценка была несмещенная, то и получившаяся оценка должна оставаться несмещенной. Это и правда так.

**Лемма 2.2.2** Пусть  $\hat{\theta}_n$  – несмещенная оценка параметра  $\theta$ , то есть

$$\mathbb{E}_\theta \hat{\theta}_n = \theta.$$

Тогда  $\hat{\theta}_{jack}$  – тоже несмещенная оценка параметра  $\theta$ .

**Доказательство.** Так как

$$\hat{\theta}_{jack} - \hat{\theta}_n = (n-1)(\hat{\theta}_n - \hat{\theta}_{(\bullet)}) = \widehat{\text{bias}}_{jack},$$

то

$$\widehat{\theta}_{jack} = \widehat{\theta}_n + \widehat{\text{bias}}_{jack}.$$

Тогда

$$\mathsf{E}_\theta \widehat{\theta}_{jack} = \mathsf{E}_\theta \widehat{\theta}_n + \mathsf{E}_\theta \widehat{\text{bias}}_{jack} = \theta + \mathsf{E}_\theta \widehat{\text{bias}}_{jack},$$

где последнее равенство справедливо в силу несмещенности оценки  $\widehat{\theta}_n$ . Но

$$\begin{aligned} \mathsf{E}_\theta \widehat{\text{bias}}_{jack} &= \mathsf{E}_\theta \left( (n-1)(\widehat{\theta}_n - \widehat{\theta}_{(\bullet)}) \right) = (n-1)(\mathsf{E}_\theta \widehat{\theta}_n - \mathsf{E}_\theta \widehat{\theta}_{(\bullet)}) = \\ &= (n-1) \left( \theta - \mathsf{E}_\theta \widehat{\theta}_{(\bullet)} \right). \end{aligned}$$

В то же время,

$$\mathsf{E}_\theta \widehat{\theta}_{(\bullet)} = \mathsf{E}_\theta \left( \frac{1}{n} \sum_{i=1}^n \widehat{\theta}_{(-i)} \right) = \frac{1}{n} \sum_{i=1}^n \mathsf{E}_\theta \widehat{\theta}_{(-i)} = \theta.$$

Значит, в предположении несмещенности  $\widehat{\theta}_n$ , получаем, что  $\mathsf{E}_\theta \widehat{\text{bias}}_{jack} = 0$ . А тогда  $\mathsf{E}_\theta \widehat{\theta}_{jack} = \theta$ .  $\square$

Что же происходит с оценкой, если она смещенная? Давайте для примера рассмотрим оценку  $S^2$  выборочной дисперсии

$$S^2 = \frac{1}{n} \sum_{i=1}^n (X_i - \bar{X})^2.$$

Как мы знаем, она смещенная, причем если дисперсия генеральной совокупности  $D\xi$  равна  $\sigma^2$ , то

$$\mathsf{E} S^2 = \frac{n-1}{n} \sigma^2,$$

а значит смещение, то есть разность между математическим ожиданием оценки и истинным значением параметра, равно

$$\mathsf{E} S^2 - \sigma^2 = -\frac{\sigma^2}{n},$$

и смещение убывает с ростом объема выборки  $n$  со скоростью  $n^{-1}$ . Теперь вычислим смещение оценки, построенной методом складного ножа. Так как

$$\begin{aligned} \mathsf{E}_\theta \widehat{\text{bias}}_{jack} &= (n-1) \left( \mathsf{E}_\theta \widehat{\theta}_n - \mathsf{E}_\theta \widehat{\theta}_{(\bullet)} \right) = (n-1) \left( \frac{n-1}{n} \sigma^2 - \frac{1}{n} \sum_{i=1}^n \frac{n-2}{n-1} \sigma^2 \right) = \\ &= (n-1) \sigma^2 \left( \frac{n-1}{n} - \frac{n-2}{n-1} \right) = \frac{\sigma^2}{n}, \end{aligned}$$

то в итоге

$$\mathbb{E}_\theta \widehat{\theta}_{jack} = \mathbb{E}_\theta \widehat{\theta}_n + \mathbb{E}_\theta \widehat{\text{bias}}_{jack} = \frac{n-1}{n} \sigma^2 + \frac{\sigma^2}{n} = \sigma^2.$$

И какой вывод? А такой, что джекнайф-оценка, построенная по оценке  $S^2$  оказывается несмешенной. Можно показать, что она совпадает с несмешенной оценкой  $S_0^2$ , равной, как известно,

$$\widehat{\theta}_{jack} = S_0^2 = \frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X})^2.$$

**Пример 2.2.4** Пусть известно, что генеральная совокупность  $\xi$  имеет распределение  $U_{\theta, \theta+1}$  с неизвестным параметром  $\theta$ . Тогда оценка параметра  $\theta$  (согласно методу максимального правдоподобия) по выборке  $X_1, X_2, \dots, X_n$  задается следующим аналитическим выражением:

$$\widehat{\theta} = \widehat{\theta}_n(X_1, X_2, \dots, X_n) = X_{(1)},$$

где  $X_{(1)}$  – первый член построенного по выборке вариационного ряда. Пусть  $\theta = 2.5$ , а набор

$$(2.60, 3.26, 2.75, 2.64, 2.83, 2.58, 3.17, 3.31, 3.48, 3.14),$$

– выборка объема 10 из равномерного распределения  $U_{2.5, 3.5}$ , элементы которой окружены до сотых. По данной выборке, согласно выбранной статистике,  $\theta$  оценивается через минимальный элемент исходной выборки:

$$\widehat{\theta} = X_{(1)} = 2.58.$$

Теперь сформируем  $n = 10$  джекнайф-наборов. Первый набор  $X_{[1]}$  будет состоять из следующих элементов (исключен первый элемент из исходной выборки):

$$(3.26, 2.75, 2.64, 2.83, 2.58, 3.17, 3.31, 3.48, 3.14),$$

Найдем частичную оценку  $\widehat{\theta}_{(-1)}$ , которая, по сути, выдает минимальный элемент набора  $X_{[1]}$ . Тогда

$$\widehat{\theta}_{(-1)} = 2.58.$$

Выполнив аналогичные шаги для оставшихся девяти псевдовыборок получим, что джекнайф-оценка будет равна:

$$\widehat{\theta}_{jack} = n\widehat{\theta}_1 - (n-1)\frac{1}{n} \sum_{i=1}^n \widehat{\theta}_{(-i)} \approx 2.56.$$

Как видим, полученная джекнайф-оценка оказалась ближе к истинному значению  $\theta = 2.5$ , нежели исходная.

Проведем моделирование для выборок большего объема. Из рисунка 2 можно заметить, что обе оценки приближаются к истинному значению  $\theta$  с ростом объема выборки  $n$ . Однако исходная оценка  $\hat{\theta} = X_{(1)}$  (синие точки на рисунке) всегда оказывается больше истинного значения, равного 2.5 (ему отвечает зеленая прямая). Джекнайф-оценки же располагаются по разные стороны от истинного значения и в большинстве своем находятся ближе к истинному значению, чем исходные.

В итоге мы исправили смещение со «всегда положительного» до «меньшего и разного по знаку». Смысл проделанного может быть описан на таком бытовом примере: если весы в магазине ошибаются, но всегда в пользу покупателя (то есть показывают меньшую массу, чем есть на самом деле), то при каждой покупке магазин несет убытки. Если же весы периодически «играют» в пользу магазина, а периодически – в пользу покупателя, то в среднем не проигрывает никто.

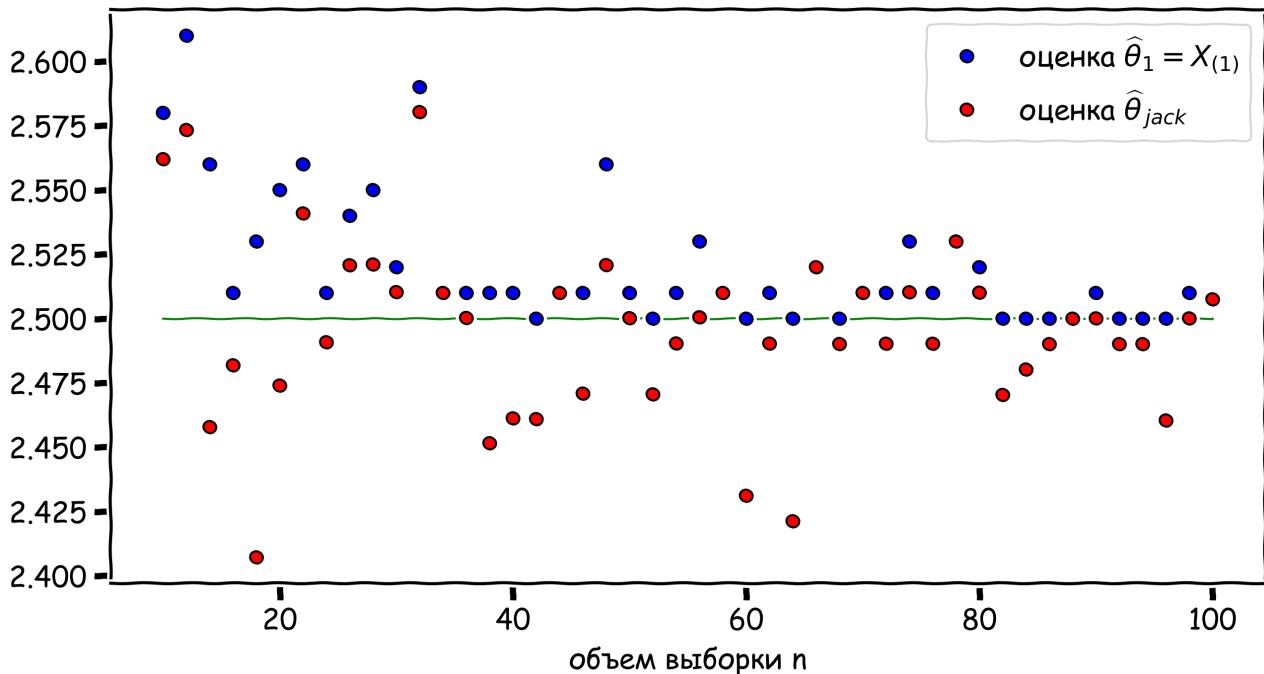


Рис. 2: Зависимость оценок  $\hat{\theta}_1$  и  $\hat{\theta}_{(jack)}$  от объема выборки  $n$

Оказывается, справедливо и более общее утверждение.

**Теорема 2.2.1** Пусть  $\hat{\theta} = \hat{\theta}_n(X_1, X_2, \dots, X_n)$  – оценка параметра  $\theta$ , причем

$$\mathbb{E}_\theta \hat{\theta} = \theta + \frac{a_1(\theta)}{n} + \frac{a_2(\theta)}{n^2} + O(n^{-3}).$$

Тогда

$$\mathbb{E}_\theta \widehat{\theta}_{jack} = \theta - \frac{a_2(\theta)}{n(n-1)} + O(n^{-2}).$$

Иными словами, теорема утверждает, что если смещение  $\mathbb{E}_\theta \widehat{\theta} - \theta$  убывало с ростом  $n$  со скоростью  $n^{-1}$ , то смещение джекнайф-оценки станет убывать со скоростью  $n^{-2}$ .

### Доказательство.

Вычислим

$$\mathbb{E}_\theta \widehat{\text{bias}}_{jack} = (n-1) \left( \mathbb{E}_\theta \widehat{\theta}_n - \mathbb{E}_\theta \widehat{\theta}_{(\bullet)} \right).$$

Так как

$$\mathbb{E}_\theta \widehat{\theta}_{(-i)} = \theta + \frac{a_1(\theta)}{n-1} + \frac{a_2(\theta)}{(n-1)^2} + O(n^{-3}),$$

то

$$\mathbb{E}_\theta \widehat{\theta}_{(\bullet)} = \frac{1}{n} \sum_{i=1}^n \left( \theta + \frac{a_1(\theta)}{n-1} + \frac{a_2(\theta)}{(n-1)^2} + O(n^{-3}) \right) = \theta + \frac{a_1(\theta)}{n-1} + \frac{a_2(\theta)}{(n-1)^2} + O(n^{-3}).$$

Тогда

$$\begin{aligned} \mathbb{E}_\theta \widehat{\text{bias}}_{jack} &= (n-1) \left( a_1(\theta) \left( \frac{1}{n} - \frac{1}{n-1} \right) + a_2(\theta) \left( \frac{1}{n^2} - \frac{1}{(n-1)^2} \right) + O(n^{-3}) \right) = \\ &= -\frac{a_1(\theta)}{n} - \frac{(2n-1)a_2(\theta)}{n^2(n-1)} + O(n^{-2}). \end{aligned}$$

В итоге,

$$\begin{aligned} \mathbb{E}_\theta \widehat{\theta}_{jack} &= \mathbb{E}_\theta \widehat{\theta} + \mathbb{E}_\theta \widehat{\text{bias}}_{jack} = \\ &= \theta + \frac{a_1(\theta)}{n} + \frac{a_2(\theta)}{n^2} + O(n^{-3}) - \frac{a_1(\theta)}{n} - \frac{(2n-1)a_2(\theta)}{n^2(n-1)} + O(n^{-2}) = \\ &= \theta - \frac{a_2(\theta)}{n(n-1)} + O(n^{-2}). \end{aligned}$$

□

Итак, джекнайф-оценка позволяет корректировать смещение, ускоряя его стремление к нулю.

**Замечание 2.2.3** Полезно также заметить, что условие на математическое ожидание, сформулированное в теореме, встречается на практике достаточно часто. В частности, как уже упоминалось,

$$S^2 = \frac{n-1}{n} \sigma^2 = \sigma^2 - \frac{\sigma^2}{n}$$

удовлетворяет условию теоремы. В данном случае  $a_1(\theta) = -\sigma^2$  (так как параметром является именно  $\sigma$ , ну или  $\sigma^2$ ), а  $a_2(\theta) = 0$ . Как мы убедились выше, джекнайф-оценка, построенная на основе  $S^2$ , оказывается несмещенной и совпадает с  $S_0^2$ .

### 2.2.3 Джекнайф-оценка дисперсии

Оценим дисперсию  $\widehat{\theta}_n$ . Ясно, что несмещенная выборочная дисперсия псевдозначений, определяемых равенствами

$$\tilde{\theta}_i = n\widehat{\theta}_n - (n-1)\widehat{\theta}_{(-i)},$$

равна

$$S_0^2(\tilde{\theta}) = \frac{1}{n-1} \sum_{i=1}^n (\tilde{\theta}_i - \widehat{\theta}_{jack})^2,$$

так как

$$\widehat{\theta}_{jack} = \frac{1}{n} \sum_{i=1}^n \tilde{\theta}_i.$$

**Определение 2.2.6** Джекнайф-оценкой дисперсии  $\widehat{\theta}_n$  называют величину

$$\widehat{\text{Var}}_{jack} = \frac{S_0^2(\tilde{\theta})}{n} = \frac{1}{n(n-1)} \sum_{i=1}^n (\tilde{\theta}_i - \widehat{\theta}_{jack})^2.$$

**Замечание 2.2.4** Заметим, что в случае, когда  $\theta$  – это математическое ожидание генеральной совокупности, а в качестве его оценки берется  $\widehat{\theta} = \widehat{\theta}_n(X_1, X_2, \dots, X_n) = \overline{X}$ , то

$$\tilde{\theta}_i = (X_1 + X_2 + \dots + X_n) - (X_1 + \dots + X_{i-1} + X_{i+1} + \dots + X_n) = X_i.$$

Кроме того, как было вычислено ранее,

$$\widehat{\theta}_{jack} = \overline{X},$$

а значит

$$\widehat{\text{Var}}_{jack} = \frac{1}{n(n-1)} \sum_{i=1}^n (X_i - \overline{X})^2 = \frac{\sigma^2}{n} = D_\theta \overline{X}.$$

Оказывается, выражение для джекнайф-оценки дисперсии может быть переписано следующим образом:

$$\widehat{\text{Var}}_{jack} = \frac{1}{n(n-1)} \sum_{i=1}^n (\tilde{\theta}_i - \widehat{\theta}_{jack})^2 = \frac{n-1}{n} \sum_{i=1}^n \left( \widehat{\theta}_{(-i)} - \widehat{\theta}_{(\bullet)} \right)^2.$$

Без дополнительных пояснений отметим, что в случае, если  $\theta_n$  удовлетворяет достаточно регулярным условиям, например, если  $\theta_n$  – гладкая функция выборочного среднего  $\bar{X}$ , то джекнайф-оценка дисперсии  $\widehat{\theta}_n$  является состоятельной. Если же изначальная статистика не является гладкой, как, например, выборочная медиана, то джекнайф использовать совершенно бесполезно – его оценки почти наверняка будут несостоятельны.

## 2.3 Бутстрэп

Давайте рассмотрим еще один из методов генерации повторных псевдовыборок – бутстрэп. Этот метод был предложен в 1979 году Брэдли Эфроном (Bradley Efron), и с тех пор заработал заслуженную популярность. Он, как и джекнайф, позволяет оценивать различные характеристики рассматриваемой статистики такие, как смещение и дисперсию, позволяет строить доверительные интервалы и так далее. В то же время, границы применимости этого метода намного шире, чем метода джекнайф, естественно, за счет большей вычислительной сложности.

### 2.3.1 Общая идея оценок методом бутстрэп

Итак, попробуем самостоятельно, из логических соображений прийти к методу бутстрэп. Пусть, как обычно,  $\theta$  – некоторый неизвестный параметр (или характеристика) распределения генеральной совокупности  $\xi$ ,  $X_1, X_2, \dots, X_n$  – выборка из генеральной совокупности  $\xi$ , а

$$\widehat{\theta} = \widehat{\theta}_n(X_1, X_2, \dots, X_n)$$

– некоторая (состоятельная) оценка  $\theta$ . На конкретной выборке (в результате эксперимента) мы можем получить лишь конкретное (одно) значение нашей оценки  $\widehat{\theta}$ , а значит никак не можем оценить ни среднее, ни разброс, ни, тем более, построить какой-либо доверительный интервал для  $\theta$ . Как же поступать в таком случае?

Сначала представим себе «идеальную ситуацию». Пусть мы можем получить не одну выборку из генеральной совокупности, а, скажем,  $B$  штук. Тогда у нас в руках  $B$  наборов

$$X_1^{(j)}, X_2^{(j)}, \dots, X_n^{(j)}, \quad j \in \{1, 2, \dots, B\}$$

независимых и одинаково (с  $\xi$ ) распределенных случайных величин, на каждом из которых мы можем вычислить значение нашей статистики  $\widehat{\theta}$

$$\widehat{\theta}^j = \widehat{\theta}_n^j(X_1^{(j)}, X_2^{(j)}, \dots, X_n^{(j)}), \quad j \in \{1, 2, \dots, B\}.$$

Совершенно ясно, что разумной оценкой как математического ожидания, так и дисперсии  $\widehat{\theta}$  являются

$$\widehat{\theta}_{(\bullet),B} = \frac{1}{B} \sum_{j=1}^B \widehat{\theta}^j$$

и

$$\widehat{\text{Var}}_B = \frac{1}{B-1} \sum_{j=1}^B \left( \widehat{\theta}^j - \frac{1}{B} \sum_{j=1}^B \widehat{\theta}^j \right)^2 = \frac{1}{B-1} \sum_{j=1}^B \left( \widehat{\theta}^j - \widehat{\theta}_{(\bullet),B} \right)^2.$$

Все сказанное ранее подтверждается и теоретически – изложенное, по сути своей, – это хорошо известный в статистике метод Монте-Карло. Все бы было хорошо, но перед нами маячит вот какая проблема: обычно мы не можем получать сколько угодно выборок из генеральной совокупности. Что же в таком случае делать? Как применить, вроде бы, совершенно логичные рассуждения, которые мы привели? Тут и возникает бутстрэп.

### 2.3.2 Метод бутстрэп

Метод бутстрэп основывается вот на какой идее. Пусть  $X_1, X_2, \dots, X_n$  – выборка. Оираясь на нее, мы можем смоделировать распределение  $\xi$  и в последствии получить необходимые выборки из смоделированного распределения. Такой подход позволит нам применить аппарат, описанный ранее.

Как известно из статистики, истинное распределение  $\xi$  по выборке  $X_1, X_2, \dots, X_n$  разумно приближается эмпирическим с функцией распределения

$$F_n^*(x) = \frac{1}{n} \sum_{i=1}^n \mathbf{I}(X_i < x),$$

где  $\mathbf{I}(A)$  – индикатор события  $A$ , аналитическое задание которого следующее:

$$\mathbf{I}(A) = \begin{cases} 1, & A \text{ произошло} \\ 0, & A \text{ не произошло} \end{cases}.$$

Имея же функцию распределения, мы легко можем моделировать выборки нужного нам объема. Но чему, по сути дела, это эквивалентно? Конечно, формированию выборок путем случайного выбора  $n$  объектов с возвращением из исходной выборки.

**Определение 2.3.1** *Бутстрэп-выборками объема  $n$  из выборки  $X_1, X_2, \dots, X_n$  называются наборы*

$$X_1^{(j)}, X_2^{(j)}, \dots, X_n^{(j)},$$

где  $X_i^{(j)} \in \{X_1, X_2, \dots, X_n\}$ .

Понятно, что число таким образом полученных выборок чрезвычайно велико! Индекс  $j$  может принимать значения из диапазона  $\{1, 2, \dots, n^n\}$  и, например, для выборки объема 12 количество бутстрэп-выборок равно

$$12^{12} = 8\ 916\ 100\ 448\ 256$$

– числу, даже названия разрядов которого знает далеко не каждый. Поэтому на практике, конечно, берут куда меньшее бутстрэп-выборок, чем существует на самом деле.

**Замечание 2.3.1** Так как выбор производится с возвращением, то в бутстрэп-выборках могут встречаться повторяющиеся элементы исходной выборки. Интересно, что с ростом  $n$  доля уникальных элементов исходной выборки, встречающихся в бутстрэп-выборке, стремится к  $\approx 0.632$ .

Пусть мы «набираем» выборку  $S_1, S_2, \dots, S_n$ . Для каждого элемента исходной выборки  $X_1, X_2, \dots, X_n$  вероятность, что  $S_i \neq X_j$  равна

$$\mathbb{P}(S_i \neq X_j) = 1 - \frac{1}{n}.$$

Так как выбор производится независимо, то

$$\mathbb{P}(S_1 \neq X_j, S_2 \neq X_j, \dots, S_n \neq X_j) = \left(1 - \frac{1}{n}\right)^n \xrightarrow[n \rightarrow +\infty]{} e^{-1} \approx 0.368.$$

Значит, вероятность того, что элемент  $X_j$  встретится в набираемой нами выборке равна

$$1 - e^{-1} \approx 0.632,$$

и это справедливо для любого элемента исходной выборки, то есть при всех  $j \in \{1, 2, \dots, n\}$ .

Итак, после генерации бутстрэп-выборок, алгоритм получения оценок математического ожидания и дисперсии оценки  $\hat{\theta}$  следующий.

1. Пусть сгенерировано  $B$  бутстрэп-выборок

$$X_1^{(j)}, X_2^{(j)}, \dots, X_n^{(j)}, \quad j \in \{1, 2, \dots, B\}.$$

2. На каждой из них вычисляется значение  $\hat{\theta}$ ,

$$\hat{\theta}^j = \hat{\theta}_n^j(X_1^{(j)}, X_2^{(j)}, \dots, X_n^{(j)}), \quad j \in \{1, 2, \dots, B\}.$$

3. В качестве оценок математического ожидания и дисперсии  $\hat{\theta}$  берутся

$$\hat{\theta}_{(\bullet),B} = \frac{1}{B} \sum_{j=1}^B \hat{\theta}^j$$

и

$$\widehat{\text{Var}}_B = \frac{1}{B-1} \sum_{j=1}^B \left( \hat{\theta}^j - \frac{1}{B} \sum_{j=1}^B \hat{\theta}^j \right)^2 = \frac{1}{B-1} \sum_{j=1}^B \left( \hat{\theta}^j - \hat{\theta}_{(\bullet),B} \right)^2,$$

соответственно.

**Определение 2.3.2** Введенные выше величины

$$\hat{\theta}_{(\bullet),B} = \frac{1}{B} \sum_{j=1}^B \hat{\theta}^j$$

и

$$\widehat{\text{Var}}_B = \frac{1}{B-1} \sum_{j=1}^B \left( \hat{\theta}^j - \frac{1}{B} \sum_{j=1}^B \hat{\theta}^j \right)^2 = \frac{1}{B-1} \sum_{j=1}^B \left( \hat{\theta}^j - \hat{\theta}_{(\bullet),B} \right)^2,$$

называют бутстрэп-оценками математического ожидания и дисперсии оценки  $\hat{\theta}$ , соответственно.

**Пример 2.3.1** Рассмотрим уже знакомый пример, когда генеральная совокупность имеет распределение  $U_{\theta, \theta+1}$  с параметром  $\theta = 2.5$ . Выполним моделирование и найдем истинное значения дисперсии оценки, а также джекнайф и бутстрэп-оценки дисперсии  $\hat{\theta} = X_{(1)}$ . Можно заметить, что при малых значениях  $n$  результаты варьируются, однако зачастую джекнайф-оценка дисперсии меньше. И наоборот, начиная с некоторого  $n$ , бутстрэп-оценка дисперсии меньше или совпадает с джекнайф-оценкой дисперсии. Практика показывает, что метод джекнайф лучше показывает себя на малых выборках, а бутстрэп предпочтителен для больших объемов данных.

### 2.3.3 Построение доверительных интервалов

Построение доверительных интервалов для  $\theta$  опирается на ровно те же самые идеи, что озвучены ранее. Рассмотрим здесь лишь самый простой случай. Если допустить, что

$$\frac{\hat{\theta} - \theta}{\sqrt{D_{\theta} \hat{\theta}}} \xrightarrow[n \rightarrow +\infty]{d} \eta \sim N_{0,1},$$

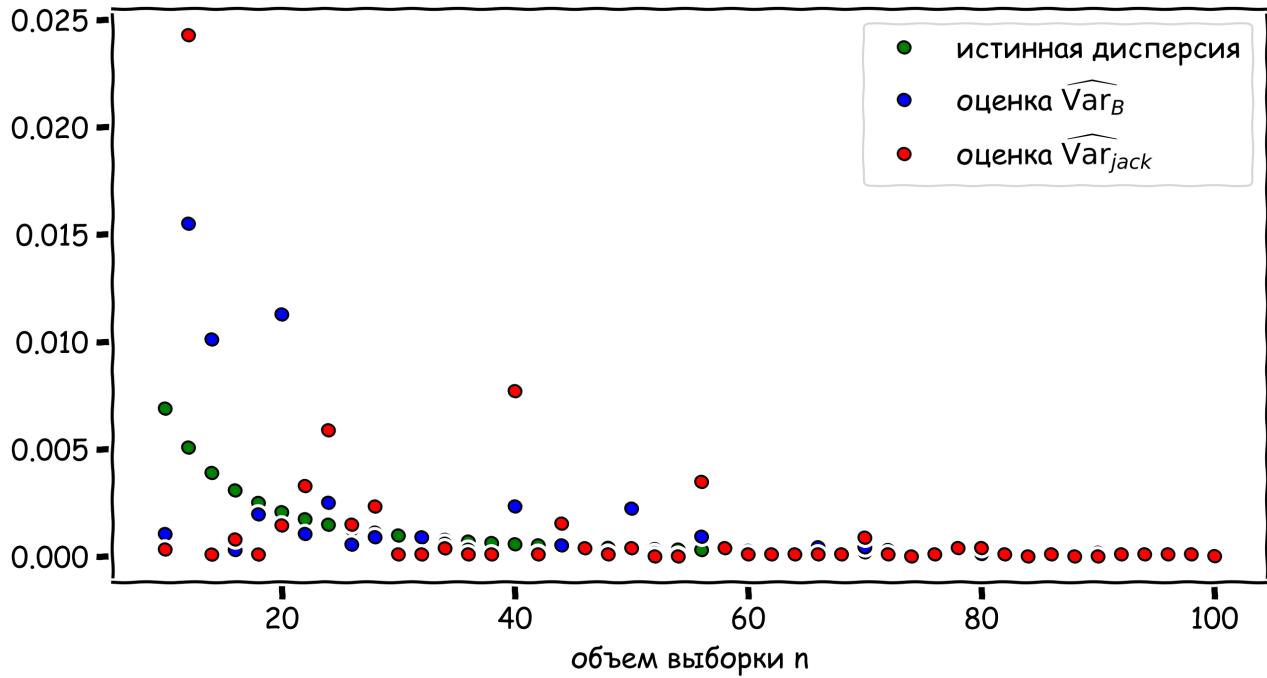


Рис. 3: Зависимость оценок дисперсии  $\widehat{\text{Var}}_{\text{jack}}$  и  $\widehat{\text{Var}}_B$  от объема выборки  $n$

или близко к нормальному, а  $\widehat{\text{Var}}_B$  – состоятельная оценка  $D_\theta \widehat{\theta}$ , то асимптотический доверительный интервал уровня  $(1 - \varepsilon)$  для параметра  $\theta$  строится обычным образом и имеет вид

$$\left( \widehat{\theta} - \tau_{1-\varepsilon/2} \sqrt{\widehat{\text{Var}}_B}, \widehat{\theta} + \tau_{1-\varepsilon/2} \sqrt{\widehat{\text{Var}}_B} \right),$$

где  $\tau_{1-\varepsilon/2}$  – квантиль уровня  $(1 - \varepsilon/2)$  стандартного нормального распределения.

### 2.3.4 Некоторые финальные замечания

Рассмотрению методов ресемплинга можно посвятить не одну лекцию. Рассмотренные нами моменты и подходы – это лишь часть тех бонусов, что дает исследователю бутстрэп. Аналогично тому, как было сделано в методе джекнайф, бутстрэп может оценивать и улучшать смещение исходной оценки  $\widehat{\theta}$  и многое другое.

Кроме того, все, что обсуждалось ранее, относится к непараметрическому бутстрэпу – мы не выдвигаем никаких предположений о распределении  $\xi$ . В то же время, если известно, каким образом распределение  $\xi$  зависит от интересующего нас параметра, метод может быть существенно усилен. Бутстрэп выборки в этом случае генерируются из эмпирической функции распределения, в которую вместо неизвестного параметра  $\theta$  может быть подставлена

его оценка. В этом случае количество бутстрэп выборок может быть сколь угодно большим.

Итак, теперь мы готовы посмотреть, как изученные методы применяются на практике к построению ансамблей моделей.

## 3 Ансамбли моделей

### 3.1 Общее понятие ансамбля

Как мы уже упоминали ранее, ансамбль моделей – это подход машинного обучения, в котором несколько моделей, называемых еще часто **слабыми учениками**, обучаются для решения одинаковой задачи, а затем объединяются для получения лучших результатов. И, конечно, главный вопрос здесь – как объединять таких учеников?

Итак, предположим, что рассматривается обучение с учителем, и нам дана (конкретная) обучающая выборка  $x_1, x_2, \dots, x_n$  с откликами  $y_1, y_2, \dots, y_n$ , где  $x_i \in X, y_i \in Y, i \in \{1, 2, \dots, n\}$ .

**Замечание 3.1.1** *Немного поясним введенные обозначения. Например, если  $X_1, X_2, \dots, X_p$  – числовые предикторы (ситуация, которая у нас обычно и возникает), то в качестве множества  $X$  резонно рассматривать  $\mathbb{R}^p$ . Конкретное наблюдение тогда может быть отождествлено с вектором*

$$x_i = (x_{i1}, x_{i2}, \dots, x_{ip}) \in \mathbb{R}^p.$$

*При рассмотрении задачи классификации, множество  $Y$  обычно имеет вид  $Y = \{1, 2, 3, \dots, M\}$ , где  $M$  – количество классов (конечно, классы могут быть занумерованы как-то иначе). При рассмотрении же задачи регрессии,  $Y$  часто либо совпадает с  $\mathbb{R}$ , либо является его «непрерывным» подмножеством.*

Итак, давайте дадим определение слову «алгоритм», используемому в дальнейшем.

**Определение 3.1.1** *Пусть  $b(x) : X \rightarrow R$  – базовый алгоритм, обучающийся на тренировочном наборе данных,  $R$  – вспомогательное множество, а  $C : R \rightarrow Y$  – так называемое решающее правило. Алгоритмом, основанном на базовом алгоритме  $b$  с решающим правилом  $C$ , назовем отображение*

$$a(x) = C(b(x)).$$

Поясним, что имеется в виду под базовым алгоритмом, а также что такое множество  $R$  на примерах.

**Пример 3.1.1** Пусть, например,  $Y = \{-1, 1\}$ , и решается задача двухклассовой классификации. Предположим, что  $b(x)$  – базовый алгоритм – алгоритм логистической регрессии. Тогда  $b(x)$  выдает число в диапазоне от 0 до 1, то есть вероятность отнесения объекта к классу «+1», и множество  $R$  – это  $[0, 1]$ . Если выбрать порог отсечения, равный 0.5, то решающее правило может быть следующим: «если  $b(x) \geq 0.5$ , то присвоим наблюдению класс «+1», иначе – класс «-1»». Итого, алгоритм на основе логистической регрессии с выбранным решающим правилом может быть описан следующим образом:

$$a(x) = \begin{cases} 1, & b(x) \geq 0.5 \\ -1, & b(x) < 0.5 \end{cases}.$$

Решающее правило может быть выбрано и как-то иначе – это, конечно же, задача исследователя.

**Замечание 3.1.2** Например, с граничным значением 0.5 из предыдущего примера поступать можно по-разному. Часто используется следующий подход – добавление дополнительного класса «0» для объектов, которые непонятно как классифицировать. В этом случае алгоритм действует по правилу «лучше промолчать, чем соврать» и задается, например, так

$$a(x) = \begin{cases} 1, & b(x) > 0.5 \\ 0, & b(x) = 0.5 \\ -1, & b(x) < 0.5 \end{cases}.$$

Впрочем, выбор порога отсечения, равного 0.5, тоже весьма субъективно.

Приведем еще один пример.

**Пример 3.1.2** Пусть решается задача двухклассовой классификации,  $Y = \{-1, 1\}$ , а  $b(x)$  – базовый алгоритм – алгоритм SVM. В этом случае  $R = \mathbb{R}$ , так как  $b(x)$  выдает, вообще говоря, произвольные числа, а решающее правило с фиктивным классом «0» может быть задано так: «если  $b(x) > 0$ , то наблюдению  $x$  присваивается класс «+1», если  $b(x) < 0$ , то «-1», а если  $b(x) = 0$ , то «0»». Аналитически алгоритм может быть задан так:

$$a(x) = \text{sign } b(x) = \begin{cases} 1, & b(x) > 0 \\ 0, & b(x) = 0 \\ -1, & b(x) < 0 \end{cases}.$$

**Пример 3.1.3** Пусть, например, решается задача  $M$ -классовой классификации,  $Y = \{1, 2, \dots, M\}$ ,  $b_i(x)$  – снова базовый алгоритм – алгоритм логистической регрессии, выдающий вероятность отнесения объекта к классу с номером  $i \in \{1, 2, \dots, M\}$  (то, что  $b_i(x) = p \in [0, 1]$  означает, что объект  $x$  относится к классу  $i$  с вероятностью  $p$ , и не относится к классу  $i$  с вероятностью  $(1 - p)$ ). Вспомогательное множество  $R$  – это снова отрезок  $[0, 1]$ . Тогда решающее правило может быть, например, таким: «отнести объект к тому классу, базовый алгоритм отнесения к которому выдает наибольшее число». Иными словами,

$$a(x) = \arg \max_{i \in \{1, 2, \dots, M\}} b_i(x).$$

В случае, если подходит несколько классов, можно либо отнести наблюдение к любому из подходящих классов, либо отнести его к фиктивному классу «0» (конечно, введя его в рассмотрение).

**Пример 3.1.4** Пусть решается задача регрессии,  $b(x)$  – базовый алгоритм множественной линейной регрессии. Тогда решающее правило, вообще говоря, не нужно (оно является тождественным отображением), и

$$a(x) = C(b(x)) = b(x),$$

а значит  $R = \mathbb{R}$ .

Теперь, зная, что такое алгоритм, можно определить и ансамбль (или композицию) алгоритмов.

**Определение 3.1.2** Пусть  $b_i(x) : X \rightarrow R$  – базовые алгоритмы,  $i \in \{1, 2, \dots, k\}$ . Пусть, кроме того,  $F : R^k \rightarrow R$  – корректирующее правило и  $C : R \rightarrow Y$  – решающее правило. Тогда ансамблем алгоритмов  $b_1, \dots, b_k$  с корректирующим правилом  $F$  и решающим правилом  $C$  называется алгоритм

$$a(x) = C(F(b_1(x), b_2(x), \dots, b_k(x))).$$

Поясним введенное определение на примере.

**Пример 3.1.5** Пусть решается задача регрессии и  $b_i(x)$  – некоторый базовый алгоритм,  $i \in \{1, 2, \dots, k\}$  (например, алгоритмы полиномиальной регрессии с полиномами разных степеней). Тогда в качестве корректирующего правила можно рассмотреть, например, так называемое простое голосование (*Simple Voting*):

$$F(b_1(x), b_2(x), \dots, b_k(x)) = \frac{1}{k} \sum_{i=1}^k b_i(x).$$

– *базальное усреднение ответов*. В этом случае решающего правила не требуется (оно тождественно), а предсказание может даваться следующим алгоритмом:

$$a(x) = \frac{1}{k} \sum_{i=1}^k b_i(x).$$

В данном примере все алгоритмы имеют одинаковый вес, ни один не выделяется. Приведем еще один пример, обобщающий предыдущий.

**Пример 3.1.6** Пусть решается задача регрессии и  $b_i(x)$  – базовый алгоритм,  $i \in \{1, 2, \dots, k\}$ . Тогда в качестве корректирующего правила можно рассмотреть, например, так называемое *взвешенное голосование* (*Weighted Voting*). Пусть  $\omega_i$ ,  $i \in \{1, 2, \dots, n\}$  – веса, то есть  $\omega_i \geq 0$  и

$$\sum_{i=1}^k \omega_i = 1.$$

Тогда зададим корректирующее правило, как

$$F(b_1(x), b_2(x), \dots, b_k(x)) = \sum_{i=1}^k \omega_i b_i(x).$$

Решающее правило снова тождественно, а предсказание может даваться следующим алгоритмом:

$$a(x) = \sum_{i=1}^k \omega_i b_i(x).$$

Ясно, что в рассматриваемом примере больший вес имеет смысл сопоставлять алгоритму, которому мы больше «доверяем», или который по той или иной причине мы предпочитаем остальным.

**Пример 3.1.7** Пусть решается задача  $M$ -классовой классификации,  $Y = \{1, 2, \dots, M\}$  и  $a_i(x)$ ,  $i \in \{1, 2, \dots, k\}$  – алгоритмы, построенные на основе каких-то базовых алгоритмов, то есть алгоритмы, выдающие класс. Рассмотрим в качестве корректирующего правила так называемое голосование большинством:

$$F(a_1(x), a_2(x), \dots, a_k(x)) = \text{mode}(a_1(x), a_2(x), \dots, a_k(x)),$$

где **mode** выдает моду набора данных – значение, которое встречается наиболее часто. Если таких значений несколько, то в качестве значения  $F$

можно либо взять какую-то из мод, либо присвоить фиктивный класс «0». Тогда ансамбль (с тождественным решающим правилом) можно определить, как

$$a(x) = \text{mode}(a_1(x), a_2(x), \dots, a_k(x)).$$

Аналогично рассмотренному ранее, можно предложить и взвешенное голосование. Без дополнительных пояснений отметим, что ансамбль может быть определен, как

$$a(x) = \arg \max_{m \in \{1, 2, \dots, M\}} \sum_{i=1}^k w_i \mathbf{1}(a_i(x) = m),$$

причем в случае, если подходит несколько классов, то можно поступать любым из способов, описанных ранее. Описанное правило выбирает тот класс, которому отвечает большая сумма, при этом решению каждого классификатора приписан некоторый вес, конечно же влияющий на всю сумму.

Приведенные корректирующие функции – далеко не все возможные, но часто встречающиеся. Выбор корректирующей функции – сложная задача, часто зависящая от предметной области.

Базовые алгоритмы и алгоритмы, на них построенные, могут быть получены как с использованием различных алгоритмов машинного обучения, так и с помощью разных обучающих выборок. Перейдем к рассмотрению конкретных алгоритмов, и начнем с бэггинга. Он как раз-таки и использует разобранный статистический аппарат бутстрэпа.

## 3.2 Бэггинг

Бэггинг (от англ. Bagging = Bootstrap **aggregating**) – один из подходов к формированию ансамблей, основанный на независимом обучении базовых алгоритмов на различных выборках. Опишем алгоритм формально, стараясь использовать ранее введенные обозначения.

1. Пусть  $X_1, X_2, \dots, X_n$  – выборка объема  $n$ ,  $b_1(x), b_2(x), \dots, b_t(x)$  – базовые алгоритмы.
2. По выборке  $X_1, X_2, \dots, X_n$  создать  $t$  независимых подвыборок

$$X_1^{(j)}, X_2^{(j)}, \dots, X_l^{(j)}, \quad j \in \{1, 2, \dots, t\}$$

объема  $l \leq n$  методом бутстрэп.

3. Для каждого  $j \in \{1, 2, \dots, t\}$  обучить базовый алгоритм  $b_j(x)$  на выборке  $X_1^{(j)}, X_2^{(j)}, \dots, X_l^{(j)}$ .

#### 4. Сформировать ансамбль

$$a(x) = C(F(b_1(x), b_2(x), \dots, b_t(x))).$$

**Замечание 3.2.1** Часто бэггинг используют для уменьшения дисперсии конкретного базового алгоритма  $b(x)$ . Для этого полагают  $b_1(x) = b_2(x) = \dots = b_t(x) = b(x)$  то есть, по сути, обучают один и тот же алгоритм, но на разных подвыборках исходной выборки. Все шаги описанного алгоритма сохраняются.

**Замечание 3.2.2** Отметим также, что выбор объема подвыборки  $l$  – отдельная задача. С одной стороны, получившаяся выборка должна быть представительной для обучения, а с другой – не позволять модели переобучиться. Кроме того, чем больше объем выборки, тем больше обучается модель. Если, к тому же, велико и число  $t$ , то вычислительных мощностей может не хватить и вовсе.

### 3.2.1 Пример решения задачи классификации

Применим бэггинг к решению задачи классификации. Рассмотрим урезанные до двух предикторов данные по футбольной статистике:  $X_1$  – количество ударов в створ ворот;  $X_2$  – процент владения мячом. Урезание проводится лишь для того, чтобы можно было удобно визуализировать получающиеся результаты на плоскости. Объем исходной выборки невелик, всего 34 наблюдения, а данные выглядят следующим образом (отклик – первый столбец таблицы – равен 1, если команда выиграла, и равен 0, если проиграла):

Победа или проигрыш	Количество ударов в створ	Процент владения мячом
1	7	40
0	0	60
0	3	43
1	4	57
...	...	...

Наша цель – имея данные о количестве ударов в створ ворот и проценте владения мячом, классифицировать команду в одну из двух групп: победившие или проигравшие. В качестве базового алгоритма будем рассматривать алгоритм, основанный на логистической регрессии. Результат применения логистической регрессии на исходных данных представлен на рисунке 4 – вся плоскость поделена на два класса синей прямой. Желтые объекты – это победы команд, а зеленые – проигрыши.

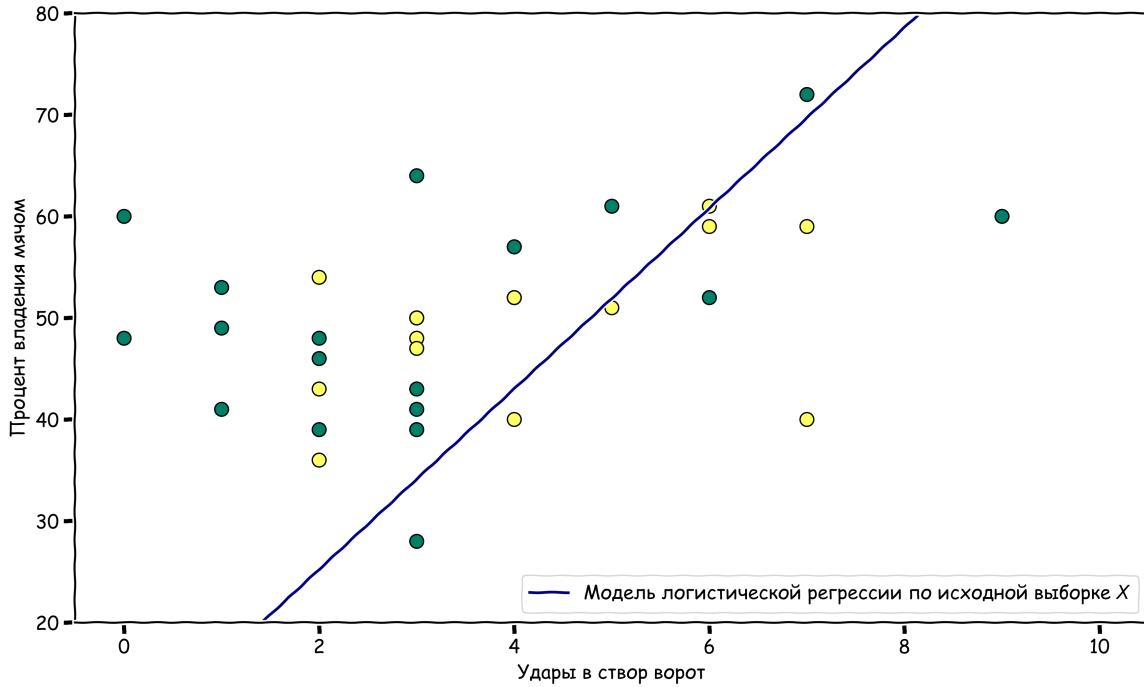


Рис. 4: Классификация с помощью логистической регрессии.

Теперь воспользуемся бэггингом, сформировав  $t = 5$  выборок (для наглядности визуализации) объема 34, выбрав в качестве ансамбля «голосование большинством», то есть

$$a(x) = \text{mode}(b_1(x), b_2(x), b_3(x), b_4(x), b_5(x)),$$

где

$$b_i(x) = \begin{cases} 1, & a_i(x) \geq 0.5 \\ 0, & a_i(x) < 0.5 \end{cases},$$

а  $a_i(x)$  – базовый алгоритм логистической регрессии, выдающий числовую вероятность в диапазоне  $[0, 1]$ .

Полученные результаты легко сравнить, если отобразить разделяющие плоскости прямые, полученные в результате логистической регрессии, на плоскости. На рисунке 5 видно, что результаты достаточно сильно отличаются. Нормали всех гиперплоскостей направлены в сторону класса желтых (вниз и вправо).

Рассмотрим новое тестовое наблюдение  $Z = (5, 65)$  (рисунок 6), отвечающее вот какой ситуации: команда ударила в створ ворот пять раз и владела мячом 65% игрового времени.

Сравним результат классификации с помощью логистической регрессии и построенного ансамбля. Алгоритм логистической регрессии дает вероятность

$$P_+ \approx 0.39$$

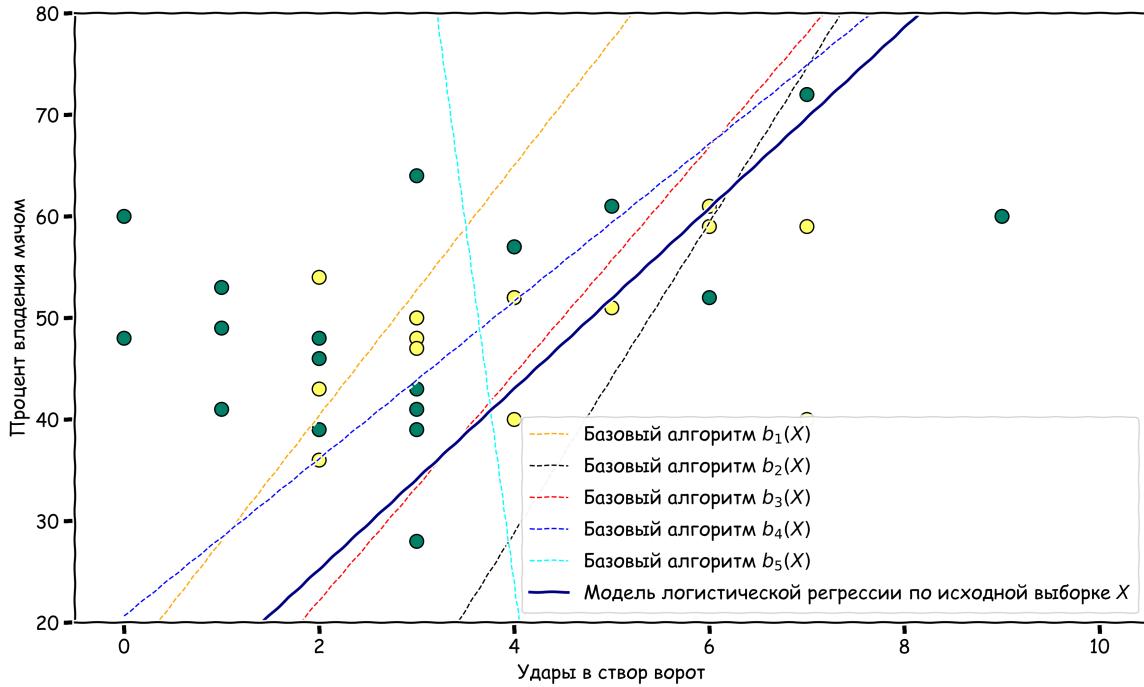


Рис. 5: Базовые алгоритмы на основе бэггинга

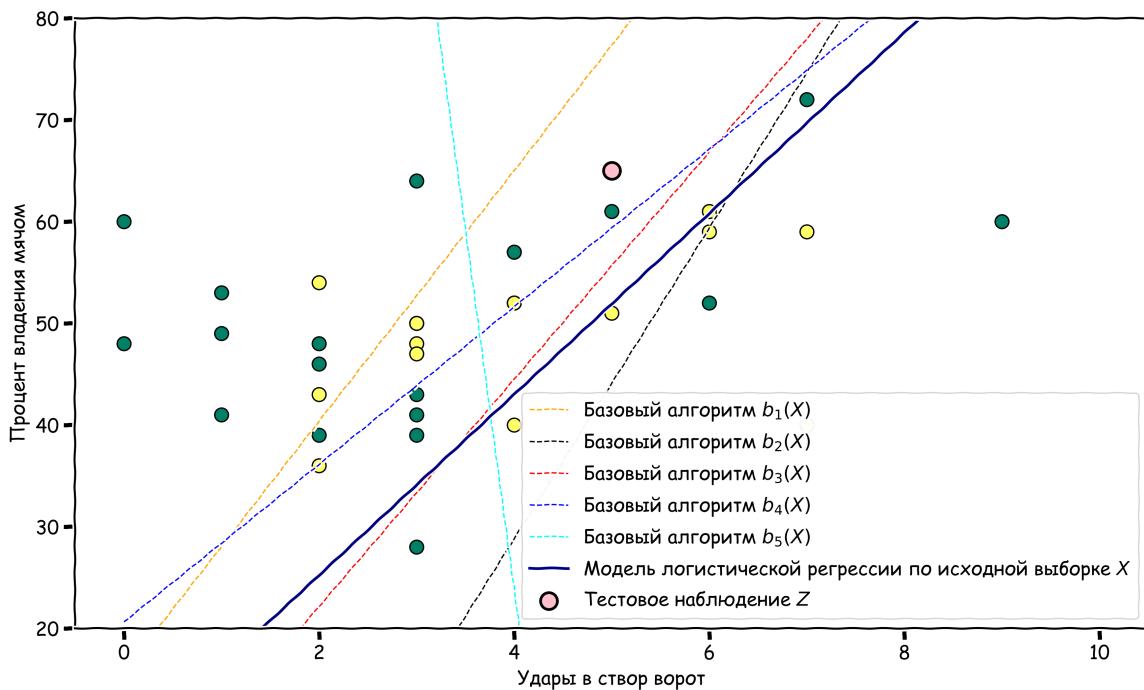


Рис. 6: Предсказание результата.

отнесения нового объекта к классу желтых, а значит, при пороге отсечения 0.5 относит его к классу зеленых (проигравших). Это понятно и из геометрических соображений – рассматриваемый объект лежит в той части плоскости, которая отвечает классу «зеленых».

Аналогичные вычисления выполняются для каждого базового алгорит-

ма  $b_i(Z)$ , а прогнозы равны

$$P_{b_1+} \approx 0.63, P_{b_2+} \approx 0.39, P_{b_3+} \approx 0.45, P_{b_4+} \approx 0.38, P_{b_5+} \approx 0.75.$$

В итоге, два алгоритма относят объект к классу желтых (1ый и 5ый), а остальные три – к классу зеленых. Это, опять же, понятно и без вычислений, просто из рисунка – по расположению тестового объекта относительно разделяющих прямых. Поскольку в качестве ансамбля выбрано «голосование большинством», то и ансамбль относит объект к классу зеленых – проигравших.

### 3.2.2 Пример задачи регрессии

Теперь приведем пример решения задачи регрессии с использованием ансамблей. В качестве тренировочного набора данных рассмотрим уже встречавшиеся нам ранее данные, показывающие время, проведенное в магазине, в зависимости от количества выбранных товаров. Данные представлены в виде таблицы. В качестве предиктора  $X_1$  выберем количество товаров, которое купил человек, а в качестве отклика  $Y$  – время, проведенное в магазине.

№	Время в магазине (мин.)	Количество выбранных товаров
1	10	15
2	5	12
3	12	18
4	25	30
5	1	3
6	18	20
7	11	14
8	7	10
9	19	20
10	15	13

Уравнение регрессии  $Y$  на  $X_1$  (с округленными коэффициентами) имеет следующий вид:

$$Y = 4.06 + 0.93X_1.$$

Соответствующая прямая построена на плоскости (рисунок 7), а предсказание необходимого времени для покупки 27 товаров вычисляется подстановкой значения  $X_1 = 27$  в полученную функцию, то есть:

$$4.06 + 0.93 \cdot 27 = 29.17.$$

Теперь, аналогично предыдущему примеру, воспользуемся бэггингом, где в качестве базовых алгоритмов выступает линейная регрессия, а в качестве

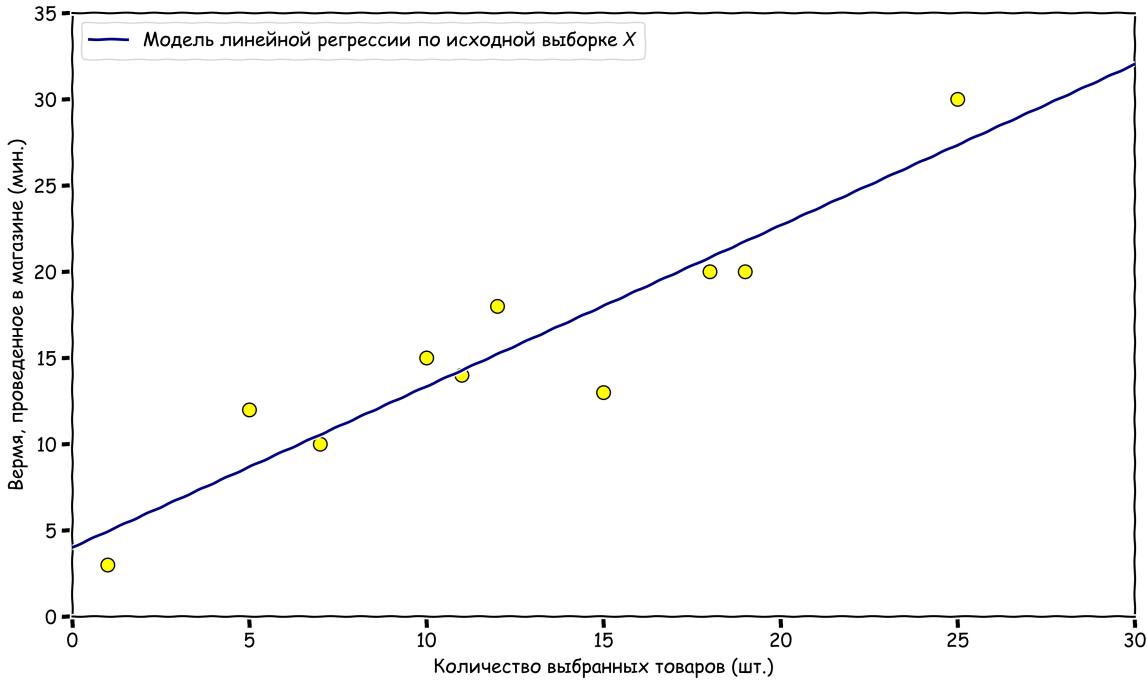


Рис. 7: Зависимость времени, проведенного в магазине, от количества выбранных товаров.

корректирующего правила выбрано простое голосование – усреднение результатов. Моделируем  $B = 1000$  бутстрэп выборок, для пяти из которых на рисунке 8 отображены прямые, соответствующие алгоритмам  $b_i(x)$ , каждый из которых может быть использован для предсказания.

Отметим на рисунке 9 вертикальной пунктирной прямой значение  $X_1 = 27$ , для которого будем искать предсказание. Так, базовый алгоритм  $b_2(x)$ , как видно из рисунка, дает предсказание около 24 минут. Из рисунка также виден значительный разброс результатов, от 24 до, примерно, 31 минуты.

Что же насчет среднего, среди всей тысячи результатов? Оно составит около 28.66 минут:

$$\frac{1}{1000} (31.21 + 23.94 + 27.48 + 30.84 + 25.49 + \dots) = 28.66,$$

что на пол минуты больше, чем прогноз, который нам давала исходная модель.

### 3.3 Адаптивный бустинг для двухклассовой классификации (AdaBoost)

Перейдем к рассмотрению еще одного метода построения ансамблей – бустингу. Основное отличие этого метода от ранее рассмотренного бэггинга заключается в том, что слабые ученики (базовые алгоритмы) обучаются теперь не независимо, а последовательно, учитывая опыт предыдущего. Так,

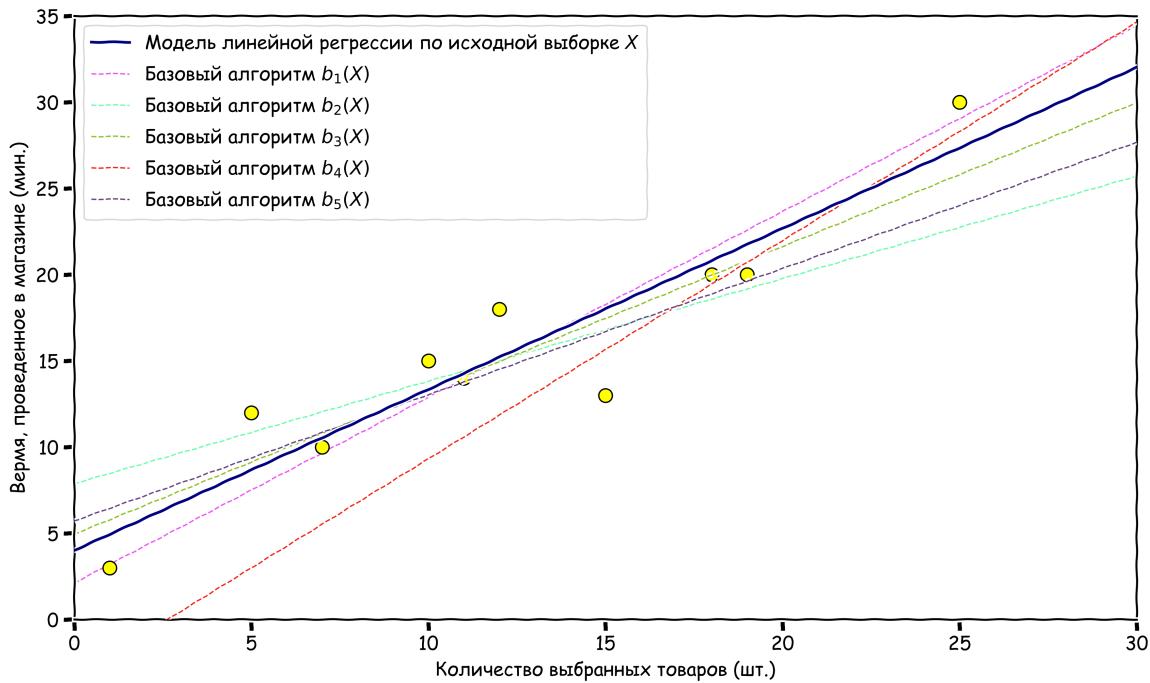


Рис. 8: Базовые алгоритмы на основе бутстрэпа.

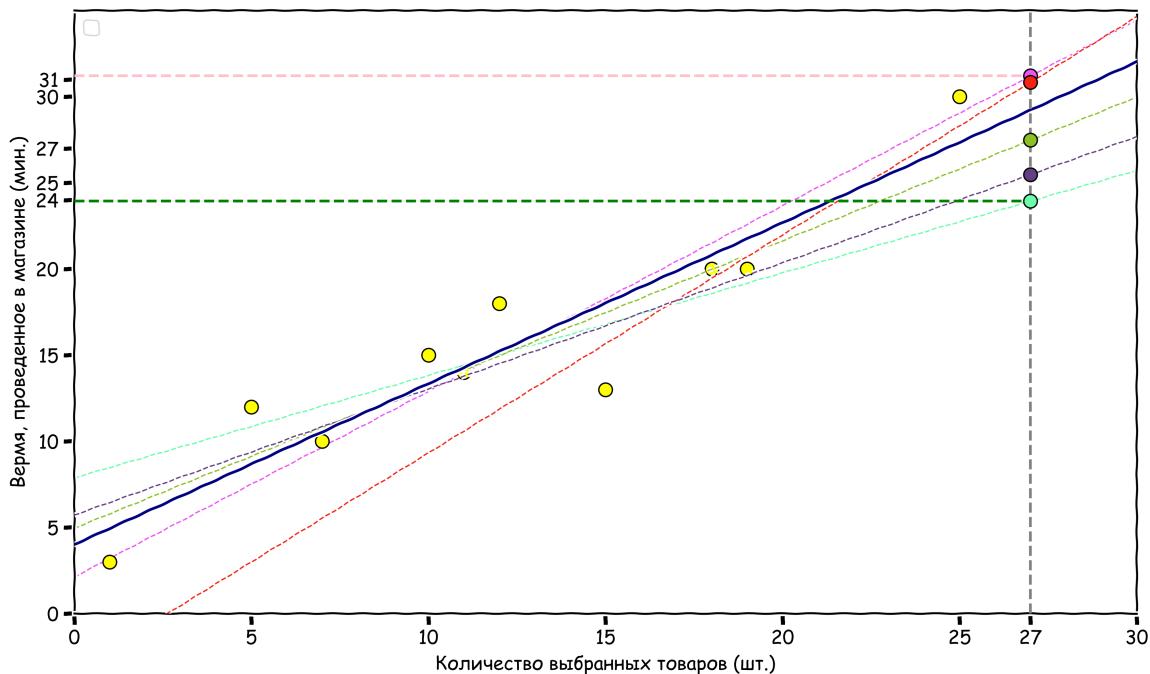


Рис. 9: Предсказание результата.

каждый последующий ученик начинает придавать большее значение тем наблюдениям из тренировочного набора данных, на которых ошиблись предыдущие ученики.

Рассмотрим алгоритм адаптивного бустинга, причем применительно только к задаче двухклассовой классификации. Итак, предположим, что

$Y = \{-1, 1\}$ ,  $x_1, x_2, \dots, x_n$  – выборка (тренировочные данные) объема  $n$ ,  $x_i \in X$ ,  $i \in \{1, 2, \dots, n\}$  и  $B$  – множество алгоритмов классификации, то есть

$$B = \{b \mid b : X \rightarrow \{-1, 1\}\}.$$

При построении ансамбля будем рассматривать корректирующее правило взвешенного голосования, тогда ансамбль может быть задан аналитическим выражением вида

$$a(x) = \text{sign} \left( \sum_{t=1}^T \omega_t b_t(x) \right),$$

где  $b_t(x) \in B$ ,  $\omega_t$  – веса (не обязательно нормированные),  $t \in \{1, 2, \dots, T\}$ ,  $T$  – количество обученных алгоритмов. Подбор же как алгоритмов классификации, так и весов перед ними, будем производить итеративно, исходя из требования минимизации количества ошибок на тренировочных данных. Количество ошибок может быть задано следующим аналитическим выражением:

$$Q_T = \sum_{i=1}^n \mathbf{I}(a(x_i) \neq y_i) = \sum_{i=1}^n \mathbf{I} \left( y_i \sum_{t=1}^T \omega_t b_t(x_i) < 0 \right),$$

где  $x_i \in X$ ,  $\mathbf{I}$  – индикатор. Действительно, каждое слагаемое в первом выражении равно единице только в том случае, когда ансамбль выдает класс, отличный от истинного класса тренировочного объекта. Конечно, наша задача – в минимизации функции  $Q_T$  на тренировочных данных, ведь чем меньше число ошибок, тем лучше.

Минимизация написанного выражения – задача затруднительная, так как написанная функция не является гладкой. Однако, если мы найдем какую-то «хорошую» функцию  $\tilde{Q}_T$ , для которой  $Q_T \leq \tilde{Q}_T$ , то, минимизируя последнюю, будет уменьшаться и исходная функция. Ясно, что так как

$$Q_T = \sum_{i=1}^n \mathbf{I} \left( y_i \sum_{t=1}^T \omega_t b_t(x_i) < 0 \right) \leq \sum_{i=1}^n \exp \left( -y_i \sum_{t=1}^T \omega_t b_t(x_i) \right) = \tilde{Q}_T,$$

то написанная функция  $\tilde{Q}_T$  подходит. Именно она и используется в алгоритме адаптивного бустинга. Перепишем ее в виде

$$\tilde{Q}_T = \sum_{i=1}^n \exp \left( -y_i \sum_{t=1}^{T-1} \omega_t b_t(x_i) \right) e^{-y_i \omega_T b_T(x_i)}$$

и обозначим

$$\alpha_i = \exp \left( -y_i \sum_{t=1}^{T-1} \omega_t b_t(x_i) \right).$$

Нормируем коэффициенты так, чтобы их сумма была равна единице, тогда получим

$$\alpha_0 = \sum_{i=1}^n \alpha_i, \quad \alpha_i = \frac{\alpha_i}{\alpha_0}, \quad i \in \{1, 2, \dots, n\}.$$

**Замечание 3.3.1** Последнее равенство следует понимать так: сначала вычисляется отношение  $\alpha_i/\alpha_0$ , а затем новое значение присваивается переменной  $\alpha_i$ . Часто это обозначают либо

$$\alpha_i \leftarrow \frac{\alpha_i}{\alpha_0}, \quad \text{либо } \alpha_i := \frac{\alpha_i}{\alpha_0}.$$

**Замечание 3.3.2** Полезно заметить, что выражение для  $\tilde{Q}_T$  переписывается в виде

$$\tilde{Q}_T = \sum_{i=1}^n \alpha_i e^{-y_i \omega_T b_T(x_i)}$$

и коэффициенты  $\alpha_i$  не зависят от алгоритма  $b_T$ , а зависят только от алгоритмов  $b_i$  с номерами  $i < T$ , а значит могут быть найдены итерационно.

Конечно, нужно пояснить, откуда берутся коэффициенты  $\omega_i$ ,  $i \in \{1, 2, \dots, T\}$ , но мы это сделаем непосредственно в алгоритме.

Итак, запишем алгоритм для тренировочного набора данных  $x_1, x_2, \dots, x_n$  объема  $n$ , заданного множества алгоритмов  $B$  и числа используемых алгоритмов  $T$  по шагам:

1. Инициализировать начальные веса следующим образом:

$$\alpha_1 = \alpha_2 = \dots = \alpha_n = \frac{1}{n}.$$

2. Для всех  $t \in \{1, 2, \dots, T\}$

- (a) Найти алгоритм  $b_t$  из множества  $B$ , минимизирующий количество ошибок на тренировочном наборе данных, с весами  $\alpha_1, \alpha_2, \dots, \alpha_n$ :

$$b_t = \arg \min_{b \in B} \sum_{i=1}^n \alpha_i \cdot \mathbb{I}(b(x_i) \neq y_i).$$

Если таких алгоритмов несколько, выбрать любой.

- (b) Пусть

$$\varepsilon_t = \sum_{i=1}^n \alpha_i \cdot \mathbb{I}(b_t(x_i) \neq y_i)$$

– найденное минимальное значение на алгоритме  $b_t$ . Положить

$$\omega_t = \frac{1}{2} \ln \frac{1 - \varepsilon_t}{\varepsilon_t}.$$

(c) Пересчитать  $\alpha_i = \alpha_i e^{-y_i \omega_t b_t(x_i)}$ ,  $i \in \{1, 2, \dots, n\}$ .

(d) Нормировать полученные коэффициенты:

$$\alpha_0 = \sum_{i=1}^n \alpha_i, \quad \alpha_i = \frac{\alpha_i}{\alpha_0}, \quad i \in \{1, 2, \dots, n\}.$$

3. Составить итоговый ансамбль в виде

$$a(x) = \text{sign} \left( \sum_{t=1}^T \omega_t b_t(x) \right).$$

Можно доказать, что при некоторых необременительных ограничениях на множество алгоритмов  $B$ , минимум  $\tilde{Q}_T$  достигается именно при таком выборе параметров  $\omega_i$ ,  $i \in \{1, 2, \dots, T\}$

### 3.3.1 Пример

Давайте рассмотрим следующий «игрушечный пример», чтобы пройтись по шагам описанного алгоритма. Мы опустим подбор оптимального алгоритма  $b_t$  на шаге 2а, подбирая его «на глаз», лишь для упрощения изложения.

Итак, имеется выборка объема  $n = 12$ , желтые точки соответствуют классу «+1», зеленые – классу «-1». Пусть базовые алгоритмы классификации способны разделять плоскость либо горизонтальными, либо вертикальными прямыми. Легко увидеть, что ни одна из таких прямых не разделит плоскость так, чтобы не было ошибок классификации. Пусть  $T = 3$ , тем самым наша задача – корректным образом построить три алгоритма классификации, снабдив каждый корректным весом. Рассмотрим подробно каждую итерацию:

1. На первом инициализируем значения  $\alpha_1, \alpha_2, \dots, \alpha_{12}$ , они равны

$$\alpha_1 = \frac{1}{12}, \quad \alpha_2 = \frac{1}{12}, \dots, \quad \alpha_{12} = \frac{1}{12}.$$

Пусть базовый алгоритм  $b_1(x)$  представлен горизонтальной линией (верхнюю полуплоскость он относит к зеленым, а нижнюю – к желтым). Мы видим, что два зеленых объекта классифицированы неверно, они выделены в таблице синим. Найдем  $\varepsilon_1$  из соотношения

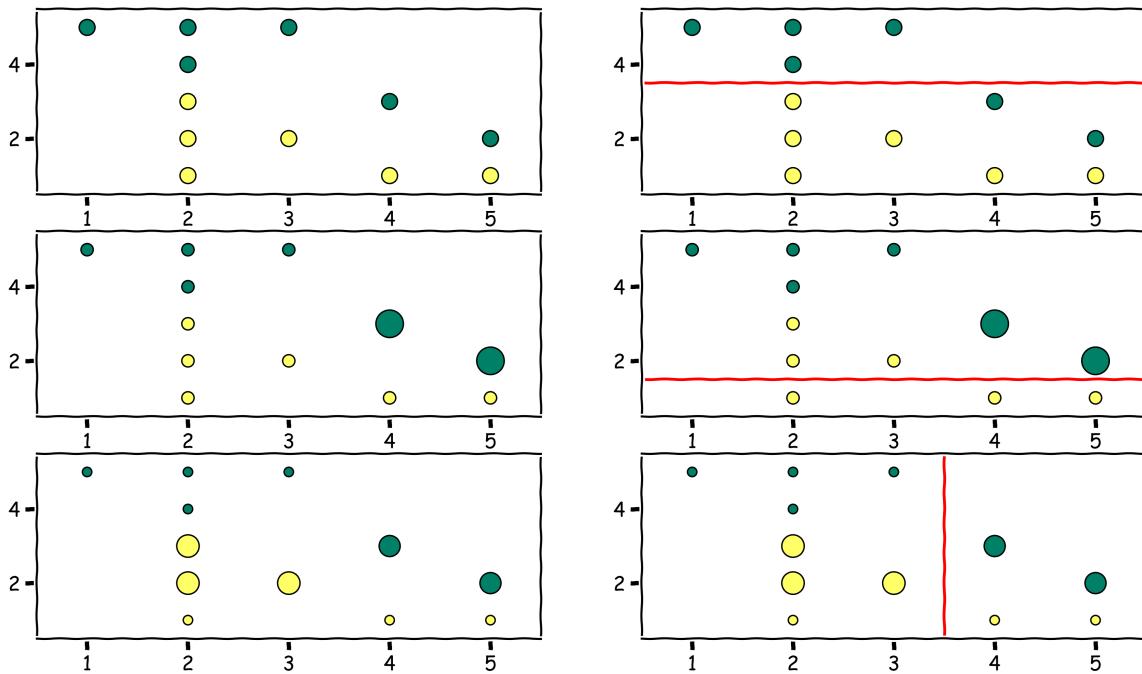


Рис. 10: Три итерации алгоритма AdaBoost.

$X$	(4, 1)	(2, 4)	(5, 1)	(3, 2)	(1, 5)	(3, 5)	(2, 1)	(2, 2)	(2, 5)	(4, 3)	(2, 3)	(5, 2)
$y_i$	+1	-1	+1	+1	-1	-1	+1	+1	-1	-1	+1	-1
$b_1(x_i)$	+1	-1	+1	+1	-1	-1	+1	+1	-1	+1	+1	+1

$$\varepsilon_1 = \frac{1}{12} \sum_{i=1}^{12} \mathbb{I}(b_1(x_i) \neq y_i) = \frac{1}{12} \cdot 2 \approx 0.167.$$

Теперь найдем значение  $\omega_1$ :

$$\omega_1 = \frac{1}{2} \ln \left( \frac{1 - \varepsilon_1}{\varepsilon_1} \right) \approx \frac{1}{2} \ln \left( \frac{1 - 0.167}{0.167} \right) \approx 0.805.$$

Найдем новые веса для объектов, пересчитав их, используя соотношение

$$\alpha_i = \alpha_i e^{-y_i \omega_1 b_1(x_i)}, i \in \{1, 2, \dots, n\}.$$

Например,  $\alpha_1$  находится, как

$$\alpha_1 \approx \frac{1}{12} \cdot e^{-1 \cdot 0.805} \approx 0.037.$$

Остальные значения получаются аналогичным образом и представлены в таблице. Теперь нормируем найденные веса.

Зная значение всех весов, находим их сумму  $\alpha_0 \approx 0.745$  и делим каждый из весов на  $\alpha_0$ , получим значения указанные в последней строке таблицы – это новые веса, используемые на следующей итерации (конечно, округленные).

$X$	(4, 1)	(2, 4)	(5, 1)	(3, 2)	(1, 5)	(3, 5)	(2, 1)	(2, 2)	(2, 5)	(4, 3)	(2, 3)	(5, 2)
$y_i$	+1	-1	+1	+1	-1	-1	+1	+1	-1	-1	+1	-1
$b_1(x_i)$	+1	-1	+1	+1	-1	-1	+1	+1	-1	+1	+1	+1
$\alpha_i \cdot e^{-y_i \omega_t b_1(x_i)}$	0.037	0.037	0.037	0.037	0.037	0.037	0.037	0.037	0.037	0.186	0.037	0.186
$\frac{\alpha_i \cdot e^{-y_i \omega_t b_1(x_i)}}{\alpha_0}$	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.25	0.05	0.25

Как видим, почти все веса стали меньше начальных ( $0.05 < 0.083$ ), кроме двух, которые, наоборот, увеличились. Не сложно догадаться, что это и есть те объекты, которые были классифицированы неверно. На следующей итерации следующий базовый алгоритм придаст им большее внимание.

2. Переходим ко второму шагу. Теперь веса таковы:

$$\alpha_i \approx \begin{cases} 0.25, & i \in \{10, 12\} \\ 0.05, & i \in \{1, 2, \dots, 12\} \setminus \{10, 12\} \end{cases}.$$

Для наглядности изменим размер точек на рисунке 10 в соответствии с изменившимися весами. Пусть теперь выбранный базовый алгоритм  $b_2(x)$  делит плоскость так, как показано на среднем рисунке (верх снова относится к зеленому классу, а низ – к желтому). Классифицируем каждый объект, используя выбранный алгоритм, данные представлены в таблице.

$X$	(4, 1)	(2, 4)	(5, 1)	(3, 2)	(1, 5)	(3, 5)	(2, 1)	(2, 2)	(2, 5)	(4, 3)	(2, 3)	(5, 2)
$y_i$	+1	-1	+1	+1	-1	-1	+1	+1	-1	-1	+1	-1
$b_2(x_i)$	+1	-1	+1	-1	-1	-1	+1	-1	-1	-1	-1	-1
$\alpha_i$	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.25	0.05	0.25

Вес элементов, на которых допущена ошибка, равен 0.05, а значит

$$\varepsilon_2 \approx 3 \cdot 0.05 = 0.15.$$

и тогда:

$$\omega_2 \approx \frac{1}{2} \ln \left( \frac{1 - 0.15}{0.15} \right) \approx 0.867.$$

Повторяем расчет аналогично проделанному в первом пункте, нормируем веса ( $\alpha_0 \approx 0.714$ ) и представим данные в виде таблицы:

$X$	(4, 1)	(2, 4)	(5, 1)	(3, 2)	(1, 5)	(3, 5)	(2, 1)	(2, 2)	(2, 5)	(4, 3)	(2, 3)	(5, 2)
$y_i$	+1	-1	+1	+1	-1	-1	+1	+1	-1	-1	+1	-1
$b_2(X)$	+1	-1	+1	-1	-1	-1	+1	-1	-1	-1	-1	-1
$\alpha_i \cdot e^{-y_i \omega_t b_2(x_i)}$	0.021	0.021	0.021	0.119	0.021	0.021	0.021	0.119	0.021	0.105	0.119	0.105
$\frac{\alpha_i \cdot e^{-y_i \omega_t b_2(x_i)}}{\alpha_0}$	0.029	0.029	0.029	0.167	0.029	0.029	0.029	0.167	0.029	0.147	0.167	0.147

3. Остался последний шаг алгоритма. Как мы видим, теперь у нас имеется три категории весов: 0.029 – для точек, которые все рассмотренные ранее базовые алгоритмы классифицировали верно (вес для них, кстати,

продолжает уменьшаться), 0.167 — для трех точек, которые были классифицированы неверно (веса перед ними увеличились), и 0.147 — для точек, при которых на прошлой итерации вес был увеличен, но которые на этот раз были классифицированы верно, поэтому их веса немного, но уменьшились.

Теперь веса:

$$\alpha_i \approx \begin{cases} 0.029, & i \in \{1, 2, 3, 5, 6, 7, 9\} \\ 0.147, & i \in \{10, 12\} \\ 0.167, & i \in \{4, 8, 11\} \end{cases}.$$

Пусть выбранный алгоритм  $b_3(x)$  делит плоскость вертикальной прямой, как показано на нижнем рисунке (классификация такая: слева — желтые, справа — зеленые). Видно, что классификатор ошибается в шести случаях (соответствующие столбцы закрашены в таблице и все имеют одинаковый вес).

$X$	(4, 1)	(2, 4)	(5, 1)	(3, 2)	(1, 5)	(3, 5)	(2, 1)	(2, 2)	(2, 5)	(4, 3)	(2, 3)	(5, 2)
$y_i$	+1	-1	+1	+1	-1	-1	+1	+1	-1	-1	+1	-1
$b_3(x_i)$	-1	+1	-1	+1	+1	+1	+1	+1	+1	-1	+1	-1
$\omega_3$	0.029	0.029	0.029	0.167	0.029	0.029	0.029	0.167	0.029	0.147	0.167	0.147

Тогда доля ошибок составит

$$\varepsilon_3 \approx 6 \cdot 0.029 \approx 0.174,$$

а значение  $\omega_3 \approx 0.779$ . Все результаты (пересчет и нормирование весов) представим в таблице:

$X$	(4, 1)	(2, 4)	(5, 1)	(3, 2)	(1, 5)	(3, 5)	(2, 1)	(2, 2)	(2, 5)	(4, 3)	(2, 3)	(5, 2)
$y_i$	+1	-1	+1	+1	-1	-1	+1	+1	-1	-1	+1	-1
$b_3(X)$	-1	+1	-1	+1	+1	+1	+1	+1	+1	-1	+1	-1
$\omega_3(i) \cdot e^{-\alpha_3 y_i b_3(X)}$	0.064	0.064	0.064	0.076	0.064	0.064	0.013	0.076	0.064	0.067	0.076	0.067
$\frac{\omega_3(i) \cdot e^{-\alpha_3 y_i b_3(X)}}{Z_3}$	0.084	0.084	0.084	0.1	0.084	0.084	0.018	0.1	0.084	0.089	0.1	0.089

Итак, финальный алгоритм строится следующим образом:

$$a(x) = \text{sign} \left( \sum_{t=1}^3 \omega_t b_t(x) \right) \approx \text{sign} (0.805 \cdot b_1(x) + 0.867 \cdot b_2(x) + 0.779 \cdot b_3(x)).$$

Тогда классификация тестового объекта, скажем,  $x = (4, 1)$  с помощью ансамбля может быть получена как:

$$a(x) = \text{sign} (0.805 \cdot (+1) + 0.867 \cdot (+1) + 0.779 \cdot (-1)) = \text{sign} (0.893) = +1.$$

Ансамбль его относит к желтым.

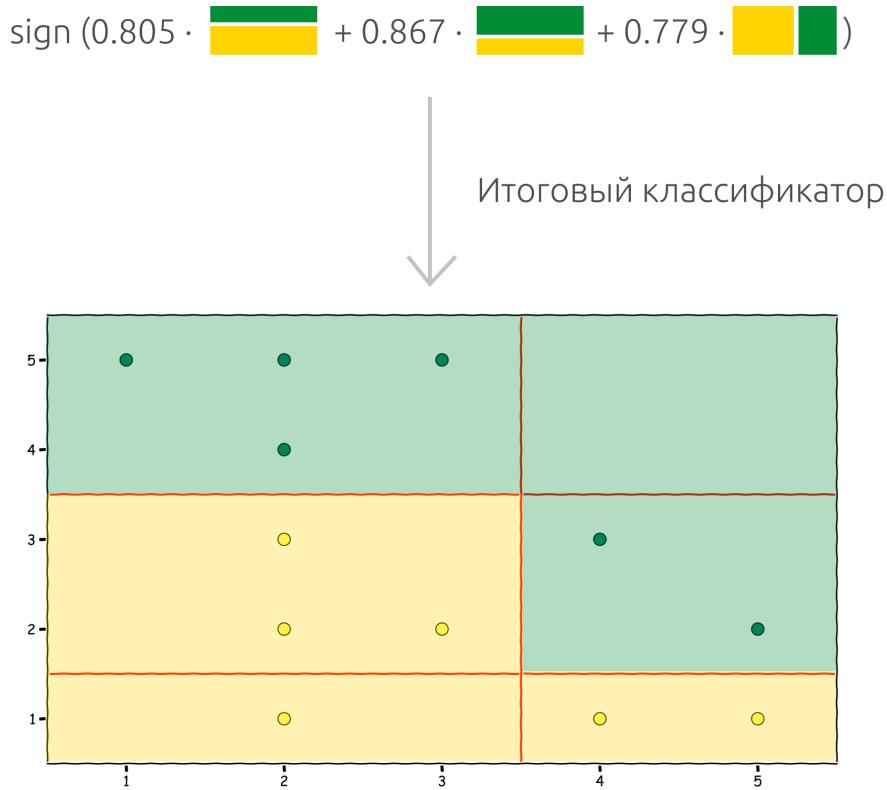


Рис. 11: Итоговый классификатор, полученный алгоритмом AdaBoost.

### 3.4 Стекинг (Stacking)

В традиционных подходах ансамблевых моделей, что рассматривались ранее, мы использовали несколько базовых моделей на различных комбинациях входных данных и объединяли их результаты с помощью голосования, взвешенного голосования, усреднения результатов и т.д. В то же время резонно возникает следующий вопрос: почему, собственно, для создания композиции используются такие простые операции как усреднение или голосование? Может, подбор решающего правила доверить очередному алгоритму (т.н. «метаалгоритму») машинного обучения.

Идея стекинга в этом и состоит: обучить несколько разных базовых моделей и объединить их путем обучения так называемой метамодели. Метамодель является, по своей сути, еще одной моделью, которая служит для вывода прогнозов на основе нескольких прогнозов, возвращенных базовыми моделями. В итоге, мы используем не заранее спланированное решение, вроде голосования, а даем возможность выбрать корректирующее правило новой модели.

Существует несколько возможных алгоритмов стекинга, мы рассмотрим лишь один из них. Обратите внимание, в этом алгоритме обозначения несколько отличаются от обозначений в предыдущих пунктах.

1. Пусть  $X = \{x_1, x_2, \dots, x_n\}$  – обучающая выборка с откликами  $Y =$

$\{y_1, y_2, \dots, y_n\}$ ,  $b_1, b_2, \dots, b_k$  – базовые алгоритмы.

2. Разделить  $X$  на  $k$  непересекающихся почти одинаковых по размеру блоков (folds)

$$X = X_1 \cup X_2 \cup \dots \cup X_k.$$

Обычно  $k$  – это не очень большое число (до 10).

3. Для каждого  $i \in \{1, 2, \dots, k\}$  и  $j \in \{1, 2, \dots, k\}$  обучить базовый алгоритм  $b_i$  на наборе данных

$$X_{[-j]} = X \setminus X_j,$$

и протестировать на  $X_j$ , получив набор откликов  $Y_{ji}$ .

4. Сформировать метаданные (метапредикторы) для обучения метамодели следующим образом

$$X_{meta} = \begin{pmatrix} Y_{11} & Y_{12} & \dots & Y_{1k} \\ Y_{21} & Y_{22} & \dots & Y_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{k1} & Y_{k2} & \dots & Y_{kk} \end{pmatrix}.$$

В качестве откликов для обучения метамодели взять отклики исходные отклики  $Y$ .

5. Обученная метамодель считается ансамблем, построенным на основе стекинга.

**Замечание 3.4.1** Для применения построенной метамодели к тестовым данным, часто поступают следующим образом:

- Базовые алгоритмы обучаются на всей тестовой выборке.
- Тестовое данное подается на вход каждому базовому алгоритму, тем самым получая строку метаданных.
- Стока полученных метаданных подается обученному метаалгоритму.

### 3.4.1 Пример на пальцах

Чтобы продемонстрировать все этапы алгоритма, рассмотрим следующие данные. Пусть двое играют в дартс. Мишень – круг с центром в точке  $(0, 0)$  единичного радиуса. Пусть  $X_1$  откладывается по горизонтали, а  $X_2$  – по вертикали, тогда пара  $(X_1, X_2)$  – координаты дротика, попавшего в мишень.

Пусть отклик 0 соответствует первому игроку (зеленые точки на рисунке 12), а отклик 1 – второму (желтые точки).

Составим модель классификации, которая будет предсказывать, кто из игроков выполнил бросок, основываясь на координатах попадания дротика.

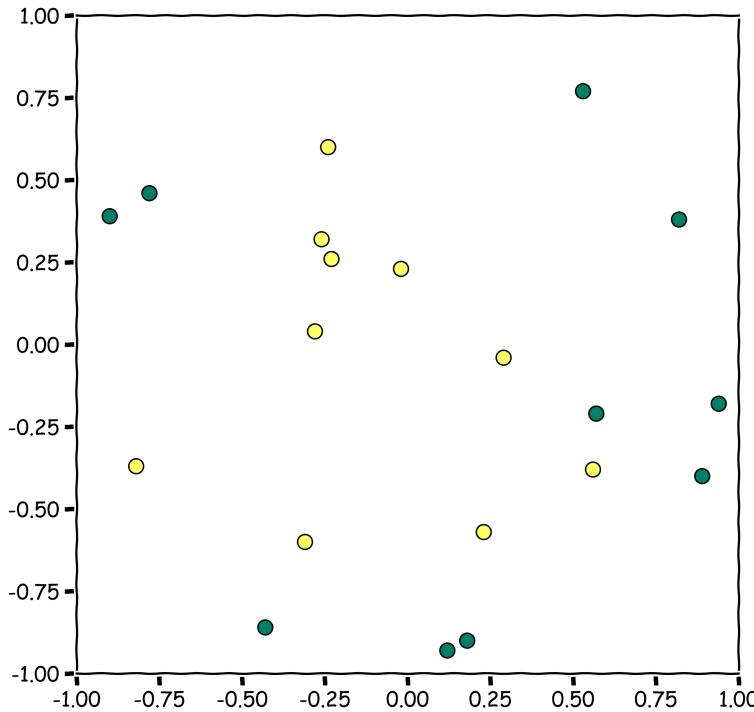


Рис. 12: Исходные данные попаданий в дартс.

Для стекинга будем использовать разделение на 4 непересекающихся блока случайным образом. В качестве базовых алгоритмов выберем:  $k$ -ближайших соседей ( $k = 3$ ) и дерево принятия решений (максимальная глубина 4, критерий разбиения – энтропия). Для сравнения результатов, применим алгоритмы к исходным данным. Из рисунков видно, что алгоритм  $k$ -ближайших соседей (рисунок 13а) допустил три ошибки, а дерево принятия решений (рисунок 13б) – две.

Выполнив четыре итерации алгоритма стекинга, получаем отклики выбранных базовых алгоритмов для каждого из разбиений (блока). Полученные результаты могут быть записаны в табличном виде: В принятых обозначениях получены метаданные (метапредикторы) для обучения метамодели

$$X_{meta} = \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ \vdots & \vdots \\ 1 & 1 \\ 1 & 0 \\ \vdots & \vdots \\ 1 & 1 \end{pmatrix}.$$

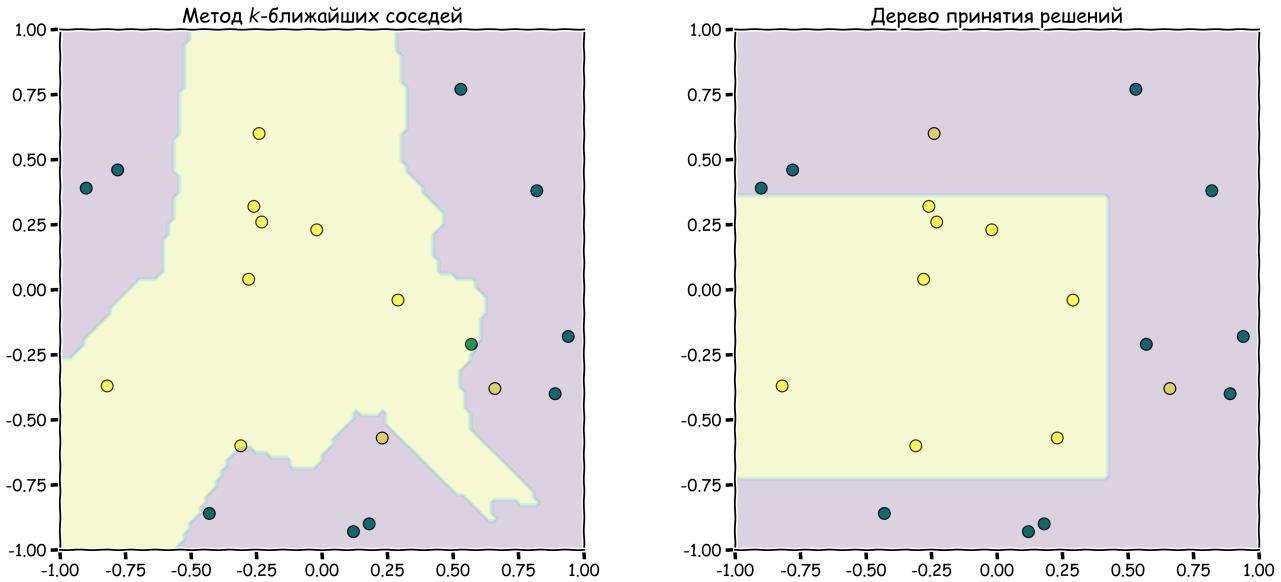


Рис. 13: Сравнение моделей классификации

Номер блока	$X_1$	$X_2$	$Y_{kNN}$	$Y_{DT}$	$Y$
4	0.94	-0.18	0	0	0
3	0.12	-0.93	1	1	0
...	...	...	...	...	...
3	0.18	-0.90	1	1	0
2	-0.82	-0.37	1	0	1
...	...	...	...	...	...
1	0.29	-0.04	1	1	1

Метаалгоритм  $a(x)$  обучим на предикторах  $X_{meta}$  и отклике  $Y$ . В качестве метаалгоритма используем логистическую регрессию. Результат представлен на рисунке 14 из которого можно заметить, что итоговая модель вобрала черты двух моделей.

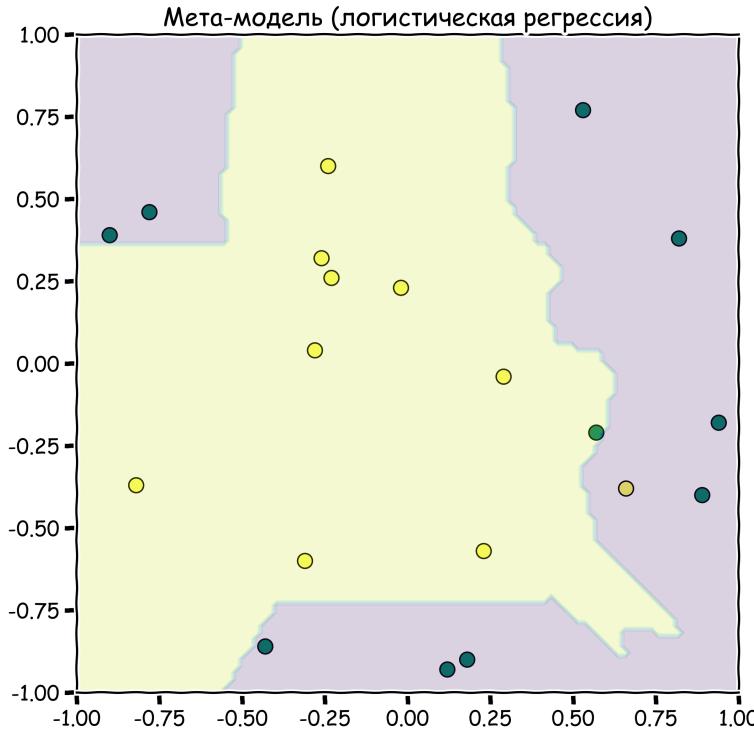


Рис. 14: Модель классификации: стекинг.

### 3.5 Случайный лес

Ранее мы познакомились с алгоритмом деревьев принятия решений. Этот алгоритм очень удобно модифицировать, используя бэггинг: построенное ДПР, как мы видели, может достигать нулевой ошибки на любой обучающей выборке, но, в то же время, может быть очень неустойчивым на тестовой.

Случайный лес – это ансамбль деревьев принятия решений, основанный на бэггинге, но с одним важным дополнением – добавлением еще одной случайности – случайности по признакам. Таким образом, каждое дерево в лесу будет обучено на случайной бутстрэп выборке и некотором наборе признаков из исходного тренировочного набора данных. Все это делает деревья менее похожими между собой, а итоговую модель более устойчивой. Обычно для задачи регрессии рекомендуется использовать  $m = \lceil \frac{p}{3} \rceil$ , а в классификации  $m = \lceil \sqrt{p} \rceil$  признаков (квадратные скобки обозначают целую часть числа).

Резюмируя, запишем алгоритм построения случайного леса. Пусть имеется обучающая выборка  $X$  объема  $n$  с  $p$  предикторами, тогда для каждого дерева  $t = \{1, 2, \dots, T\}$  необходимо:

1. Сформировать бутстрэп выборку  $X_t$ .
2. Из исходных  $p$  предикторов, случайнym образом выбрать  $m$  предикторов.

3. Сформировать дерево принятия решений на основе выборки  $X_t$ , оставив  $m$  предикторов.
4. Повторить шаги 1-3 заданное число  $T$  раз.
5. В зависимости от задачи, сформировать решающее правило и ансамбль.

Еще одно достоинство случайног леса заключается в том, что для оценки вероятности ошибочной классификации нет необходимости использовать кросс-проверку или тестовую выборку. Ведь при формировании деревьев используются бутстрэп, а как мы знаем бутстрэп выборка состоит из около 63% объектов исходной выборки. А значит, на оставшихся, можно выполнить оценку.

Обратимся к синтетическим данным и рассмотрим задачу классификации, сравнив три подхода: деревья принятия решений, бэггинг над ними и случайный лес (рисунок 15). Как мы видим (впрочем, мы это и так знали), деревья решений дают нам переобучение, что отражается на рисунке в виде ровных угловатых границ, уходящих далеко от объектов. В случае бэггинга и случайног леса границы уже более гладкие, нет явных выступов: по сути, область выглядит более цельной. В целом проблема переобучения практически не свойственна случайному лесу, в особенности, если в лесу достаточно деревьев.

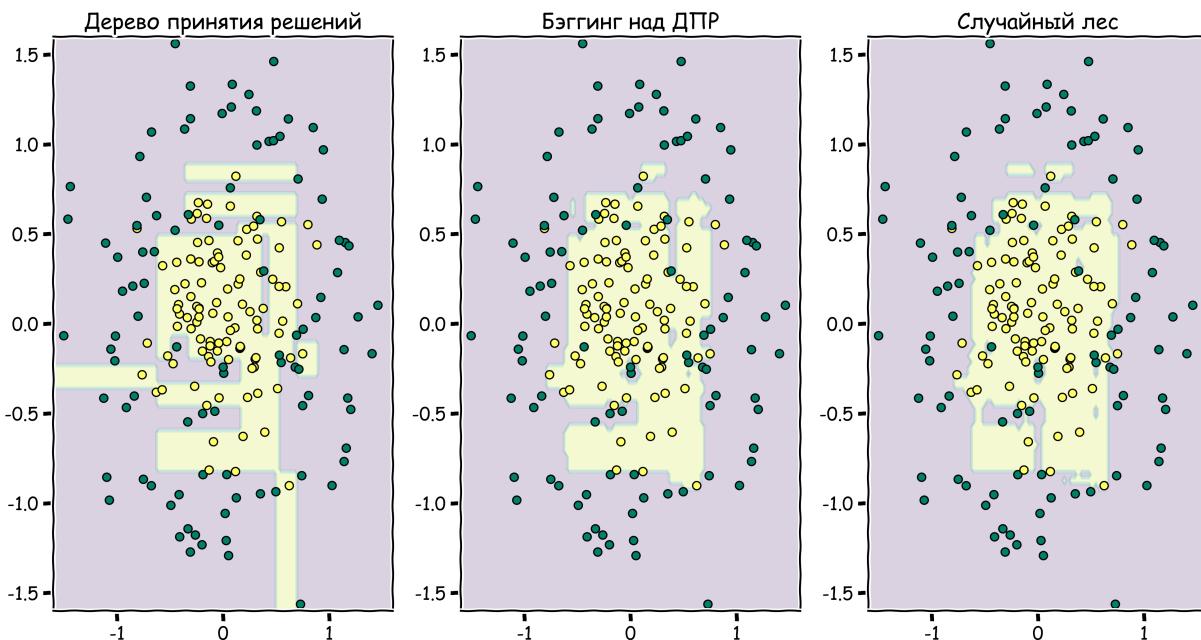


Рис. 15: Сравнение моделей.

Однако основным ограничением случайног леса является то, что большое количество деревьев может сделать алгоритм медленным и неэффек-

тивным для предсказаний в реальном времени. Как правило, эти алгоритмы быстро обучаются, но довольно медленно создают прогнозы после обучения.

## 4 Многоклассовая классификация

Еще одним важным разделом, который мы до сих пор толком не обсудили, является многоклассовая классификация с использованием алгоритмов логистической регрессии или SVM. На самом деле, некоторые подходы к решению этой задачи мы уже осветили, когда рассматривали алгоритмы и ансамбли несколько ранее.

Итак, вопрос следующий: как уже изученные методы двухклассовой классификации применить в том случае, если классов больше? На самом деле, существует достаточно много способов сделать это – мы перечислим лишь два самых популярных: один против всех и все против всех.

### 4.1 Один против всех (one-vs-all)

Начнем с подхода один против всех. Принцип, скорее всего, ясен из названия: так как мы умеем выполнять классификацию объекта к одному из двух возможных классов, то почему бы в случае, когда классов много, не обучить модель отвечать на вопрос: «Принадлежит ли объект  $m$ -ому классу?»,  $m \in \{1, 2, \dots, M\}$ , а затем сформировать какое-то решающее правило на основе ответов алгоритма?

Озвученный подход, использующий двухклассовый классификатор, по своей сути, реализуется следующим образом: при обучении модели на определение принадлежности объекта к классу  $m$ , элементам  $m$ -ого класса устанавливается отклик «+1», а всем остальным – отклик «−1». Перейдем к формальному алгоритму.

1. Пусть решается задача  $M$ -классовой классификации,  $Y = \{1, \dots, M\}$ , а  $b_i(x)$ ,  $i \in \{1, 2, \dots, M\}$  – базовые алгоритмы логистической регрессии, выдающие число в диапазоне  $[0, 1]$ . Тогда для каждого  $i \in \{1, 2, \dots, M\}$ 
  - (a) Всем объектам, принадлежащим классу  $i$ , назначить класс «+1», остальным – класс «−1».
  - (b) Обучить бинарный классификатор  $b_i(x)$  на наборе данных с новыми откликами.
2. Итоговый классификатор  $a(x)$  будет выдавать класс, соответствующий номеру самого уверенного из бинарных классификаторов  $b_i(x)$ :

$$a(x) = \arg \max_{i \in \{1, 2, \dots, M\}} b_i(x).$$

В случае, если подходит несколько значений  $i$ , можно присвоить объекту любой из подходящих классов, или фиктивный класс «0».

Ясно, что вместо базовых алгоритмов логистической регрессии можно использовать, например, базовые алгоритмы SVM или какие-то другие.

**Замечание 4.1.1** *Иначе говоря, в случае, если классификатор строится на основе логистической регрессии, мы выбираем тот класс  $i$ , базовый алгоритм  $b_i(x)$  на котором выдает наибольшую вероятность. Если рассматривается, например, SVM, то тот класс, на котором больше значение классификатора  $b_i(x)$ .*

**Замечание 4.1.2** *Конечно, описанное решающее правило – лишь одно из возможных. Скажем, если логистическая регрессия при каждом  $i$  выдает значение меньшее, чем 0.5, то не всегда имеет смысл присваивать объекту какой-то класс – слишком уж большая «неуверенность». В этом случае решающее правило можно поменять и алгоритм многоклассовой классификации  $a(x)$  может быть, например, таким (с фиктивным классом «0» и принципом «лучше промолчать, чем сорвать»):*

$$a(x) = \begin{cases} b(x), & b(x) \geq 0.5 \\ 0, & b(x) < 0.5 \end{cases}, \quad b(x) = \arg \max_{i \in \{1,2,\dots,M\}} b_i(x).$$

*Конечно, выбор решающего правила – свобода и ответственность, предоставляемые исследователю.*

Простота описанного алгоритма влечет за собой и его недостатки. Например, при обучении модели может оказаться, что все выборки имеют примерно равные объемы. Но тогда при обучении каждого алгоритма  $b_i(x)$  представители отрицательного класса будут сильно перевешивать (хотя бы количеством) представителей положительного, тем самым положительный класс будет недооценен, и классификатор будет «неуверенным».

### 4.1.1 Пример

Применим алгоритм к уже знакомым данным о сладости и хрусткости различных продуктов. Как видно из таблицы, данные разделены на три класса. Кроме того, визуально, исходя из рисунка, понятно, что достаточно хорошо делятся на группы, так что алгоритм многоклассовой классификации, построенный на основе базовых алгоритмов логистической регрессии, должен показать себя не с худшей стороны.

Таблица 1: Тренировочные данные.

Продукт	Сладость	Хруст	Класс
банан	10	1	фрукт
апельсин	7	4	фрукт
виноград	8	3	фрукт
креветка	2	2	протеин
бекон	1	5	протеин
орехи	3	3	протеин
сыр	2	1	протеин
рыба	3	2	протеин
огурец	2	8	овощ
яблоко	9	8	фрукт
морковь	4	10	овощ
сельдерей	2	9	овощ
салат айсберг	3	7	овощ
груша	8	7	фрукт

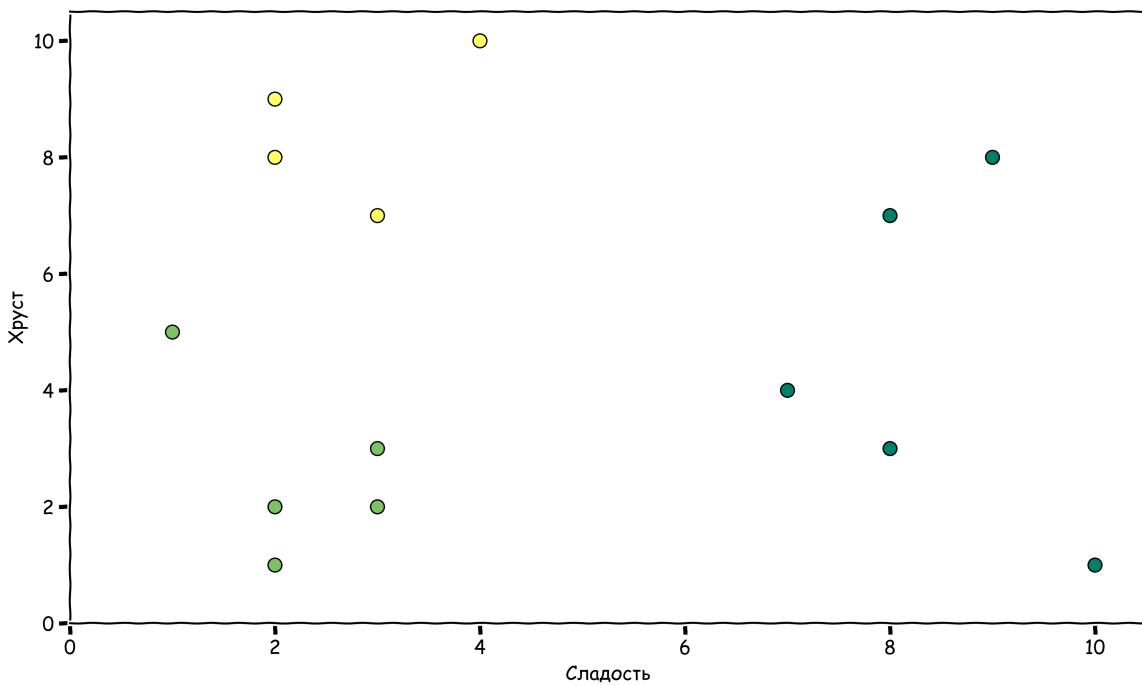


Рис. 16: Визуализация значений таблицы.

Во-первых, для удобства занумеруем исходные классы следующим образом: фрукты – 1, овощи – 2, протеины – 3. В таком случае, на первой итерации, при  $i = 1$ , мы будем обучать классификатор отличать фрукты от прочих объектов. На основе сформулированного в алгоритме правила, дан-

ным сначала назначается новый отклик  $Y^*$ . Результаты вы можете видеть в таблице 2.

Таблица 2: Итерация  $k = 1$ .

Продукт	Сладость	Хруст	Класс $Y$	Класс $Y^*$
банан	10	1	1	+1
апельсин	7	4	1	+1
виноград	8	3	1	+1
креветка	2	2	3	-1
бекон	1	5	3	-1
орехи	3	3	3	-1
сыр	2	1	3	-1
рыба	3	2	3	-1
огурец	2	8	2	-1
яблоко	9	8	1	+1
морковь	4	10	2	-1
сельдерей	2	9	2	-1
салат айсберг	3	7	2	-1
груша	8	7	1	+1

Обучив модель, используя алгоритм логистической регрессии, приходим к разделяющей прямой, задаваемой уравнением (коэффициенты округлены до сотых)

$$-6.23 + 1.22X_1 - 0.12X_2 = 0.$$

Аналогичные шаги проделываются при  $i = 2$  и  $i = 3$ , что приводит нас к еще двум прямым, которые задаются уравнениями

$$7.52 - 0.95X_1 - 0.97X_2 = 0,$$

$$-5.53 - 0.57X_1 + 1.09X_2 = 0.$$

Отобразив прямые на плоскости, легко увидеть, что каждая из построенных моделей справляется со своей задаче и не допускает ошибок на тренировочных данных.

Теперь классификация нового, тестового наблюдения, не составляет труда: опираясь на алгоритм, необходимо выбрать тот класс, которому соответствует самый уверенный базовый классификатор. Возьмем объект «Перец» с координатами  $(6, 9)$ , найдем значение  $\Psi_i$  для каждого из классификаторов:

$$\Psi_1 = -6.23 + 1.22 \cdot 6 - 0.12 \cdot 9 = 0.01.$$

$$\Psi_2 = 7.52 - 0.95 \cdot 6 - 0.97 \cdot 9 = -6.91,$$

$$\Psi_3 = -5.53 - 0.57 \cdot 6 + 1.09 \cdot 9 = 0.86.$$

Теперь найдем вероятности отнесения объекта к классу «плюс» (каждый раз – к своему), согласно формуле

$$P_+ = \frac{1}{1 + e^{-\Psi}}.$$

Подставив в нее полученные значения, найдем вероятности отнесения объекта «Перец» к классам фрукты, протеины и овощи, соответственно:

$$P_+(\Psi_1) = \frac{1}{1 + e^{-0.01}} \approx 0.503,$$

$$P_+(\Psi_2) = \frac{1}{1 + e^{6.91}} \approx 0.001,$$

$$P_+(\Psi_3) = \frac{1}{1 + e^{-0.86}} \approx 0.703.$$

В результате чего, «Перец» будет классифицирован, как овощ!

Отметим также, что если тестовый объект попадает в треугольник (на рисунке), образованный пересечением трех прямых, то классификация относительно каждого из обученных алгоритмов будет неуверенной (выдаваемые вероятности будут меньше, чем 0.5), и именно для таких объектов следует обдумать финальное решение: классифицировать ли их вовсе.

## 4.2 Все против всех (all-vs-all)

Второй подход схож с предыдущим и основан все на том же умении определять принадлежность объекта к одному из двух классов. Однако, если раньше мы «сталкивались» каждый класс со всеми остальными, то теперь мы «столкнемся» при обучении каждый класс с каждым. Перейдем к алгоритму.

1. Пусть решается задача  $M$ -классовой классификации,  $Y = \{1, 2, \dots, M\}$ ,  $a(x)$  – алгоритм бинарной логистической регрессии, выдающий класс «+1» или «-1». Тогда для каждого  $i, j \in \{1, 2, \dots, M\}$ ,  $i < j$ 
  - (a) Составить подвыборку  $X_{ij}$  исходной выборки  $X$ , содержащую все объекты из классов  $i$  и  $j$ .
  - (b) Сопоставить элементам класса  $i$  отклик «+1», а элементам класса  $j$  – отклик «-1».

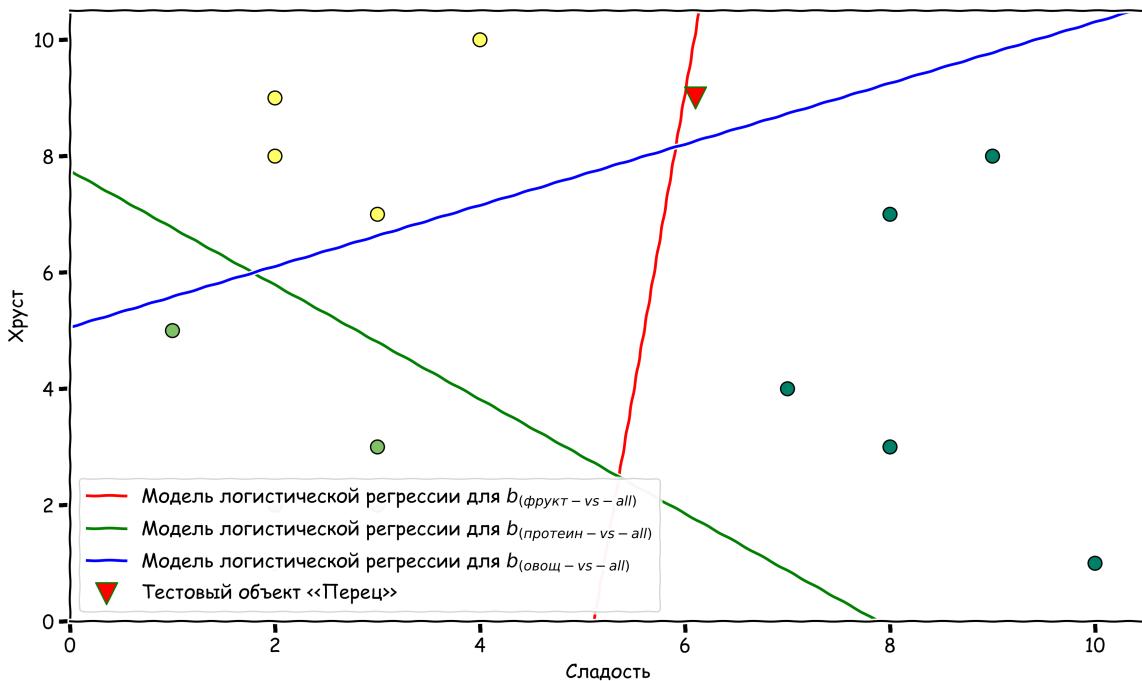


Рис. 17: Базовые модели логистических регрессий.

(c) Обучить алгоритм  $a(x)$  и создать новый алгоритм  $a_{ij}(x)$  присваивающий корректные номера классам

$$a_{ij}(x) = \begin{cases} i, & a(x) = +1 \\ j, & a(x) = -1 \end{cases}.$$

(d) Положить  $a_{ji}(x) = a_{ij}(x)$ ,  $a_{ii}(x) = 0$ .

2. Итоговый классификатор  $a(x)$  выдает класс, соответствующий голосованию большинства среди всех обученных алгоритмов, то есть:

$$a(x) = \arg \max_{m \in \{1, 2, \dots, M\}} \sum_{i=1}^K \sum_{j=1}^K I(a_{ij}(X) = m).$$

В случае, если подходит несколько значений  $m$ , можно присвоить объекту любой из подходящих классов, или фиктивный класс «0».

#### 4.2.1 Пример

Вернемся все к тем же данным, они представлены в таблице 1. Отклик состоит из трех уникальных значений: фрукты, овощи и протеины, а значит пары «сталкиваемых» классов такие: (фрукт, овощ), (фрукт, протеин), (овощ, протеин). Составим подвыборку для первой пары (таблица 3).

Таблица 3: Обучающая подвыборка  $X_{(\text{фрукт, овощ})}$ .

Продукт	Сладость	Хруст	Класс
банан	10	1	фрукт
апельсин	7	4	фрукт
виноград	8	3	фрукт
огурец	2	8	овощ
яблоко	9	8	фрукт
морковь	4	10	овощ
сельдерей	2	9	овощ
салат айсберг	3	7	овощ
груша	8	7	фрукт

Будем использовать логистическую регрессию в качестве алгоритма классификации, моделирование приводит к уравнению разделяющей прямой:

$$-2.11 + 0.95X_1 - 0.44X_2 = 0,$$

которая отделяет фрукты от овощей (ведь информация о протеинах, которым соответствуют светло-зеленые точки в левом нижнем углу, не была использована при обучении). Нормаль прямой имеет координаты  $(0.95, -0.44)$ , и указывает в сторону зеленых точек (фруктов).

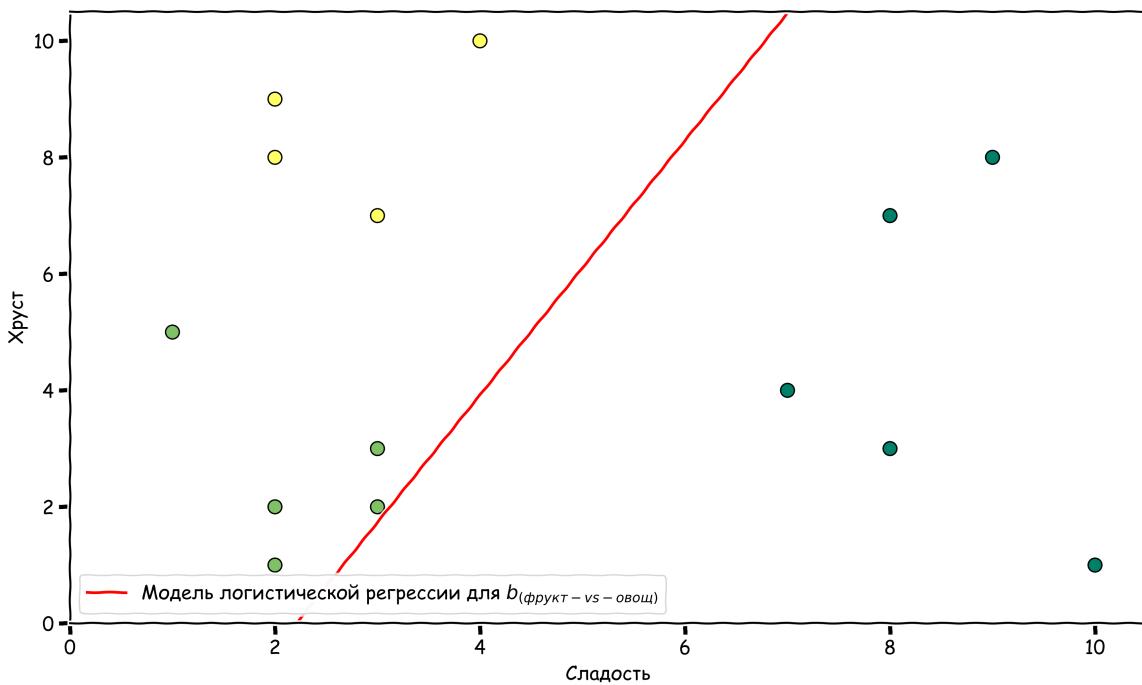


Рис. 18: Тренировочные данные и регрессия для классов фрукты и овощи.

Аналогичным образом формируются подвыборки  $X_{(\text{фрукт, протеин})}$  и  $X_{(\text{овощ, протеин})}$ , для которых получены уравнения следующих разделяющих

прямых:

$$\begin{aligned} -6.36 + 1.04X_1 + 0.28X_2 &= 0, \\ -6.96 + 0.33X_1 + 1.07X_2 &= 0. \end{aligned}$$

Все прямые отображены на рисунке 19.

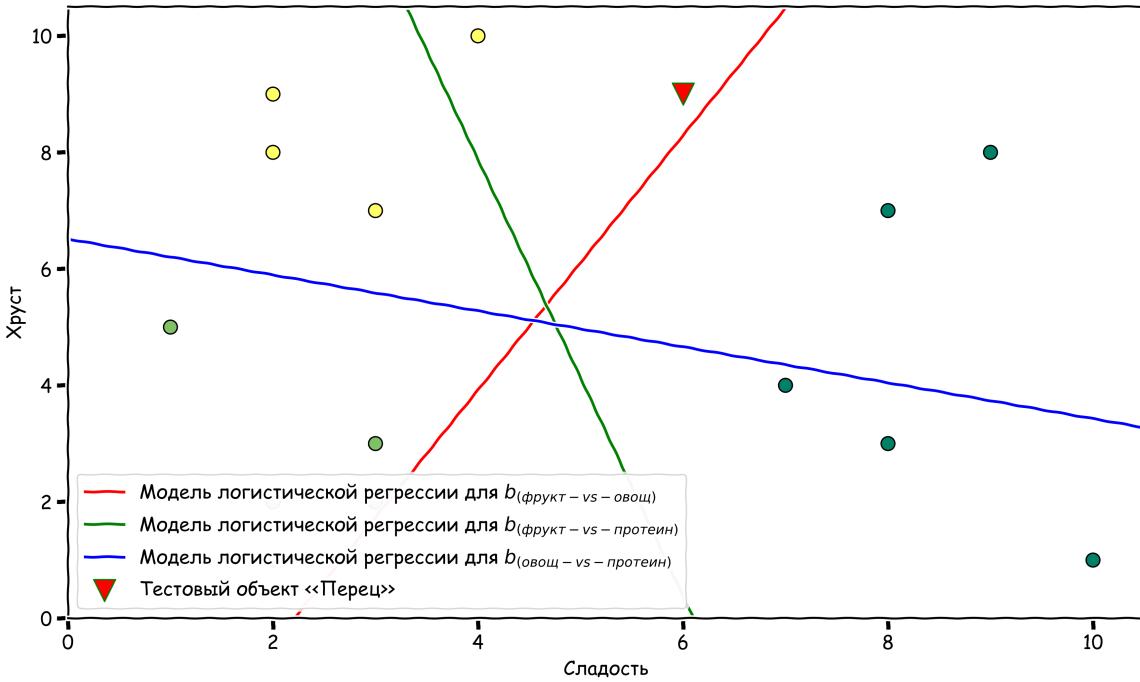


Рис. 19: Тренировочные данные и базовые алгоритмы.

Теперь проверим нашу модель, возьмем объект «Перец» с координатами (6,9) и найдем значения  $\Psi_i$  для каждого из классификаторов и соответствующие вероятности:

$$\Psi_1 = -2.11 + 0.95 \cdot 6 - 0.44 \cdot 9 = -0.37, \quad P_+(\Psi_1) = \frac{1}{1 + e^{-0.37}} \approx 0.41,$$

$$\Psi_2 = -6.36 + 1.04 \cdot 6 + 0.28 \cdot 9 = 2.4, \quad P_+(\Psi_2) = \frac{1}{1 + e^{-2.4}} \approx 0.92,$$

$$\Psi_3 = -6.96 + 0.33 \cdot 6 + 1.07 \cdot 9 = 4.65, \quad P_+(\Psi_3) = \frac{1}{1 + e^{-4.65}} \approx 0.99.$$

Итак, первая модель относит объект «Перец» к классу овощи ( $P_+ < 0.5$ ), вторая модель к классу фруктов и третья к классу овощи. Таким образом класс «овощ» назначается объекту «Перец» большинством голосов.