Теоретические основы информатики (концептуальные модели и математические основы)

Лекция № 3. Вводные примеры

А.С. Шундеев

Содержание

Содержание

Основные понятия

Через X всегда будет обозначаться непустое множество объектов, а через Y будет обозначаться непустое множество меток.

Основные понятия

Через X всегда будет обозначаться непустое множество объектов, а через Y будет обозначаться непустое множество меток.

Пара вида (x, y), где $x \in X$ и $y \in Y$, называется примером.

Основные понятия

Через X всегда будет обозначаться непустое множество объектов, а через Y будет обозначаться непустое множество меток.

Пара вида (x, y), где $x \in X$ и $y \in Y$, называется примером.

Относительно примера (x, y) будем говорить, что объекту x приписана метка y.

Основные понятия

Через X всегда будет обозначаться непустое множество объектов, а через Y будет обозначаться непустое множество меток.

Пара вида (x, y), где $x \in X$ и $y \in Y$, называется примером.

Относительно примера (x, y) будем говорить, что объекту x приписана метка y.

Множество всех примеров $X \times Y$ будет обозначаться через Z.

Основные понятия

Через X всегда будет обозначаться непустое множество объектов, а через Y будет обозначаться непустое множество меток.

Пара вида (x, y), где $x \in X$ и $y \in Y$, называется примером.

Относительно примера (x, y) будем говорить, что объекту x приписана метка y.

Множество всех примеров $X \times Y$ будет обозначаться через Z.

Элементы множества Z^* будут называться наборами обучающих примеров или обучающими выборками.

Основные понятия

Через X всегда будет обозначаться непустое множество объектов, а через Y будет обозначаться непустое множество меток.

Пара вида (x, y), где $x \in \mathsf{X}$ и $y \in \mathsf{Y}$, называется примером.

Относительно примера (x, y) будем говорить, что объекту x приписана метка y.

Множество всех примеров $\mathsf{X} \times \mathsf{Y}$ будет обозначаться через Z.

Элементы множества Z^* будут называться наборами обучающих примеров или обучающими выборками.

Под алгоритмом обучения понимается произвольное отображение вида

$$\mathcal{A} : \mathsf{Z}^* \longrightarrow \mathcal{H}, \ \mathbf{z} \longmapsto h_{\mathbf{z}}, \qquad \mathcal{H} \subseteq \mathsf{Y}^{\mathsf{X}}.$$

Основные понятия

Через X всегда будет обозначаться непустое множество объектов, а через Y будет обозначаться непустое множество меток.

Пара вида (x, y), где $x \in \mathsf{X}$ и $y \in \mathsf{Y}$, называется примером.

Относительно примера (x, y) будем говорить, что объекту x приписана метка y.

Множество всех примеров $\mathsf{X} \times \mathsf{Y}$ будет обозначаться через Z.

Элементы множества Z^* будут называться наборами обучающих примеров или обучающими выборками.

Под алгоритмом обучения понимается произвольное отображение вида

$$\mathcal{A} : \mathsf{Z}^* \longrightarrow \mathcal{H}, \ \mathbf{z} \longmapsto h_{\mathbf{z}}, \qquad \mathcal{H} \subseteq \mathsf{Y}^{\mathsf{X}}.$$

При этом класс функций $\mathcal{H}$ называется моделью, а его элементы называются гипотезами или предикторами.

Основные понятия

В так называемом реализуемом случае задачи обучения предполагается наличие неизвестной целевой функции $g : X \longrightarrow Y$.

Основные понятия

В так называемом реализуемом случае задачи обучения предполагается наличие неизвестной целевой функции $g : X \longrightarrow Y$.

С помощью алгоритма обучения $\mathcal{A}$ и обучающей выборки $\mathbf{z} \in Z^*$, описывающей поведение функции g, строится приближение $h_{\mathbf{z}} \approx g$.

Основные понятия

В так называемом реализуемом случае задачи обучения предполагается наличие неизвестной целевой функции $g : X \longrightarrow Y$.

С помощью алгоритма обучения $\mathcal{A}$ и обучающей выборки $\mathbf{z} \in Z^*$, описывающей поведение функции g, строится приближение $h_{\mathbf{z}} \approx g$.

В идеальном случае, на который не всегда приходится расчитывать, обучающая выборка имеет вид $\mathbf{z} = g \circ \mathbf{x}$, где $\mathbf{x} \in X^*$.

Основные понятия

В так называемом реализуемом случае задачи обучения предполагается наличие неизвестной целевой функции $g : X \longrightarrow Y$.

С помощью алгоритма обучения $\mathcal{A}$ и обучающей выборки $\mathbf{z} \in Z^*$, описывающей поведение функции g, строится приближение $h_{\mathbf{z}} \approx g$.

В идеальном случае, на который не всегда приходится расчитывать, обучающая выборка имеет вид $\mathbf{z} = g \circ \mathbf{x}$, где $\mathbf{x} \in X^*$.

В случае, когда множество меток конечно $|Y| < \infty$, говорят о задаче классификации. При этом элементы класса $\mathcal{H}$ дополнительно называются классификаторами.

Основные понятия

В так называемом реализуемом случае задачи обучения предполагается наличие неизвестной целевой функции $g : X \longrightarrow Y$.

С помощью алгоритма обучения $\mathcal{A}$ и обучающей выборки $\mathbf{z} \in Z^*$, описывающей поведение функции g, строится приближение $h_{\mathbf{z}} \approx g$.

В идеальном случае, на который не всегда приходится расчитывать, обучающая выборка имеет вид $\mathbf{z} = g \circ \mathbf{x}$, где $\mathbf{x} \in X^*$.

В случае, когда множество меток конечно $|Y| < \infty$, говорят о задаче классификации. При этом элементы класса $\mathcal{H}$ дополнительно называются классификаторами.

Если множество меток $Y = \mathbb{R}$, то говорят о задаче регрессии.

Основные понятия

В так называемом реализуемом случае задачи обучения предполагается наличие неизвестной целевой функции $g : X \longrightarrow Y$.

С помощью алгоритма обучения $\mathcal{A}$ и обучающей выборки $\mathbf{z} \in Z^*$, описывающей поведение функции g, строится приближение $h_{\mathbf{z}} \approx g$.

В идеальном случае, на который не всегда приходится расчитывать, обучающая выборка имеет вид $\mathbf{z} = g \circ \mathbf{x}$, где $\mathbf{x} \in X^*$.

В случае, когда множество меток конечно $|Y| < \infty$, говорят о задаче классификации. При этом элементы класса $\mathcal{H}$ дополнительно называются классификаторами.

Если множество меток $Y = \mathbb{R}$, то говорят о задаче регрессии.

Дополнительно выделим ещё два частных случая.

Основные понятия

В так называемом реализуемом случае задачи обучения предполагается наличие неизвестной целевой функции $g : X \longrightarrow Y$.

С помощью алгоритма обучения $\mathcal{A}$ и обучающей выборки $\mathbf{z} \in Z^*$, описывающей поведение функции g, строится приближение $h_{\mathbf{z}} \approx g$.

В идеальном случае, на который не всегда приходится расчитывать, обучающая выборка имеет вид $\mathbf{z} = g \circ \mathbf{x}$, где $\mathbf{x} \in X^*$.

В случае, когда множество меток конечно $|Y| < \infty$, говорят о задаче классификации. При этом элементы класса $\mathcal{H}$ дополнительно называются классификаторами.

Если множество меток $Y = \mathbb{R}$, то говорят о задаче регрессии.

Дополнительно выделим ещё два частных случая. Случай $Y = \{0, 1\}$ соответствует задаче бинарной классификации, а случай $Y = [0, 1]$ соответствует задаче обучения функций.

Основные понятия

Задача бинарной классификации эквивалентна так называемой задаче обучения концептов.

Основные понятия

Задача бинарной классификации эквивалентна так называемой задаче обучения концептов.

Произвольное подмножество $\mathcal{C} \subseteq 2^X$ будем называть классом концептов.

Основные понятия

Задача бинарной классификации эквивалентна так называемой задаче обучения концептов.

Произвольное подмножество $\mathcal{C} \subseteq 2^X$ будем называть классом концептов.

Таким образом, можно говорить о целевом концепте и алгоритмах обучения, которые по обучающим выборкам строят концепты.

Основные понятия

Задача бинарной классификации эквивалентна так называемой задаче обучения концептов.

Произвольное подмножество $\mathcal{C} \subseteq 2^X$ будем называть классом концептов.

Таким образом, можно говорить о целевом концепте и алгоритмах обучения, которые по обучающим выборкам строят концепты.

Заметим, что существует взаимно однозначное соответствие между концептами и их характеристическими функциями.

Основные понятия

Задача бинарной классификации эквивалентна так называемой задаче обучения концептов.

Произвольное подмножество $\mathcal{C} \subseteq 2^X$ будем называть классом концептов.

Таким образом, можно говорить о целевом концепте и алгоритмах обучения, которые по обучающим выборкам строят концепты.

Заметим, что существует взаимно однозначное соответствие между концептами и их характеристическими функциями.

В дальнейшем условие $\mathcal{H} = \{\mathbf{1}_C : C \in \mathcal{C}\}$ будем для краткости записывать через $\mathcal{C} \simeq_{\mathbf{1}} \mathcal{H}$.

Основные понятия

Перейдём к рассмотрению вопроса оценки точности алгоритма обучения.

Основные понятия

Перейдём к рассмотрению вопроса оценки точности алгоритма обучения.

Функцией потерь называется ограниченная функция вида $l : Y^2 \longrightarrow \mathbb{R}_+$, для которой выполняется условие $l(y, y) = 0$ для любого $y \in Y$.

Основные понятия

Перейдём к рассмотрению вопроса оценки точности алгоритма обучения.

Функцией потерь называется ограниченная функция вида $l : Y^2 \longrightarrow \mathbb{R}_+$, для которой выполняется условие $l(y, y) = 0$ для любого $y \in Y$.

Примерами таких функций могут служить

$$l_{01}(y_1, y_2) := \begin{cases} 0, & \text{если} \quad y_1 = y_2; \\ 1, & \text{иначе} \end{cases}$$

и

$$l_{\text{sq}}(y_1, y_2) := |y_1 - y_2|^2,$$

где $y_1, y_2 \in Y$.

Основные понятия

Перейдём к рассмотрению вопроса оценки точности алгоритма обучения.

Функцией потерь называется ограниченная функция вида $l : Y^2 \longrightarrow \mathbb{R}_+$, для которой выполняется условие $l(y, y) = 0$ для любого $y \in Y$.

Примерами таких функций могут служить

$$l_{01}(y_1, y_2) := \begin{cases} 0, & \text{если} \quad y_1 = y_2; \\ 1, & \text{иначе} \end{cases}$$

и

$$l_{\text{sq}}(y_1, y_2) := |y_1 - y_2|^2,$$

где $y_1, y_2 \in Y$.

Функция l_{01} используется в задаче классификации, а функция l_{sq} используется в задаче регрессии.

Основные понятия

Перейдём к рассмотрению вопроса оценки точности алгоритма обучения.

Функцией потерь называется ограниченная функция вида $l : Y^2 \longrightarrow \mathbb{R}_+$, для которой выполняется условие $l(y, y) = 0$ для любого $y \in Y$.

Примерами таких функций могут служить

$$l_{01}(y_1, y_2) := \begin{cases} 0, & \text{если} \quad y_1 = y_2; \\ 1, & \text{иначе} \end{cases}$$

и

$$l_{\text{sq}}(y_1, y_2) := |y_1 - y_2|^2,$$

где $y_1, y_2 \in Y$.

Функция l_{01} используется в задаче классификации, а функция l_{sq} используется в задаче регрессии. В последнем случае предполагается, что множество меток Y является ограниченным подмножеством в $\mathbb{R}$.

Основные понятия

Функция потерь l позволяет оценить, на сколько некоторая гипотеза $h \in \mathcal{H}$ соответствует индивидуальному примеру $z = (x, y) \in \mathrm{Z}$.

Основные понятия

Функция потерь l позволяет оценить, на сколько некоторая гипотеза $h \in \mathcal{H}$ соответствует индивидуальному примеру $z = (x, y) \in Z$.

В качестве соответствующей числовой характеристики выступает величина $l(h(x), y)$.

Основные понятия

Функция потерь l позволяет оценить, на сколько некоторая гипотеза $h \in \mathcal{H}$ соответствует индивидуальному примеру $z = (x, y) \in Z$.

В качестве соответствующей числовой характеристики выступает величина $l(h(x), y)$.

Для оценки соответствия гипотезы h целому набору примеров $\mathbf{z} = (z_1, \ldots, z_n) \in Z^n$, где $z_i = (x_i, y_i)$ $(i = 1, \ldots, n; n \in \mathbb{N})$, используется функция эмпирического риска, задаваемая по правилу

$$r(l; h, \mathbf{z}) := \frac{1}{n} \sum_{i=1}^{n} l(h(x_i), y_i).$$

Основные понятия

Функция потерь l позволяет оценить, на сколько некоторая гипотеза $h \in \mathcal{H}$ соответствует индивидуальному примеру $z = (x, y) \in Z$.

В качестве соответствующей числовой характеристики выступает величина $l(h(x), y)$.

Для оценки соответствия гипотезы h целому набору примеров $\mathbf{z} = (z_1, \ldots, z_n) \in Z^n$, где $z_i = (x_i, y_i)$ $(i = 1, \ldots, n; n \in \mathbb{N})$, используется функция эмпирического риска, задаваемая по правилу

$$r(l; h, \mathbf{z}) := \frac{1}{n} \sum_{i=1}^{n} l(h(x_i), y_i).$$

Если из контекста понятно, о какой функции потерь идёт речь, то будет использоваться сокращение $r(h, \mathbf{z})$.

Основные понятия

Таким образом, числовая характеристика $r(h_{\mathbf{z}}, \mathbf{z})$ может использоваться для оценки точности алгоритма обучения $\mathcal{A}$.

Основные понятия

Таким образом, числовая характеристика $r(h_{\mathbf{z}}, \mathbf{z})$ может использоваться для оценки точности алгоритма обучения $\mathcal{A}$.

К сожалению, она оценивает близость построенной гипотезы $h_{\mathbf{z}}$ к целевой функции g только на объектах из обучающей выборки $\mathbf{z}$.

Основные понятия

Таким образом, числовая характеристика $r(h_{\mathbf{z}}, \mathbf{z})$ может использоваться для оценки точности алгоритма обучения $\mathcal{A}$.

К сожалению, она оценивает близость построенной гипотезы $h_{\mathbf{z}}$ к целевой функции g только на объектах из обучающей выборки $\mathbf{z}$.

Одной из целей настоящего курса является иследование применимости подобной оценки.

Основные понятия

Среди всех алгоритмов обучения особую роль играют методы минимизации эмпирического риска, которые удовлетворяют условию

$$h_{\mathbf{z}} \in \arg\min_{h \in \mathcal{H}} r(h, \mathbf{z}) \qquad (\mathbf{z} \in \mathrm{Z}^*).$$

Основные понятия

Среди всех алгоритмов обучения особую роль играют методы минимизации эмпирического риска, которые удовлетворяют условию

$$h_{\mathbf{z}} \in \arg\min_{h \in \mathcal{H}} r(h, \mathbf{z}) \qquad (\mathbf{z} \in \mathrm{Z}^*).$$

Заметим, что для любой задачи классификации, использующей функцию потерь l_{01} всегда существует такой метод.

Основные понятия

Функция потерь l позволяет сопоставить классу гипотез $\mathcal{H}$ класс функций

$$\mathcal{F} := \{(x, y) \longmapsto l(h(x), y) \ : \ h \in \mathcal{H}\}.$$

Основные понятия

Функция потерь l позволяет сопоставить классу гипотез $\mathcal{H}$ класс функций

$$\mathcal{F} := \{(x, y) \longmapsto l(h(x), y) \ : \ h \in \mathcal{H}\}.$$

Подобное соответствие будем обозначать через $\mathcal{H} \simeq_l \mathcal{F}$.

Основные понятия

Функция потерь l позволяет сопоставить классу гипотез $\mathcal{H}$ класс функций

$$\mathcal{F} := \{(x, y) \longmapsto l(h(x), y) \ : \ h \in \mathcal{H}\}.$$

Подобное соответствие будем обозначать через $\mathcal{H} \simeq_l \mathcal{F}$.

В дальнейшем, будет удобно вычислять эмпирический риск сразу для соответствующей функции $f \in \mathcal{F}$. По определению положим

$$r(f, \mathbf{z}) := \frac{1}{n} \sum_{i=1}^{n} f(z_i).$$

Содержание

Деревья решений

Деревья решений являются одним из наиболее понятных для понимания и популярных методов анализа данных. Существует несколько подходов для их определения и построения.

Деревья решений

Деревья решений являются одним из наиболее понятных для понимания и популярных методов анализа данных. Существует несколько подходов для их определения и построения.

Одним из таких подходов является алгоритм CART (Classification and Regression Tree), который и будет нами рассмотрен применительно к задаче классификации.

Деревья решений

Деревья решений являются одним из наиболее понятных для понимания и популярных методов анализа данных. Существует несколько подходов для их определения и построения.

Одним из таких подходов является алгоритм CART (Classification and Regression Tree), который и будет нами рассмотрен применительно к задаче классификации.

В качесте иллюстрации будем использовать следующий пример.

Содержание

Деревья решений

Определение дерева решений

В качестве множества объектов X будет выступать подмножество в $\mathbb{R}_+ \times \{\textbf{куб}, \textbf{шар}\}$.

Множеством меток Y будет $\{\textbf{чёрный}, \textbf{белый}\}$.

Целевая функция g устроена следующим образом. Предположим, что x представляет собой пару значений признаков *вес* и *форма* некоторого камня в коллекции. Тогда $x \in X$ и $g(x)$ – это цвет, которым обладает этот камень.

Деревья решений

Определение дерева решений

В качестве множества объектов X будет выступать подмножество в $\mathbb{R}_+ \times \{\textbf{куб}, \textbf{шар}\}$.

Множеством меток Y будет $\{\textbf{чёрный}, \textbf{белый}\}$.

Целевая функция g устроена следующим образом. Предположим, что x представляет собой пару значений признаков *вес* и *форма* некоторого камня в коллекции. Тогда $x \in X$ и $g(x)$ – это цвет, которым обладает этот камень.

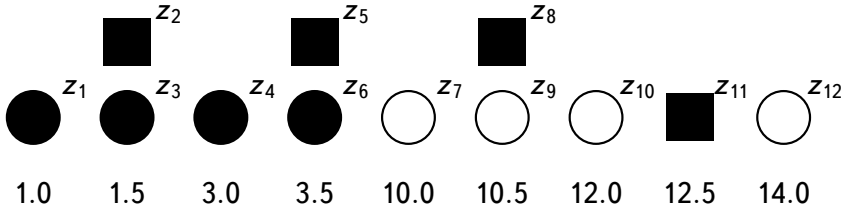
Класс гипотез $\mathcal{H}$ образуют функции, каждая из которых может быть задана с помощью некоторого дерева решений. Алгоритм обучения $\mathcal{A}$ строит по обучающей выборке $\mathbf{z} \in Z^*$ такое дерево.

Прежде, чем непосредственно перейти к описанию алгоритма обучения, разберём структуру дерева решений и правило вычисления с его помощью значений соответствующей функции.

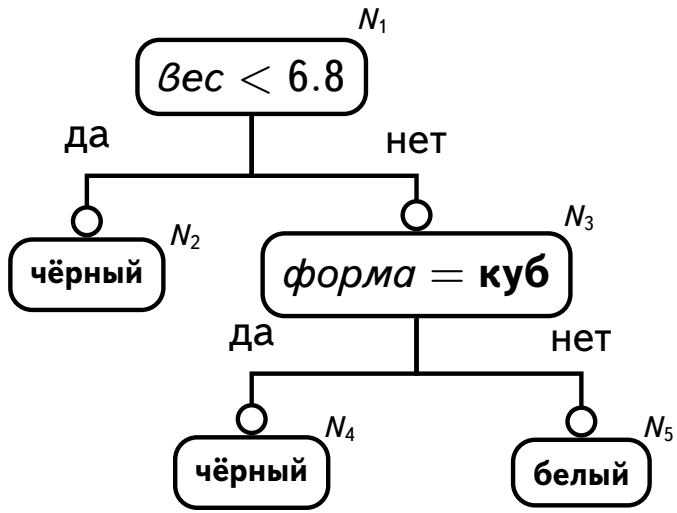
Деревья решений

Определение дерева решений

Каждой внутренней вершине такого дерева приписан некоторый предикат, заданный на множестве X, а каждой листовой вершине приписана метка из Y.



Приписанный вершине предикат проверяет значение некоторого признака объекта.



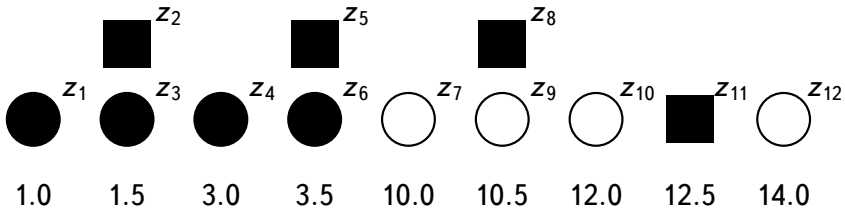
Для количественных признаков проверка заключается в сравнении с некоторым пороговым значением.

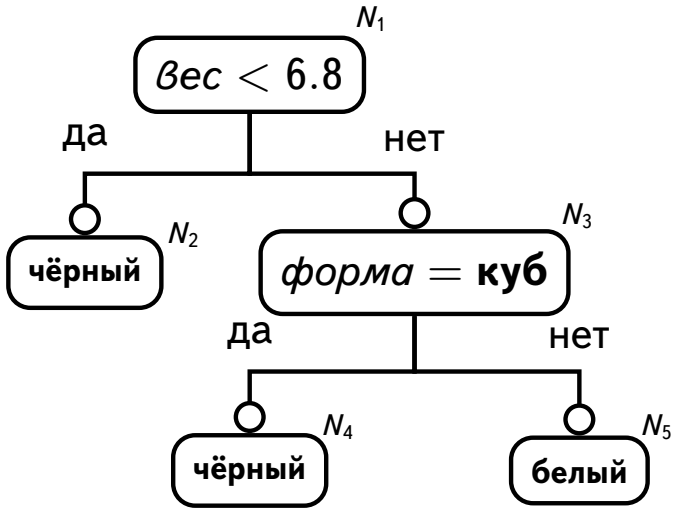
Деревья решений

Определение дерева решений

Качественный признак принимает свои значения из некоторого конечного множества. Для него проверяется принадлежность некоторому фиксированному подмножеству этих допустимых значений.

Корневой вершине N_1 приписан предикат, проверяющий количественный признак *вес*. В качестве порогового значения выступает число 6.8.



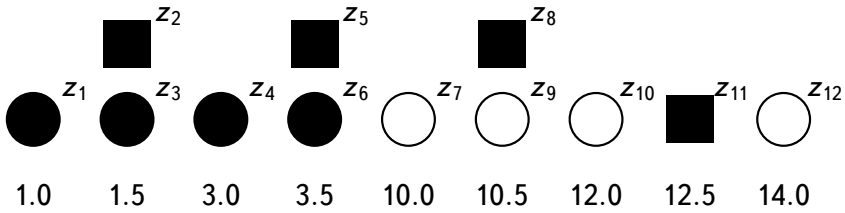


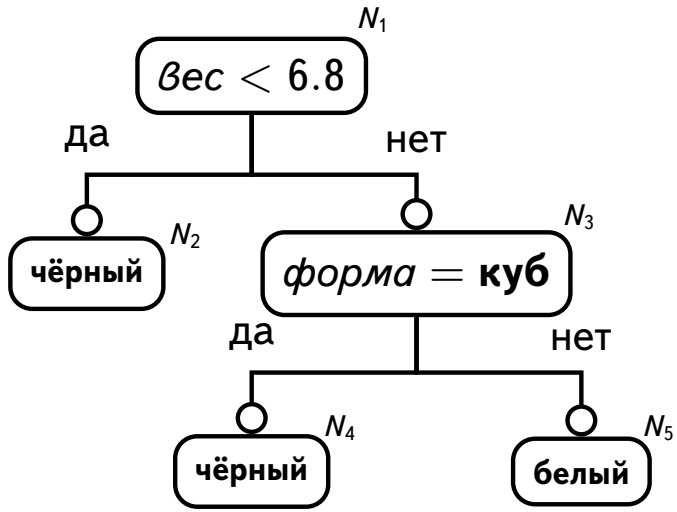
Деревья решений

Определение дерева решений

Внутренней вершине N_3 приписан предикат, проверяющий значение качественного признака *форма* на его принадлежность одноэлементному множеству $\{\textbf{куб}\}$.

Листовым вершинам N_2, N_4 и N_5 приписаны, соответственно, метки **чёрный**, **чёрный** и **белый**.



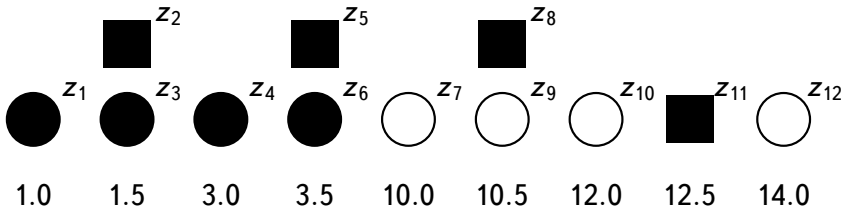


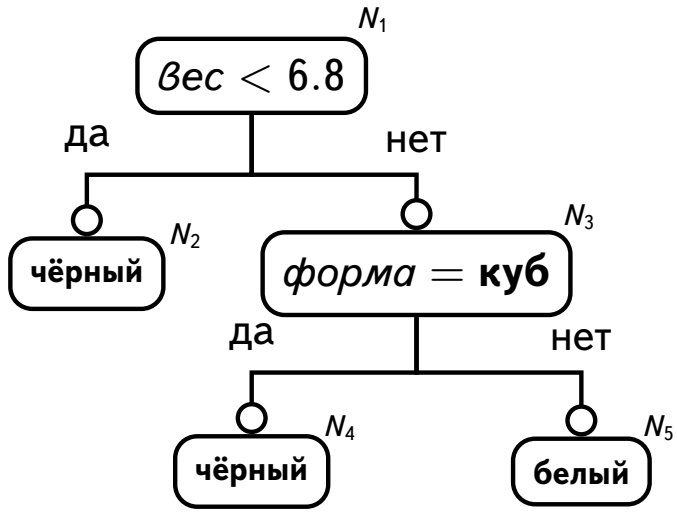
Деревья решений

Определение дерева решений

Каждому объекту $x \in X$ однозначно соответствует некоторый путь, начинающийся в корне дерева и заканчивающийся в его листовой вершине.

Тем самым, задаётся гипотеза $h : X \longrightarrow Y$, где в качестве $h(x)$ выступает метка, приписанная соответствующей листовой вершине.





Деревья решений

Определение дерева решений

Подобный путь строится рекурсивно, начиная с корня дерева.

Деревья решений

Определение дерева решений

Подобный путь строится рекурсивно, начиная с корня дерева.

Предположим, что уже была построена начальная часть такого пути. Концевую вершину начальной части пути назовём текущей.

Деревья решений

Определение дерева решений

Подобный путь строится рекурсивно, начиная с корня дерева.

Предположим, что уже была построена начальная часть такого пути. Концевую вершину начальной части пути назовём текущей.

Если текущая вершина является листовой, то построение завершается.

Деревья решений

Определение дерева решений

Подобный путь строится рекурсивно, начиная с корня дерева.

Предположим, что уже была построена начальная часть такого пути. Концевую вершину начальной части пути назовём текущей.

Если текущая вершина является листовой, то построение завершается.

В противном случае текущая вершина является внутренней, и у неё имеется левый и правый потомок.

Деревья решений

Определение дерева решений

Подобный путь строится рекурсивно, начиная с корня дерева.

Предположим, что уже была построена начальная часть такого пути. Концевую вершину начальной части пути назовём текущей.

Если текущая вершина является листовой, то построение завершается.

В противном случае текущая вершина является внутренней, и у неё имеется левый и правый потомок.

С помощью предиката, приписанного текущей вершине, проверяется объект x.

Деревья решений

Определение дерева решений

Подобный путь строится рекурсивно, начиная с корня дерева.

Предположим, что уже была построена начальная часть такого пути. Концевую вершину начальной части пути назовём текущей.

Если текущая вершина является листовой, то построение завершается.

В противном случае текущая вершина является внутренней, и у неё имеется левый и правый потомок.

С помощью предиката, приписанного текущей вершине, проверяется объект x.

Если он проходит проверку, то новой текущей вершиной становится левый потомок, иначе ею становится правый потомок.

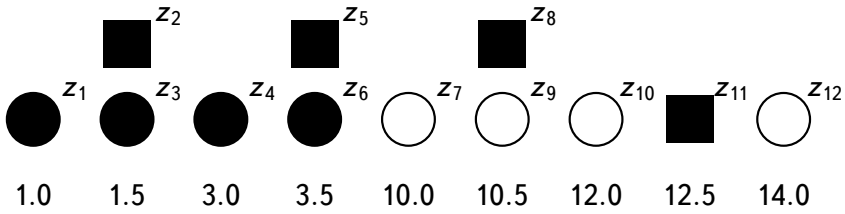
Деревья решений

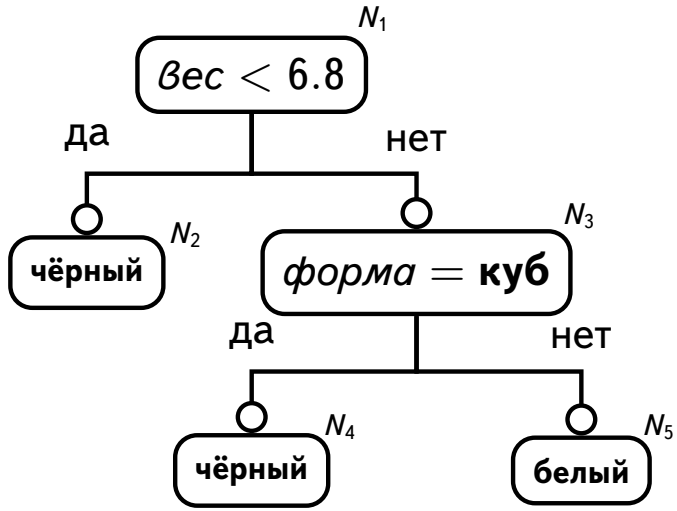
Определение дерева решений

В рассматриваемом примере объекту $(2.0, \textbf{куб})$ в дереве соответствует путь N_1, N_2, а объекту $(11.5, \textbf{куб})$ соответствует путь N_1, N_3, N_4.

Для этих двух объектов будет вычислена метка **чёрный.**

Объекту $(11.5, \textbf{шар})$ в дереве соответствует путь N_1, N_3, N_5, и для него будет вычислена метка **белый.**

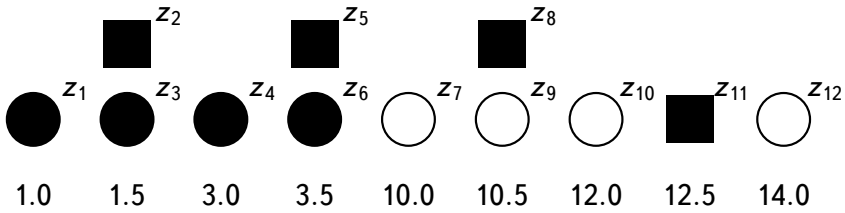


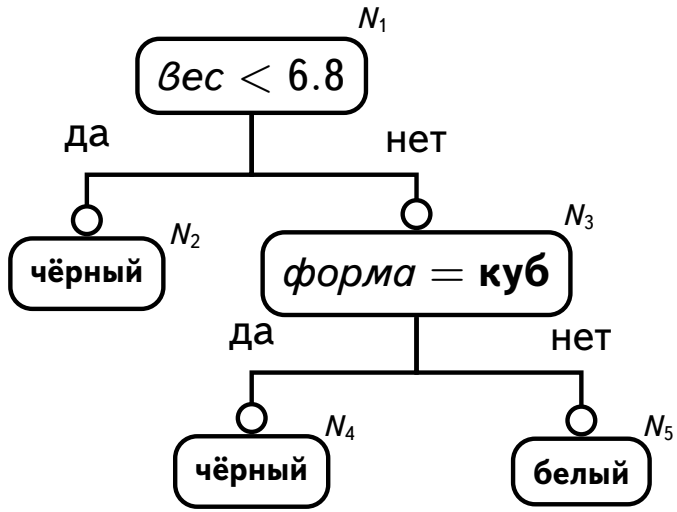


Деревья решений

Определение дерева решений

Заметим также, что в рассматриваемом примере для любого объекта из обучающей выборки приведённое дерево решений корректно вычисляет его метку.





Содержание

Деревья решений

Процедура построения дерева решений

Процедура построения дерева решений носит рекурсивный характер и начинается с его корня.

Деревья решений

Процедура построения дерева решений

Процедура построения дерева решений носит рекурсивный характер и начинается с его корня.

В процессе построения каждой вершине ставится в соответствие набор примеров.

Деревья решений

Процедура построения дерева решений

Процедура построения дерева решений носит рекурсивный характер и начинается с его корня.

В процессе построения каждой вершине ставится в соответствие набор примеров.

На основе этого набора принимается решение о том, будет ли эта вершина внутренней или листовой.

Деревья решений

Процедура построения дерева решений

Процедура построения дерева решений носит рекурсивный характер и начинается с его корня.

В процессе построения каждой вершине ставится в соответствие набор примеров.

На основе этого набора принимается решение о том, будет ли эта вершина внутренней или листовой.

Если вершина становится внутренней, то её набор разделяется на два поднабора, соответственно, для её будущего левого и правого потомка.

Деревья решений

Процедура построения дерева решений

Процедура построения дерева решений носит рекурсивный характер и начинается с его корня.

В процессе построения каждой вершине ставится в соответствие набор примеров.

На основе этого набора принимается решение о том, будет ли эта вершина внутренней или листовой.

Если вершина становится внутренней, то её набор разделяется на два поднабора, соответственно, для её будущего левого и правого потомка.

После этого, процедура построения рекурсивно применяется к этим потомкам.

Деревья решений

Процедура построения дерева решений

Процедура построения дерева решений носит рекурсивный характер и начинается с его корня.

В процессе построения каждой вершине ставится в соответствие набор примеров.

На основе этого набора принимается решение о том, будет ли эта вершина внутренней или листовой.

Если вершина становится внутренней, то её набор разделяется на два поднабора, соответственно, для её будущего левого и правого потомка.

После этого, процедура построения рекурсивно применяется к этим потомкам.

Корню дерева соответствует исходная обучающая выборка.

Деревья решений

Процедура построения дерева решений

Опишем механизм выбора предиката для внутренней вершины.

Для этого нам потребуется ввести понятие индекса неоднородности набора примеров.

Деревья решений

Процедура построения дерева решений

Опишем механизм выбора предиката для внутренней вершины.

Для этого нам потребуется ввести понятие индекса неоднородности набора примеров.

Интуитивно, этот индекс должен принимать малые значения для наборов, в которых объекты преимущественно принадлежат одному классу.

Существует несколько видов индексов неоднородности.

Деревья решений

Процедура построения дерева решений

Одним из них является индекс Джини, который вычисляется по формуле

$$\gamma_{\mathrm{gini}}(\mathbf{z}) \coloneqq 1 - p_{\mathrm{w}}(\mathbf{z})^2 - p_{\mathrm{b}}(\mathbf{z})^2 \qquad (\mathbf{z} \in Z^*),$$

где $p_{\mathrm{w}}(\mathbf{z})$ – доля объектов в наборе $\mathbf{z}$ с меткой **белый**, а $p_{\mathrm{b}}(\mathbf{z})$ – доля объектов с меткой **чёрный**.

Деревья решений

Процедура построения дерева решений

Одним из них является индекс Джини, который вычисляется по формуле

$$\gamma_{\mathrm{gini}}(\mathbf{z}) \coloneqq 1 - p_{\mathrm{w}}(\mathbf{z})^2 - p_{\mathrm{b}}(\mathbf{z})^2 \qquad (\mathbf{z} \in Z^*),$$

где $p_{\mathrm{w}}(\mathbf{z})$ – доля объектов в наборе $\mathbf{z}$ с меткой **белый**, а $p_{\mathrm{b}}(\mathbf{z})$ – доля объектов с меткой **чёрный**.

Если всем объектам в наборе примеров приписана одна и та же метка, то индекс Джини примет своё минимальное значение 0.

Деревья решений

Процедура построения дерева решений

Одним из них является индекс Джини, который вычисляется по формуле

$$\gamma_{\mathrm{gini}}(\mathbf{z}) \coloneqq 1 - p_{\mathrm{w}}(\mathbf{z})^2 - p_{\mathrm{b}}(\mathbf{z})^2 \qquad (\mathbf{z} \in Z^*),$$

где $p_{\mathrm{w}}(\mathbf{z})$ – доля объектов в наборе $\mathbf{z}$ с меткой **белый**, а $p_{\mathrm{b}}(\mathbf{z})$ – доля объектов с меткой **чёрный**.

Если всем объектам в наборе примеров приписана одна и та же метка, то индекс Джини примет своё минимальное значение 0.

Если в наборе примеров поровну объектов с метками **белый** и **чёрный**, то индекс Джини примет своё максимальное значение $\frac{1}{2}$.

Деревья решений

Процедура построения дерева решений

С помощью предиката, приписанного внутренней вершине, набор примеров $\mathbf{z}$ может быть разбит на два поднабора $\mathbf{z}^l$ и $\mathbf{z}^r$.

Деревья решений

Процедура построения дерева решений

С помощью предиката, приписанного внутренней вершине, набор примеров $\mathbf{z}$ может быть разбит на два поднабора $\mathbf{z}^l$ и $\mathbf{z}^r$.

Набор $\mathbf{z}^l$ состоит из всех примеров, которые проходят проверку с помощью этого предиката, а набор $\mathbf{z}^r$ образуют оставшиеся примеры.

Деревья решений

Процедура построения дерева решений

С помощью предиката, приписанного внутренней вершине, набор примеров $\mathbf{z}$ может быть разбит на два поднабора $\mathbf{z}^l$ и $\mathbf{z}^r$.

Набор $\mathbf{z}^l$ состоит из всех примеров, которые проходят проверку с помощью этого предиката, а набор $\mathbf{z}^r$ образуют оставшиеся примеры.

Впоследствии набор $\mathbf{z}^l$ будет приписан левому потомку, а набор $\mathbf{z}^r$ – правому потомку рассматриваемой вершины.

Деревья решений

Процедура построения дерева решений

С помощью предиката, приписанного внутренней вершине, набор примеров $\mathbf{z}$ может быть разбит на два поднабора $\mathbf{z}^l$ и $\mathbf{z}^r$.

Набор $\mathbf{z}^l$ состоит из всех примеров, которые проходят проверку с помощью этого предиката, а набор $\mathbf{z}^r$ образуют оставшиеся примеры.

Впоследствии набор $\mathbf{z}^l$ будет приписан левому потомку, а набор $\mathbf{z}^r$ – правому потомку рассматриваемой вершины.

При выборе предиката следует стремиться к тому, чтобы получившиеся поднаборы $\mathbf{z}^l$ и $\mathbf{z}^r$ были максимально однородными.

Деревья решений

Процедура построения дерева решений

Формализовать эту стратегию можно с помощью использования следующей числовой характеристики

$$\Delta_{\text{gini}}(\mathbf{z}, \mathbf{z}^l, \mathbf{z}^r) := \gamma_{\text{gini}}(\mathbf{z}) - \frac{|\mathbf{z}^l|}{|\mathbf{z}|}\gamma_{\text{gini}}(\mathbf{z}^l) - \frac{|\mathbf{z}^r|}{|\mathbf{z}|}\gamma_{\text{gini}}(\mathbf{z}^r),$$

которая называется уменьшением среднего индекса неоднородности.

Деревья решений

Процедура построения дерева решений

Формализовать эту стратегию можно с помощью использования следующей числовой характеристики

$$\Delta_{\text{gini}}(\mathbf{z}, \mathbf{z}^l, \mathbf{z}^r) := \gamma_{\text{gini}}(\mathbf{z}) - \frac{|\mathbf{z}^l|}{|\mathbf{z}|}\gamma_{\text{gini}}(\mathbf{z}^l) - \frac{|\mathbf{z}^r|}{|\mathbf{z}|}\gamma_{\text{gini}}(\mathbf{z}^r),$$

которая называется уменьшением среднего индекса неоднородности.

Эта характеристика используется следующим образом.

Деревья решений

Процедура построения дерева решений

Формализовать эту стратегию можно с помощью использования следующей числовой характеристики

$$\Delta_{\text{gini}}(\mathbf{z}, \mathbf{z}^l, \mathbf{z}^r) := \gamma_{\text{gini}}(\mathbf{z}) - \frac{|\mathbf{z}^l|}{|\mathbf{z}|}\gamma_{\text{gini}}(\mathbf{z}^l) - \frac{|\mathbf{z}^r|}{|\mathbf{z}|}\gamma_{\text{gini}}(\mathbf{z}^r),$$

которая называется уменьшением среднего индекса неоднородности.

Эта характеристика используется следующим образом.

Последовательно просматриваются все признаки.

Деревья решений

Процедура построения дерева решений

Для количественных признаков просматриваются пороговые значения. Заметим, что всегда можно ограничиться конечным множеством таких значений.

Деревья решений

Процедура построения дерева решений

Для количественных признаков просматриваются пороговые значения. Заметим, что всегда можно ограничиться конечным множеством таких значений.

Для качественных признаков просматриваются все подмножества множества принимаемых признаком значений. Таких подмножеств конечное число.

Деревья решений

Процедура построения дерева решений

Для количественных признаков просматриваются пороговые значения. Заметим, что всегда можно ограничиться конечным множеством таких значений.

Для качественных признаков просматриваются все подмножества множества принимаемых признаком значений. Таких подмножеств конечное число.

В результате такого перебора выбирается предикат, для которого характеристика Δ_{gini} примет наибольшее значение.

Деревья решений

Процедура построения дерева решений

Для количественных признаков просматриваются пороговые значения. Заметим, что всегда можно ограничиться конечным множеством таких значений.

Для качественных признаков просматриваются все подмножества множества принимаемых признаком значений. Таких подмножеств конечное число.

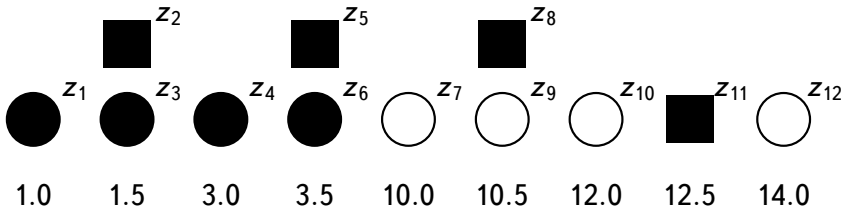
В результате такого перебора выбирается предикат, для которого характеристика Δ_{gini} примет наибольшее значение.

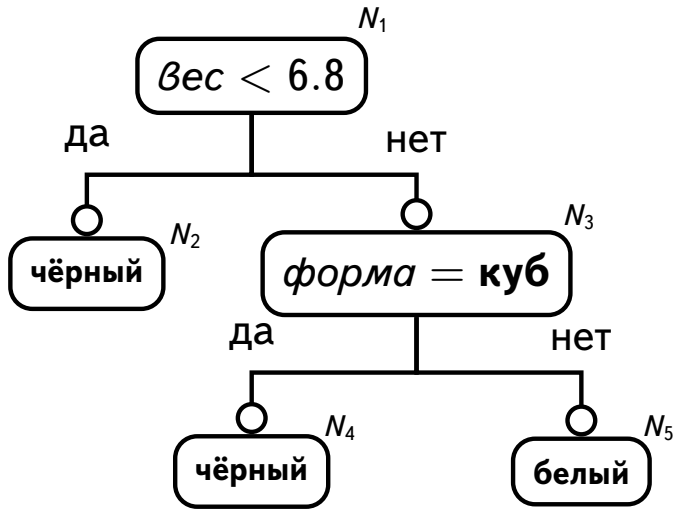
В рассматриваемом примере предикат, приписанный корневой вершине N_1, разбивает обучающую выборку $\mathbf{z} = (z_1, \ldots, z_{12})$ на поднаборы $\mathbf{z}^l = (z_1, \ldots, z_5)$ и $\mathbf{z}^r = (z_6, \ldots, z_{12})$. При этом уменьшение среднего индекса неоднородности достигает значения $\Delta_{\text{gini}} \approx 0.22$.

Деревья решений

Процедура построения дерева решений

Если разбиение сделать менее однородным, например, перенести объект из $\mathbf{z}^l$ в $\mathbf{z}^r$ или перенести объект с меткой **белый** из $\mathbf{z}^r$ в $\mathbf{z}^l$, то величина Δ_{gini} уменьшится и будет равна, соответственно, ≈ 0.16 и ≈ 0.10.



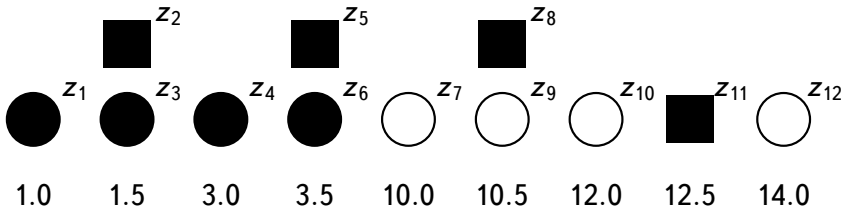


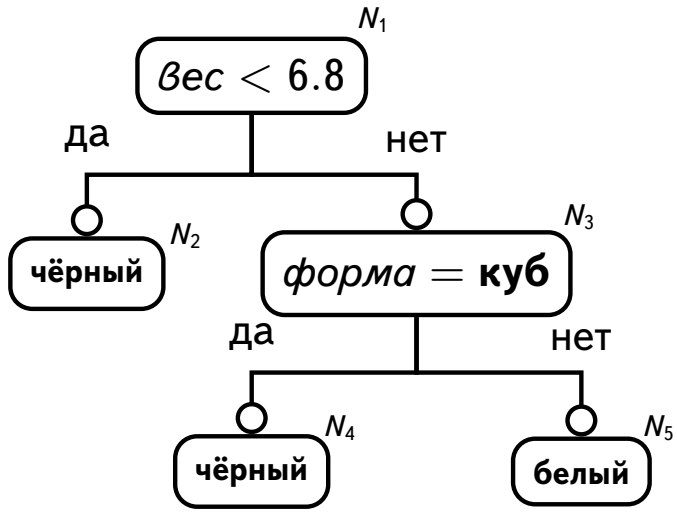
Деревья решений

Процедура построения дерева решений

В процессе построения вершинам N_2, N_4 и N_5 будут приписаны, соответственно, наборы $(z_1, \ldots, z_5)$, (z_8, z_{11}) и $(z_7, z_9, z_{10}, z_{12})$.

Эти наборы однородны. Они содержат только объекты с одинаковыми метками.





Деревья решений

Процедура построения дерева решений

Однородность набора примеров, приписанного вершине, является одним из возможных критериев остановки процедуры построения дерева решений и признания вершины листовой.

В качестве дополнительного критерия может выступать ограничение, накладываемое на высоту строящегося дерева.

Корректировка высоты дерева решений может также осуществляться с помощью процедуры, получившей название подрезка (pruning). Эта процедура применяется к уже целиком построенному дереву решений и заключается в слиянии его листовых вершин.

Содержание

Деревья решений

Некоторые наблюдения и выводы

Рассмотренный пример позволяет сделать несколько интересных наблюдений.

Деревья решений

Некоторые наблюдения и выводы

Рассмотренный пример позволяет сделать несколько интересных наблюдений.

Например, изначальное предположение о существовании целевой функции не всегда является выполнимым.

Деревья решений

Некоторые наблюдения и выводы

Рассмотренный пример позволяет сделать несколько интересных наблюдений.

Например, изначальное предположение о существовании целевой функции не всегда является выполнимым.

Если отказаться от рассмотрения признака форма, то в обучающей выборке появятся два примера, в которых одному объекту с весом 10.5 будут соответствовать две разные метки.

Деревья решений

Некоторые наблюдения и выводы

Рассмотренный пример позволяет сделать несколько интересных наблюдений.

Например, изначальное предположение о существовании целевой функции не всегда является выполнимым.

Если отказаться от рассмотрения признака форма, то в обучающей выборке появятся два примера, в которых одному объекту с весом 10.5 будут соответствовать две разные метки.

Безусловно, даже в этом случае остаётся определённая закономерность между объектами и их метками, которая может быть приближенно вычислена с помомощью алгоритма обучения.

Деревья решений

Некоторые наблюдения и выводы

В подобной ситуации, вместо предположения о существовании неизвестной целевой функции делается предположение о существовании некоторого неизвестного, но фиксированного вероятностного распределения, заданного на Z.

Деревья решений

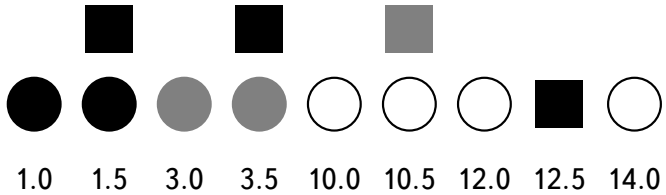
Некоторые наблюдения и выводы

В подобной ситуации, вместо предположения о существовании неизвестной целевой функции делается предположение о существовании некоторого неизвестного, но фиксированного вероятностного распределения, заданного на Z.

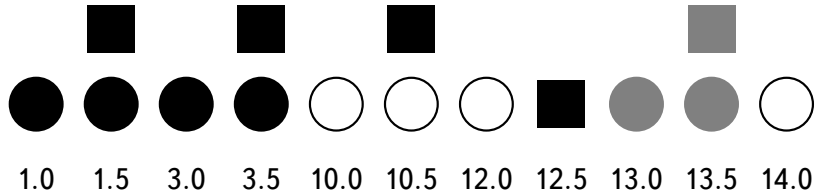
При этом обучающая выборка трактуется как реализация случайной выборки, порождённой этим распределением.

Деревья решений

Некоторые наблюдения и выводы



а)

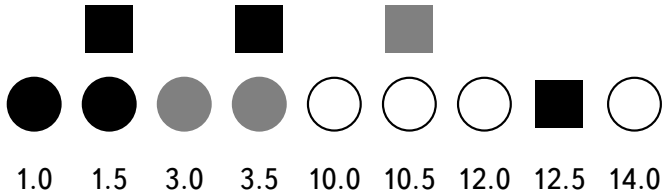


б)

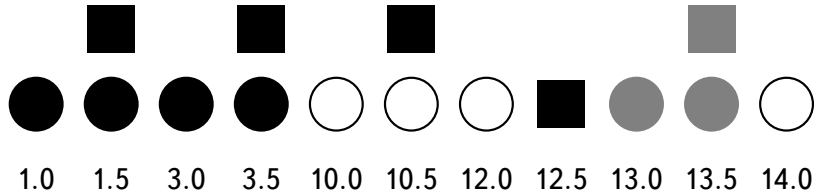
Введём дополнительную метку **серый** и рассмотрим две обучающие выборки $\mathbf{z}'$ (рис. а) и $\mathbf{z}''$ (рис. б), содержащие примеры с такой меткой.

Деревья решений

Некоторые наблюдения и выводы



а)



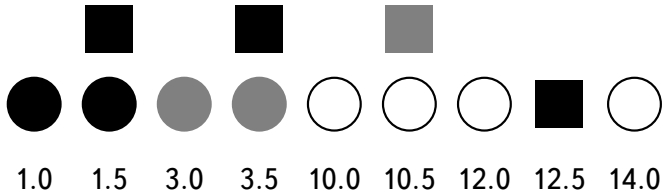
б)

Введём дополнительную метку **серый** и рассмотрим две обучающие выборки $\mathbf{z}'$ (рис. а) и $\mathbf{z}''$ (рис. б), содержащие примеры с такой меткой.

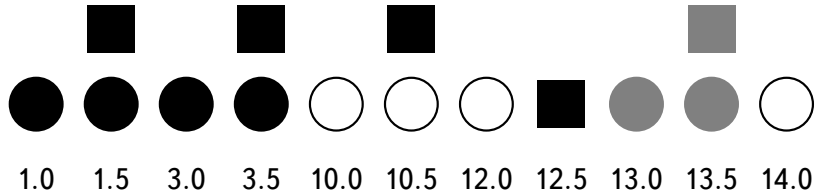
Обучающая выборка $\mathbf{z}'$ получается из исходной выборки $\mathbf{z}$ путём замены у некоторых объектов значения метки **чёрный** на **серый**.

Деревья решений

Некоторые наблюдения и выводы



а)



б)

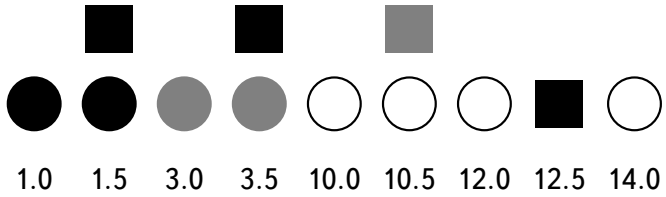
Введём дополнительную метку **серый** и рассмотрим две обучающие выборки $\mathbf{z}'$ (рис. а) и $\mathbf{z}''$ (рис. б), содержащие примеры с такой меткой.

Обучающая выборка $\mathbf{z}'$ получается из исходной выборки $\mathbf{z}$ путём замены у некоторых объектов значения метки **чёрный** на **серый**.

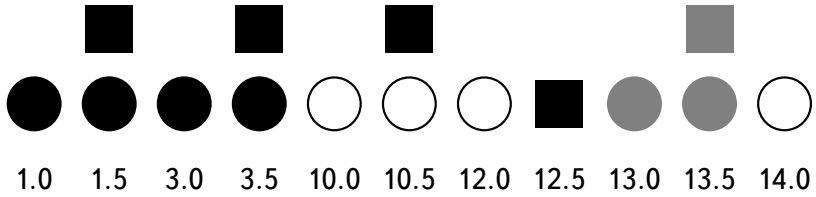
Вспомним, что изначально значения признаков объектов и значения поставленных им в соответствие меток формировались путём измерения соответствующих характеристик у камней из коллекции.

Деревья решений

Некоторые наблюдения и выводы



а)

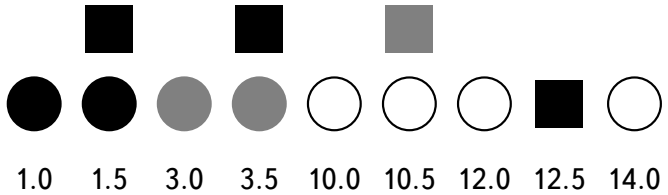


б)

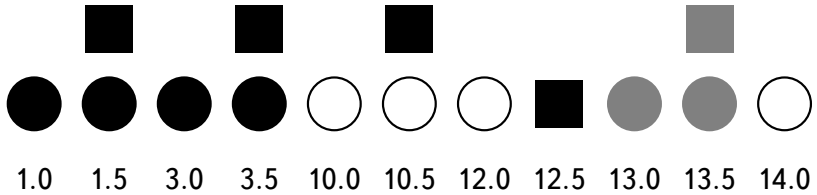
Вполне можно допустить, что в некоторых случаях подобные измерения могут содержать ошибки. Например, истинный цвет камня **серый** мог быть измерен как **чёрный** или наоборот.

Деревья решений

Некоторые наблюдения и выводы



а)



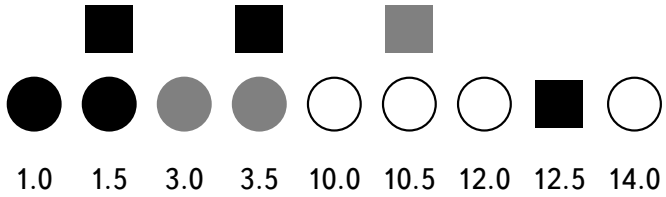
б)

Вполне можно допустить, что в некоторых случаях подобные измерения могут содержать ошибки. Например, истинный цвет камня **серый** мог быть измерен как **чёрный** или наоборот.

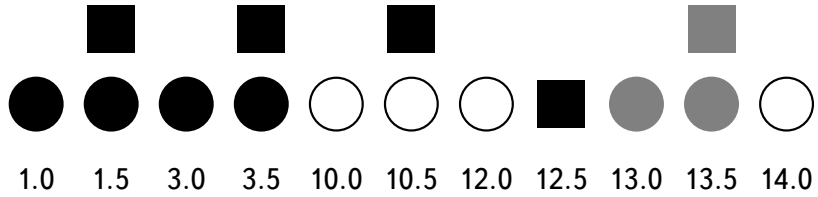
Это показывает, что предположение о том, что обучающая выборка имеет вид $\mathbf{z} = g \circ \mathbf{x}$, где $\mathbf{x} \in X^*$, не всегда может выполняться.

Деревья решений

Некоторые наблюдения и выводы



а)

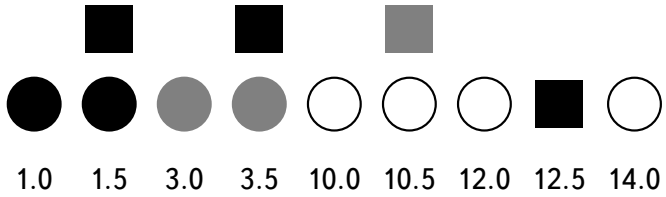


б)

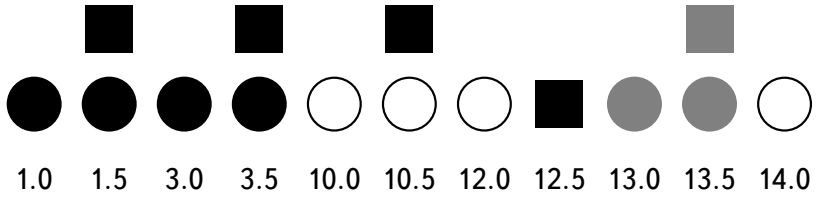
Обучающая выборка $\mathbf{z}''$ получается из исходной выборки $\mathbf{z}$ путём добавления новых объектов с меткой **серый.**

Деревья решений

Некоторые наблюдения и выводы



а)



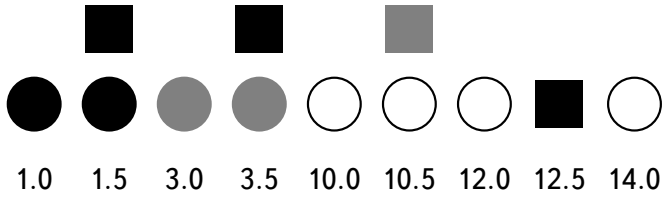
б)

Обучающая выборка $\mathbf{z}''$ получается из исходной выборки $\mathbf{z}$ путём добавления новых объектов с меткой **серый**.

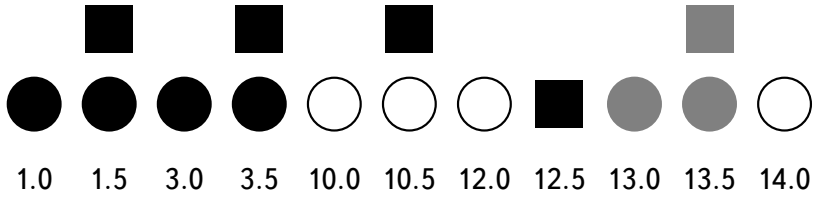
Заметим, что построенное нами дерево решений не учитывает существование таких объектов.

Деревья решений

Некоторые наблюдения и выводы



а)



б)

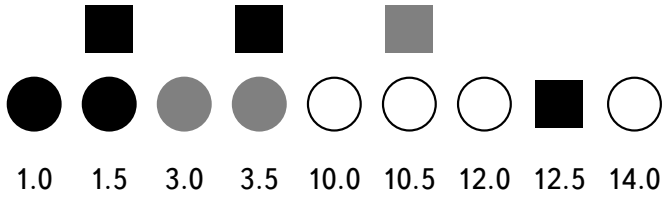
Обучающая выборка $\mathbf{z}''$ получается из исходной выборки $\mathbf{z}$ путём добавления новых объектов с меткой **серый**.

Заметим, что построенное нами дерево решений не учитывает существование таких объектов.

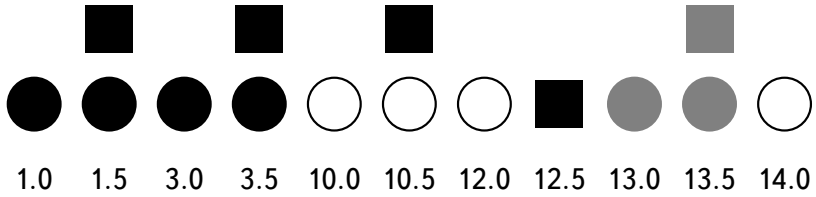
Оно может точно классифицировать объекты с метками **белый** и **чёрный**, и оно всегда будет делать ошибочное предсказание для объектов с меткой **серый**.

Деревья решений

Некоторые наблюдения и выводы



а)



б)

Обучающая выборка $\mathbf{z}''$ получается из исходной выборки $\mathbf{z}$ путём добавления новых объектов с меткой **серый**.

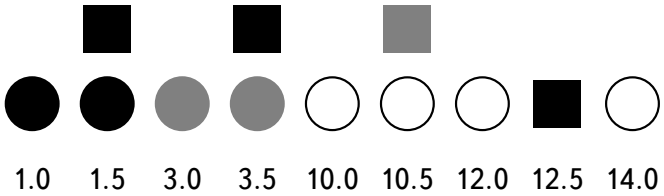
Заметим, что построенное нами дерево решений не учитывает существование таких объектов.

Оно может точно классифицировать объекты с метками **белый** и **чёрный**, и оно всегда будет делать ошибочное предсказание для объектов с меткой **серый**.

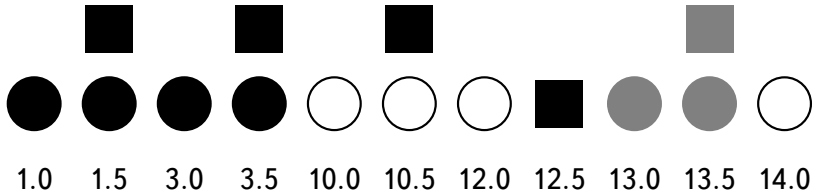
Это позволяет сделать важный заключительный вывод.

Деревья решений

Некоторые наблюдения и выводы



а)

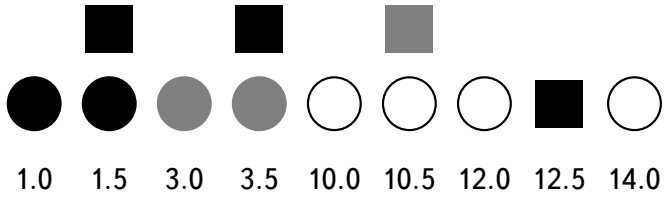


б)

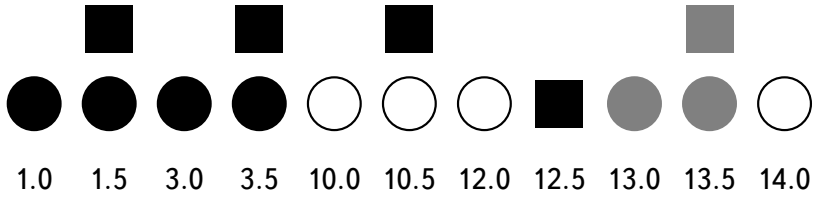
Выбор примеров для формирования обучающей выборки не может быть произвольным.

Деревья решений

Некоторые наблюдения и выводы



а)



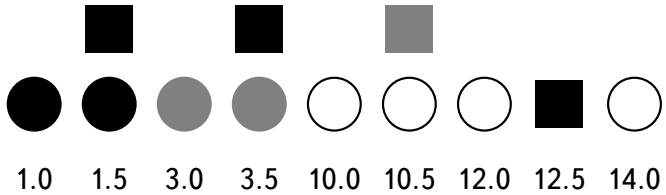
б)

Выбор примеров для формирования обучающей выборки не может быть произвольным.

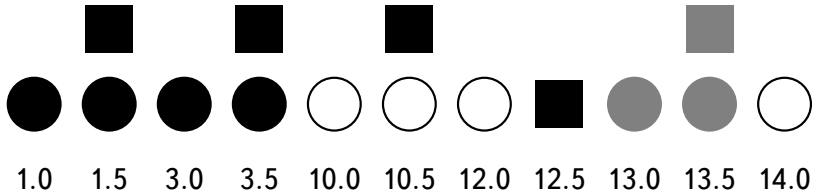
Объекты из обучающей выборки и объекты, метки которых впоследствии будут вычисляться с помощью построенной алгоритмом обучения гипотезы, должны быть похожими. Они должны удовлетворять некоторой общей закономерности.

Деревья решений

Некоторые наблюдения и выводы



а)

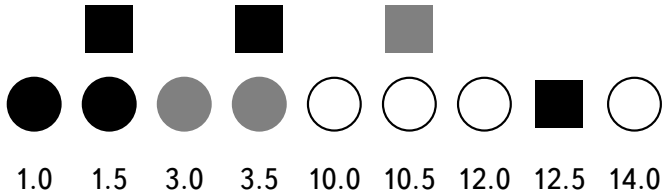


б)

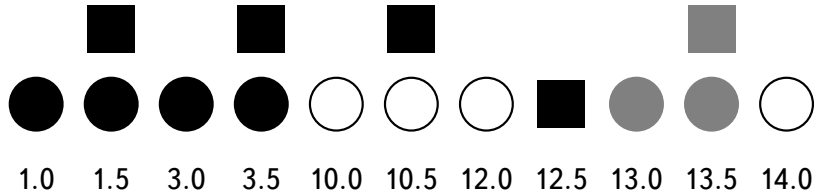
На языке теории вероятностей это условие будет формализовано следующм образом.

Деревья решений

Некоторые наблюдения и выводы



а)



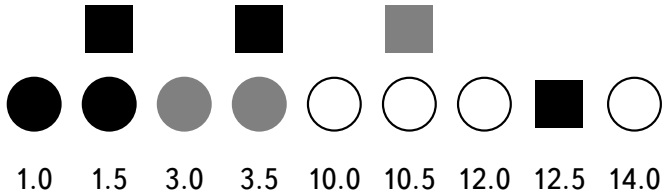
б)

На языке теории вероятностей это условие будет формализовано следующм образом.

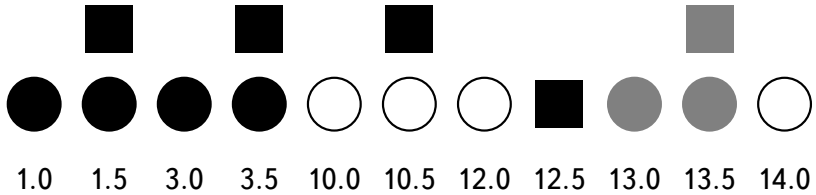
Примеры для обучающей выборки должны выбираться в соответствии с некоторым фиксированным, но возможно неизвестным, вероятностным распределением, заданным на Z.

Деревья решений

Некоторые наблюдения и выводы



а)



б)

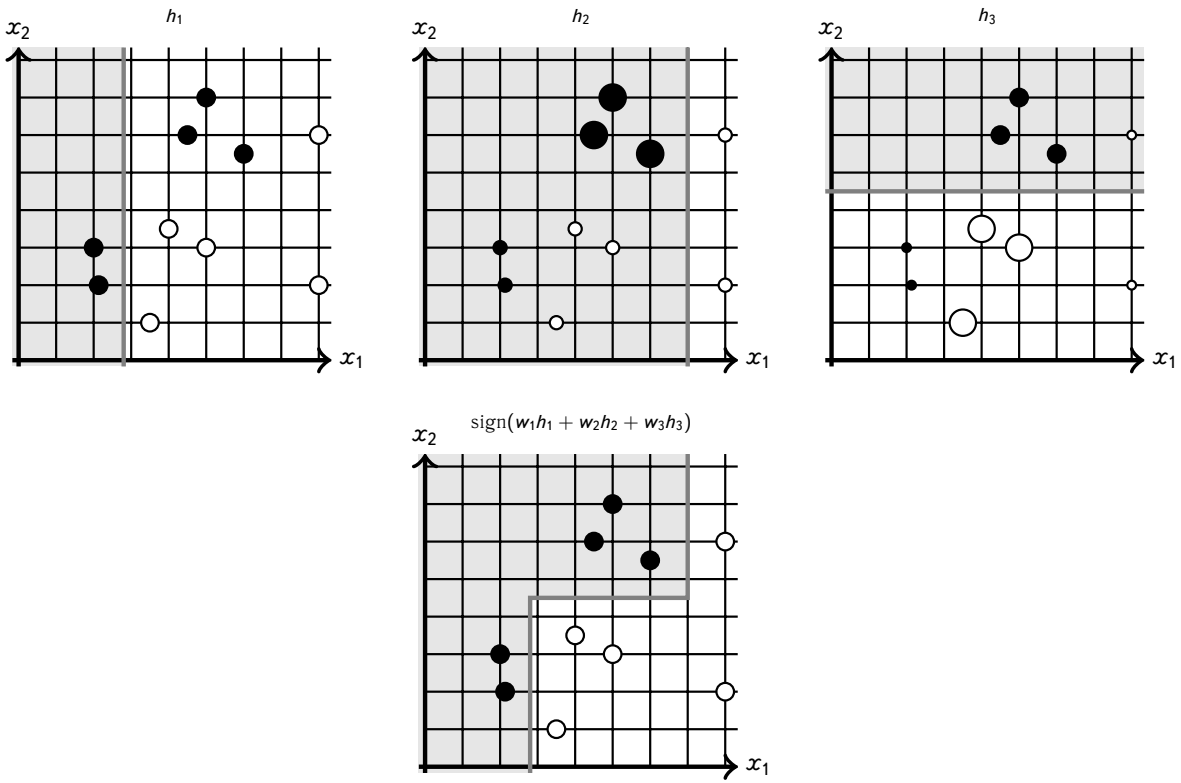
На языке теории вероятностей это условие будет формализовано следующм образом.

Примеры для обучающей выборки должны выбираться в соответствии с некоторым фиксированным, но возможно неизвестным, вероятностным распределением, заданным на Z.

Объекты, к которым будет применяться построенная гипотеза, должны удовлетворять соответствующему маргинальному распределению.

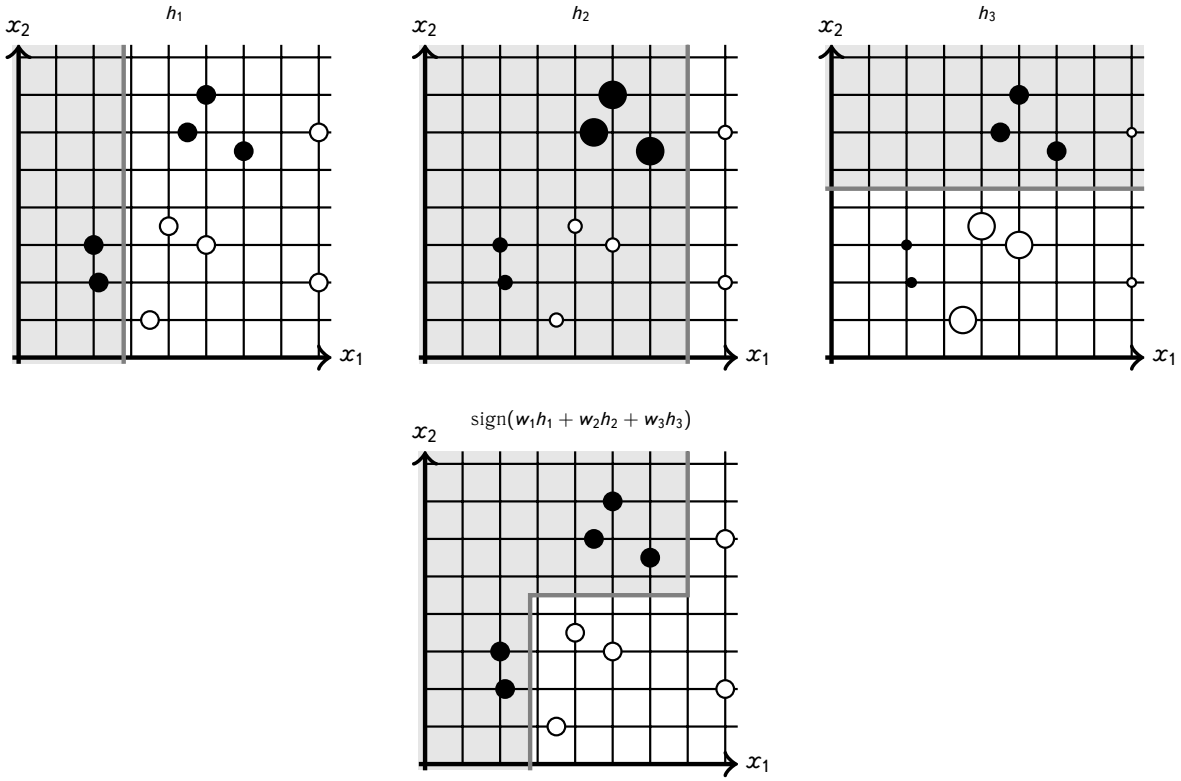
Содержание

Адаптивный бустинг



Прежде, чем перейти к описанию и анализу алгоритма адаптивного бустинга, обсудим пример его работы, схематично изображённый на этом рисунке.

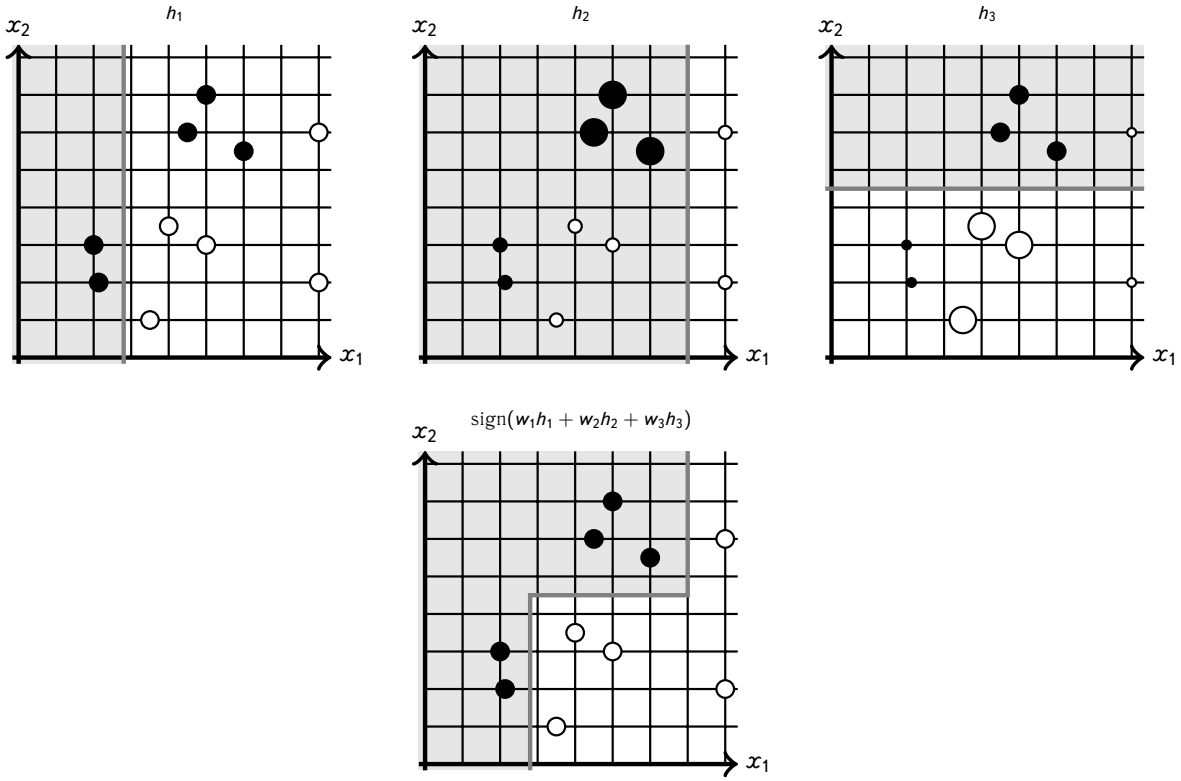
Адаптивный бустинг



В качестве множества объектов X выступает $\mathbb{R}^2$, а в качестве множества меток Y выступает $\{-1, 1\}$.

На рисунке метке 1 соответствует чёрный и серый цвет, а метке -1 соответствует белый цвет.

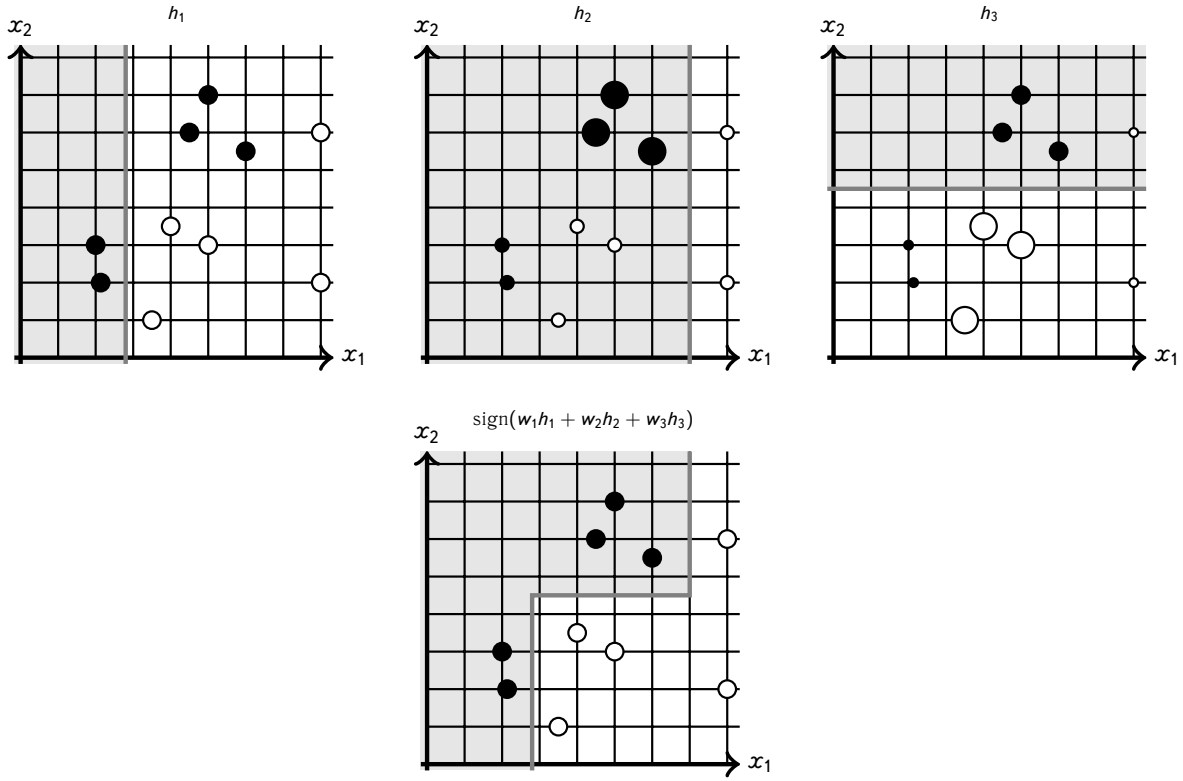
Адаптивный бустинг



Используются два класса гипотез $\mathcal{H}$ и $\mathcal{H}'$.

Класс $\mathcal{H}$ состоит из деревьев решений единичной длины. С помощью таких деревьев можно задавать полуплоскости, границы которых параллельны координатным осям.

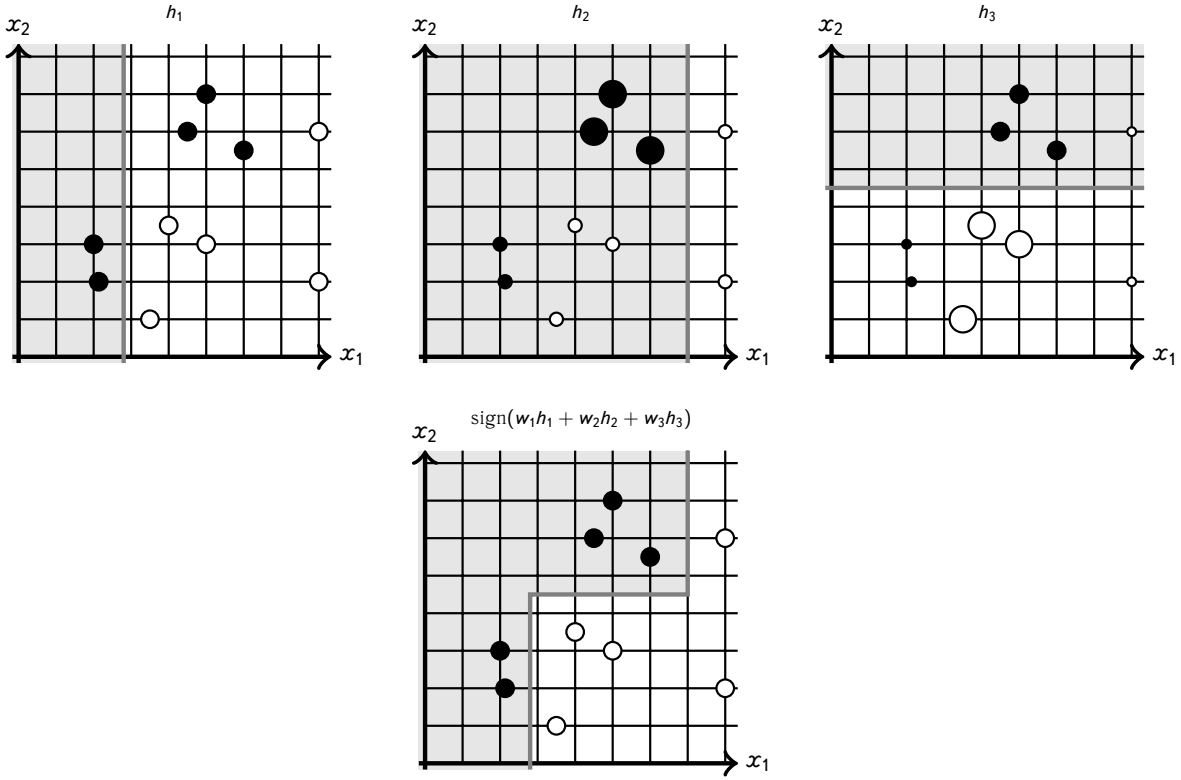
Адаптивный бустинг



Гипотезы h_1, h_2 и h_3, изображённые в верхней части рисунка принадлежат классу $\mathcal{H}$.

Класс $\mathcal{H}'$ состоит из взвешенных сумм рассматриваемых деревьев решений, к которым применяется функция взятия знака. Подобные гипотезы принято называть агрегированными классификаторами.

Адаптивный бустинг

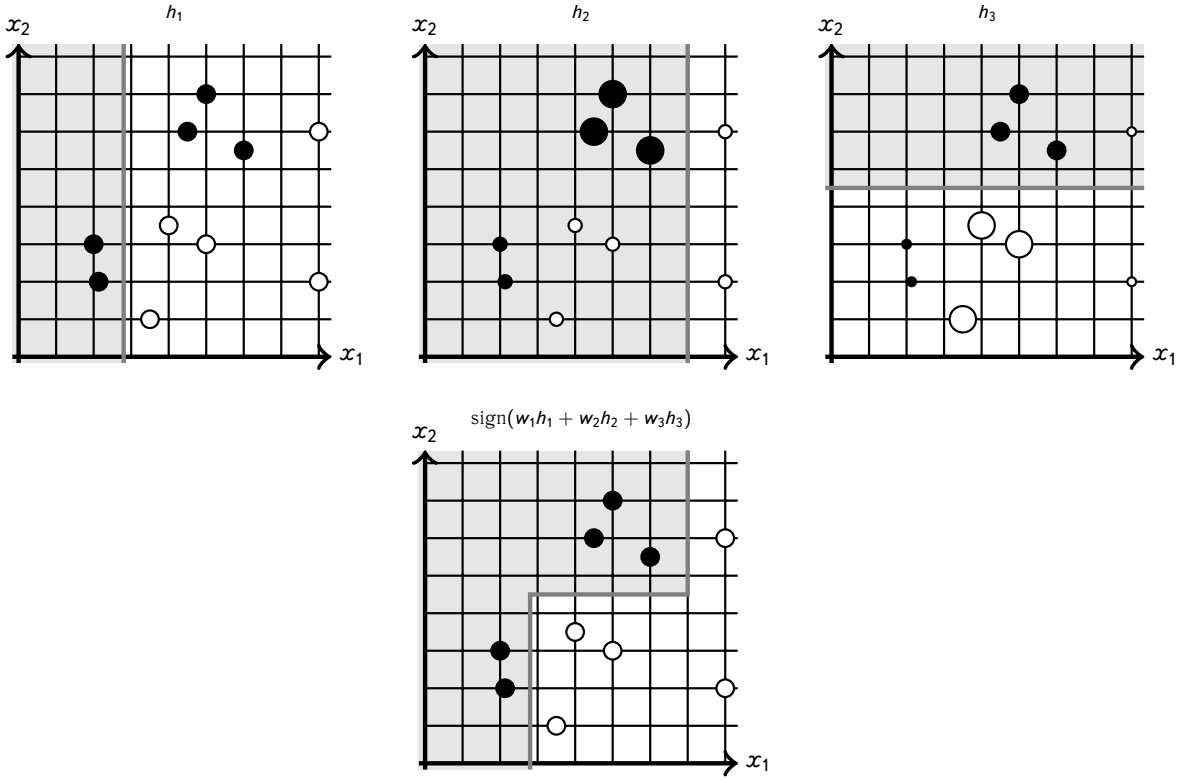


Примеры из обучающей выборки представлены на рисунке кружками.

С помощью рассматриваемых деревьев решений отделить примеры с разными метками, не представляется возможным.

В то же время агрегированному классификатору сделать это удаётся.

Адаптивный бустинг

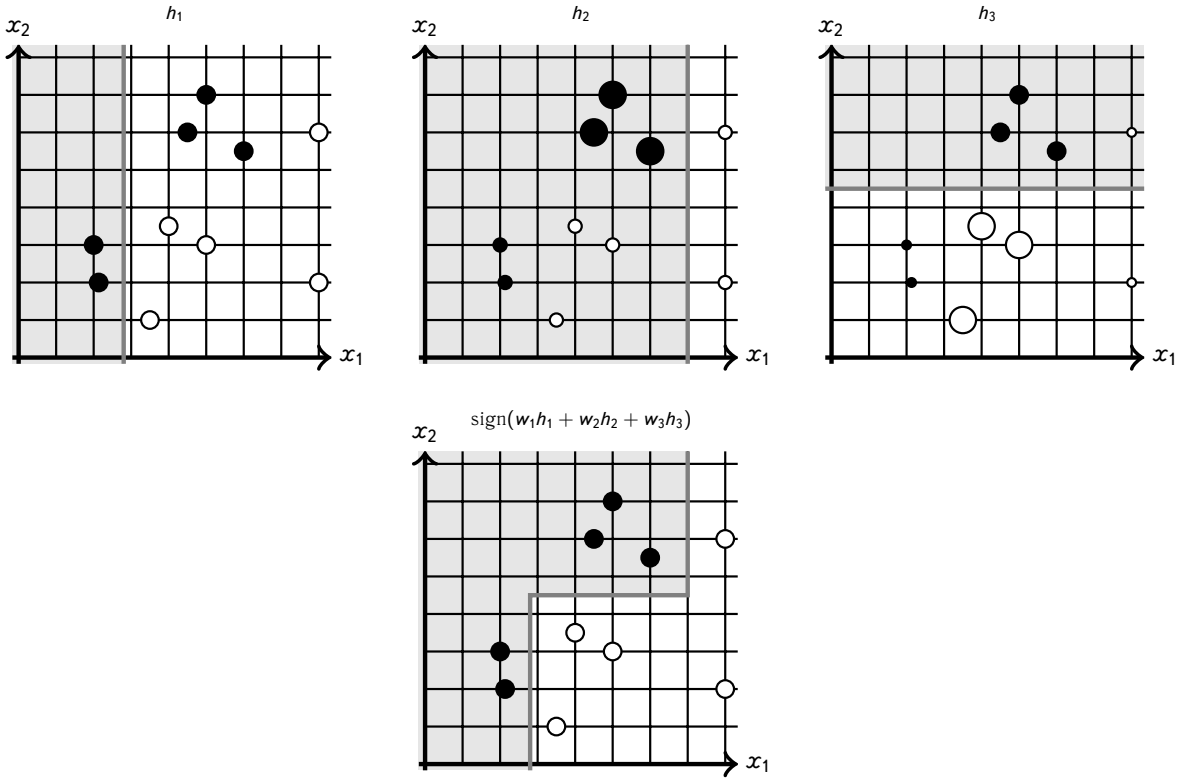


Заметим, что на рисунке примеры изображены кружками разного диаметра.

Размер диаметра отражает вес (значимость) примера.

В левой верхней части рисунка все примеры имеют одинаковый вес.

Адаптивный бустинг



При этом гипотеза h_1 ошибочно классифицирует три примера с меткой 1.

На соседнем рисунке вес этих примеров увеличен. Гипотеза h_2 классифицирует эти примеры правильно.

Содержание

Адаптивный бустинг

Идея алгоритма

Используется некоторый вспомогательный «слабый» алгоритм обучения. В данном случае в его роли выступает метод построения дерева решений с единичной высотой.

Адаптивный бустинг

Идея алгоритма

Используется некоторый вспомогательный «слабый» алгоритм обучения. В данном случае в его роли выступает метод построения дерева решений с единичной высотой.

Слабый алгоритм обучения на вход получает исходную обучающую выборку и набор весов примеров из этой выборки. Строится последовательность гипотез.

Адаптивный бустинг

Идея алгоритма

Используется некоторый вспомогательный «слабый» алгоритм обучения. В данном случае в его роли выступает метод построения дерева решений с единичной высотой.

Слабый алгоритм обучения на вход получает исходную обучающую выборку и набор весов примеров из этой выборки. Строится последовательность гипотез.

После построения очередной гипотезы производится перестройка весов примеров. Увеличиваются веса примеров, на которых эта гипотеза даёт ошибочный ответ.

Адаптивный бустинг

Идея алгоритма

Используется некоторый вспомогательный «слабый» алгоритм обучения. В данном случае в его роли выступает метод построения дерева решений с единичной высотой.

Слабый алгоритм обучения на вход получает исходную обучающую выборку и набор весов примеров из этой выборки. Строится последовательность гипотез.

После построения очередной гипотезы производится перестройка весов примеров. Увеличиваются веса примеров, на которых эта гипотеза даёт ошибочный ответ.

В конце все построенные гипотезы агрегируются в итоговый классификатор.

Адаптивный бустинг

Идея алгоритма

В определении алгоритма обучения не подразумевалось, что обучающая выборка может содержать примеры, которым будут приписаны некоторые веса.

Адаптивный бустинг

Идея алгоритма

В определении алгоритма обучения не подразумевалось, что обучающая выборка может содержать примеры, которым будут приписаны некоторые веса.

Однако, с точки зрения минимизации эмпирического риска алгоритму обучения не должен быть важен порядок примеров в обучающей выборке.

Адаптивный бустинг

Идея алгоритма

В определении алгоритма обучения не подразумевалось, что обучающая выборка может содержать примеры, которым будут приписаны некоторые веса.

Однако, с точки зрения минимизации эмпирического риска алгоритму обучения не должен быть важен порядок примеров в обучающей выборке.

Кроме того, на результат его работы должна влиять только частота появления каждого примера в выборке, а не количество таких появлений.

Адаптивный бустинг

Идея алгоритма

В определении алгоритма обучения не подразумевалось, что обучающая выборка может содержать примеры, которым будут приписаны некоторые веса.

Однако, с точки зрения минимизации эмпирического риска алгоритму обучения не должен быть важен порядок примеров в обучающей выборке.

Кроме того, на результат его работы должна влиять только частота появления каждого примера в выборке, а не количество таких появлений.

Поэтому, например, обучающая выборка с повторяющимися примерами $(z_1, z_1, z_2, z_3, z_1, z_3)$ может быть заменена на уменьшенный набор примеров (z_1, z_2, z_3) и набор частот их появления $(\frac{1}{2}, \frac{1}{6}, \frac{1}{3})$.

Адаптивный бустинг

Идея алгоритма

Введём вспомогательную функцию, которая будет служить обобщением понятия эмпирического риска для наборов, содержащих примеры с весами. По определению положим

$$\widehat{r}(\pi; h, \mathbf{z}) := \pi\{i : h(x_i) \neq y_i\},$$

где $\pi \in \mathcal{M}^1_+(\mathbb{N}_n)$, $h \in \mathcal{H}$ и $\mathbf{z} \in Z^n$ $(n \in \mathbb{N})$.

Адаптивный бустинг

Идея алгоритма

Введём вспомогательную функцию, которая будет служить обобщением понятия эмпирического риска для наборов, содержащих примеры с весами. По определению положим

$$\widehat{r}(\pi; h, \mathbf{z}) := \pi\{i : h(x_i) \neq y_i\},$$

где $\pi \in \mathcal{M}^1_+(\mathbb{N}_n)$, $h \in \mathcal{H}$ и $\mathbf{z} \in Z^n$ $(n \in \mathbb{N})$.

Заметим, что $r(l_{01}; h, \mathbf{z}) = \widehat{r}(\pi_{\text{un}}; h, \mathbf{z})$, где π_{un} – равномерная вероятностная мера на $\mathbb{N}_n$.

Адаптивный бустинг

Описание алгоритма

Настраиваемые параметры:

$\mathcal{A}_{\text{weak}}$ – слабый алгоритм обучения, который по обучающей выборке $\mathbf{z} \in Z^n$ и распределению $\pi \in \mathcal{M}_+^1(\mathbb{N}_n)$ строит гипотезу из $\mathcal{H}$;

$T \in \mathbb{N}$ – число итераций.

Вход:

$\mathbf{z} \in Z^n$.

Основной блок:

$\pi_1 := \pi_{\text{un}}$.

Цикл $(t = 1, \ldots, T)$**:**

с помощью слабого учителя $\mathcal{A}_{\text{weak}}$ на основе обучающей выборки $\mathbf{z}$ и распределения частот примеров π_t построим новую гипотезу

$$h_t := \mathcal{A}_{\text{weak}}(\mathbf{z}, \pi_t);$$

вычислим эмпирический риск построенной гипотезы

$$\varepsilon_t := \widehat{r}(\pi_t; h_t, \mathbf{z});$$

Адаптивный бустинг

Описание алгоритма

вычислим коэффициент, с которым построенная гипотеза будет входить в итоговый классификатор,

$$w_t := \frac{1}{2} \ln \frac{1 - \varepsilon_t}{\varepsilon_t};$$

вычислим нормирующий делитель

$$F_t := \sum_{i=1}^{n} \pi_t\{i\} \cdot \exp\big(- w_t \, y_i \, h_t(x_i)\big);$$

вычислим новое распределение частот

$$\pi_{t+1}\{i\} := \pi_t\{i\} \cdot \frac{\exp\big(- w_t \, y_i \, h_t(x_i)\big)}{F_t} \qquad (i = 1, \ldots, n).$$

Выход:

$$h_{\mathbf{z}} := \operatorname{sign}\left(\sum_{t=1}^{T} w_t \cdot h_t \right).$$

Адаптивный бустинг

Описание алгоритма

Замечание 3.1.

В алгоритме адаптивного бустинга при обновлении частот примеров фигурирует выражение

$$\exp\left(-w_t\, y_i\, h_t(x_i)\right) = \begin{cases} e^{-w_t}, & \text{если} \quad y_i = h_t(x_i); \\ e^{w_t}, & \text{если} \quad y_i \neq h_t(x_i). \end{cases}$$

Адаптивный бустинг

Описание алгоритма

Замечание 3.1.

В алгоритме адаптивного бустинга при обновлении частот примеров фигурирует выражение

$$\exp\big(-w_t\, y_i\, h_t(x_i)\big) = \begin{cases} e^{-w_t}, & \text{если} \quad y_i = h_t(x_i); \\ e^{w_t}, & \text{если} \quad y_i \neq h_t(x_i). \end{cases}$$

Это выражение означает, что если i-й пример не правильно классифицируется с помощью построенной гипотезы h_t, то его вес увеличивается.

Содержание

Адаптивный бустинг

Анализ алгоритма

Теорема 3.1.

Пусть $\mathbf{z} \in Z^n$ $(n \in \mathbb{N})$ и $\gamma \in (0, \frac{1}{2})$. Предположим, что на каждой итерации цикла алгоритма адаптивного бустинга эмпирический риск построенной гипотезы $\varepsilon_t \leqslant \frac{1}{2} - \gamma$ $(t = 1, \ldots, T)$. Тогда

$$r(l_{01}; h_{\mathbf{z}}, \mathbf{z}) \leqslant \exp(-2\gamma^2 T). \tag{1}$$

Адаптивный бустинг

Анализ алгоритма

Теорема 3.1.

Пусть $\mathbf{z} \in Z^n$ $(n \in \mathbb{N})$ и $\gamma \in (0, \frac{1}{2})$. Предположим, что на каждой итерации цикла алгоритма адаптивного бустинга эмпирический риск построенной гипотезы $\varepsilon_t \leqslant \frac{1}{2} - \gamma$ $(t = 1, \ldots, T)$. Тогда

$$r(l_{01}; h_{\mathbf{z}}, \mathbf{z}) \leqslant \exp(-2\gamma^2 T). \tag{1}$$

◀ Обозначим

$$f_0 := 0, \qquad f_t := \sum_{j=1}^{t} w_j h_j, \qquad \widehat{h}_t := \operatorname{sign} \circ f_t \qquad (t = 1, \ldots, T)$$

и заметим, что $h_{\mathbf{z}} = \widehat{h}_T$.

Адаптивный бустинг

Анализ алгоритма

Учитывая неравенство $\mathbf{1}_{\{u \leqslant 0\}} \leqslant e^{-u}$ $(u \in \mathbb{R})$ и предположение о том, что $Y = \{-1, 1\}$, получим

$$r(\widehat{h}_t, \mathbf{z}) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{\{y_i \widehat{h}_t(x_i) \leqslant 0\}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{\{y_i f_t(x_i) \leqslant 0\}} \leqslant \frac{1}{n} \sum_{i=1}^{n} e^{-y_i f_t(x_i)} \qquad (2)$$

$(t = 1, \ldots, T)$.

Адаптивный бустинг

Анализ алгоритма

Учитывая неравенство $\mathbf{1}_{\{u \leqslant 0\}} \leqslant e^{-u}$ $(u \in \mathbb{R})$ и предположение о том, что $Y = \{-1, 1\}$, получим

$$r(\widehat{h}_t, \mathbf{z}) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{\{y_i \widehat{h}_t(x_i) \leqslant 0\}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{\{y_i f_t(x_i) \leqslant 0\}} \leqslant \frac{1}{n} \sum_{i=1}^{n} e^{-y_i f_t(x_i)} \qquad (2)$$

$(t = 1, \ldots, T)$.

Обозначим

$$Z_t := \frac{1}{n} \sum_{i=1}^{n} e^{-y_i f_t(x_i)} \quad (t = 1, \ldots, T).$$

Адаптивный бустинг

Анализ алгоритма

Учитывая неравенство $\mathbf{1}_{\{u \leqslant 0\}} \leqslant e^{-u}$ $(u \in \mathbb{R})$ и предположение о том, что $Y = \{-1, 1\}$, получим

$$r(\widehat{h}_t, \mathbf{z}) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{\{y_i \widehat{h}_t(x_i) \leqslant 0\}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{\{y_i f_t(x_i) \leqslant 0\}} \leqslant \frac{1}{n} \sum_{i=1}^{n} e^{-y_i f_t(x_i)} \qquad (2)$$

$(t = 1, \ldots, T)$.

Обозначим

$$Z_t := \frac{1}{n} \sum_{i=1}^{n} e^{-y_i f_t(x_i)} \quad (t = 1, \ldots, T).$$

Из (2) следует, что для доказательства неравенства (1) достаточно показать, что

$$Z_T \leqslant \exp(-2\gamma^2 T). \qquad (3)$$

Адаптивный бустинг

Анализ алгоритма

Представим

$$Z_T = \frac{Z_T}{Z_0} = \frac{Z_T}{Z_{T-1}} \cdot \frac{Z_{T-1}}{Z_{T-2}} \cdot \ldots \cdot \frac{Z_2}{Z_1} \cdot \frac{Z_1}{Z_0}.$$

Адаптивный бустинг

Анализ алгоритма

Представим

$$Z_T = \frac{Z_T}{Z_0} = \frac{Z_T}{Z_{T-1}} \cdot \frac{Z_{T-1}}{Z_{T-2}} \cdot \ldots \cdot \frac{Z_2}{Z_1} \cdot \frac{Z_1}{Z_0}.$$

Следовательно, неравенство (3) будет выполнено, если

$$\frac{Z_{t+1}}{Z_t} \leqslant e^{-2\gamma^2} \qquad (t = 0, \ldots, T-1). \tag{4}$$

Адаптивный бустинг

Анализ алгоритма

Представим

$$Z_T = \frac{Z_T}{Z_0} = \frac{Z_T}{Z_{T-1}} \cdot \frac{Z_{T-1}}{Z_{T-2}} \cdot \ldots \cdot \frac{Z_2}{Z_1} \cdot \frac{Z_1}{Z_0}.$$

Следовательно, неравенство (3) будет выполнено, если

$$\frac{Z_{t+1}}{Z_t} \leqslant e^{-2\gamma^2} \qquad (t = 0, \ldots, T-1). \tag{4}$$

Докажем представление

$$\pi_{t+1}\{i\} = \frac{e^{-y_i f_t(x_i)}}{\sum\limits_{k=1}^{n} e^{-y_k f_t(x_k)}} =: \frac{e^{-y_i f_t(x_i)}}{Q_t} \qquad (t = 0, \ldots, T-1;\ i = 1, \ldots, n) \tag{5}$$

индукцией по параметру t.

Адаптивный бустинг

Анализ алгоритма

Основание индукции $t = 0$. Имеем

$$\pi_1\{i\} = \frac{e^{-y_i f_0(x_i)}}{\sum\limits_{k=1}^{n} e^{-y_k f_0(x_k)}} = \frac{e^{-y_i \cdot 0}}{\sum\limits_{k=1}^{n} e^{-y_k \cdot 0}} = \frac{1}{n} \quad (i = 1, \ldots, n).$$

Адаптивный бустинг

Анализ алгоритма

Основание индукции $t = 0$. Имеем

$$\pi_1\{i\} = \frac{e^{-y_i f_0(x_i)}}{\sum\limits_{k=1}^{n} e^{-y_k f_0(x_k)}} = \frac{e^{-y_i \cdot 0}}{\sum\limits_{k=1}^{n} e^{-y_k \cdot 0}} = \frac{1}{n} \quad (i = 1, \ldots, n).$$

Индуктивный переход. Предположим, что (5) верно для всех $t < q$ $(q = 1, \ldots, T-1)$. Покажем, что (5) будет верно и для $t = q$.

Адаптивный бустинг

Анализ алгоритма

Запишем

$$\pi_{q+1}\{i\} = \pi_q\{i\}\,\frac{\exp\bigl(-w_q\,y_i\,h_q(x_i)\bigr)}{\sum\limits_{k=1}^{n}\pi_q\{k\}\exp\bigl(-w_q\,y_k\,h_q(x_k)\bigr)}$$

$$= \frac{\exp\bigl(-y_i\,f_{q-1}(x_i)\bigr)}{Q_{q-1}}\cdot\frac{\exp\bigl(-w_q\,y_i\,h_q(x_i)\bigr)}{\sum\limits_{k=1}^{n}\frac{\exp\bigl(-y_k\,f_{q-1}(x_k)\bigr)}{Q_{q-1}}\exp\bigl(-w_q\,y_k\,h_q(x_k)\bigr)}$$

$$= \frac{\exp\bigl(-y_i\,f_{q-1}(x_i) - w_q\,y_i\,h_q(x_i)\bigr)}{\sum\limits_{k=1}^{n}\exp\bigl(-y_k\,f_{q-1}(x_k) - w_q\,y_k\,h_q(x_k)\bigr)}$$

$$= \frac{\exp\bigl(-y_i\,f_q(x_i)\bigr)}{\sum\limits_{k=1}^{n}\exp\bigl(-y_k\,f_q(x_k)\bigr)} \quad (i = 1,\ldots,n).$$

Адаптивный бустинг

Анализ алгоритма

Для каждого $t = 0, \ldots, T-1$ получим

$$\frac{Z_{t+1}}{Z_t} = \frac{\sum\limits_{i=1}^{n} e^{-y_i f_{t+1}(x_i)}}{\sum\limits_{k=1}^{n} e^{-y_k f_t(x_k)}} = \frac{\sum\limits_{i=1}^{n} e^{-y_i f_t(x_i)}\, e^{-y_i w_{t+1} h_{t+1}(x_i)}}{\sum\limits_{k=1}^{n} e^{-y_k f_t(x_k)}}$$

$$= \sum_{i=1}^{n} \pi_{t+1}\{i\}\, e^{-y_i w_{t+1} h_{t+1}(x_i)}$$

$$= e^{-w_{t+1}} \sum_{i: y_i h_{t+1}(x_i) = 1} \pi_{t+1}\{i\} + e^{w_{t+1}} \sum_{i: y_i h_{t+1}(x_i) = -1} \pi_{t+1}\{i\}$$

$$= e^{-w_{t+1}} (1 - \varepsilon_{t+1}) + e^{w_{t+1}} \varepsilon_{t+1} = 2\sqrt{\varepsilon_{t+1}(1 - \varepsilon_{t+1})}.$$

Адаптивный бустинг

Анализ алгоритма

Функция $u \longmapsto u(1-u)$ возрастает на отрезке $\left[0, \frac{1}{2}\right]$.

Адаптивный бустинг

Анализ алгоритма

Функция $u \longmapsto u(1-u)$ возрастает на отрезке $\left[0, \frac{1}{2}\right]$.

Поэтому

$$2\sqrt{\varepsilon_{t+1}(1-\varepsilon_{t+1})} \leqslant 2\sqrt{\left(\frac{1}{2}-\gamma\right)\left(\frac{1}{2}+\gamma\right)} = \sqrt{1-4\gamma^2}$$

$$\leqslant \left| \begin{array}{c} 1+u \leqslant e^u \\ (u \in \mathbb{R}) \end{array} \right| \leqslant e^{-\frac{4\gamma^2}{2}} = e^{-2\gamma^2},$$

Адаптивный бустинг

Анализ алгоритма

Функция $u \longmapsto u(1-u)$ возрастает на отрезке $\left[0, \frac{1}{2}\right]$.

Поэтому

$$2\sqrt{\varepsilon_{t+1}(1-\varepsilon_{t+1})} \leqslant 2\sqrt{\left(\frac{1}{2}-\gamma\right)\left(\frac{1}{2}+\gamma\right)} = \sqrt{1-4\gamma^2}$$
$$\leqslant \left| \begin{array}{c} 1+u \leqslant e^u \\ (u \in \mathbb{R}) \end{array} \right| \leqslant e^{-\frac{4\gamma^2}{2}} = e^{-2\gamma^2},$$

а значит верно искомое неравенство (1).

■

Содержание

Линейные модели

В 1958 году американский физиолог Ф. Розенблатт, изучая феномен восприятия, предложил математическую модель нейрона, которая получила название персептрон.

Линейные модели

В 1958 году американский физиолог Ф. Розенблатт, изучая феномен восприятия, предложил математическую модель нейрона, которая получила название персептрон.

Персептрон можно рассматривать как некоторое техническое устройство, имеющее несколько пронумерованных входов и один выход. На каждый вход j $(j = 1, \ldots, m)$ подаётся сигнал в виде числового значения $u_j \in \mathbb{R}$. Эти входные сигналы преобразуются в один выходной сигнал вида

Линейные модели

В 1958 году американский физиолог Ф. Розенблатт, изучая феномен восприятия, предложил математическую модель нейрона, которая получила название персептрон.

Персептрон можно рассматривать как некоторое техническое устройство, имеющее несколько пронумерованных входов и один выход. На каждый вход j $(j = 1, \ldots, m)$ подаётся сигнал в виде числового значения $u_j \in \mathbb{R}$. Эти входные сигналы преобразуются в один выходной сигнал вида

$$\operatorname{sign}\left(w_0 + \sum_{j=1}^{m} w_j u_j\right). \tag{6}$$

Линейные модели

Как мы видим в (6), сигнал, поступивший на j-й вход, усиливается или ослабляется с помощью некоторого фиксированного для этого входа *веса* $w_j \in \mathbb{R}$.

Линейные модели

Как мы видим в (6), сигнал, поступивший на j-й вход, усиливается или ослабляется с помощью некоторого фиксированного для этого входа *веса* $w_j \in \mathbb{R}$.

После этого, взвешенная сумма входных сигналов сравнивается с фиксированным пороговым значением $w_0 \in \mathbb{R}$.

Линейные модели

Как мы видим в (6), сигнал, поступивший на j-й вход, усиливается или ослабляется с помощью некоторого фиксированного для этого входа *веса* $w_j \in \mathbb{R}$.

После этого, взвешенная сумма входных сигналов сравнивается с фиксированным пороговым значением $w_0 \in \mathbb{R}$.

В результате такого сравнения персептрон формирует выходной сигнал равный 1 или -1.

Линейные модели

Как мы видим в (6), сигнал, поступивший на j-й вход, усиливается или ослабляется с помощью некоторого фиксированного для этого входа *веса* $w_j \in \mathbb{R}$.

После этого, взвешенная сумма входных сигналов сравнивается с фиксированным пороговым значением $w_0 \in \mathbb{R}$.

В результате такого сравнения персептрон формирует выходной сигнал равный 1 или -1.

В контексте работы персептрона функцию sign принято называть функцией активации.

Линейные модели

Понятие персептрона позволяет рассмотреть задачу классификации, где в качестве множества объектов X выступает пространство $\mathbb{R}^m$, в качестве множества меток Y выступает множество $\{-1, 1\}$, а класс гипотез образуют все функции, которые могут быть вычислены с помощью некоторого персептрона.

Линейные модели

Понятие персептрона позволяет рассмотреть задачу классификации, где в качестве множества объектов X выступает пространство $\mathbb{R}^m$, в качестве множества меток Y выступает множество $\{-1, 1\}$, а класс гипотез образуют все функции, которые могут быть вычислены с помощью некоторого персептрона.

Использование функции потерь l_{01} гарантирует наличие метода минимизации эмпирического риска у этой задачи.

Линейные модели

Понятие персептрона позволяет рассмотреть задачу классификации, где в качестве множества объектов X выступает пространство $\mathbb{R}^m$, в качестве множества меток Y выступает множество $\{-1, 1\}$, а класс гипотез образуют все функции, которые могут быть вычислены с помощью некоторого персептрона.

Использование функции потерь l_{01} гарантирует наличие метода минимизации эмпирического риска у этой задачи.

Ранее, мы не акцентировали особого внимания на вычислительном аспекте решения задачи обучения.

Линейные модели

Понятие персептрона позволяет рассмотреть задачу классификации, где в качестве множества объектов X выступает пространство $\mathbb{R}^m$, в качестве множества меток Y выступает множество $\{-1, 1\}$, а класс гипотез образуют все функции, которые могут быть вычислены с помощью некоторого персептрона.

Использование функции потерь l_{01} гарантирует наличие метода минимизации эмпирического риска у этой задачи.

Ранее, мы не акцентировали особого внимания на вычислительном аспекте решения задачи обучения.

Построение перспетрона с помощью метода эмпирического риска, использующего функцию потерь l_{01}, является сложной комбинаторной задачей.

Линейные модели

Поэтому важно, либо исследовать частные случаи, когда поиск решения осуществляется с приемлемыми вычислительными затратами, либо перейти к другим постановкам задач с изначально более слабыми условиями.

Содержание

Линейные модели

Алгоритм Розенблатта

Нашей ближайшей целью является изложение классического результата, касающегося оценки применимости и сложности алгоритма Розенблатта, предназначенного для обучения персептрона.

Линейные модели

Алгоритм Розенблатта

Нашей ближайшей целью является изложение классического результата, касающегося оценки применимости и сложности алгоритма Розенблатта, предназначенного для обучения персептрона.

Этот результат известен как теорема Новикова. В настоящее время он прежде всего интересен тем, что отражает идеи и подходы начального этапа в области исследований проблем искусственного интеллекта.

Линейные модели

Алгоритм Розенблатта

Нашей ближайшей целью является изложение классического результата, касающегося оценки применимости и сложности алгоритма Розенблатта, предназначенного для обучения персептрона.

Этот результат известен как теорема Новикова. В настоящее время он прежде всего интересен тем, что отражает идеи и подходы начального этапа в области исследований проблем искусственного интеллекта.

Далее, нам будет удобно использовать следующее определение персептрона.

Линейные модели

Алгоритм Розенблатта

Нашей ближайшей целью является изложение классического результата, касающегося оценки применимости и сложности алгоритма Розенблатта, предназначенного для обучения персептрона.

Этот результат известен как теорема Новикова. В настоящее время он прежде всего интересен тем, что отражает идеи и подходы начального этапа в области исследований проблем искусственного интеллекта.

Далее, нам будет удобно использовать следующее определение персептрона.

Под персептроном будут пониматься функции вида

$$\mathbf{u} \longmapsto \operatorname{sign}\left(\langle \mathbf{w}, \mathbf{u} \rangle\right) \qquad \left(\mathbf{w}, \mathbf{u} \in \mathbb{R}^m\right). \tag{7}$$

Линейные модели

Алгоритм Розенблатта

С формальной точки зрения два определения (6) и (7) отличаются друг от друга. В первом случае дополнительно фигурирует пороговое значение w_0.

Линейные модели

Алгоритм Розенблатта

С формальной точки зрения два определения (6) и (7) отличаются друг от друга. В первом случае дополнительно фигурирует пороговое значение w_0.

Однако это отличие не носит принципиального характера.

Линейные модели

Алгоритм Розенблатта

С формальной точки зрения два определения (6) и (7) отличаются друг от друга. В первом случае дополнительно фигурирует пороговое значение w_0.

Однако это отличие не носит принципиального характера.

Действительно, если в определении (6) положить пороговое значение равным нулю, то мы придём к определению (7).

Линейные модели

Алгоритм Розенблатта

С формальной точки зрения два определения (6) и (7) отличаются друг от друга. В первом случае дополнительно фигурирует пороговое значение w_0.

Однако это отличие не носит принципиального характера.

Действительно, если в определении (6) положить пороговое значение равным нулю, то мы придём к определению (7).

С другой стороны, при использовании определения (7) можно ограничиться рассмотрением только тех объектов, у которых первая координата равна единице. В этом случае весовой коэффициент, приписанный первому входу персептрона, будет выполнять роль порогового значения.

Линейные модели

Алгоритм Розенблатта

С формальной точки зрения два определения (6) и (7) отличаются друг от друга. В первом случае дополнительно фигурирует пороговое значение w_0.

Однако это отличие не носит принципиального характера.

Действительно, если в определении (6) положить пороговое значение равным нулю, то мы придём к определению (7).

С другой стороны, при использовании определения (7) можно ограничиться рассмотрением только тех объектов, у которых первая координата равна единице. В этом случае весовой коэффициент, приписанный первому входу персептрона, будет выполнять роль порогового значения.

Далее, по умолчанию будет использоваться определение (7).

Линейные модели

Алгоритм Розенблатта

Персептрон (7) тесно связан с гиперплоскостью, задаваемой уравнением

$$\langle \mathbf{w}, \mathbf{u} \rangle = 0 \qquad \left(\mathbf{u} \in \mathbb{R}^m \right).$$

Такую гиперплоскость принято называть разделяющей.

Линейные модели

Алгоритм Розенблатта

Персептрон (7) тесно связан с гиперплоскостью, задаваемой уравнением

$$\langle \mathbf{w}, \mathbf{u} \rangle = 0 \qquad \left(\mathbf{u} \in \mathbb{R}^m \right).$$

Такую гиперплоскость принято называть разделяющей.

Она разделяет пространство $\mathbb{R}^m$ на два полупространства, лежащих «выше» и «ниже» неё.

Линейные модели

Алгоритм Розенблатта

Персептрон (7) тесно связан с гиперплоскостью, задаваемой уравнением

$$\langle \mathbf{w}, \mathbf{u} \rangle = 0 \qquad \left(\mathbf{u} \in \mathbb{R}^m \right).$$

Такую гиперплоскость принято называть разделяющей.

Она разделяет пространство $\mathbb{R}^m$ на два полупространства, лежащих «выше» и «ниже» неё.

Понятно, что персептрон корректно классифицирует множество объектов в том и только том случае, когда объекты с одной меткой вместе лежат в одном из этих двух полупространств.

Линейные модели

Алгоритм Розенблатта

Персептрон (7) тесно связан с гиперплоскостью, задаваемой уравнением

$$\langle \mathbf{w}, \mathbf{u} \rangle = 0 \qquad \left(\mathbf{u} \in \mathbb{R}^m \right).$$

Такую гиперплоскость принято называть разделяющей.

Она разделяет пространство $\mathbb{R}^m$ на два полупространства, лежащих «выше» и «ниже» неё.

Понятно, что персептрон корректно классифицирует множество объектов в том и только том случае, когда объекты с одной меткой вместе лежат в одном из этих двух полупространств.

Поэтому, далее, мы будем отождествлять персептрон с его разделяющей гиперплоскостью и использовать эти понятия как взаимозаменяемые.

Линейные модели

Алгоритм Розенблатта

Опишем предложенный Розенблаттом алгоритм, который будем обозначать как $\mathcal{A}_{\text{ros}}$.

Линейные модели

Алгоритм Розенблатта

Опишем предложенный Розенблаттом алгоритм, который будем обозначать как $\mathcal{A}_{\text{ros}}$.

Вход: Набор примеров вида

$$(\mathbf{u}_1, y_1), \ldots, (\mathbf{u}_n, y_n) \qquad (n \in \mathbb{N}). \tag{8}$$

Основной блок:

$\mathbf{w}_0 := (0, \ldots, 0)$.

Цикл $(t = 0, 1, \ldots)$**:**

Если: существует индекс i_t $(1 \leqslant i_t \leqslant n)$ такой, что $\langle \mathbf{w}_t, y_{i_t} \mathbf{u}_{i_t} \rangle \leqslant 0$,
то:

$$\mathbf{w}_{t+1} := \mathbf{w}_t + y_{i_t}\, \mathbf{u}_{i_t}; \tag{9}$$

иначе: выход из цикла.

Выход: $\mathbf{w}_t$.

Линейные модели

Алгоритм Розенблатта

Прежде всего заметим, что алгоритм $\mathcal{A}_{\text{ros}}$ представляет собой частично определенную функцию и, строго говоря, не удовлетворяет определению алгоритма обучения, которое было дано ранее.

Линейные модели

Алгоритм Розенблатта

Прежде всего заметим, что алгоритм $\mathcal{A}_{\text{ros}}$ представляет собой частично определенную функцию и, строго говоря, не удовлетворяет определению алгоритма обучения, которое было дано ранее.

Относительно этого алгоритма с уверенностью можно сказать следующее.

Линейные модели

Алгоритм Розенблатта

Прежде всего заметим, что алгоритм $\mathcal{A}_{\text{ros}}$ представляет собой частично определенную функцию и, строго говоря, не удовлетворяет определению алгоритма обучения, которое было дано ранее.

Относительно этого алгоритма с уверенностью можно сказать следующее.

Если за конечное число шагов он завершит свою работу, то для набора входных примеров будет построена разделяющая гиперплоскость.

Линейные модели

Алгоритм Розенблатта

Прежде всего заметим, что алгоритм $\mathcal{A}_{\text{ros}}$ представляет собой частично определенную функцию и, строго говоря, не удовлетворяет определению алгоритма обучения, которое было дано ранее.

Относительно этого алгоритма с уверенностью можно сказать следующее.

Если за конечное число шагов он завершит свою работу, то для набора входных примеров будет построена разделяющая гиперплоскость.

Возникает закономерный вопрос.

Линейные модели

Алгоритм Розенблатта

Прежде всего заметим, что алгоритм $\mathcal{A}_{\text{ros}}$ представляет собой частично определенную функцию и, строго говоря, не удовлетворяет определению алгоритма обучения, которое было дано ранее.

Относительно этого алгоритма с уверенностью можно сказать следующее.

Если за конечное число шагов он завершит свою работу, то для набора входных примеров будет построена разделяющая гиперплоскость.

Возникает закономерный вопрос.

Способен ли алгоритм $\mathcal{A}_{\text{ros}}$ построить разделяющую гиперплоскость для любого набора входных примеров, у которых такая гиперплоскость существует?

Линейные модели

Алгоритм Розенблатта

Положительный ответ на этот вопрос даёт теорема Новикова.

Линейные модели

Алгоритм Розенблатта

Положительный ответ на этот вопрос даёт теорема Новикова.

Для формулировки этой теоремы нам потребуется выразить условие существования разделяющей гиперплоскости в аналитическом виде. Обозначим

$$\rho_* := \max_{\|\mathbf{w}\|_2=1} \rho(\mathbf{w}), \quad \rho(\mathbf{w}) := \min_{i=1,\ldots,n} \langle \mathbf{w}, y_i \mathbf{u}_i \rangle.$$

Линейные модели

Алгоритм Розенблатта

Положительный ответ на этот вопрос даёт теорема Новикова.

Для формулировки этой теоремы нам потребуется выразить условие существования разделяющей гиперплоскости в аналитическом виде. Обозначим

$$\rho_* := \max_{\|\mathbf{w}\|_2=1} \rho(\mathbf{w}), \quad \rho(\mathbf{w}) := \min_{i=1,\ldots,n} \langle \mathbf{w}, y_i \mathbf{u}_i \rangle.$$

Очевидно, что условие $\rho_* > 0$ эквивалентно существованию разделяющей гиперплоскости для набора примеров (8).

Линейные модели

Алгоритм Розенблатта

Положительный ответ на этот вопрос даёт теорема Новикова.

Для формулировки этой теоремы нам потребуется выразить условие существования разделяющей гиперплоскости в аналитическом виде. Обозначим

$$\rho_* := \max_{\|\mathbf{w}\|_2=1} \rho(\mathbf{w}), \quad \rho(\mathbf{w}) := \min_{i=1,\ldots,n} \langle \mathbf{w}, y_i \mathbf{u}_i \rangle.$$

Очевидно, что условие $\rho_* > 0$ эквивалентно существованию разделяющей гиперплоскости для набора примеров (8).

В дальнейшем, под шагами алгоритма $\mathcal{A}_{\text{ros}}$ будем понимать те итерации его внутреннего цикла, на которых осуществляется попытка вычисления (9) набора коэффициентов разделяющей гиперплоскости.

Теорема 3.2 (Новиков).

Предположим, что для входного набора примеров (8) выполняется условие $\rho_* > 0$. Тогда алгоритм $\mathcal{A}_{\text{ros}}$ за t шагов, где

$$t \leqslant \frac{D^2}{\rho_*^2}, \qquad D := \max_{i=1,\ldots,n} \|\mathbf{u}_i\|_2. \tag{10}$$

построит разделяющую гиперплоскость.

Теорема 3.2 (Новиков).

Предположим, что для входного набора примеров (8) выполняется условие $\rho_* > 0$. Тогда алгоритм $\mathcal{A}_{\text{ros}}$ за t шагов, где

$$t \leqslant \frac{D^2}{\rho_*^2}, \qquad D := \max_{i=1,\ldots,n} \|\mathbf{u}_i\|_2. \tag{10}$$

построит разделяющую гиперплоскость.

◀ Предположим, что алгоритмом было выполнено t шагов.

Теорема 3.2 (Новиков).

Предположим, что для входного набора примеров (8) выполняется условие $\rho_* > 0$. Тогда алгоритм $\mathcal{A}_{\text{ros}}$ за t шагов, где

$$t \leqslant \frac{D^2}{\rho_*^2}, \qquad D := \max_{i=1,\ldots,n} \|\mathbf{u}_i\|_2. \tag{10}$$

построит разделяющую гиперплоскость.

◀ Предположим, что алгоритмом было выполнено t шагов.

Изначально мы не предполагаем, что после их выполнения алгоритм завершит свою работу.

Линейные модели

Алгоритм Розенблатта

Запишем

$$\begin{aligned}\|\mathbf{w}_t\|_2^2 &= \|\mathbf{w}_{t-1}\|_2^2 + 2\langle \mathbf{w}_{t-1}, y_{i_{t-1}}\mathbf{u}_{i_{t-1}}\rangle + \|\mathbf{u}_{i_{t-1}}\|_2^2 \leqslant \|\mathbf{w}_{t-1}\|_2^2 + D^2 \\ &\leqslant \|\mathbf{w}_{t-2}\|_2^2 + 2\langle \mathbf{w}_{t-2}, y_{i_{t-2}}\mathbf{u}_{i_{t-2}}\rangle + \|\mathbf{u}_{i_{t-2}}\|_2^2 + D^2 \leqslant \|\mathbf{w}_{t-2}\|_2^2 + 2D^2 \\ &\leqslant \ldots \leqslant \|\mathbf{w}_0\|_2^2 + t\,D^2 = t\,D^2.\end{aligned} \tag{11}$$

Линейные модели

Алгоритм Розенблатта

Запишем

$$\begin{aligned}\|\mathbf{w}_t\|_2^2 &= \|\mathbf{w}_{t-1}\|_2^2 + 2\langle \mathbf{w}_{t-1}, y_{i_{t-1}}\mathbf{u}_{i_{t-1}}\rangle + \|\mathbf{u}_{i_{t-1}}\|_2^2 \leqslant \|\mathbf{w}_{t-1}\|_2^2 + D^2 \\ &\leqslant \|\mathbf{w}_{t-2}\|_2^2 + 2\langle \mathbf{w}_{t-2}, y_{i_{t-2}}\mathbf{u}_{i_{t-2}}\rangle + \|\mathbf{u}_{i_{t-2}}\|_2^2 + D^2 \leqslant \|\mathbf{w}_{t-2}\|_2^2 + 2D^2 \\ &\leqslant \ldots \leqslant \|\mathbf{w}_0\|_2^2 + t\,D^2 = t\,D^2.\end{aligned} \tag{11}$$

Непрерывная функция, заданная на компакте, достигает на нём своего максимального значения.

Линейные модели

Алгоритм Розенблатта

Запишем

$$\begin{aligned}\|\mathbf{w}_t\|_2^2 &= \|\mathbf{w}_{t-1}\|_2^2 + 2\langle \mathbf{w}_{t-1}, y_{i_{t-1}}\mathbf{u}_{i_{t-1}}\rangle + \|\mathbf{u}_{i_{t-1}}\|_2^2 \leqslant \|\mathbf{w}_{t-1}\|_2^2 + D^2 \\ &\leqslant \|\mathbf{w}_{t-2}\|_2^2 + 2\langle \mathbf{w}_{t-2}, y_{i_{t-2}}\mathbf{u}_{i_{t-2}}\rangle + \|\mathbf{u}_{i_{t-2}}\|_2^2 + D^2 \leqslant \|\mathbf{w}_{t-2}\|_2^2 + 2D^2 \\ &\leqslant \ldots \leqslant \|\mathbf{w}_0\|_2^2 + t\,D^2 = t\,D^2.\end{aligned} \tag{11}$$

Непрерывная функция, заданная на компакте, достигает на нём своего максимального значения.

Поэтому существует вектор $\mathbf{w}_* \in \mathbb{R}^m$ такой, что

$$\rho(\mathbf{w}_*) = \rho_*, \quad \|\mathbf{w}_*\|_2 = 1.$$

Линейные модели

Алгоритм Розенблатта

С его помощью запишем

$$\begin{aligned}\langle \mathbf{w}_t, \mathbf{w}_* \rangle &= \langle y_{i_{t-1}} \mathbf{u}_{i_{t-1}} + \ldots + y_{i_0} \mathbf{u}_{i_0}, \mathbf{w}_* \rangle \\ &= \langle y_{i_{t-1}} \mathbf{u}_{i_{t-1}}, \mathbf{w}_* \rangle + \ldots + \langle y_{i_0} \mathbf{u}_{i_0}, \mathbf{w}_* \rangle \geqslant t\, \rho_*.\end{aligned} \tag{12}$$

Линейные модели

Алгоритм Розенблатта

С его помощью запишем

$$\begin{aligned}\langle \mathbf{w}_t, \mathbf{w}_* \rangle &= \langle y_{i_{t-1}} \mathbf{u}_{i_{t-1}} + \ldots + y_{i_0} \mathbf{u}_{i_0}, \mathbf{w}_* \rangle \\ &= \langle y_{i_{t-1}} \mathbf{u}_{i_{t-1}}, \mathbf{w}_* \rangle + \ldots + \langle y_{i_0} \mathbf{u}_{i_0}, \mathbf{w}_* \rangle \geqslant t\, \rho_*.\end{aligned} \tag{12}$$

С другой стороны, с помощью неравенства Коши-Буняковского получим

$$\langle \mathbf{w}_t, \mathbf{w}_* \rangle \leqslant \|\mathbf{w}_t\|_2 \|\mathbf{w}_*\|_2 = \|\mathbf{w}_t\|_2. \tag{13}$$

Линейные модели

Алгоритм Розенблатта

С его помощью запишем

$$\begin{aligned}\langle \mathbf{w}_t, \mathbf{w}_* \rangle &= \langle y_{i_{t-1}} \mathbf{u}_{i_{t-1}} + \ldots + y_{i_0} \mathbf{u}_{i_0}, \mathbf{w}_* \rangle \\ &= \langle y_{i_{t-1}} \mathbf{u}_{i_{t-1}}, \mathbf{w}_* \rangle + \ldots + \langle y_{i_0} \mathbf{u}_{i_0}, \mathbf{w}_* \rangle \geqslant t\, \rho_*.\end{aligned} \tag{12}$$

С другой стороны, с помощью неравенства Коши-Буняковского получим

$$\langle \mathbf{w}_t, \mathbf{w}_* \rangle \leqslant \|\mathbf{w}_t\|_2 \|\mathbf{w}_*\|_2 = \|\mathbf{w}_t\|_2. \tag{13}$$

Объединяя (11), (12) и (13), установим требуемую оценку (10), а значит и конечность числа шагов работы алгоритма в рассматриваемом случае.

■

Линейные модели

Алгоритм Розенблатта

Рассмотренный вариант алгоритма Розенблатта несколько отличается от изначально предложенного.

Линейные модели

Алгоритм Розенблатта

Рассмотренный вариант алгоритма Розенблатта несколько отличается от изначально предложенного.

Первоначальный вариант был, скорее, ориентирован на работу с последовательно поступающими ему на вход индивидуальными обучающими примерами. Соответственно, на каждой итерации цикла либо происходило вычисление очередного значения $\mathbf{w}_{t+1}$ по правилу (9), либо вектор, задающий строющуюся гиперплоскость, не менялся $\mathbf{w}_{t+1} = \mathbf{w}_t$.

Линейные модели

Алгоритм Розенблатта

Рассмотренный вариант алгоритма Розенблатта несколько отличается от изначально предложенного.

Первоначальный вариант был, скорее, ориентирован на работу с последовательно поступающими ему на вход индивидуальными обучающими примерами. Соответственно, на каждой итерации цикла либо происходило вычисление очередного значения $\mathbf{w}_{t+1}$ по правилу (9), либо вектор, задающий строющуюся гиперплоскость, не менялся $\mathbf{w}_{t+1} = \mathbf{w}_t$.

В настоящее время такой подход оформился в отдельное направление и получил название онлайн обучения. В рамках этого подхода алгоритм накапливает и оценивает свой предыдущий опыт обучения, и осуществляет его корректировку с каждым вновь поступившим обучающим примером.

Содержание

Линейные модели

Логистическая регрессия (наводящий пример)

Предположим, что перед нами стоит задача оценки надёжности коммерческих организаций на предмет их возможного банкротства.

Линейные модели

Логистическая регрессия (наводящий пример)

Предположим, что перед нами стоит задача оценки надёжности коммерческих организаций на предмет их возможного банкротства.

Каждая организация может быть предствлена в виде фиксированного набора числовых значений её финансовых показателей за некоторый отчётный период. К таким показателям можно, например, отнести денежные резервы организации, её доход, затраты на оплату труда сотрудников.

Линейные модели

Логистическая регрессия (наводящий пример)

Предположим, что перед нами стоит задача оценки надёжности коммерческих организаций на предмет их возможного банкротства.

Каждая организация может быть предствлена в виде фиксированного набора числовых значений её финансовых показателей за некоторый отчётный период. К таким показателям можно, например, отнести денежные резервы организации, её доход, затраты на оплату труда сотрудников.

Если предположить, что между этими показателями существует линейная зависимость, то с помощью персептрона можно пытаться решить задачу бинарной классификации.

Линейные модели

Логистическая регрессия (наводящий пример)

Предположим, что перед нами стоит задача оценки надёжности коммерческих организаций на предмет их возможного банкротства.

Каждая организация может быть предствлена в виде фиксированного набора числовых значений её финансовых показателей за некоторый отчётный период. К таким показателям можно, например, отнести денежные резервы организации, её доход, затраты на оплату труда сотрудников.

Если предположить, что между этими показателями существует линейная зависимость, то с помощью персептрона можно пытаться решить задачу бинарной классификации.

Отнести каждую организацию либо к классу надёжных, либо к классу потенциальных банкротов.

Линейные модели

Логистическая регрессия (наводящий пример)

В некоторых случая требуется произвести более глубокий анализ и для каждой организации оценить вероятность её банкротова.

Линейные модели

Логистическая регрессия (наводящий пример)

В некоторых случая требуется произвести более глубокий анализ и для каждой организации оценить вероятность её банкротова.

Очевидно, что персептрон для этой задачи не годится.

Линейные модели

Логистическая регрессия (наводящий пример)

В некоторых случая требуется произвести более глубокий анализ и для каждой организации оценить вероятность её банкротова.

Очевидно, что персептрон для этой задачи не годится.

Заметим, что возможность получения оценок вероятности банкротова организаций позволяет найти решение и для задачи классификации. Если полученная оценка больше некоторого порогового значения (например, 0.5), то организацию логично отнести к классу потенциальных банкротов.

Линейные модели

Логистическая регрессия (наводящий пример)

В некоторых случая требуется произвести более глубокий анализ и для каждой организации оценить вероятность её банкротова.

Очевидно, что персептрон для этой задачи не годится.

Заметим, что возможность получения оценок вероятности банкротова организаций позволяет найти решение и для задачи классификации. Если полученная оценка больше некоторого порогового значения (например, 0.5), то организацию логично отнести к классу потенциальных банкротов.

Построим обобщение модели персептрона, позволяющее решать задачу оценки вероятности принадлежности объекта к одному из двух рассматриваемых классов. Для этого поменяем активирующую функцию персептрона.

Линейные модели

Логистическая регрессия

Как и прежде, в качестве множества объектов X будет выступать пространство $\mathbb{R}^m$, а в качестве множества меток Y будет выступать множество $\{-1, 1\}$.

Линейные модели

Логистическая регрессия

Как и прежде, в качестве множества объектов X будет выступать пространство $\mathbb{R}^m$, а в качестве множества меток Y будет выступать множество $\{-1, 1\}$.

Функцию

$$\sigma_{\text{sig}}(u) := \frac{e^u}{1 + e^u} \quad (u \in \mathbb{R})$$

будем называть логистической функцией активации.

Линейные модели

Логистическая регрессия

Как и прежде, в качестве множества объектов X будет выступать пространство $\mathbb{R}^m$, а в качестве множества меток Y будет выступать множество $\{-1, 1\}$.

Функцию

$$\sigma_{\text{sig}}(u) := \frac{e^u}{1 + e^u} \quad (u \in \mathbb{R})$$

будем называть логистической функцией активации.

Эту функцию принято также называть сигмоидой потому, что её график своей формой напоминает латинскую букву S.

Линейные модели

Логистическая регрессия

Как и прежде, в качестве множества объектов X будет выступать пространство $\mathbb{R}^m$, а в качестве множества меток Y будет выступать множество $\{-1, 1\}$.

Функцию

$$\sigma_{\text{sig}}(u) := \frac{e^u}{1 + e^u} \quad (u \in \mathbb{R})$$

будем называть логистической функцией активации.

Эту функцию принято также называть сигмоидой потому, что её график своей формой напоминает латинскую букву S.

Сигмоида является гладкой и монотонно возрастающей функцией. В нуле она принимает значение $1/2$.

Линейные модели

Логистическая регрессия

Как и прежде, в качестве множества объектов X будет выступать пространство $\mathbb{R}^m$, а в качестве множества меток Y будет выступать множество $\{-1, 1\}$.

Функцию

$$\sigma_{\text{sig}}(u) := \frac{e^u}{1 + e^u} \quad (u \in \mathbb{R})$$

будем называть логистической функцией активации.

Эту функцию принято также называть сигмоидой потому, что её график своей формой напоминает латинскую букву S.

Сигмоида является гладкой и монотонно возрастающей функцией. В нуле она принимает значение $1/2$.

При $u \to \infty$ она стремится к 1, а при $u \to -\infty$ она стремится к 0.

Линейные модели

Логистическая регрессия

Для любого $u \in \mathbb{R}$ выполняется равенство

$$\sigma_{\text{sig}}(u) + \sigma_{\text{sig}}(-u) = 1. \tag{14}$$

Линейные модели

Логистическая регрессия

Для любого $u \in \mathbb{R}$ выполняется равенство

$$\sigma_{\text{sig}}(u) + \sigma_{\text{sig}}(-u) = 1. \tag{14}$$

Логистическим нейроном будем называть произвольную функцию вида

$$h_{\mathbf{w}} : \mathbf{u} \longmapsto \sigma_{\text{sig}}\left(\langle \mathbf{w}, \mathbf{u} \rangle\right) \qquad \left(\mathbf{w}, \mathbf{u} \in \mathbb{R}^m\right).$$

Линейные модели

Логистическая регрессия

Для любого $u \in \mathbb{R}$ выполняется равенство

$$\sigma_{\text{sig}}(u) + \sigma_{\text{sig}}(-u) = 1. \tag{14}$$

Логистическим нейроном будем называть произвольную функцию вида

$$h_{\mathbf{w}} : \mathbf{u} \longmapsto \sigma_{\text{sig}}\left(\langle \mathbf{w}, \mathbf{u} \rangle\right) \qquad \left(\mathbf{w}, \mathbf{u} \in \mathbb{R}^m\right).$$

По аналогии с персептроном, функция $h_{\mathbf{w}}$ и вектор $\mathbf{w}$ будут отождествляться, а гиперплоскость, задаваемая вектором $\mathbf{w}$, будет называться разделяющей.

Линейные модели

Логистическая регрессия

Относительно любой рассматриваемой обучающей выборки

$$(\mathbf{u}_1, y_1), \ldots, (\mathbf{u}_n, y_n) \qquad (n \in \mathbb{N})$$

будем предполагать, что она получена в ходе независимых испытаний над некоторым случайным элементом (X, Y).

Линейные модели

Логистическая регрессия

Относительно любой рассматриваемой обучающей выборки

$$(\mathbf{u}_1, y_1), \ldots, (\mathbf{u}_n, y_n) \qquad (n \in \mathbb{N})$$

будем предполагать, что она получена в ходе независимых испытаний над некоторым случайным элементом (X, Y).

Предполагается, что этот случайный элемент задан на некотором вероятностном пространстве и принимает свои значения в измеримом пространстве $\left(\mathbb{R}^m \times \{-1, 1\}, \mathcal{B}(\mathbb{R}^m) \otimes 2^{\{-1,1\}}\right)$.

Линейные модели

Логистическая регрессия

Относительно любой рассматриваемой обучающей выборки

$$(\mathbf{u}_1, y_1), \ldots, (\mathbf{u}_n, y_n) \qquad (n \in \mathbb{N})$$

будем предполагать, что она получена в ходе независимых испытаний над некоторым случайным элементом (X, Y).

Предполагается, что этот случайный элемент задан на некотором вероятностном пространстве и принимает свои значения в измеримом пространстве $\left(\mathbb{R}^m \times \{-1, 1\}, \mathcal{B}(\mathbb{R}^m) \otimes 2^{\{-1,1\}}\right)$.

В качестве целевой функции будем рассматривать условную вероятность вида

$$\eta(\mathbf{u}) := \mathsf{P}\big[\{Y = 1\} \,|\, X = \mathbf{u}\big] \quad \big(\mathbf{u} \in \mathbb{R}^m\big). \tag{15}$$

Линейные модели

Логистическая регрессия

Для этой функции выполняется условие

$$\mathsf{P}\big[\{Y = -1\} \,|\, X = \mathbf{u}\big] = 1 - \eta(\mathbf{u}) \quad \big(\mathbf{u} \in \mathbb{R}^m\big). \tag{16}$$

Линейные модели

Логистическая регрессия

Для этой функции выполняется условие

$$\mathsf{P}\big[\{Y = -1\} \,|\, X = \mathbf{u}\big] = 1 - \eta(\mathbf{u}) \quad \big(\mathbf{u} \in \mathbb{R}^m\big). \tag{16}$$

Целевую функцию η будем пытаться аппроксимировать с помощью логистических нейронов.

Линейные модели

Логистическая регрессия

Для этой функции выполняется условие

$$\mathsf{P}\big[\{Y = -1\} \,|\, X = \mathbf{u}\big] = 1 - \eta(\mathbf{u}) \quad \big(\mathbf{u} \in \mathbb{R}^m\big). \tag{16}$$

Целевую функцию η будем пытаться аппроксимировать с помощью логистических нейронов.

Заметим, что если выбрано приближение $\eta \approx h_{\mathbf{w}}$, то выполнение условий (14), (15) и (16), позволяет считать, что

$$\mathsf{P}\big[\{Y = y\} \,|\, X = \mathbf{u}\big] \approx h_{\mathbf{w}}(y\mathbf{u}) \quad \big(\mathbf{u} \in \mathbb{R}^m; y \in \{-1, 1\}\big). \tag{17}$$

Линейные модели

Логистическая регрессия

Пусть $(X_1, Y_1), \ldots, (X_n, Y_n)$ – независимые копии случайного элемента (X, Y).

Линейные модели

Логистическая регрессия

Пусть $(X_1, Y_1), \ldots, (X_n, Y_n)$ – независимые копии случайного элемента (X, Y).

Тогда

$$\begin{aligned}
\mathsf{P}\big[\{Y_1 = y_1, \ldots, Y_n = y_n\} \,|\, X_1 = \mathbf{u}_1, \ldots, X_n = \mathbf{u}_n\big] &= \prod_{i=1}^{n} \mathsf{P}\big[\{Y_i = y_i\} \,|\, X_i = \mathbf{u}_i\big] \\
&= \prod_{i=1}^{n} \mathsf{P}\big[\{Y = y_i\} \,|\, X = \mathbf{u}_i\big] \\
&\approx \prod_{i=1}^{n} h_{\mathbf{w}}(y_i \mathbf{u}_i).
\end{aligned} \tag{18}$$

Линейные модели

Логистическая регрессия

Пусть $(X_1, Y_1), \ldots, (X_n, Y_n)$ – независимые копии случайного элемента (X, Y).

Тогда

$$\begin{aligned}
\mathsf{P}\big[\{Y_1 = y_1, \ldots, Y_n = y_n\} \,|\, X_1 = \mathbf{u}_1, \ldots, X_n = \mathbf{u}_n\big] &= \prod_{i=1}^{n} \mathsf{P}\big[\{Y_i = y_i\} \,|\, X_i = \mathbf{u}_i\big] \\
&= \prod_{i=1}^{n} \mathsf{P}\big[\{Y = y_i\} \,|\, X = \mathbf{u}_i\big] \\
&\approx \prod_{i=1}^{n} h_{\mathbf{w}}(y_i \mathbf{u}_i).
\end{aligned} \tag{18}$$

Метод максимального правдоподобия предписывает максимизировать правую часть в (18).

Линейные модели

Логистическая регрессия

Вектор $\mathbf{w}_* \in \mathbb{R}^m$, на котором достигается этот максимум, следует использовать для построения приближения к целевой функции η.

Линейные модели

Логистическая регрессия

Вектор $\mathbf{w}_* \in \mathbb{R}^m$, на котором достигается этот максимум, следует использовать для построения приближения к целевой функции η.

По определению положим

$$r_{\mathrm{sig}}(\mathbf{w}) := -\ln\left(\prod_{i=1}^{n} h_{\mathbf{w}}(y_i \mathbf{u}_i)\right) = \sum_{i=1}^{n} \ln\left(1 + e^{-y_i \langle \mathbf{w}, \mathbf{u}_i \rangle}\right) \quad \left(\mathbf{w} \in \mathbb{R}^m\right).$$

Линейные модели

Логистическая регрессия

Вектор $\mathbf{w}_* \in \mathbb{R}^m$, на котором достигается этот максимум, следует использовать для построения приближения к целевой функции η.

По определению положим

$$r_{\text{sig}}(\mathbf{w}) := -\ln\left(\prod_{i=1}^{n} h_{\mathbf{w}}(y_i \mathbf{u}_i)\right) = \sum_{i=1}^{n} \ln\left(1 + e^{-y_i \langle \mathbf{w}, \mathbf{u}_i \rangle}\right) \quad \left(\mathbf{w} \in \mathbb{R}^m\right).$$

Тогда

$$\mathbf{w}_* = \arg\min_{\mathbf{w} \in \mathbb{R}^m} r_{\text{sig}}(\mathbf{w}).$$

Линейные модели

Логистическая регрессия

Вектор $\mathbf{w}_* \in \mathbb{R}^m$, на котором достигается этот максимум, следует использовать для построения приближения к целевой функции η.

По определению положим

$$r_{\text{sig}}(\mathbf{w}) := -\ln\left(\prod_{i=1}^{n} h_{\mathbf{w}}(y_i \mathbf{u}_i)\right) = \sum_{i=1}^{n} \ln\left(1 + e^{-y_i \langle \mathbf{w}, \mathbf{u}_i \rangle}\right) \quad \left(\mathbf{w} \in \mathbb{R}^m\right).$$

Тогда

$$\mathbf{w}_* = \arg\min_{\mathbf{w} \in \mathbb{R}^m} r_{\text{sig}}(\mathbf{w}).$$

Использование гладкой функции активации σ_{sig} позволило свести задачу обучения логистического нейрона к задаче минимизации гладкой и выпуклой функции r_{sig}.

Линейные модели

Логистическая регрессия

Для функции r_{sig} можно явно в аналитическом виде выразить градиент и применить для решения задачи минимизации метод градиентного спуска.

Линейные модели

Логистическая регрессия

Для функции r_{sig} можно явно в аналитическом виде выразить градиент и применить для решения задачи минимизации метод градиентного спуска.

Этот метод позволяет построить приближение к локальному минимуму функции в случае существования последнего.

Линейные модели

Логистическая регрессия

Для функции r_{sig} можно явно в аналитическом виде выразить градиент и применить для решения задачи минимизации метод градиентного спуска.

Этот метод позволяет построить приближение к локальному минимуму функции в случае существования последнего.

Для функции r_{sig} существование локального минимума будет означать, что он одновременно является и глобальным минимумом.

Линейные модели

Логистическая регрессия

Для функции r_{sig} можно явно в аналитическом виде выразить градиент и применить для решения задачи минимизации метод градиентного спуска.

Этот метод позволяет построить приближение к локальному минимуму функции в случае существования последнего.

Для функции r_{sig} существование локального минимума будет означать, что он одновременно является и глобальным минимумом.

Поэтому в этом случае вектор $\mathbf{w}_*$ будет корректно опредлён.

Линейные модели

Логистическая регрессия

Расположение объектов относительно разделяющей гиперплоскости, задаваемой вектором $\mathbf{w}_*$, обладает следующим свойством.

Линейные модели

Логистическая регрессия

Расположение объектов относительно разделяющей гиперплоскости, задаваемой вектором $\mathbf{w}_*$, обладает следующим свойством.

Чем больше расстояние от объекта до разделяющей гиперплоскости, те больше вероятность принадлежности объекта к соответствующему классу.

Линейные модели

Логистическая регрессия

Расположение объектов относительно разделяющей гиперплоскости, задаваемой вектором $\mathbf{w}_*$, обладает следующим свойством.

Чем больше расстояние от объекта до разделяющей гиперплоскости, те больше вероятность принадлежности объекта к соответствующему классу.

Вблизи разделяющей гиперплоскости такая вероятность будет ≈ 0.5.

Содержание

Линейные модели

Обобщения

Модель персептрона оказалась очень плодотворной в то смысле, что послужила отправной точкой для возникновения целого ряда направлений исследований.

Линейные модели

Обобщения

Модель персептрона оказалась очень плодотворной в то смысле, что послужила отправной точкой для возникновения целого ряда направлений исследований.

Отметим три из них.

Линейные модели

Обобщения

Модель персептрона оказалась очень плодотворной в то смысле, что послужила отправной точкой для возникновения целого ряда направлений исследований.

Отметим три из них.

Первое направление связано с изучением линейных моделей обучения.

Линейные модели

Обобщения

Модель персептрона оказалась очень плодотворной в то смысле, что послужила отправной точкой для возникновения целого ряда направлений исследований.

Отметим три из них.

Первое направление связано с изучением линейных моделей обучения.

Второе направление, крайне востребованное в настоящее время, посвящено изучению моделей нейронных сетей. Оно будет отдельно рассмотрено.

Линейные модели

Обобщения

Модель персептрона оказалась очень плодотворной в то смысле, что послужила отправной точкой для возникновения целого ряда направлений исследований.

Отметим три из них.

Первое направление связано с изучением линейных моделей обучения.

Второе направление, крайне востребованное в настоящее время, посвящено изучению моделей нейронных сетей. Оно будет отдельно рассмотрено.

Здесь же более подробно обсудим третье направление.

Линейные модели

Обобщения

Линейные модели обучения используют следующее предположение.

Линейные модели

Обобщения

Линейные модели обучения используют следующее предположение.

Объекты представляются в виде наборов числовых признаков. Между этими признаками имеется некоторая линейная зависимость, обнаружение которой позволяет решить задачу обучения.

Линейные модели

Обобщения

Линейные модели обучения используют следующее предположение.

Объекты представляются в виде наборов числовых признаков. Между этими признаками имеется некоторая линейная зависимость, обнаружение которой позволяет решить задачу обучения.

На практике предположение о наличии скрытой линейной зависимости между признаками не всегда является обоснованным.

Линейные модели

Обобщения

Линейные модели обучения используют следующее предположение.

Объекты представляются в виде наборов числовых признаков. Между этими признаками имеется некоторая линейная зависимость, обнаружение которой позволяет решить задачу обучения.

На практике предположение о наличии скрытой линейной зависимости между признаками не всегда является обоснованным.

В качестве примера рассмотрим те точки (u, v) на плоскости, которые удовлетворяют уравнению $u^2 + v^2 = 1$.

Линейные модели

Обобщения

Линейные модели обучения используют следующее предположение.

Объекты представляются в виде наборов числовых признаков. Между этими признаками имеется некоторая линейная зависимость, обнаружение которой позволяет решить задачу обучения.

На практике предположение о наличии скрытой линейной зависимости между признаками не всегда является обоснованным.

В качестве примера рассмотрим те точки (u, v) на плоскости, которые удовлетворяют уравнению $u^2 + v^2 = 1$.

С одной стороны, явно существует зависимость между координатами точек, позволяющая определять их принадлежность окружности.

Линейные модели

Обобщения

Линейные модели обучения используют следующее предположение.

Объекты представляются в виде наборов числовых признаков. Между этими признаками имеется некоторая линейная зависимость, обнаружение которой позволяет решить задачу обучения.

На практике предположение о наличии скрытой линейной зависимости между признаками не всегда является обоснованным.

В качестве примера рассмотрим те точки (u, v) на плоскости, которые удовлетворяют уравнению $u^2 + v^2 = 1$.

С одной стороны, явно существует зависимость между координатами точек, позволяющая определять их принадлежность окружности.

С другой стороны, эта зависимость не носит линейный характер.

Линейные модели

Обобщения

Обобщая этот пример, можно сделать вывод.

Линейные модели

Обобщения

Обобщая этот пример, можно сделать вывод.

Теоретически исследованные и аппробированные линейные модели обучения имеют существенное ограничение для своего применения на практике.

Линейные модели

Обобщения

Обобщая этот пример, можно сделать вывод.

Теоретически исследованные и аппробированные линейные модели обучения имеют существенное ограничение для своего применения на практике.

Обойти это ограничение можно следующим образом.

Линейные модели

Обобщения

Обобщая этот пример, можно сделать вывод.

Теоретически исследованные и аппробированные линейные модели обучения имеют существенное ограничение для своего применения на практике.

Обойти это ограничение можно следующим образом.

В нашем примере рассмотрим отображение $(u_1, u_2) \longmapsto (u_1^2, u_2^2)$.

Линейные модели

Обобщения

Обобщая этот пример, можно сделать вывод.

Теоретически исследованные и аппробированные линейные модели обучения имеют существенное ограничение для своего применения на практике.

Обойти это ограничение можно следующим образом.

В нашем примере рассмотрим отображение $(u_1, u_2) \longmapsto (u_1^2, u_2^2)$.

О таких отображениях говорят, что они преобразуют исходное признаковое пространство в некоторое спрямляющее пространство.

Линейные модели

Обобщения

Обобщая этот пример, можно сделать вывод.

Теоретически исследованные и аппробированные линейные модели обучения имеют существенное ограничение для своего применения на практике.

Обойти это ограничение можно следующим образом.

В нашем примере рассмотрим отображение $(u_1, u_2) \longmapsto (u_1^2, u_2^2)$.

О таких отображениях говорят, что они преобразуют исходное признаковое пространство в некоторое спрямляющее пространство.

В спремляющем пространстве между новыми признаками u_1^2 и u_2^2 уже существует линейная зависимость.

Линейные модели

Обобщения

Таким образом, первым шагом для устранения ограничений по использованию линейных моделей является конструирование спремляющего пространства и соответствующего отображения в него.

Линейные модели

Обобщения

Таким образом, первым шагом для устранения ограничений по использованию линейных моделей является конструирование спремляющего пространства и соответствующего отображения в него.

В принципе на этом шаге можно было бы и остановиться.

Линейные модели

Обобщения

Таким образом, первым шагом для устранения ограничений по использованию линейных моделей является конструирование спремляющего пространства и соответствующего отображения в него.

В принципе на этом шаге можно было бы и остановиться.

Однако в силу разных причин напрямую работать с отображением в спремляющее пространство не всегда представляется удобным.

Линейные модели

Обобщения

Таким образом, первым шагом для устранения ограничений по использованию линейных моделей является конструирование спремляющего пространства и соответствующего отображения в него.

В принципе на этом шаге можно было бы и остановиться.

Однако в силу разных причин напрямую работать с отображением в спремляющее пространство не всегда представляется удобным.

Желательно иметь возможность работать с исходным набором признаков объектов из обучающей выборки.

Линейные модели

Обобщения

Таким образом, первым шагом для устранения ограничений по использованию линейных моделей является конструирование спремляющего пространства и соответствующего отображения в него.

В принципе на этом шаге можно было бы и остановиться.

Однако в силу разных причин напрямую работать с отображением в спремляющее пространство не всегда представляется удобным.

Желательно иметь возможность работать с исходным набором признаков объектов из обучающей выборки.

Этого можно достичь, используя понятие ядра.

Линейные модели

Обобщения

Как правило, в линейных моделях работа с обучающей выборкой сводится к использованию попарных скалярных произведений её объектов.

Линейные модели

Обобщения

Как правило, в линейных моделях работа с обучающей выборкой сводится к использованию попарных скалярных произведений её объектов.

Предположим, что задано отображение φ в спремляющее пространство.

Линейные модели

Обобщения

Как правило, в линейных моделях работа с обучающей выборкой сводится к использованию попарных скалярных произведений её объектов.

Предположим, что задано отображение φ в спремляющее пространство.

Под ядром понимается функция K, обладающая следующим свойством

$$K(\mathbf{u}, \mathbf{u}') = \langle \varphi(\mathbf{u}), \varphi(\mathbf{u}') \rangle.$$

Линейные модели

Обобщения

Как правило, в линейных моделях работа с обучающей выборкой сводится к использованию попарных скалярных произведений её объектов.

Предположим, что задано отображение φ в спремляющее пространство.

Под ядром понимается функция K, обладающая следующим свойством

$$K(\mathbf{u}, \mathbf{u}') = \langle \varphi(\mathbf{u}), \varphi(\mathbf{u}') \rangle.$$

Функция вычисляет скалярное произведение в спрямляющем пространстве.

Линейные модели

Обобщения

Как правило, в линейных моделях работа с обучающей выборкой сводится к использованию попарных скалярных произведений её объектов.

Предположим, что задано отображение φ в спремляющее пространство.

Под ядром понимается функция K, обладающая следующим свойством

$$K(\mathbf{u}, \mathbf{u}') = \langle \varphi(\mathbf{u}), \varphi(\mathbf{u}') \rangle.$$

Функция вычисляет скалярное произведение в спрямляющем пространстве.

При этом использует только значения исходных признаков.

Линейные модели

Обобщения

Приведём пример построения ядра.

Линейные модели

Обобщения

Приведём пример построения ядра.

Пусть в качестве исходного признакового пространства выступает $\mathbb{R}^2$, а спрямляющим пространством является является $\mathbb{R}^3$.

Линейные модели

Обобщения

Приведём пример построения ядра.

Пусть в качестве исходного признакового пространства выступает $\mathbb{R}^2$, а спрямляющим пространством является является $\mathbb{R}^3$.

Отображение в спремляющее пространство задаётся по правилу $\varphi : (u_1, u_2) \longmapsto (u_1^2, u_2^2, \sqrt{2}u_1u_2)$.

Линейные модели

Обобщения

Приведём пример построения ядра.

Пусть в качестве исходного признакового пространства выступает $\mathbb{R}^2$, а спрямляющим пространством является является $\mathbb{R}^3$.

Отображение в спремляющее пространство задаётся по правилу $\varphi : (u_1, u_2) \longmapsto (u_1^2, u_2^2, \sqrt{2}u_1u_2)$.

Тогда

$$\begin{aligned}\langle \varphi(\mathbf{u}), \varphi(\mathbf{u}') \rangle &= \langle (u_1^2, u_2^2, \sqrt{2}u_1u_2), (u'^2_1, u'^2_2, \sqrt{2}u'_1u'_2) \rangle \\ &= u_1^2u'^2_1 + u_2^2u'^2_2 + 2u_1u_2u'_1u'_2 \\ &= \langle \mathbf{u}, \mathbf{u}' \rangle^2.\end{aligned}$$

Линейные модели

Обобщения

Приведём пример построения ядра.

Пусть в качестве исходного признакового пространства выступает $\mathbb{R}^2$, а спрямляющим пространством является является $\mathbb{R}^3$.

Отображение в спремляющее пространство задаётся по правилу $\varphi : (u_1, u_2) \longmapsto (u_1^2, u_2^2, \sqrt{2}u_1u_2)$.

Тогда

$$\begin{aligned}\langle \varphi(\mathbf{u}), \varphi(\mathbf{u}') \rangle &= \langle (u_1^2, u_2^2, \sqrt{2}u_1u_2), (u'^2_1, u'^2_2, \sqrt{2}u'_1u'_2) \rangle \\ &= u_1^2u'^2_1 + u_2^2u'^2_2 + 2u_1u_2u'_1u'_2 \\ &= \langle \mathbf{u}, \mathbf{u}' \rangle^2.\end{aligned}$$

Следовательно, ядром является функция $K(\mathbf{u}, \mathbf{u}') = \langle \mathbf{u}, \mathbf{u}' \rangle^2$.