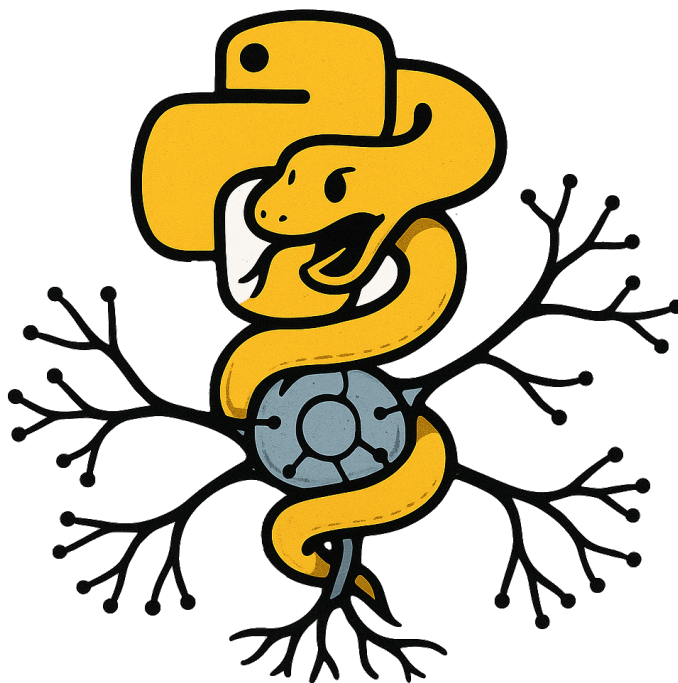


Основы Машинного Обучения



Автор: Марьичев Алексей

Нижегородский государственный университет им. Лобачевского

Предисловие

Словарь

- Вектор: $X = [x_1, x_2, \dots, x_n]$
- Транспонированный вектор(обозначается X^T) : $X^T = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix}$
-

Содержание:

Глава 1. Математический анализ

Глава 2. Линейная Алгебра

Глава 3. Теория Вероятности

Глава 4. Введение в Машинное обучение

Основные задачи ML:

- **Классификация** — определение объектов к определённым классам по общим признакам
 - **Регрессия** — прогнозирование величин, функций или событий
 - **Ранжирование** — упорядочивание входного набора данных
-

§1.1 Обучающая выборка

Представление объектов в виде различных векторов данных:

$$X = [x_1, x_2, \dots, x_n]^T = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

X - вектор входных данных

Допустим, у нас дана матрица:

$$A = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \dots & x_{mn} \end{bmatrix}$$

Здесь A - матрица входных данных, n — количество признаков объекта, а m — количество самих объектов.

Таким же видом представлены и выходные данные:

$$Y = [y_1, y_2, \dots, y_n]^T = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_m \end{bmatrix}$$

Y - вектор выходных данных

Теперь мы рассмотрим важный вопрос: как же такие объекты как изображения, звук и т. д. могут представляться в виде векторов?

Допустим, на вход задаче подается Изображение:



Теперь важное замечание-**размерность вектора n будет зависеть от количества пикселей в изображении**

Например, если изображение 1024 на 256, то размерность вектора будет $1024 * 256 = 262144$

$$X = [x_1, x_2, \dots, x_{262144}]$$

Теперь **объединим** эти понятия:

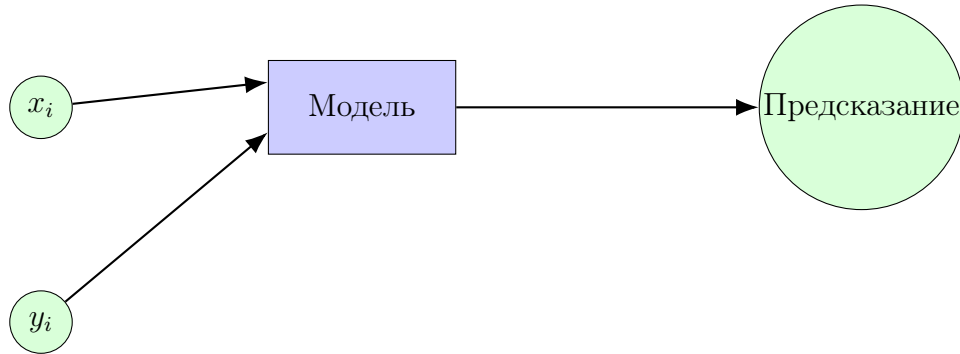
$$X' = \{(x_i, y_i) | 0 < i < m\} - \text{размеченные данные (обучающая выборка)}$$

Это и является одним из важнейших понятий в области машинного обучения, с которым вы будете встречаться всюду.

§1.2 Постановка задачи для модели

А теперь разберемся с тем, как же модель будет "обучаться":

Допустим, у нас есть размеченные данные (x_i, y_i) ,



В результате из исходных данных мы получили некое предсказание, которое на первых этапах обучения может не иметь ничего общего с правильным ответом.

Теперь представим нашу модель как линейную функцию:

$$y(x) = \phi(x, \Delta) \quad (1)$$

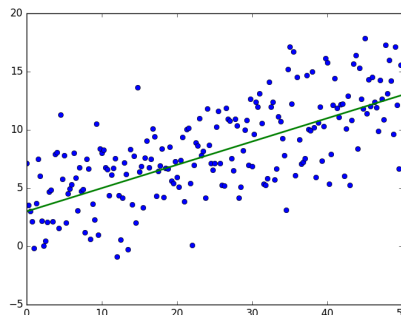
Здесь Δ — **постоянно меняющийся параметр**

Его мы будем подстраивать для наиболее точного ответа нашей модели

Для лучшего понимания перейдем к задаче линейной регрессии. Задана функция:

$$y(x, k, b) = kx + b + \psi$$

Здесь k и b - параметры от которых зависит угол поворота прямой а так же ее сдвиг Т.е. получается, что эта прямая может проходить как угодно, но за счет размеченных данных мы задаем модели желаемый результат:



И получается, что во время обучения модель дает прогнозы все точнее и точнее к желаемому результату.

Но как же наш алгоритм понимает, что ответ надо корректировать?
Сейчас мы подошли к еще одному очень важному определению в области ML:

Функция потерь - функция, которая характеризует потери при неправильном предсказании модели. Примеры таких функций:

$$L(x, a) = |a(x) - y(x)| \quad \text{— абсолютная ошибка}$$

$$L(x, a) = (a(x) - y(x))^2 \quad \text{— квадратичная ошибка}$$

Так же введем связанное понятие:

Средний эмпирический риск

$$Q(a) = \frac{1}{l} \sum_{i=1}^l L(a(x_i), y_i)$$

Среднее значение функции потерь на обучающей выборке.

Это среднее арифметическое по всем потерям в текущем цикле обучения модели.

Вспомним формулу (1): Наша задача - минимизировать средний эмпирический риск за счет изменения параметра Δ

§1.3 Линейная модель

Рассмотрим функцию $y = kx + b + \psi$.

В процессе обучения модели на данной функции перед нами будет стоять задача подобрать такие k' и b' , чтобы сама функция y наиболее точно отображала желаемый результат.

Но что если мы попробуем выразить функцию через характеристики объекта? Допустим, у нас есть ϕ_1, ϕ_2 :

$$y = f_1(x)\phi_1 + f_2(x)\phi_2 + \psi$$

Здесь $f_1(x)$ — первая характеристика объекта, а $f_2(x)$ — вторая.

Очевидно, что если $f_1(x) = x$, $f_2(x) = 1$, а $\phi_1 = k'$, $\phi_2 = b'$, то формула сводится к изначальной:

$$y(x) = k'x + b'$$

Итак, линейная модель:

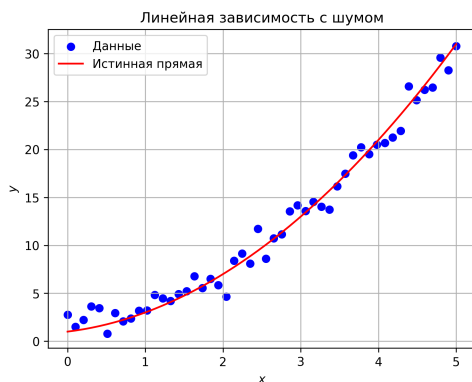
$$a(x) = \sum_{i=0}^n f_i(x)\phi_i$$

Рассмотрим конкретный пример

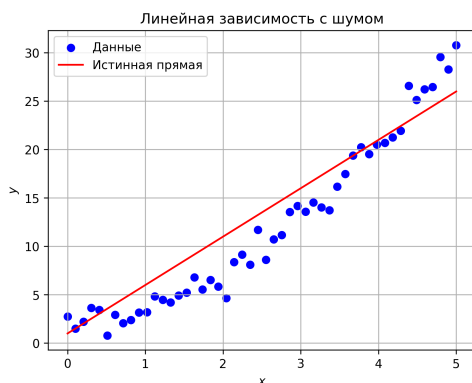
§1.3 БЕТА Переобучение

Нам дана функция:

$$f(x) = x^2 + x + 1$$



Сначала может показаться, что мы можем описать данную функцию с помощью одной характеристики: $a(x) = f(x)\phi$



Но в таком случае получится прямая линия, лишь по очертаниям похожая на нашу кривую.

В таком случае нам поможет полином:

$$\begin{cases} f_0(x) = \text{const} \\ f_1(x) = x \\ f_2(x) = x^2 \\ \vdots \\ f_n(x) = x^n \end{cases} \quad - \text{ все характеристики}$$

Таким образом наша модель (при $f_0 = 1$): $a(x) = \sum_{i=0}^n f_i(x)\phi_i = \phi_0 + x\phi_1 + \dots + x^n\phi_n$

Важно

Система характеристик является Линейно Независимой

Почему характеристики не могут быть линейно зависимыми?

Допустим задана система:

$$\begin{cases} f_0(x) = 1 \\ f_1(x) = x \\ f_2(x) = x + 5 \end{cases}$$

Получается, что $f_2(x) = 5f_1(x) + f_0(x)$. Следовательно, если одну из характеристик можно выразить через другие, то зачем же он вообще нужен? Получается, что он просто является лишним в нашей системе и можно справиться без него.

ПЕРЕПИСАТЬ ГЛАВУ

§1.4 Степень переобучения модели

1.4.1 Оценка по отложенной выборке(hold-out)

Для данного метода размеченные данные делят на две части:

- Обучающие
- Отложенная(hold-out)

Обычно данные делят в соотношении 70:30.

Цель: сравнить качество модели на данных, используемых при обучении, с новыми данными того же характера. Для этого строят два новых параметра: $Q(a, X)$ - для обучающих данных и $Q'(a, X')$ - для отложенных данных.

В результате, если средним эмпирический риск для обучающих данных меньше, чем для отложенных, то модель следует подкорректировать для лучшего показателя.

1.4.2 Скользящий контроль (leave-one-out)

Допустим, нам даны n различных размеченных данных $:= (x_1, x_2, \dots, x_n)$.
Данный метод основан на том, что мы построим n таких моделей, что :

[illegible]

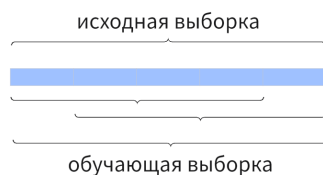
Т.е. мы построили n различных моделей $a_i(x)$, таких, что каждая из них обучалась на наборе данных размерностью $n - 1$ (для a_i — модели убрали x_i -ый вектор данных)

Ну а конечная модель $a(x) = F(a_1, a_2, \dots, a_n)$

На больших наборах данных этот способ требует огромной вычислительной мощи, потому он почти не используется на практике.

1.4.3 Кросс-валидация(cross-validation, k-fold)

Очень похожий на скользящий контроль метод, но различие состоит в том, что здесь мы разбиваем входные данные на некоторые группы и составляем из них модели:



Этот метод позволяет строить модели, которые будут обладать **лучшими обобщающими способностями, при меньшем количестве вычислений.**

1.4.4 Обобщение моделей

В прошлых частях мы встречались с обобщением модели:

$a(x) = F(a_1(x), a_2(x), \dots, a_n(x))$, но не упоминалось, как же это делается на самом деле. В основном используют два метода для обобщения моделей, познакомимся с ними поближе:

1) Выбор одной модели с лучшими показателями

Допустим у нас есть модели:

$$\begin{cases} a_1(x) \\ a_2(x) \\ \dots \\ a_n(x) \end{cases}$$

Среди них мы выбираем ту модель, у которой **средний эмпирический риск минимальный**. Но на самом деле, даже если этот показатель и наименьший, это не означает, что модель лучше остальных.

2) Выбор наиболее часто встречающегося результата

Допустим у нас есть модели, которые выдают определенный результат $\in (0, 1)$:

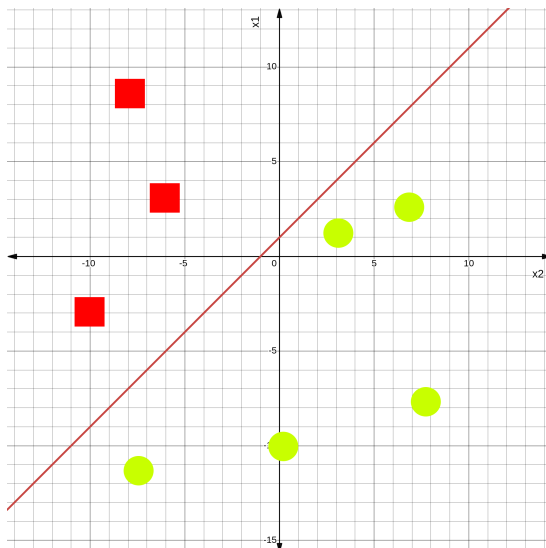
$$\begin{cases} a_1(x) = 1 \\ a_2(x) = 1 \\ \dots \\ a_{n-1}(x) = 1 \\ a_n(x) = 0 \end{cases}$$

Чаще всего встречается ответ 1, а значит мы его примем за верный.

Но, данный способ **требует большой вычислительной мощи**, т.к. предсказания нам будет давать уже не одна а n моделей.

§1.5 Уравнение гиперплоскости

Рассмотрим Здесь прямая в двумерном пространстве делит два класса предметов,

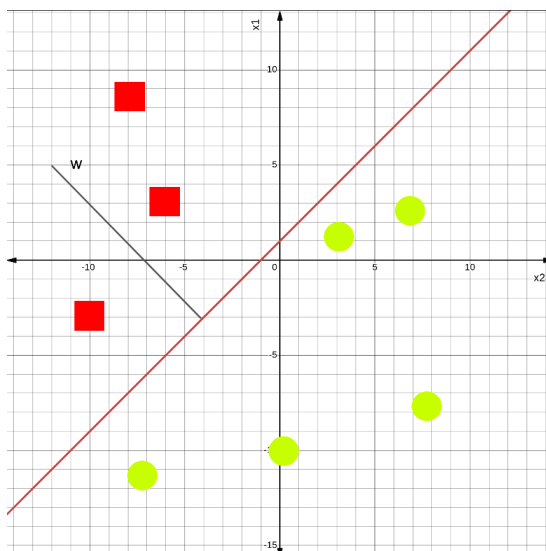


т.е. по левую часть от прямой располагаются предметы, относящиеся к одному классу, а по правую-к другому классу.

Обратимся к линейному уравнению:

$$w_1x_1 + w_2x_2 + w_0 = 0$$

Вектор $w = (w_1, w_2)^T$ является нормалью к гиперплоскости, т.е. ортогонален всем векторам, лежащим в этой плоскости.



Докажем это:

Пусть x — произвольный вектор, лежащий в гиперплоскости. Тогда:

$$w \cdot x = \|w\| \|x\| \cos(\theta)$$

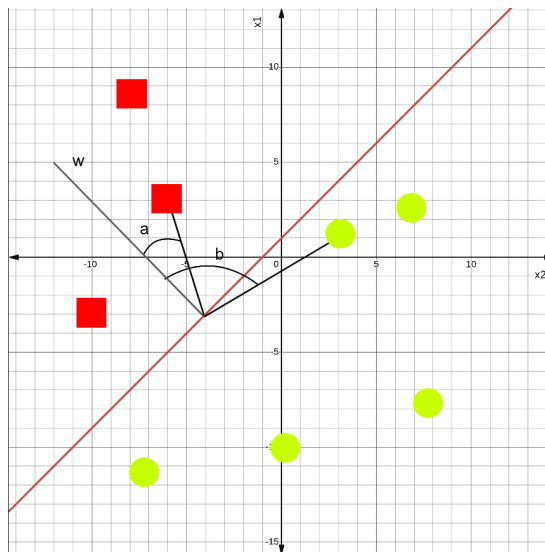
Поскольку x лежит в гиперплоскости, а w — нормаль, угол между ними 90° , следовательно:

$$\cos(\theta) = 0 \Rightarrow w \cdot x = 0$$

■

Но как же нам теперь отличать объекты одного класса от объектов другого класса с использованием полученных знаний?

Все очень просто, рассмотрим углы между ортогональной к плоскости прямой и прямой до объекта.



Заметим, что угол a является **острым углом**, как и любой другой угол между нормальную к гиперплоскости и прямой до объекта такого же класса. ($\cos(a) > 0$)

а вот угол b является **тупым**, как и любой другой угол между нормальную к гиперплоскости и прямой до объекта такого же класса. ($\cos(a) < 0$)

А значит скалярное произведение (w, x) для одного класса будет положительным, а для другого отрицательным (следует из знака \cos)

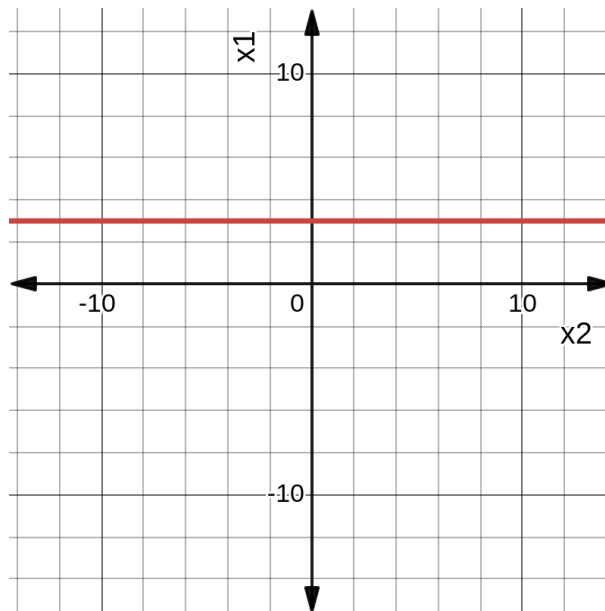
Более того, мы можем описать такую функцию как:

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

При нуле объект не будет определен к классу.

Решение задач

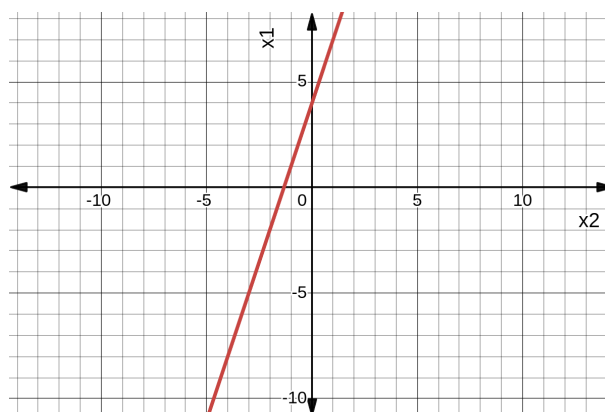
1) На графике представлена разделяющая линия в пространстве двух признаков



Требуется найти вектор коэффициентов $[w_0, w_1, w_2]^T$, удовлетворяющий линейной системе:

$$w_1 x_1 + w_2 x_2 + w_0 = 0$$

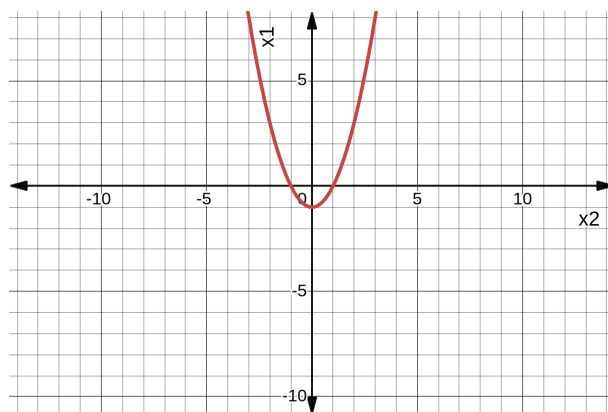
2) На графике представлена разделяющая линия в пространстве двух признаков



Требуется найти вектор коэффициентов $[w_0, w_1, w_2]^T$, удовлетворяющий линейной системе:

$$w_1 x_1 + w_2 x_2 + w_0 = 0$$

3) На графике представлена разделяющая линия в пространстве двух признаков



Требуется найти вектор коэффициентов $[w_0, w_1, w_2]^T$, удовлетворяющий линейной системе:

$$w_1x_1 + w_2x_2 + w_0 = 0$$