Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных средств

М. И. Вашкевич, Н. А. Петровский

ВСТРАИВАЕМЫЕ СИСТЕМЫ ОБРАБОТКИ МЕДИАДАННЫХ. ЛАБОРАТОРНЫЙ ПРАКТИКУМ

Рекомендовано УМО по образованию в области информатики и радиоэлектроники в качестве пособия для специальности 6-05-0611-05 «Компьютерная инженерия»

Минск БГУИР 2025

УДК 004.93(076.5)
ББК 32.972я73
В23

Р е ц е н з е н т ы:

кафедра радиофизики и цифровых медиатехнологий
Белорусского государственного университета (протокол № 15 от 21.05.2024);

декан инженерно-педагогического факультета
Белорусского национального технического университета
кандидат технических наук, доцент А. А. Дробыш

Вашкевич, М. И.

В23 Встраиваемые системы обработки медиаданных. Лабораторный практикум : пособие / М. И. Вашкевич, Н. А. Петровский. – Минск : БГУИР, 2025. – 74 с. : ил.

ISBN 978-985-543-810-7.

Содержит лабораторные работы по основам обработки медиаданных. В основе построения систем обработки медиаданных лежат фундаментальные понятия теории вероятностей и математической статистки. Первая лабораторная работа посвящена изучению методов корреляционной обработки данных с использованием языка Python. Вторая лабораторная работа по теме «Линейная регрессия» содержит подробное изложение данного фундаментального метода моделирования зависимостей в данных. В третьей лабораторной работе рассмотрена задача бинарной классификации на примере изучения базового линейного классификатора и логистической регрессии. Четвертая и пятая лабораторные работы посвящены обработке изображений. Рассмотрены задачи: преобразования яркости, бинаризации, растяжения контрастности, обработки на основе гистограмм, а также пространственной фильтрации.

УДК 004.93(076.5)
ББК 32.972я73

ISBN 978-985-543-810-7

Содержание

ЛАБОРАТОРНАЯ РАБОТА № 1. КОРРЕЛЯЦИОННЫЕ МЕТОДЫ АНАЛИЗА ДАННЫХ

ЦЕЛЬ РАБОТЫ – изучение основ корреляционного анализа данных с использованием языка Python.

1.1. Теоретические сведения

При практическом анализе данных большое значение имеет правильное понимание некоторых понятий теории вероятностей, а именно – понятий математического ожидания, ковариации и корреляции случайных величин.

1.1.1. Математическое ожидание

Математическое ожидание случайной величины X, называемое также средним значением, определяется как

$$E\{X\} = \int_{\mathcal{X}} xp(x)dx, \tag{1.1}$$

где $p(x)$ – плотность вероятности.

Определение (1.1) справедливо для непрерывных случайных величин. В данном случае интеграл берется по всей области определения $\mathcal{X}$ случайной величины X. В качестве $p(x)$ может выступать любая интегрируемая неотрицательная функция, для которой $\int p(x)dx = 1$. Важным примером непрерывного распределения является нормальное распределение с функцией плотности

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{(x-\mu)^2}{2\sigma^2}\right), \tag{1.2}$$

где $-\infty < \mu < \infty$; $0 < \sigma < \infty$; $-\infty < x < \infty$.

На рис. 1.1 приведен график плотности вероятности нормального распределения для значений параметров $\mu = 1$, $\sigma = 0{,}5$.

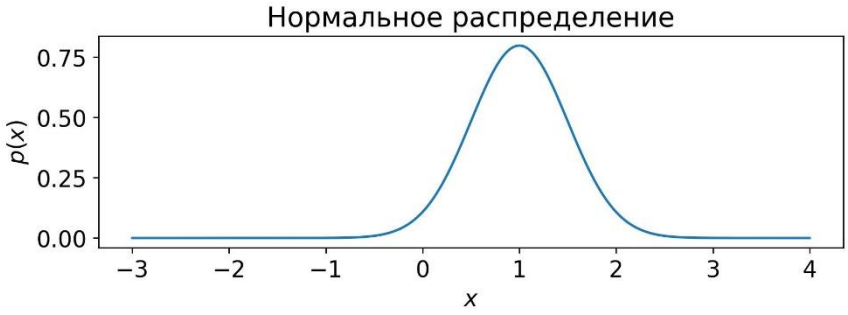


Рис. 1.1. Плотность вероятности нормального распределения

Для дискретных случайных величин математическое ожидание определяется как

$$E\{X\} = \sum_{x_i \in \mathcal{X}} x_i P(X = x_i),$$

где $P(X = x_i)$ – вероятность того, что случайная величина X примет значение x_i; $\mathcal{X}$ – множество всех возможных значений X (***фазовое пространство***).

Заметим также, что математическое ожидание применяется не только к случайным величинам, но и к функциям от случайных величин. Пусть, например, $g(X)$ – функция от случайной величины X, тогда

$$E\{g(X)\} = \int_{\mathcal{X}} g(x)p(x)dx \tag{1.3}$$

в случае, если X – непрерывная случайная величина и

$$E\{g(X)\} = \sum_{x \in \mathcal{X}} g(x)p(x), \tag{1.4}$$

если X – дискретная случайная величина.

Отметим важное свойство математического ожидания, а именно

$$E\{X + a\} = E\{X\} + a. \tag{1.5}$$

Это свойство говорит, что если к случайной величине прибавить константу a, то математическое ожидание величины также увеличится на константу a.

Имея дело с конкретной выборкой $x_1, x_2, \dots, x_N$ данных, которой представлена случайная величина X, мы, как правило, не знаем значения математического ожидания $E\{X\}$. Однако на основании выборки может быть рассчитано среднее арифметическое значение:

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i. \tag{1.6}$$

Значение $\bar{x}$ называют ***выборочным средним***. Можно показать, что при достаточно больших N арифметическое среднее совпадает с математическим ожиданием. Обычно говорят, что среднее арифметическое $\bar{x}$ является ***оценкой*** математического ожидания и записывают этот факт как

$$E\{X\} \approx \bar{x} \Leftrightarrow E\{X\} = \bar{x} + \epsilon. \tag{1.7}$$

Различие между $E\{X\}$ и $\bar{x}$ заключается в следующем: математическое ожидание – это число, т. е. неслучайная величина, а среднее арифметическое $\bar{x}$ может меняться от выборки к выборке, т. е. является случайной величиной.

Ниже приведен листинг на языке Python, который показывает, каким образом можно прочитать данные из таблицы и посчитать для них выборочное среднее.

```
import pandas as pd
import numpy as np

data = pd.read_csv("basketball_data.csv")
x = data['Age'].to_numpy()
x_bar = np.mean(x)
print(f'x_bar = {x_bar}')
```

После выполнения программы в консоли должна появиться строка

```
>> x_bar = 26.4
```

В данном примере в csv-таблице хранятся данные игроков баскетбольной команды. При помощи команды `x = data['Age'].to_numpy()` из прочитанной таблицы выбирается столбец данных с возрастом игроков и преобразуется в `numpy`-массив (тип `ndarray`). Далее при помощи команды вычисления среднего значения `np.mean(x)` находится выборочное среднее. Полученное в результате значение показывает, что средний возраст игроков команды приблизительно равен 26,4 годам.

1.1.2. Дисперсия

Дисперсия – мера рассеяния случайной величины. Дисперсия определяется как математическое ожидание квадрата отклонения случайной величины X от своего среднего значения:

$$\mathrm{Var}\{X\} = E\{(X - E\{X\})^2\}. \tag{1.8}$$

Размерность дисперсии совпадает с квадратом размерности случайной величины X. Для вычисления дисперсии можно использовать следующее выражение:

$$\begin{aligned} \mathrm{Var}\{X\} &= E\left\{X^2 + (E\{X\})^2 - 2X \cdot E\{X\}\right\} = \\ &= E\{X^2\} + (E\{X\})^2 - 2(E\{X\})^2 = E\{X^2\} - (E\{X\})^2. \end{aligned} \tag{1.9}$$

Стандартным отклонением σ называется значение квадратного корня из дисперсии:

$$\sigma = \sqrt{\mathrm{Var}\{X\}}. \tag{1.10}$$

В случае когда случайная величина представлена своей выборкой вместо дисперсии стандартного отклонения, вычисляют ***выборочную дисперсию*** и ***выборочное стандартное отклонение***. Поскольку дисперсия определяется через оператор математического ожидания (выражение (1.8)), то выборочная дисперсия состоит в замене этого оператора усреднением:

$$s^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2. \tag{1.11}$$

Выражение (1.11) называют ***смещенной*** оценкой дисперсии и редко используют на практике, чаще используется ***несмещенная*** оценка дисперсии, вычисляемая по формуле

$$s^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2. \tag{1.12}$$

Выборочное стандартное отклонение вычисляется как

$$sd = \sqrt{s^2}. \tag{1.13}$$

В следующем листинге приведен пример вычисления выборочной дисперсии и стандартного отклонения с использованием библиотеки NumPy.

```
data = pd.read_csv("basketball_data.csv")

x = data['Age' ].to_numpy()
Var_x = np.var(x)
SD_x = np.std(x)
SD2_x = np.sqrt(Var_x)
print(f'Variance[X] = {Var_x:2.3f}')
print(f'SD[X] = {SD_x:2.3f}')
print(f'sqrt(Var[X]) = {SD2_x:2.3f}')
```

После выполнения программы в консоли отобразится результат

```
>> Variance[X] = 17.107
>> SD[X] = 4.136
>> sqrt(Var[X]) = 4.136
```

Полученный результат говорит о том, что дисперсия возраста игроков составляет приблизительно 17 единиц, однако этот показатель не очень информативен, поскольку измеряется в годах в квадрате. Характеристика стандартного

отклонения лишена этого недостатка. Согласно полученным данным стандартное отклонение от среднего возраста 26,4 года в команде составляет 4,1 года.

1.1.3. Ковариация

В данном разделе будет рассмотрена ковариация. Это важная характеристика для оценки связи между случайными величинами.

Пусть X и Y случайные величины, которые заданы своими выборками парных наблюдений $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$. Требуется определить: имеется ли между X и Y линейная связь или они независимы.

Введем следующие обозначения для математического ожидания и дисперсии случайных величин X и Y:

$$E\{X\} = \mu_X, \qquad E\{Y\} = \mu_Y, \qquad \mathrm{Var}\{X\} = \sigma_X^2, \quad \mathrm{Var}\{Y\} = \sigma_Y^2.$$

Теперь рассмотрим величину $E\{(X - \mu_X)(Y - \mu_Y)\}$. В данном случае мы вычитаем из случайных величин их средние значение, т. е. «центрируем» их, и затем формируем произведение центрированных X и Y. Мы знаем, что если две случайные величины независимы, то математическое ожидание их произведения равно произведению их математических ожиданий. Теперь перенесем это на рассматриваемый нами случай. Если предположить, что X и Y независимы, то

$$E\{(X - \mu_X)(Y - \mu_Y)\} = E\{(X - \mu_X)\} \cdot E\{(Y - \mu_Y)\}. \tag{1.14}$$

Используя свойство (1.5), мы можем переписать правую часть выражения (1.14) в следующем виде:

$$E\{(X - \mu_X)\} \cdot E\{(Y - \mu_Y)\} = (E\{X\} - \mu_X) \cdot (E\{Y\} - \mu_Y) = 0 \cdot 0 = 0. \tag{1.15}$$

Полученный результат говорит о том, что если случайные величины X и Y независимы, то значение $E\{(X - \mu_X)(Y - \mu_Y)\}$ равно нулю. Можно также сделать и обратное утверждение: **ненулевое значение величины** $E\{(X - \mu_X)(Y - \mu_Y)\}$ **свидетельствует об отсутствии независимости**. Данная величина называется ***ковариацией*** между случайными величинами X и Y и обозначается как

$$\mathrm{Cov}\{X, Y\} = E\{(X - \mu_X)(Y - \mu_Y)\}.$$

Удобно рассматривать эту величину как показатель зависимости частного вида, так как она может принимать нулевое значение, даже когда X и Y не являются независимыми.

Приведенные выше рассуждения имеют общий характер. В реальной ситуации значения μ_X и μ_Y нам не известны. Поэтому на практике всегда выполняют переход от математического ожидания $E\{\}$ к его оценке путем вычисления выборочного среднего значения:

$$E\{X\} = \mu_X \approx \frac{1}{N}\sum_{i=1}^{N} x_i. \qquad (1.16)$$

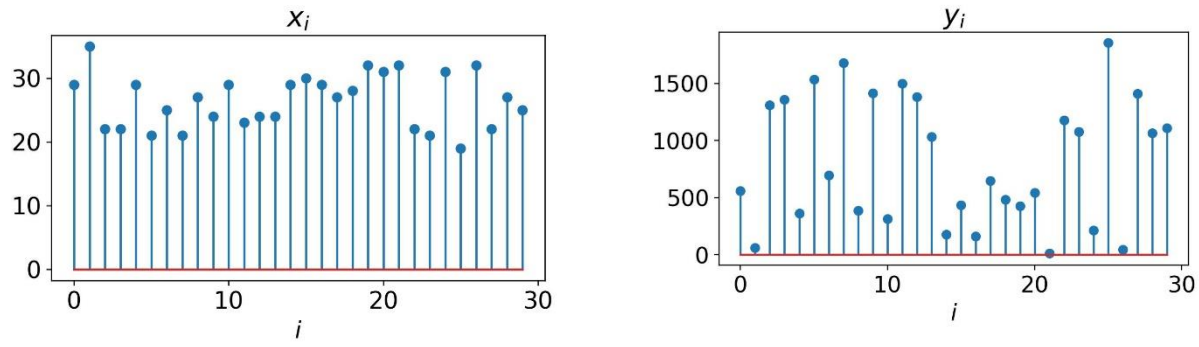
Выборочное среднее является оценкой математического ожидания μ_X, и поэтому его иногда обозначают как $\hat{\mu}_X$. В пособии мы не будем использовать данную нотацию, чтобы не запутывать читателя. Во всех рассматриваемых далее случаях под μ_X можно понимать выборочное среднее случайной величины X, представленной своей выборкой $\{x_1, x_2, \ldots, x_N\}$.

Учитывая сделанные замечания, формула для ковариации приобретает следующий вид:

$$\mathrm{Cov}(X,Y) = E\{(X-\mu_X)(Y-\mu_Y)\} \approx \frac{1}{N-1}\sum_{i=1}^{N}(x_i-\mu_X)(y_i-\mu_Y). \qquad (1.17)$$

Выражение в правой части определяет вычисление ***выборочной ковариации***.

Рассмотрим пример расчета корреляции пар случайных величин. В качестве примера возьмем набор данных, содержащий сведения о 30 игроках баскетбольной команды – их возраст и средненедельную величину финансовой поддержки. Обозначим через x_i возраст игрока с номером i, а через y_i – величину его финансовой поддержки. На рис. 1.2 представлены графики x_i и y_i.



а *б*

Рис. 1.2. Данные о баскетбольной команде:
а – возраст; *б* – финансовая поддержка

Выборочное среднее для возраста равно $\mu_X = 26{,}4$, а финансовой поддержки $\mu_Y = 814{,}4$. На рис. 1.3 представлены графики x_i и y_i после «центрирования», т. е. после вычитания из них выборочного среднего значения.

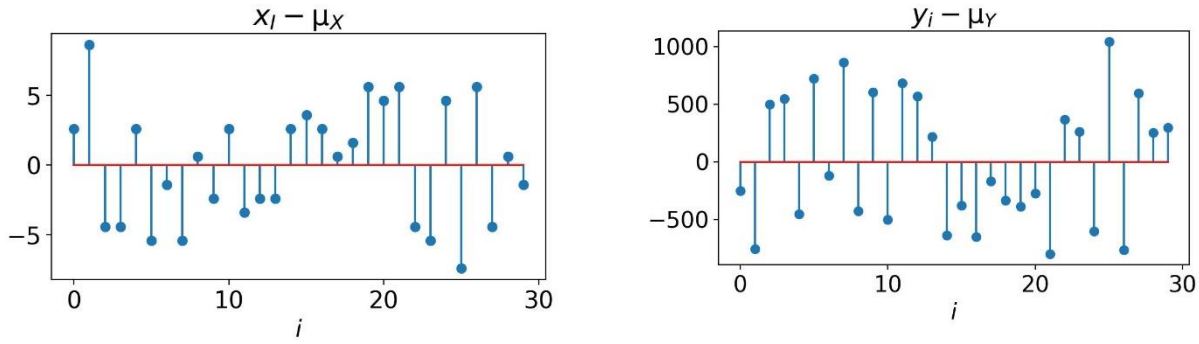


Рис. 1.3. «Центрированные» данные

На рис. 1.4 показан график произведения центрированных выборок x_i и y_i. Можно видеть, что полученный график имеет отрицательную полярность, как следствие, вычисленная по формуле (1.17) выборочная корреляция также имеет отрицательное значение $\mathrm{Cov}(X,Y) \approx -2082$.

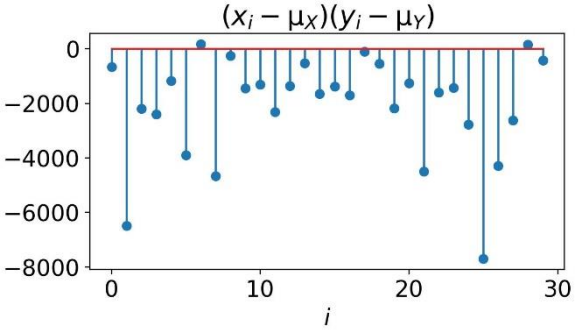


Рис. 1.4. Расчет ковариации для двух случайных величин

Отрицательное значение ковариации говорит о том, что имеет место обратная зависимость (при увеличении X значение Y уменьшается). В нашем случае это означает, что большую спонсорскую поддержку получают более молодые игроки баскетбольной команды.

Ковариация измеряется в тех же единицах, что и признаки (случайные величины X и Y). Например, величина $\mathrm{Cov}(X,Y) \approx -2082$ измеряется в единицах «доллар/год». Корреляцию удобно использовать, если нужно сравнивать аналогичные показатели в двух разных выборках.

1.1.4. Корреляция

Если в качестве показателя зависимости использовать ковариацию $\mathrm{Cov}(X,Y)$, то серьезным недостатком является то, что ее значение может меняться при изменении единиц, в которых измеряются исходные случайные величины. Данный эффект можно исключить, если разделить ковариацию на произведение среднеквадратических отклонений (СКО) $\sigma_X\sigma_Y$:

$$r_{X,Y} = \frac{\mathrm{Cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{\sum_{n=1}^{N}(x_n - \mu_X)(y_n - \mu_Y)}{\sqrt{\sum_{n=1}^{N}(x_n - \mu_X)^2 \sum_{n=1}^{N}(y_n - \mu_Y)^2}}. \quad (1.18)$$

Полученное отношение называется (выборочным) ***коэффициентом (линейной) корреляции*** между X и Y. Или просто ***корреляцией***.

Для рассмотренного выше примера

$$\sigma_X \approx 4{,}136, \quad \sigma_Y \approx 548{,}3,$$

поэтому коэффициент линейной корреляции

$$r_{X,Y} = \frac{\mathrm{Cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{-2082}{4{,}136 \cdot 548{,}3} \approx -0{,}92.$$

Можно считать, что $r_{X,Y}$ показывает, насколько близко соотношение между X и Y соответствует строго линейной зависимости. Другие формы зависимости не могут быть так хорошо измерены с помощью коэффициента корреляции $r_{X,Y}$. Полученное значение $r_{X,Y} = -0{,}92$ говорит о сильной отрицательной зависимости между признаками X (возраст) и Y (спонсорская поддержка).

Часто в решаемых задачах имеется более двух признаков (случайных величин), взаимосвязи между которыми требуется исследовать. Для этих целей используют ***ковариационную*** или ***корреляционную матрицу***.

Ковариационная матрица содержит строку и столбец для каждого признака, и каждый элемент матрицы содержит ковариацию между соответствующими парами признаков. В результате элементы главной диагонали содержат ковариацию между одноименными признаками, т. е. фактически являются дисперсиями данных признаков. Для обозначения ковариационной матрицы обычно используется греческая буква Σ. Ковариационная матрица для множества из трех признаков $\{X_1, X_2, X_3\}$ выглядит следующим образом:

$$\Sigma = \begin{bmatrix} \mathrm{Var}\{X_1\} & \mathrm{Cov}(X_1, X_2) & \mathrm{Cov}(X_1, X_3) \\ \mathrm{Cov}(X_1, X_2) & \mathrm{Var}\{X_2\} & \mathrm{Cov}(X_2, X_3) \\ \mathrm{Cov}(X_1, X_3) & \mathrm{Cov}(X_2, X_3) & \mathrm{Var}\{X_3\} \end{bmatrix}.$$

Корреляционная матрица является нормированной версией ковариационной матрицы:

$$R = \begin{bmatrix} 1 & r_{X_1,X_2} & r_{X_1,X_3} \\ r_{X_1,X_2} & 1 & r_{X_1,X_3} \\ r_{X_1,X_3} & r_{X_1,X_3} & 1 \end{bmatrix}.$$

1.1.5. Первичный анализ данных

При изучении выборки парных наблюдений $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$ случайных величин X и Y имеется два предварительных шага для определения существования и степени линейной зависимости между X и Y. Первый шаг заключается в построении ***диаграммы рассеяния***, т. е. в графическом отображении точек $(x_1, y_1), \dots, (x_N, y_N)$ на плоскости XY.

Пример построения диаграммы рассеяния на языке Python приведен ниже.

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

data = pd.read_csv("basketball_data.csv")
x = data[['Age']].to_numpy()
y = data[['Sponsorship']].to_numpy()
_,ax = plt.subplots(figsize=(6,3))
ax.scatter(x, y)
ax.grid(visible=True,linestyle='-.')
plt.xlabel('$X$')
plt.ylabel('$Y$')
```

На рис. 1.5 приведен результат выполнения данного кода.

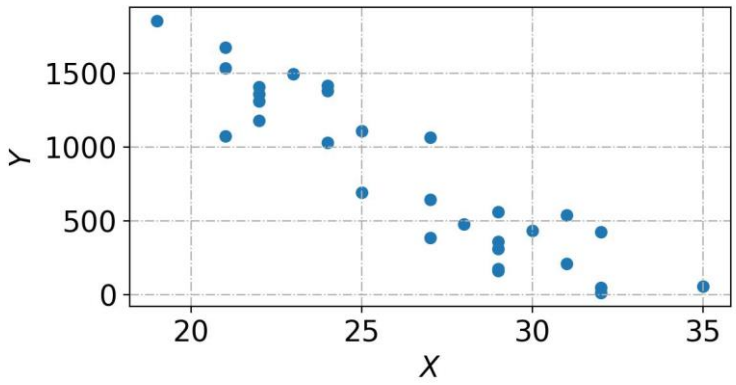


Рис. 1.5. Пример диаграммы рассеяния

Анализируя диаграмму рассеяния, можно эмпирически решить, допустимо ли предположение о линейной зависимости между X и Y. В приведенном на рис. 1.5 примере можно легко допустить наличие линейной зависимости между X и Y.

Вторым предварительным шагом является вычисление выборочного коэффициента корреляции (1.18). Если абсолютная величина коэффициента корреляции велика, это обоснованно указывает на сильную зависимость между переменными.

Иногда полезно получить корреляции и диаграмму рассеяния для различных комбинаций преобразования X и Y, например $(X, \log Y)$, $(\log X, Y)$, $(\log X, \log Y)$, $(\sqrt{X}, \log Y)$ и т. д. Преобразование, для которого получается

наибольшее по абсолютной величине значение коэффициента корреляции, будет тем преобразованием, которому соответствует наиболее сильная линейная зависимость.

Еще одним важным способом первичного анализа данных является построение ***гистограммы***. Для построения гистограммы берется набор данных $X = \{x_1, x_2, \dots, x_N\}$, представляющих выборку случайной величины X. Далее выбирается минимальный $x_{\min}$ и максимальный элемент $x_{\max}$ из выборки и формируется интервал $\text{Range} = [x_{\min}\ ;\ x_{\max}\ ]$. Интервал Range разбивается на k непересекающихся равных интервалов группировки (корзин – англ. *bins*):

$$c_0 = x_{\min} < c_1 < \cdots < c_{k-1} < c_k = x_{\max}.$$

Будем считать, что интервалы группировки имеют фиксированную длину:

$$\Delta = c_i - c_{i-1}.$$

Пусть n_i – количество значений, попадающих в i-й интервал, тогда

$$f_i = \frac{n_i}{N}$$

называется ***относительной частотой*** попаданий в i-й интервал, а

$$f_i^* = \frac{n_i}{\Delta N}$$

нормированной частотой.

Можно доказать, что сумма относительных частот равна 1.

Гистограммой называется функция $h(x)$, определенная на интервале Range, которая задается следующим образом:

$$h(x) = f_i, \qquad \text{если } x \in [c_{i-1}; c_i].$$

Нормированной гистограммой называется функция

$$h(x) = f_i^* = \frac{f_i}{\Delta}, \qquad \text{если } x \in [c_{i-1}; c_i].$$

В Python гистограмму можно построить, используя функцию histogram библиотеки NumPy. Ниже показан код, позволяющий построить гистограмму возраста игроков баскетбольной команды.

```
data = pd.read_csv("basketball_data.csv")
x = data['Age' ].to_numpy()
h,bins_ = np.histogram(x,bins=12)
fig,ax = plt.subplots(figsize=(7.5,2))
```

```
ax.bar(bins_[:-1], h)
plt.xlabel('$c_i$')
plt.ylabel('$h_i$')
```

Результат выполнения программы показан на рис. 1.6.

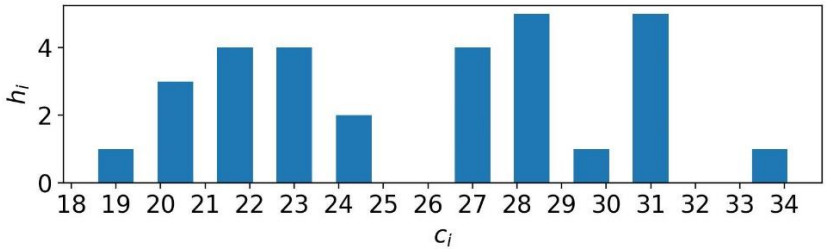


Рис. 1.6. Пример гистограммы

Можно заметить, что в функцию `np.histogram(x,bins=12)` передается выборка данных, а через именованный аргумент `bins` в переменную передается количество интервалов (корзин), на которых будет строиться гистограмма. В переменную `h` возвращается массив рассчитанных значений, а в переменную `bins_` возвращаются границы интервалов группировки.

Полученная гистограмма позволяет судить о характере распределения данных.

1.2. Порядок выполнения работы

1) Загрузите набор данных `iris`, содержащий информацию о 150 цветках ирисов, сортов Setosa, Versicolor и Virginica. В наборе данных указаны длина чашелистика (X_1), ширина чашелистика (X_2), длина лепестка (X_3) и ширина лепестка (X_4) для каждого растения. Ниже приведена ссылка на набор данных:

►`https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html`

Для выполнения последующих заданий отберите из набора данных только те элементы, которые относятся к определенному сорту в соответствии с вариантом.

Таблица 1.1. Варианты

Номер варианта (n_0)	Сорт
1, 4, 7	Setosa
2, 5, 8	Versicolor
3, 6	Virginica

Постройте диаграмму рассеяния для выбранных величин X_1 и X_3.

2) Постройте распределение вероятностей $p(x)$ длины лепестка. Для построения $p(x)$ интервал между минимальным и максимальным значением параметра разбейте на равное число интервалов (корзин), число интервалов можно задать в диапазоне от 10 до 20.

3) Опишите на языке Python, не используя сторонних библиотек, функцию `E_ave(X)` для расчета выборочного среднего значения случайной величины (по сути, эта функция будет практической реализацией оператора $E\{X\}$). Рассчитайте при помощи разработанной функции среднее значение $E\{X_1\}$ и $E\{X_3\}$. Какая из величин больше? Объясните почему.

4) Напишите на языке Python функцию `Cov_ave(X)` для расчета выборочной ковариации, используя функцию `E_ave(X)` из предыдущего задания. Рассчитайте при помощи разработанной функции выборочную ковариацию $\mathrm{Cov}\{X_1, X_3\}$. Какую размерность имеет получившаяся величина? Что она показывает?

5) Напишите на языке Python функцию `Corr_ave(X)` для расчета выборочной ковариации, используя функции `Cov_ave(X)` и `E_ave(X)`. Рассчитайте при помощи разработанной функции выборочный коэффициент корреляции $\mathrm{Corr}\{X_1, X_3\}$. Какую размерность имеет получившаяся величина? Что она показывает?

6) Представьте, что вы работаете с 4-мерной случайной величиной

$$Y = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix}.$$

Необходимо, используя разработанную функцию `Corr_ave(X)`, рассчитать корреляционную матрицу для Y.

7) Оформите отчет в соответствии с СТП 01-2017.

1.3. Дополнительные задания

1) Используя выражение (1.1), докажите, что математическое ожидание непрерывной случайной величины X с нормальным распределением (1.2) равно μ.

2) Докажите свойство (1.5) математического ожидания.

3) Найдите в интернете информацию о диаграмме размаха (англ. *boxplot*). Используя функции библиотеки `matplotlib`, постройте диаграммы размаха для признаков из набора данных `iris`.

ЛАБОРАТОРНАЯ РАБОТА № 2. ЛИНЕЙНАЯ РЕГРЕССИЯ

ЦЕЛЬ РАБОТЫ – изучение основ регрессионного анализа и реализация модели линейной регрессии на языке Python.

2.1. Теоретические сведения

2.1.1. Регрессионный анализ

В регрессионном анализе рассматривается связь между одной переменной, называемой *зависимой переменной*, и несколькими другими, называемыми *независимыми переменными*. Эта связь представляется с помощью *математической модели*, т. е. уравнения, которое связывает зависимую переменную с независимыми с учетом множества соответствующих предположений. Независимые переменные связаны с зависимой посредством *функции регрессии*, которая в свою очередь имеет набор неизвестных *параметров*. Если функция линейна относительно параметров (но необязательно линейна относительно независимых переменных), то говорят о ***линейной модели регрессии***. В противном случае модель называется *нелинейной*.

При построении модели линейной регрессии мы предполагаем, что между зависимой случайной величиной Y и независимой случайной величиной X существует связь, которую можно записать в общей форме:

$$Y = f(X) + e,$$

где f – фиксированная, но неизвестная функция от X; e – ошибка, которая не зависит от X.

Можно сказать, что функция f выражает ***систематическую*** информацию о Y, содержащуюся в X.

Регрессионный анализ используют по двум причинам. Во-первых, описание зависимости между переменными помогает установить наличие возможной причинной связи. Во-вторых, для получения предиктора (предсказателя) для зависимой переменной. Эта возможность особенно важна в тех случаях, когда прямые измерения зависимой переменной затруднены или дорого стоят.

Величина линейной зависимости между двумя переменными измеряется посредством коэффициента корреляции. Ниже будет показано, что методы регрессионного и корреляционного анализов тесно связаны между собой.

Приведем пример задачи, где возможно применение линейной регрессии. Предположим, у нас есть данные о игроках баскетбольной команды, которые включают в себя такие параметры, как рост, вес, возраст и средненедельный объем спонсорской поддержки (табл. 2.1).

Таблица 2.1. Информация о баскетбольной команде

Номер игрока	Возраст, лет	Рост, см	Вес, фунты	Спонсорская помощь, доллар/неделю
1	29	192	218	561
2	35	218	251	60
3	22	197	221	1312
4	22	192	219	1359
5	29	198	223	362
6	21	166	188	1536
7	25	195	221	694
8	21	182	199	1678
9	27	189	199	385
10	24	205	232	1416
11	29	206	246	314
12	23	185	207	1497
13	24	172	183	1383
14	24	169	183	1034
15	29	185	197	178
16	30	215	232	434
17	29	158	184	162
18	27	190	207	648
19	28	195	235	481
20	32	192	200	427
21	31	202	220	542
22	32	184	213	12
23	22	190	215	1179
24	21	178	193	1078
25	31	185	200	213
26	19	191	218	1855
27	32	196	235	47
28	22	198	221	1409
29	27	207	247	1065
30	25	201	244	1111

На основании приведенных данных, используя модель линейной регрессии, можно, например, построить функцию, которая предсказывает величину спонсорской помощи в зависимости от возраста игрока. На рис. 2.1 приведена диаграмма рассеяния для переменных возраст (X) и спонсорская помощь (Y). Из диаграммы видно, что с увеличением возраста в среднем спонсорская помощь падает. Модель линейной регрессии позволяет выразить данную взаимосвязь более строгим образом.

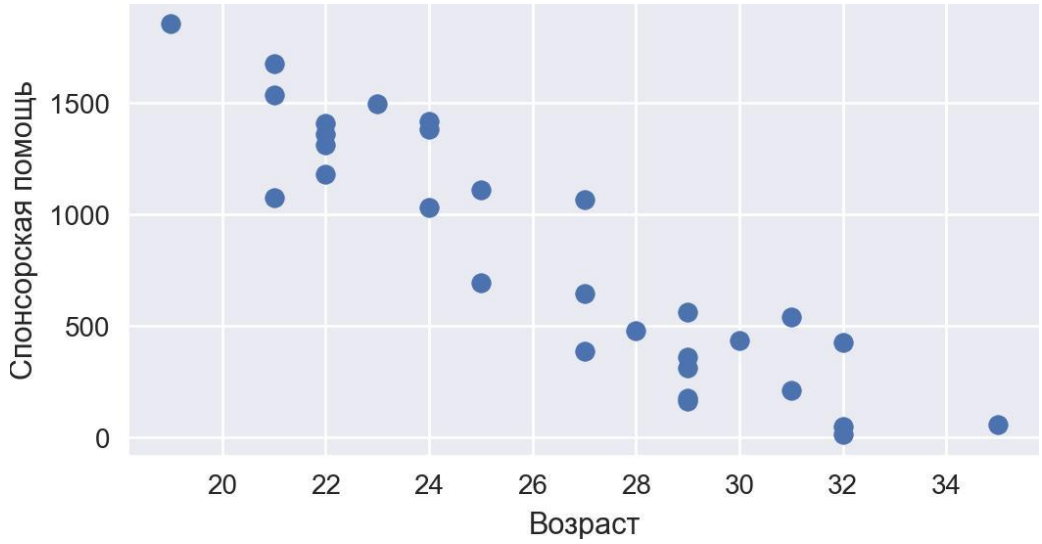


Рис. 2.1. Диаграмма рассеяния

2.1.2. Модель линейной регрессии

Если предполагается линейная зависимость между случайными величинами Y и X, то теоретическая модель задается уравнением

$$y_i = w_0 + w_1 x_i + e_i, \quad i = 1, \dots, N \tag{2.1}$$

и называется моделью ***простой линейной регрессии*** Y по X. Величины w_0 и w_1 являются неизвестными параметрами, а $e_1, \dots, e_N$ суть некоррелированные ошибки, которые можно считать выборкой случайной величины e со средним 0 и неизвестной дисперсией:

$$E\{e\} = 0, \quad \mathrm{Var}\{e\} = \sigma^2. \tag{2.2}$$

То есть для каждого значения $X = x_i$ имеется распределение Y (необязательно нормальное) со средним значением $w_0 + w_1 x_i$ и дисперсией σ^2.

Таким образом, у линейной регрессии есть три параметра: w_0 – ***пересечение*** (свободный член), w_1 – ***угловой коэффициент*** (коэффициент регрессии), σ^2 – ***дисперсия остатков.***

Таким образом, при построении модели линейной регрессии ставится задача нахождения *оценок* параметров $\hat{w}_0, \hat{w}_1, \hat{\sigma}^2$. В результате будет получена подогнанная линейная функция

$$\hat{f}(x) = \hat{w}_0 + \hat{w}_1 x,$$

которая позволит получить *оценку* зависимой переменой y_i на основании значения независимой переменной x_i:

$$\hat{y}_i = \hat{f}(x_i).$$

Зная оценки y_i, можно вычислить остатки:

$$e_i = y_i - \hat{y}_i = y_i - (\hat{w}_0 + \hat{w}_1 x_i).$$

При получении (обучении) линейной регрессии важную роль играет параметр суммы квадратов отклонений или остатков (*RSS – residual sums of squares*):

$$RSS = \sum_{i=1}^{N} {e_i}^2.$$

Каким образом можно получить оценку неизвестных значений w_0 и w_1 на основе имеющейся выборки данных объема N? Наилучшие оценки $\hat{w}_0$ и $\hat{w}_1$ для w_0 и w_1 получаются минимизацией соответственно по $\hat{w}_0$ и $\hat{w}_1$ суммы квадратов остатков:

$$RSS = \sum_{i=1}^{N} (y_i - (\hat{w}_0 + \hat{w}_1 x_i))^2. \qquad (2.3)$$

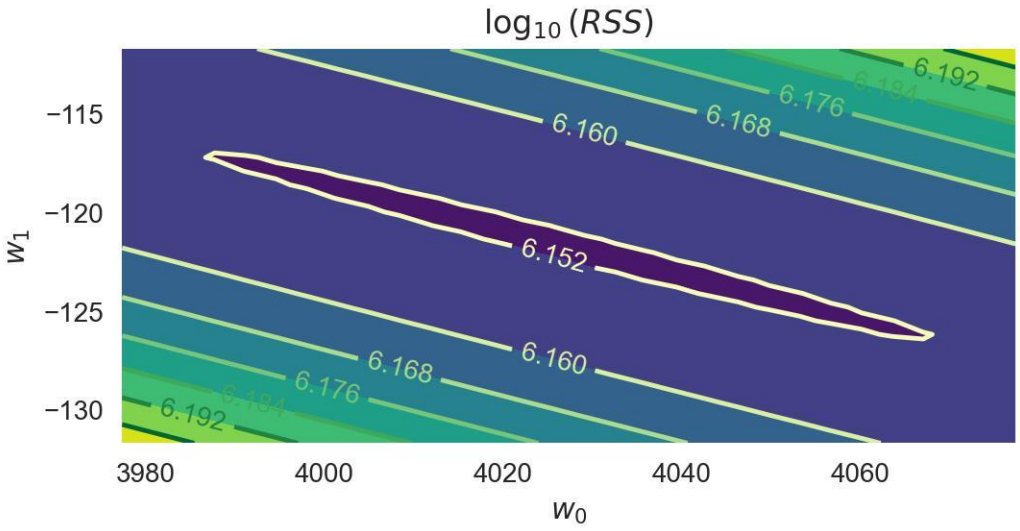


Рис. 2.2. Логарифм функции ошибки *RSS*

Из (2.3) видно, что RSS зависит от имеющейся выборки данных $(x_i, y_i)_{i=1,..,N}$, а параметрами функции являются коэффициенты регрессии $\hat{w}_0$ и $\hat{w}_1$. Для примера на рис. 2.2 показан график поверхности функции RSS модели

линейной регрессии, построенной для данных, приведенных на рис. 2.1. Поверхность ошибки RSS показывает нам, что есть определенные значения параметров $\widehat{w}_0$ и $\widehat{w}_1$, при которых функция RSS достигает минимального значения.

Поиск коэффициентов линейной регрессии

Очевидно, что RSS есть мера ошибки, возникающей при аппроксимации данных (x_i, y_i) при помощи регрессионной модели $\widehat{y_i} = \widehat{w}_0 + \widehat{w}_1 x_i$. Естественно, что нас интересуют такие значения $\widehat{w}_0$ и $\widehat{w}_1$, которые минимизируют RSS.

Для минимизации RSS найдем частные производные по $\widehat{w}_0$ и $\widehat{w}_1$ и приравняем их к нулю.

$$\frac{\partial RSS}{\partial \widehat{w}_0} = \sum_{i=1}^{N} 2(y_i - \widehat{w}_0 - \widehat{w}_1 x_i)(-1) \triangleq 0$$

$$\Rightarrow \widehat{w}_0 = \frac{1}{N}\sum_{i=1}^{N}(y_i - \widehat{w}_1 x_i),$$

т. е.

$$\widehat{w}_0 = \frac{1}{N}\sum_{i=1}^{N}(y_i - \widehat{w}_1 x_i) = \underbrace{\frac{1}{N}\sum_{i=1}^{N} y_i}_{\bar{y}} - \widehat{w}_1 \underbrace{\frac{1}{N}\sum_{i=1}^{N} x_i}_{\bar{x}} = \bar{y} - \widehat{w}_1 \bar{x}. \tag{2.4}$$

В выражении (2.4) через $\bar{y}$ и $\bar{x}$ обозначены средние значения соответствующих случайных величин.

Выполним подстановку (2.4) в (2.3) и получим

$$\begin{aligned} RSS = \sum_{i=1}^{N}\big(y_i - (\widehat{w}_0 + \widehat{w}_1 x_i)\big)^2 = \sum_{i=1}^{N}(y_i - (\bar{y} - \widehat{w}_1 \bar{x}) - \widehat{w}_1 x_i)^2 = \\ = \sum_{i=1}^{N}(y_i - \bar{y} - \widehat{w}_1 (x_i - \bar{x}))^2. \end{aligned} \tag{2.5}$$

Используя последнее выражение, найдем производную RSS по $\widehat{w}_1$:

$$\frac{\partial RSS}{\partial \widehat{w}_1} = \sum_{i=1}^{N} 2\big(y_i - \bar{y} - \widehat{w}_1 (x_i - \bar{x})\big)(-1)(x_i - \bar{x}) \triangleq 0. \tag{2.6}$$

Далее выразим $\widehat{w}_1$:

$$\widehat{w}_1 = \frac{\sum_{i=1}^{N}(y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^{N}(x_i - \bar{x})^2}. \tag{2.7}$$

Нахождение коэффициентов по формулам (2.4) и (2.7) называется ***подгонкой*** (англ. *fit*) модели линейной регрессии по ***методу наименьших квадратов***. Название обусловлено тем, что найденные коэффициенты минимизируют функцию квадратичной ошибки RSS (см. формулу (2.3)).

Анализ модели линейной регрессии

Оценкой уравнения регрессии (или прямой наименьших квадратов) будет

$$\hat{y} = \widehat{w}_0 + \widehat{w}_1 x.$$

Оценкой значения случайной величины Y при $X = x_i$ будет $\hat{y}_i = \widehat{w}_0 + \widehat{w}_1 x_i$. Разница между наблюдаемым и оцененным значением Y называется ***отклонением*** или ***остатком***

$$e_i = y_i - \hat{y}_i.$$

Прямая наименьших квадратов доставляет минимум сумме квадратов отклонений $\widehat{RSS} = \sum_{i=1}^{N} e_i^2$.

Используя (2.4) и (2.7), уравнение линейной регрессии можно записать в следующем виде:

$$\hat{y} = \underbrace{\bar{y} - \widehat{w}_1 \bar{x}}_{\widehat{w}_0} + \widehat{w}_1 x = \bar{y} + \widehat{w}_1 (x - \bar{x}). \tag{2.8}$$

Коэффициент $\widehat{w}_1$ обычно называется ***коэффициентом регрессии***, $\widehat{w}_0$ – ***свободным членом уравнения регрессии***.

Следует заметить, что коэффициент регрессии содержит умноженную на N ковариацию между X и Y и умноженную на N дисперсию X в знаменателе:

$$\widehat{w}_1 = \frac{N \cdot \mathrm{Cov}(X, Y)}{N \cdot \mathrm{Var}\{X\}} = \frac{\mathrm{Cov}(X, Y)}{\mathrm{Var}\{X\}}.$$

Отметим еще несколько полезных вещей. Согласно выражению (2.8) прямая регрессия проходит через точку $(\bar{x}, \bar{y})$. Прибавление константы ко всем значениям x (параллельный перенос) влияет только на свободный член, но не на коэффициент регрессии, поскольку тот определен в терминах отклонения от среднего значения и при параллельном переносе не изменяется. Поэтому мы можем ***центрировать значения*** x ***относительно нуля***, вычтя из каждого x_i величину $\bar{x}$, тогда свободный член будет равен $\bar{y}$. Можно даже вычесть $\bar{y}$ из всех значений y_i, чтобы получить нулевой свободный член, не изменив задачу существенным образом.

Уравнение линейной регрессии можно существенно упростить, если ***нормировать*** X, поделив все значения x_i на дисперсию $\sigma_X^2 = \mathrm{Var}\{X\}$:

$$x_i{}' = \frac{x_i}{\sigma_X^2}. \tag{2.9}$$

В этом случае $\bar{x}$ заменится на $\bar{x}' = \bar{x}/\sigma_X^2$. Для нормированных значений коэффициент регрессии имеет вид

$$\widehat{w}_1 = \frac{1}{N}\sum_{i=1}^{N}(x_i{}' - \bar{x}')(y_i - \bar{y}) = \mathrm{Cov}\{X', Y\}, \tag{2.10}$$

т. е. в качестве коэффициента регрессии можно взять ковариацию между нормированным признаком и целевой переменной. Таким образом, можно считать, что линейная регрессия состоит из двух шагов:

1. Нормировка признака (независимой переменной), см. выражение (2.9).
2. Вычисление ковариации между целевой переменной и нормированным признаком (2.10).

Важно отметить еще тот факт, что сумма отклонений ε_i, полученных методом наименьших квадратов, равна нулю:

$$\sum_{i=1}^{N}(y_i - (\widehat{w}_0 + \widehat{w}_1 x_i)) = N(\bar{y} - \widehat{w}_0 - \widehat{w}_1\,\bar{x}) = 0. \tag{2.11}$$

Это свойство интуитивно может казаться красивым, но следует помнить, что оно делает линейную регрессию чувствительной к ***выбросам***. Под выбросами понимают точки, отстоящие далеко от прямой регрессии, что нередко случается в результате ошибок измерения.

2.1.3. Множественная линейная регрессия

Описание модели

Простая модель линейной регрессии, рассмотренная выше, учитывает только один предиктор (описательный признак). Однако большинство задач аналитического прогнозирования носят многомерный характер. Распространим модель линейной регрессии на случай многомерных данных. Такая модель будет называться ***множественной линейной регрессией***.

Для начала опишем одномерную линейную регрессию в матричной форме:

$$\begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} w_0 + \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} w_1 + \begin{bmatrix} e_1 \\ \vdots \\ e_N \end{bmatrix}. \tag{2.12}$$

Теперь мы можем перейти к однородным координатам и переписать это выражение следующим образом:

$$\begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_N \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} + \begin{bmatrix} e_1 \\ \vdots \\ e_N \end{bmatrix}. \tag{2.13}$$

Теперь мы можем переписать (2.13) в матричном виде

$$\mathrm{y} = \mathbf{X}\mathrm{w} + \mathbf{e}, \tag{2.14}$$

где X – матрица размерностью $N \times 2$, а $\mathbf{y}, \mathbf{e}$ – векторы размерностью $N \times 1$, $\mathbf{w} = [w_0 \quad w_1]^T$.

В случае если предикторов становится больше, например d, то $\mathbf{X}$ просто становится матрицей $N \times (d + 1)$, первый столбец которой содержит единицы, а остальные столбцы хранят d предикторов; первым элементом $\mathbf{w}$ является свободный член, а остальные d – коэффициенты регрессии.

Легко заметить, что в формулировке (2.14) отдельное предсказание величины $\hat{y}_i$ осуществляется путем взятия i-й строки матрицы X, которую можно обозначить, как $\mathbf{X}_{i:}$, и вычисления ее *скалярного произведения* с вектором коэффициентов регрессии $\mathbf{w}$:

$$\hat{y}_i = \mathbf{X}_{i:}\mathbf{w} = \langle \mathbf{X}_{:i}, \mathbf{w} \rangle. \tag{2.15}$$

Как и в случае простой линейной регрессии, определим функцию ошибки как сумму квадратов остатков:

$$RSS = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{N} (y_i - \langle \mathbf{X}_{:i}, \mathbf{w} \rangle)^2. \tag{2.16}$$

Нормальные уравнения

Оценка $\widehat{\mathbf{w}}$ по методу наименьших квадратов (МНК) минимизирует:

$$RSS(\widehat{\boldsymbol{w}}) = \|\mathbf{y} - \hat{\mathbf{y}}\|^2 = \|\mathbf{y} - \mathbf{X}\widehat{w}\|^2. \tag{2.17}$$

МНК оценки вектора $\widehat{\mathbf{w}}$ вычисляется по формуле

$$\widehat{\mathbf{w}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \tag{2.18}$$

Давайте это докажем. Начнем с того, что перепишем (2.17) в виде

$$RSS(\widehat{\mathbf{w}}) = (\mathbf{y} - \mathbf{X}\widehat{\mathbf{w}})^T(\mathbf{y} - \mathbf{X}\widehat{\mathbf{w}}).$$

Это квадратичная функция с $d + 1$ параметрами. Продифференцируем ее по $\widehat{\mathbf{w}}$:

$$\frac{\partial RSS}{\partial \widehat{\mathbf{w}}} = -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\widehat{\mathbf{w}}), \quad \frac{\partial^2 RSS}{\partial \widehat{\mathbf{w}} \partial \widehat{\mathbf{w}}^T} = 2\mathbf{X}^T\mathbf{X}.$$

Приравняем первую производную к нулю и получим

$$-2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\widehat{\mathbf{w}}) = 0 \quad \Rightarrow \quad \widehat{\mathbf{w}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}.$$

Выражение (2.18) описывает систему нормальных уравнений.

2.1.4. Градиентный спуск

Для минимизации RSS в выражении (2.16) можно использовать метод градиентного спуска. Напомним, что градиентный спуск – это итеративный метод поиска минимума (максимума) функции многих переменных.

При инициализации в качестве w задается случайная стартовая точка в пространстве параметров модели.

На каждой итерации происходит обновление весов в соответствии со следующей схемой:

$$\mathbf{w}_t \leftarrow \mathbf{w}_{t-1} - \eta \frac{\partial RSS}{\partial \mathbf{w}}, \tag{2.19}$$

где η – скорость обучения, которая определяет, как быстро алгоритм сходится; $\mathbf{w}_t$ – значение коэффициентов модели на шаге t.

Чтобы воспользоваться (2.19), требуется вычислить частную производную по каждому отдельному параметру $\frac{\partial RSS}{\partial w_p}$. Сделаем это.

$$\begin{aligned} \frac{\partial RSS}{\partial w_p} &= \sum_{i=1}^{N} 2 \cdot \left(y_i - \langle \mathbf{X}_{:i}, \mathbf{w} \rangle\right) \cdot \frac{\partial (y_i - \langle \mathbf{X}_{:i}, \mathbf{w} \rangle)}{\partial w_p} = \\ &= \sum_{i=1}^{N} 2 \cdot \left(y_i - \langle \mathbf{X}_{:i}, \mathbf{w} \rangle\right) \cdot \left(-\mathrm{X}_{p,i}\right). \end{aligned} \tag{2.20}$$

Критерием остановки градиентного спуска может служить значение разницы минимизируемой функции на предыдущем и на текущим шаге. Если она меньше наперед заданного значения ϵ, которое обычно выбирается в диапазоне от 10^{-3} до 10^{-6}, то процесс минимизации можно завершать. Типичный диапазон для скорости обучения $\eta \in [10^{-8},\ 0{,}1]$.

2.1.5. Оценка качества линейной регрессии

Значение RSS

Значение RSS сообщает вам, какая величина отклонения (дисперсии) остается после подгонки линейной модели, которая измеряется квадратами различий между прогнозируемыми и фактическими целевыми значениями. RSS – это общая дисперсия результата.

Коэффициент детерминации R^2

Мы можем использовать коэффициент детерминации R^2 для оценки эффективности модели линейной регрессии. В основе расчета коэффициента R^2 лежит сравнение оцениваемой модели с другой, «базовой» моделью. В качестве базовой модели используется воображаемая модель, которая всегда возвращает среднее значение зависимой переменной y, которое рассчитано на тестовом наборе (либо на всем наборе данных). Таким образом, для любого x_i базовая модель всегда будет возвращать $\bar{y}$. Итак, коэффициент детерминации вычисляется как

$$R^2 = 1 - \frac{\text{сумма квадратов остатков}}{\text{общая сумма квадратов}} = 1 - \frac{RSS}{TSS}, \tag{2.21}$$

где ***общая сумма квадратов*** (англ. *total sum of squares*) рассчитывается как

$$TSS = \sum_{i=1}^{N} (y_i - \bar{y})^2. \tag{2.22}$$

Значение R^2 находится в диапазоне $[0, 1)$. Значения, близкие к единице, указывают на лучшую эффективность модели. Коэффициент детерминации R^2 также интерпретируют как величину изменения зависимой переменной y, которая объясняется независимыми (описательными) признаками x.

Заметим также, что коэффициент детерминации R^2 для случая простой линейной регрессии Y по X в точности совпадает с квадратом коэффициента корреляции r_{XY}^2.

R^2 показывает, какая часть дисперсии целевой переменной Y может быть объяснена линейной моделью. Значение R^2 колеблется между 0 для моделей, в которых модель вообще не объясняет данные, и 1 для моделей, которые объясняют все отклонения в данных.

Коэффициент R^2 позволяет оценивать эффективность модели независимо от предметной области.

Давайте рассмотрим еще один пример, где с R^2 не все так гладко. Возьмем данные о 30 игроках профессиональной баскетбольной команды (см. табл. 2.1). Допустим, что мы хотим построить линейную регрессию, чтобы предсказать размер спонсорской поддержки (y) по описательным признакам (возраст, рост, вес). Рассчитаем несколько моделей, каждый раз добавляя по одному новому признаку, и посмотрим, как будет изменяться R^2. Результаты представлены в табл. 2.2.

Таблица 2.2. Коэффициент R^2

№	Предиктор	R^2
1	Возраст	0,8427
2	Рост	0,8629
3	Вес	0,8639

Мы видим, что, если постепенно добавлять новые признаки в модель, коэффициент R^2 увеличивается.

Скорректированный коэффициент детерминации (adjusted R^2 value)

Есть один подвох в использовании коэффициента R^2. Дело в том, что R^2 всегда увеличивается (по крайней мере не уменьшается) с увеличением количества признаков в модели, даже если они вообще не содержат никакой информации о целевой переменной. Поэтому лучше использовать скорректированный R^2, который учитывает количество предикторов, используемых в модели.

$$R^2_{\text{adj}} = 1 - \left(\frac{RSS}{TSS} \times \frac{N-1}{N-K-1}\right), \tag{2.23}$$

где N – число наблюдений; K – число предикторов.

Корректировка заключается в том, что R^2_{adj} учитывает число степеней свободы. Преимущество R^2_{adj} в том, что он будет возрастать при добавлении новых предикторов только в том случае, если уменьшение ошибки RSS больше, чем мы ожидаем от добавления случайных значений. Однако недостаток R^2_{adj} в том, что мы теперь не можем интерпретировать его как долю «объясненной» дисперсии выхода, как это было в случае с R^2.

Если теперь посчитать R^2_{adj} для данных о спонсорской поддержки, то мы получим следующие значения (табл. 2.3).

Таблица 2.3. Коэффициент R^2_{adj}

№	**Предиктор**	$\boldsymbol{R^2_{\text{adj}}}$
1	Возраст	0,8427
2	Рост	0,8580
3	Вес	0,8539

Мы видим, что добавление признака «рост» позволяет нам несколько улучшить качество предсказания, однако такой фактор, как «вес», уже не улучшает работы нашей модели.

2.1.6. Значимость коэффициентов регрессии

Опишем процедуру проверки гипотезы о статистической значимости коэффициентов регрессии.

Вначале мы выдвигаем нулевую гипотезу, что данный коэффициент w_j не важен для модели (т. е. равен нулю). Далее принятие/отклонение этой гипотезы выполняется в три этапа:

1. Вычисляется тестовая статистика.

2. Вычисляется вероятность того, что значение случайной величины с распределением тестовой статистики будет больше и равно значению тестовой статистики. Эта вероятность называется ***p-значением***.

3. *P*-значение сравнивается с предопределенным порогом значимости, и если *p*-значение меньше порога, то нулевая гипотеза отклоняется. Стандартные статистические пороги равны 5 и 1 %.

Итак, что же мы будем считать?

Дело в том, что найденные нами коэффициенты регрессии w_j являются случайными величинами и, следовательно, для них можно оценить дисперсию и математическое ожидание. Известно, что w_j распределено по нормальному закону. Дисперсия коэффициента w_j линейной регрессии рассчитывается как

$$\sigma_{w_j}^2 = \frac{\sigma^2}{\sum_{i=1}^{N}(x_i - \bar{x}_i)}, \tag{2.24}$$

где σ^2 – определено в (2.2). К сожалению, σ^2 мы, как правило, не знаем, поэтому выборочная дисперсия s^2 служит ее оценкой:

$$s = \sqrt{\frac{\sum_{i=1}^{N}\left(y_i - \left(w_0 + \sum_{j=1}^{d} x_j w_j\right)\right)^2}{N-2}}. \tag{2.25}$$

В результате получаем

$$s_{w_j}^2 = \frac{s^2}{\sum_{i=1}^{N}(x_i - \bar{x}_i)^2}. \tag{2.26}$$

Величина $\frac{(N-2)s_{w_j}^2}{\sigma^2}$ распределена как χ^2 с $N-2$ степенями свободы, поэтому (по определению) статистика данного теста

$$t = \frac{w_j/\sigma}{\sqrt{\frac{(N-2)s_{w_j}^2}{\sigma^2(N-2)}}} = \frac{w_j}{\sqrt{s_{w_j}^2}} \tag{2.27}$$

подчиняется *t*-распределению с $N-2$ степенями свободы.

В критерии (2.27) мы делим числитель на σ, чтобы в числителе получилось распределение $\mathcal{N}(0,1)$. Напомним, что согласно нашей нулевой гипотезе $E\{w_j\} = 0$.

Далее, используя таблицу t-критерия, мы можем определить p-значение, и если оно меньше требуемого уровня значимости, обычно 0,05 или 0,01, то мы отклоняем нулевую гипотезу и говорим, что признак x_j оказывает значительное влияние на модель.

Рассчитаем значимость коэффициентов регрессии для данных о баскетбольной команде (табл. 2.4).

Таблица 2.4. Значимость коэффициентов модели линейной регрессии для предсказания спонсорской поддержки баскетбольного игрока

№	**Предиктор**	**Коэффициент** w_j	**Значение** t	p**-значение**
1	Возраст	–128,76	–13,935	4,0515e–14
2	Рост	8,93	3,127	0,004
3	Вес	–2,08	–1,059	0,298

Полученные данные говорят о том, что исключительное значение имеет возраст. Также высокое значение имеет рост. В свою очередь вес не имеет статистической значимости (p-значение больше 0,05).

2.2. Порядок выполнения работы

1) Загрузите набор данных согласно варианту (табл. 2.5).

Таблица 2.5. Наборы данных по вариантам

Номер варианта (n_0)	Набор данных	Зависимая переменная (Y)
1	mtcars	Пробег (миль на галлон/ mpg)
2	mtcars	Полная мощность / hp
3	wine	Цена
4	Fish (категория Bream)	Вес
5	Fish (категория Roach)	Цена
6	Fish (категория Perch)	Цена
7	Real estate (записи с 1 по 50)	Цена
8	Real estate (записи с 100 по 150)	Цена

2) Используя набор данных согласно варианту, самостоятельно выберите независимый параметр X_1, далее:

а) постройте диаграмму рассеяния для выбранных величин X_1 и Y;

б) рассчитайте коэффициент корреляции $r_{X_1,Y}$;

в) постройте линейную регрессию Y по X_1 и выведите на экран полученную линию регрессии;

г) выведите в консоль значение минимизированной суммы квадратов остатков (RSS);

д) посчитайте коэффициент детерминации R^2 для полученной модели простой линейной регрессии.

3) Выберите дополнительный независимый параметр X_2 и постройте множественную линейную регрессию Y по X_1 и X_2:

а) выведите в консоль найденные параметры множественной линейной регрессии;

б) постройте (3D) диаграмму рассеяния и плоскость, соответствующую регрессионной функции;

в) выведите в консоль значение минимизированной суммы квадратов остатков (RSS). Стало ли значение RSS меньше, чем при простой линейной регрессии?

г) посчитайте коэффициент детерминации R^2 для модели множественной линейной регрессии;

д) поочередно (один за другим) добавьте в модель множественной регрессии все имеющиеся описательные признаки. После каждого очередного добавления пересчитайте коэффициент детерминации R^2. Результаты сведите в таблицу. Что можно сказать по этой таблице?

е) повторите предыдущий пункт, только на этот раз рассчитывайте скорректированный коэффициент детерминации R^2.

4) Рассчитайте и выведите на экран p-значения для коэффициентов множественной линейной регрессии, полученной в задании № 3.

5) Оформите отчет в соответствии с СТП 01-2017.

2.3. Дополнительные задания

1) Измените одно значение в наборе данных и покажите чувствительность линейной регрессии к выбросам.

2) Постройте поверхность ошибок $RSS(w_0, w_1)$, используя сетку значений w_0 и w_1. Плоскость $w_0 - w_1$ называют весовым пространством.

3) Выполните нормализацию данных и повторите процесс обучения множественной регрессии. При фиксированной скорости обучения и числе итераций градиентного спуска в каком случае получается лучше результат: с применением нормализации или без нее?

4) Постройте график изменения функции ошибки RSS в процессе градиентного спуска.

ЛАБОРАТОРНАЯ РАБОТА № 3. БИНАРНАЯ КЛАССИФИКАЦИЯ

ЦЕЛЬ РАБОТЫ – изучение основ построения классификатора на основе базового линейного классификатора и логистической регрессии с использованием языка Python.

3.1. Теоретические сведения

3.1.1. Задача бинарной классификации

Под бинарной классификацией понимают задачу отнесения объектов к одному из двух классов. Классическим примером такой задачи может служить разделение входной почты на «спам» и «неспам». При этом объекты, которые подлежат классификации, обладают параметрами – $X_1, X_2, \dots, X_d$, которые иногда называют еще ***предикторами*** или ***описательными признаками***. С точки зрения математического описания предикторы представляют собой случайные величины. Иногда бывает удобно предикторы объединить в один вектор:

$$X \coloneqq (X_1, X_2, \dots, X_d).$$

Помимо признаков каждый объект имеет метку Y, которая называется ***целевой переменной.*** Пусть $\mathcal{Y}$ – область значений Y. В задаче классификации $\mathcal{Y}$ – это конечное множество классов (пространство меток классов). Иногда говорят, что Y относится к ***категориальному*** типу данных. В задаче бинарной классификации $\mathcal{Y} \in \{0,1\}$. Для обучения модели классификатора используется $\mathcal{D} \subset \mathcal{P}(\mathcal{X} \times \mathcal{Y})$ – множество примеров из неизвестного совместного распределения $p(X, Y)$ входов и выходов, которое называется ***данными***.

$\mathcal{D}$ обычно записывается списком:

$$\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}.$$

Таким образом, между описательными признаками и целевой переменной есть функциональная связь, выражаемая как

$$f: X \to Y.$$

Иногда $f(X)$ называются ***помечающей функцией***. Задача классификации – предсказать Y на основе X. Таким образом, под обучением классификатора понимается построение функции $\hat{f}(x)$, которая как можно лучше аппроксимирует $f(x)$.

Используя вероятностную интерпретацию, можно сказать, что необходимо оценить функцию

$$f(x) \coloneqq E\{Y|X = x\} = \int_{\mathcal{Y}} y \cdot p(y|x)dy$$

на основе имеющихся данных $\mathcal{D}$.

3.1.2. Базовый линейный классификатор

Базовый линейный классификатор относится к геометрическому типу моделей. В таких моделях используются геометрические понятия – прямые, плоскости, расстояния и др. Преимущество геометрических моделей в том, что их легко визуализировать, особенно в 2- и 3-мерном пространстве.

Рассмотрим построение простейшего ***базового линейного классификатора***. Пусть имеются два множества точек D_0 и D_1 в пространстве $\mathbb{R}^d$, соответствующие объектам двух разных классов (обозначенных метками «0» и «1»). Элементами множеств D_0 и D_1 являются d-мерные векторы $\mathbf{x} = [x_1, x_2, \dots, x_d]$. Каждый такой вектор содержит признаки объекта. На рис. 3.1 приведен пример точек, относящихся к двум разным классам в двумерном пространстве.

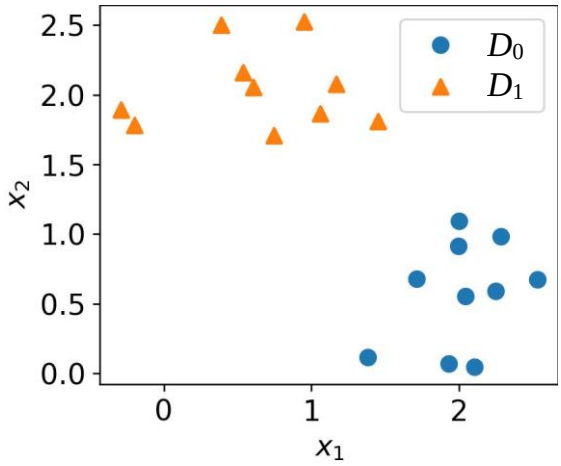


Рис. 3.1. Пример множеств объектов двух классов

Множества D_0 и D_1 называются ***линейно разделимыми***, если существует линейная разделяющая поверхность (гиперплоскость), решающая задачу классификации:

$$y(\mathrm{x}) = \sum_{i=1}^{d} x_i w_i + w_0, \tag{3.1}$$

такая, что

$$\begin{aligned} y(\mathbf{x}) &< 0, \text{если } \mathrm{x} \in D_0, \\ y(\mathbf{x}) &> 0, \text{если } \mathrm{x} \in D_1. \end{aligned} \tag{3.2}$$

В выражении (3.1) параметры w_i определяют гиперплоскость в пространстве признаков.

Сделаем два дополнения, которые помогут нам упростить нотацию и которые часто используются в литературе по машинному обучению.

1) Дополним размерность вектора $\mathbf{x}$, записав в нулевую позицию значение $x_0 = 1$. После этого все наши векторы будут иметь вид $(1, x_1, x_2, \dots, x_d)$, а выражение (3.1) упростится:

$$y(\mathrm{x}) = \sum_{i=0}^{d} x_i w_i. \tag{3.3}$$

Мы можем это сделать, поскольку теперь переменная w_0 будет умножаться на константный признак $x_0 = 1$.

2) Применим линейную алгебру и вместо суммы (3.3) будем использовать операцию скалярного произведения:

$$y(\mathbf{x}) = \mathbf{w}^T\mathbf{x}. \tag{3.4}$$

При желании мы всегда можем вернуться от формулировки (3.4) к (3.1), что иногда бывает полезно.

Гиперплоскость, разделяющая данные разных классов, также называется ***решающей границей*** и описывается уравнением

$$\mathbf{w}^T\mathbf{x} = 0. \tag{3.5}$$

Таким образом, вектор $\mathbf{w}$ перпендикулярен решающей гиперплоскости и направлен в сторону данных, помеченных как «1» (положительные объекты).

В ***базовом линейном классификаторе*** вектор $\mathbf{w}$ представляется, как вектор, соединяющий «центр масс» отрицательных (D_0) и положительных (D_1) объектов. Под центром масс здесь понимается усредненный вектор каждого класса. Так, для отрицательного класса имеем

$$\bar{\mathbf{x}}_0 = \frac{1}{|D_0|}\sum_{\mathrm{x}\in D_0} \mathbf{x},$$

аналогично для положительного класса

$$\bar{\mathbf{x}}_1 = \frac{1}{|D_1|}\sum_{\mathrm{x}\in D_1} \mathbf{x}.$$

В данных выражениях $|D_0|$ и $|D_1|$ – это число элементов каждого класса. Можно сказать, что точки множества D_0 группируются вокруг $\bar{\mathbf{x}}_0$ (условный «центр масс»). Также и точки из D_1 группируются вокруг $\bar{\mathbf{x}}_1$. Пример вычисления $\bar{\mathbf{x}}_0$ и $\bar{\mathbf{x}}_1$ показан на рис. 3.2.

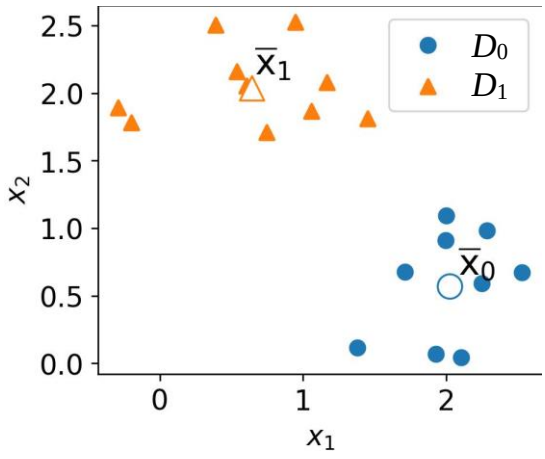


Рис. 3.2. Центры масс для двух классов

Теперь у нас уже есть точки $\bar{\mathbf{x}}_0$ и $\bar{\mathbf{x}}_1$,так что мы можем определить вектор, который их соединяет, как $\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_0$. Пример вектора $\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_0$ показан на рис. 3.3.

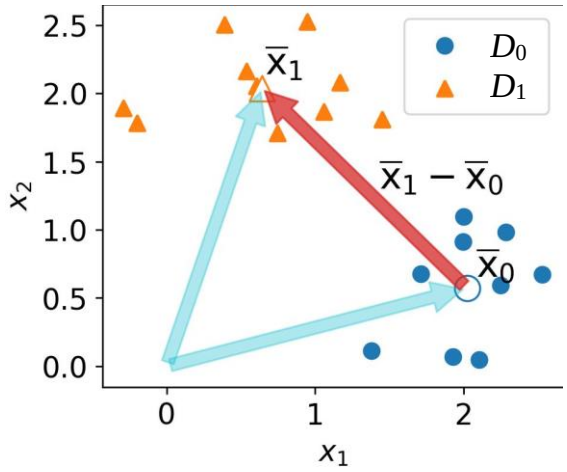


Рис. 3.3. Вектор, соединяющий центры масс

Вернемся к исходной задаче. Напомним, что мы строим базовый линейный классификатор. Критерий, по которому он работает, – это расстояние между неизвестной точкой данных $\mathbf{x}$ и центрами масс $\bar{\mathbf{x}}_0$ и $\bar{\mathbf{x}}_1$. Другими словами – точка $\mathbf{x}$ будет классифицирована как объект 0-го класса, если расстояние $|\mathbf{x} - \bar{\mathbf{x}}_0| < |\mathbf{x} - \bar{\mathbf{x}}_1|$, иначе $\mathbf{x}$ будет присвоена метка «1». По этой причине разделяющая поверхность должна проходить через середину вектора $\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_0$.

Построим базовый линейный классификатор для 2-мерного случая, для этого вернемся к выражению (3.1):

$$y(\mathbf{x}) = x_2 w_2 + x_1 w_1 + w_0. \tag{3.6}$$

Вектор $\mathbf{w} = [w_1, w_2]$, который является нормалью к искомой поверхности, мы уже нашли, он равен $\mathbf{w} = \bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_0$. Остается найти свободный член w_0, который будет фиксировать положение разделяющей прямой вдоль $\mathbf{w}$. Мы уже знаем, что разделяющая прямая должна проходить через середину «отрезка» $\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_0$. Для удобства обозначим $\bar{\mathbf{x}}_0$ как точку $N(n_1, n_2)$, а $\bar{\mathbf{x}}_1$ как точку $P(p_1, p_2)$. Найдем теперь координаты середины «отрезка» NP:

$$x_c = \left(\frac{n_1 + p_1}{2}, \frac{n_2 + p_2}{2}\right).$$

В точке x_c значение $y(x_c)$ должно быть равно нулю, т. е.

$$y(x_c) = \frac{n_2 + p_2}{2} w_2 + \frac{n_1 + p_1}{2} w_1 + w_0 = 0.$$

Вспомним, что $w_2 = p_2 - n_2$, а $w_1 = p_1 - n_1$, тогда получаем

$$\frac{n_2 + p_2}{2}(p_2 - n_2) + \frac{n_1 + p_1}{2}(p_1 - n_1) + w_0 = 0.$$

Преобразуем это выражение:

$$\frac{1}{2}(p_2^2 + p_1^2) - \frac{1}{2}(n_2^2 + n_1^2) + w_0 = 0.$$

Таким образом,

$$w_0 = \frac{1}{2}(n_2^2 + n_1^2) - \frac{1}{2}(p_2^2 + p_1^2) = \frac{1}{2}(|\bar{\mathbf{x}}_0|^2 - |\bar{\mathbf{x}}_1|^2).$$

В результате получаем следующее выражение для функции принятия решения:

$$y(\mathbf{x}) = x_2(p_2 - n_2) + x_1(p_1 - n_1) + \frac{1}{2}(|\bar{\mathbf{x}}_0|^2 - |\bar{\mathbf{x}}_1|^2).$$

Соответственно, разделяющая линия определяется как

$$x_2(p_2 - n_2) + x_1(p_1 - n_1) + \frac{1}{2}(|\bar{\mathbf{x}}_0|^2 - |\bar{\mathbf{x}}_1|^2) = 0$$

или

$$(\bar{\mathrm{x}}_1 - \bar{\mathrm{x}}_0)^T \cdot \mathbf{x} + \frac{1}{2}(|\bar{\mathrm{x}}_0|^2 - |\bar{\mathrm{x}}_1|^2) = 0.$$

Для практического отображения удобнее выразить зависимость x_2 от x_1:

$$x_2 = -x_1 \frac{(p_1 - n_1)}{(p_2 - n_2)} + \frac{1}{2}(|\bar{\mathbf{x}}_0|^2 - |\bar{\mathbf{x}}_1|^2).$$

Для рассматриваемого примера решающая граница показана на рис. 3.4.

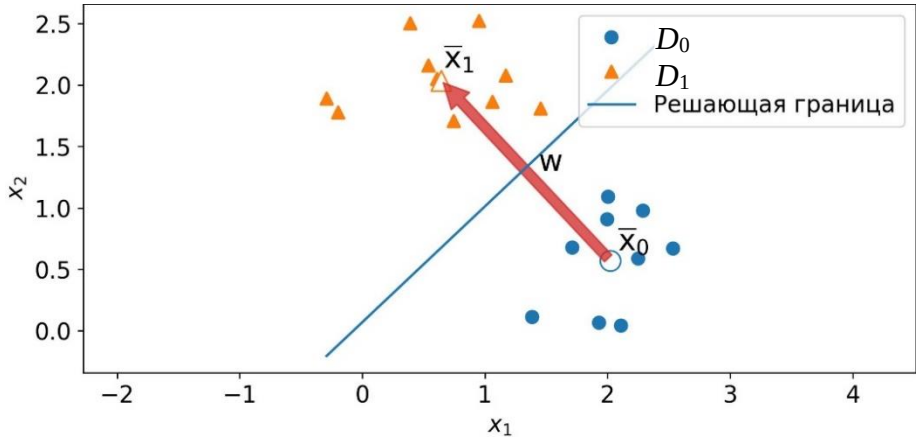


Рис. 3.4. Решающая граница

Достоинством базового линейного классификатора (БЛК) является его простота – он определен в терминах сложения, вычитания и умножения векторов-примеров. При некоторых дополнительных предположениях о данных БЛК – это лучшее, на что мы только можем надеяться.

3.1.3. Общие замечания о логистической регрессии

Задача классификации может быть разбита на два этапа: этап вывода (англ. *inference*), на котором мы используем данные, чтобы обучить модель для получения оценки вероятности $p(y_k|X)$, и последующий этап принятия решения, на котором мы используем апостериорные вероятности для оптимального назначения класса.

Альтернативный подход заключается в том, чтобы решить обе задачи сразу и просто обучить функцию, которая преобразует входы X непосредственно в решения. Такие функции называются ***дискриминантными***. Примером дискриминантной функции является БЛК, однако наиболее распространенной дискриминантной моделью является логистическая регрессия.

3.1.4. Вывод функции логистического сигмоида

Применим порождающий (генеративный) подход для задачи бинарной классификации и для вывода функции логистического сигмоида.

Пусть через C_k обозначается метка k-го класса. В генеративном подходе моделируется условная вероятность данных (зависящая от класса), т. е. $p(X|C_k)$, а также априорная вероятность классов $p(C_k)$, которые затем применяются для вычисления апостериорной вероятности $p(C_k|X)$ с использованием теоремы Байеса.

Рассмотрим случай двух классов. Апостериорную вероятность для класса C_1 можно записать как

$$p(C_1|X) = \frac{p(X|C_1)p(C_1)}{p(X|C_1)p(C_1) + p(X|C_2)p(C_2)} = \\ = \frac{1}{1+e^{-a}} = \text{logistic}(a), \tag{3.7}$$

где мы определили

$$a = \ln \frac{p(X|C_1)p(C_1)}{p(X|C_2)p(C_2)}. \tag{3.8}$$

Функция $\text{logistic}(a)$, показанная на рис. 3.5, называется логистическим сигмоидом.

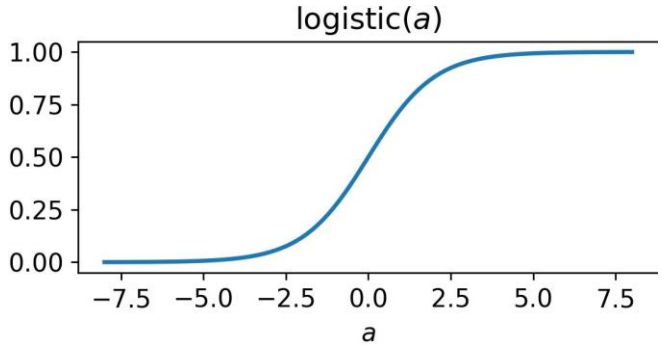


Рис. 3.5. Логистический сигмоид

Термин сигмоид означает S-образность. Логистический сигмоид обладает свойством симметрии

$$\text{logistic}(-x) = 1 - \text{logistic}(x). \tag{3.9}$$

Обратной к логистическому сигмоиду является логит-функция:

$$\text{logit}(x) = \log\left(\frac{x}{1-x}\right).$$

График логит-функции представлен на рис. 3.6.

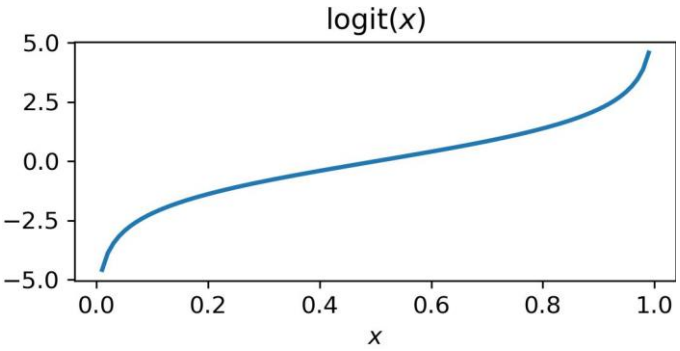


Рис. 3.6. Логит-функция

Можно выделить следующие свойства логит-функции:
1) определена в диапазоне от 0 до 1;
2) стремится к ∞ при $x \to 1$;
3) стремится к $-\infty$ при $x \to 0$;
4) симметрична относительно точки $(0, 0.5)$.

3.1.5. Логистическая регрессия

Линейная регрессия не подходит для предсказания вероятностей, поскольку предсказываемые ею значения принципиально не ограничены. Чтобы преодолеть это препятствие, используется специальный прием для преобразования неограниченной переменной с помощью функции в единичный интервал $[0, 1]$.

$$P(Y = 1|\mathbf{x}) = p_w(\mathbf{x}) = \text{logistic}(\mathbf{w}^T\mathbf{x}) + \varepsilon = \frac{1}{1 + e^{-\mathbf{w}^T\mathbf{x}}}. \quad (3.10)$$

Таким образом, у логистической регрессии есть $d + 1$ настроечный параметр w_i.

Мы будем использовать метод максимального правдоподобия для определения параметров модели логистической регрессии. Чтобы это сделать, мы также должны будем использовать производную логистического сигмоида, которая выражается в терминах этого же сигмоида:

$$\frac{d\text{logistic}}{\text{d}x} = \text{logistic}(x)(1 - \text{logistic}(x)). \quad (3.11)$$

Допустим, что у нас имеется набор данных $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)$, причем $y_i \in \{0, 1\}$.

Насколько хорошо модель (3.10) описывает наш набор данных? Ответить на этот вопрос поможет функция условного правдоподобия:

$$CL(\mathrm{w}) = \prod_{i=1}^{N} p_w(\mathbf{x}_i)^{y_i}\big(1 - p_w(\mathbf{x}_i)\big)^{1-y_i}.$$

Как обычно, проще работать с логарифмом функции правдоподобия:

$$\begin{aligned} LCL(\mathrm{w}) = \ln CL(\mathrm{w}) = \sum_{i=1}^{N} y_i \ln p_w(\mathbf{x}_i) + (1 - y_i)\ln\big(1 - p_w(\mathbf{x}_i)\big) = \\ = \sum_{\forall i \mid y_i = 1} \ln p_w(\mathbf{x}_i) + \sum_{\forall i \mid y_i = 0} \ln(1 - p_w(\mathbf{x}_i)). \end{aligned} \tag{3.12}$$

Чтобы найти оптимальные параметры логистической регрессии $\mathbf{w}$, нужно максимизировать функцию правдоподобия (3.12). Часто задачу обучения формулируют как минимизацию функции ошибки, в данном случае можно взять величину $LCL(\mathrm{w})$ со знаком минус, что приводит к функции кросс-энтропии:

$$E(\mathbf{w}) = -LCL(\mathrm{w}) = -\sum_{i=1}^{N} y_i \ln p_w(\mathbf{x}_i) + (1 - y_i)\ln(1 - p_w(\mathbf{x}_i)). \tag{3.13}$$

В точке минимума частные производные E по $\mathbf{w}$ должны быть равны нулю:

$$\nabla_w E(\mathbf{w}) = 0. \tag{3.14}$$

Сделаем некоторые подготовительные действия:

$$\ln p_w(\mathbf{x}_i) = \ln \frac{\exp(\mathbf{w}^T \mathbf{x}_i)}{\exp(\mathbf{w}^T \mathbf{x}_i) + 1} = \mathbf{w}^T \mathbf{x}_i - \ln(\exp(\mathbf{w}^T \mathbf{x}_i) + 1). \tag{3.15}$$

Вычислим производную по w_i для положительных примеров:

$$\begin{aligned} \frac{\partial}{\partial w_j} \ln p_w(\mathbf{x}_i) = x_{i,j} - \frac{\partial}{\partial w_j}(\ln(\exp(\mathbf{w}^T \mathbf{x}_i) + 1)) = \\ = x_{i,j} - \frac{1}{\exp(\mathbf{w}^T \mathbf{x}_i) + 1} \exp(\mathbf{w}^T \mathbf{x}_i)\, x_{i,j} = \end{aligned}$$

$$= x_{i,j}\left(1 - \frac{\exp(\mathbf{w}^T\mathbf{x}_i)}{\exp(\mathbf{w}^T\mathbf{x}_i) + 1}\right) = x_{i,j}\big(1 - p_w(\mathbf{x}_i)\big).$$

Если взять производную по всему вектору $\mathbf{w}$ сразу, то получим

$$\nabla_w \ln p_w(\mathbf{x}_i) = X_i\big(1 - p_w(\mathbf{x}_i)\big).$$

Для отрицательных примеров имеем

$$\ln(1 - p_w(X_i)) = \ln\frac{1}{\exp(\mathbf{w}^T\mathbf{x}_i) + 1} = -\ln(\exp(\mathbf{w}^T\mathbf{x}_i) + 1).$$

Вычислим производную по w_i для отрицательных примеров:

$$\frac{\partial}{\partial w_j}\ln\big(1 - p_w(X_i)\big) = -\frac{\partial}{\partial w_j}(\ln(\exp(\mathbf{w}^T\mathbf{x}_i) + 1)) =$$

$$= -\frac{1}{\exp(\mathbf{w}^T\mathbf{x}_i) + 1}\exp(\mathbf{w}^T\mathbf{x}_i)\, x_{i,j} =$$

$$-x_{i,j}\frac{\exp(\mathbf{w}^T\mathbf{x}_i)}{\exp(\mathbf{w}^T\mathbf{x}_i) + 1} = -x_{i,j}p_w(\mathbf{x}_i).$$

Если взять производную по всему вектору $\mathbf{w}$ сразу для отрицательных примеров, то получим

$$\nabla_w\,(1 - \ln p_w(\mathbf{x}_i)) = -\mathbf{x}_i p_w(\mathbf{x}_i)).$$

Вычислим теперь градиент кросс-энтропийной функции ошибки:

$$\nabla_w E(w) = -\nabla_w\left(\sum_{i=1}^{N} y_i \ln p_w(\mathbf{x}_i) + (1 - y_i)\ln\big(1 - p_w(\mathbf{x}_i)\big)\right) =$$

$$= -\sum_{i=1}^{N} y_i \nabla_w \ln p_w(\mathbf{x}_i) + (1 - y_i)\nabla_w\,(1 - \ln p_w(\mathbf{x}_i)) =$$

$$= -\sum_{i=1}^{N} y_i\mathbf{x}_i(1 - p_w(\mathbf{x}_i)) + (1 - y_i)(-\mathbf{x}_i p_w(\mathbf{x}_i))) =$$

$$= -\sum_{i=1}^{N} y_i \mathbf{x}_i - y_i \mathbf{x}_i p_w(\mathbf{x}_i) - \mathbf{x}_i p_w(\mathbf{x}_i) + y_i \mathbf{x}_i p_w(\mathbf{x}_i) =$$

$$= \sum_{i=1}^{N} (p_w(\mathbf{x}_i) - y_i)\mathbf{x}_i.$$

Таким образом, вклад в градиент от данных $\mathbf{x}_i$ задается ошибкой $(p_w(\mathbf{x}_i) - y_i)$ между целевой переменной и предсказанным моделью значением. Более того, если внимательно присмотреться, то можно видеть, что выражение для градиента полностью совпадает с таким же выражением для обучения линейно регрессии.

Поскольку градиент $\nabla_w E$ уже найден, то обучать логистическую регрессию можно все тем же методом градиентного спуска:

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla_w E. \tag{3.16}$$

3.1.6. Интерпретация логистической регрессии как нейронной сети

Базовая модель нейрона имеет набор соединяющих весов w_i (соответствующих синапсам в биологическом нейроне), суммирующее устройство и функцию активации, как показано на рис. 3.7.

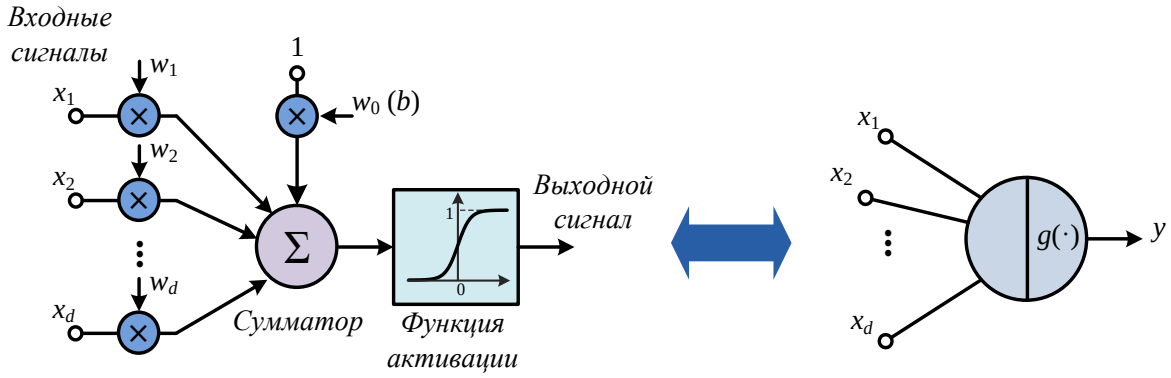


Рис. 3.7. Базовая модель нейрона

Представленная на рисунке модель нейрона описывается выражением

$$y = g\left(\sum_{i=1}^{d} w_i x_i + b\right), \tag{3.17}$$

где x_i – выходные данные; w_i – веса; b – смещение; $g(\cdot)$ – функция активации.

Рассмотрим описание (3.17) более детально. На вход функции активации $g()$ подается взвешенная комбинация входных данных (признаков):

$$v = \sum_{i=1}^{d} w_i x_i + b. \tag{3.18}$$

Каждый нейрон имеет поляризацию (связь с весом b, по которой поступает единичный сигнал), а также множество связей с весами w_i, по которым поступают входные данные. Для упрощения можно представить, что входные данные представляют собой вектор $\mathbf{x}$ с компонентами $x_1, x_2, \ldots, x_d$. В этом случае (3.18) перепишется в виде

$$v = \mathrm{w}^T \mathbf{x} + b, \tag{3.19}$$

где $\mathbf{x} = [x_1 \quad x_2 \quad \cdots \quad x_d]^T$ – вектор данных; $\mathbf{w} = [w_1 \quad w_2 \quad \cdots \quad w_d]^T$ – вектор весов.

Как и в случае линейной регрессии, для унификации к вектору данных можно добавить искусственную переменную со значением 1, а смещение b добавить к вектору весов, т. е.

$$\mathbf{x} = [1 \quad x_1 \quad x_2 \quad \cdots \quad x_d]^T, \qquad \mathbf{w} = [b \quad w_1 \quad w_2 \quad \cdots \quad w_d]^T.$$

В этом случае выражение (3.19) упростится до вида

$$v = \mathbf{w}^T \mathbf{x}.$$

Функция активации $g(v)$ может быть как линейной, так и нелинейной. Популярной нелинейной функцией является логистический сигмоид:

$$g(v) = \frac{1}{1 + e^{-v}}.$$

Логистический сигмоид имеет два полезных свойства: 1) непрерывность, что означает возможность вычислить производную, а значит, находить от него градиент; 2) его производная выражается через саму функцию:

$$\frac{dg}{dv} = \frac{e^{-v}}{(1 + e^{-v})^2} = \underbrace{\frac{1}{(1 + e^{-v})}}_{g(v)} \underbrace{\frac{e^{-v}}{(1 + e^{-v})}}_{1-g(v)} = g(v)\big(1 - g(v)\big).$$

На рис. 3.8 показан график производной логистического сигмоида.

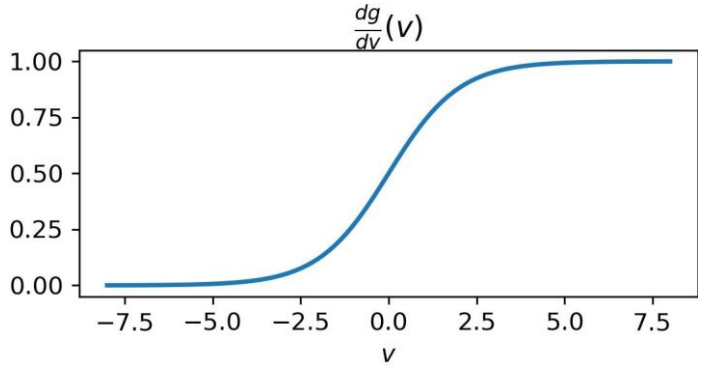


Рис. 3.8. Производная логистического сигмоида

Описанная модель нейрона (3.17) полностью совпадает с описанием логистической регрессии. Обучение классификатора на базе нейронной сети (3.17) полностью совпадает с обучением логистической регрессии.

3.1.7. Методы градиентного спуска

Рассмотрим более подробно вопрос использования градиентного спуска (3.16) для обучения как логистической регрессии, так и для нейронных сетей.

Согласно приведенному выражению обновление параметров модели $\mathbf{w}$ в момент на итерации t выполняется по правилу

$$\mathbf{w}_t = \mathbf{w}_{t-1} - \eta \nabla_{\mathbf{w}} E, \tag{3.20}$$

где $\nabla_{\mathbf{w}} E$ – градиент функции ошибки по параметрам $\mathbf{w}$, а η – скорость обучения.

Скорость обучения – весьма важный параметр, и его правильный выбор далеко не такая простая задача, как может показаться.

Трудность прямого использования выражения (3.20) заключается в том, что прежде, чем выполнить обновление параметров модели, необходимо рассчитать функцию ошибки $E(\mathbf{w})$ для всего набора данных, что достаточно затруднительно, поскольку используемые в реальных задачах наборы данных могут быть очень большими. Решить эту проблему можно воспользовавшись ***стохастическим градиентным спуском*** (СГС).

Идея данного метода проста: чтобы получить оценку вектора градиента, можно из всего набора данных выбрать случайным образом m примеров (этот набор называют мини-пакетом или батчем – от англ. *batch*), далее функция ошибки и ее градиент рассчитываются на основании мини-пакета. Обозначим через

$$f(\mathbf{x}_i; \mathbf{w}) = \hat{y}_i$$

модель (например, нейронную сеть), получающую на вход пример $\mathbf{x}_i$, и зависящую от параметров $\mathbf{w}$, и выдающую на выход оценку $\hat{y}_i$ (в нашем случае это метка класса). В этом случае можно ввести функцию потерь

$$L(f(\mathbf{x}_i; \mathbf{w}), y_i),$$

которая отражает расхождение между правильным откликом y_i и выдаваемым моделью $\hat{y}_i$.

Согласно (3.13) в качестве функции потерь можно использовать

$$L(f(\mathbf{x}_i; \mathbf{w}), y_i) = -\big(y_i \ln f(\mathbf{x}_i; \mathbf{w}) + (1 - y_i) \ln\big(1 - f(\mathbf{x}_i; \mathbf{w})\big)\big).$$

Используя введенные обозначения, метод стохастического градиентного спуска можно сформулировать следующим образом. На очередном шаге t выбрать из обучающего набора мини-пакет из m примеров $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\}$ и соответствующие им метки. Далее вычислить оценку градиента

$$\mathbf{g}_t = \frac{1}{m} \nabla_{\mathbf{w}} \sum_{i=1}^{m} L(f(\mathbf{x}_i; \mathbf{w}), y_i).$$

И выполнить обновление параметров модели

$$\mathbf{w}_t = \mathbf{w}_{t-1} - \eta \mathbf{g}_t.$$

Еще одним важным методом, позволяющим ускорить процесс обучения, является алгоритм Adam (от англ. *Adaptive Moment Estimation*). Данный алгоритм оптимизации сочетает в себе идеи из алгоритмов градиентного спуска и адаптивного шага. Согласно методу Adam необходимо вычислить сглаженную (усредненную) версию градиента

$$\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1) \mathbf{g}_t,$$

где $\mathbf{g}_t$ – градиент функции потерь, вычисленный на шаге t; $\mathbf{m}_t$ – сглаженный градиент на шаге t; β_1 – параметр, отвечающий за «скорость» усреднения.

Аналогичным образом вычисляется сглаженная версия среднеквадратичного градиента функции потерь:

$$\mathbf{v}_t = \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2) \mathbf{g}_t^2,$$

где $\mathbf{v}_t$ – среднеквадратичное значение градиента функции потерь на шаге t; β_2 – параметр, отвечающий за «скорость» усреднения значений $\mathbf{g}_t^2$.

Обновление параметров нейронной сети $\mathbf{w}_t$ выполняется по следующему правилу:

$$\mathbf{w}_t = \mathbf{w}_{t-1} - \frac{\eta}{\sqrt{\mathbf{v}_t + \epsilon}} \mathbf{m}_t,$$

где ϵ – малая константа, вводимая для численной стабильности.

В оригинальной статье, где был представлен метод Adam рекомендуется брать следующие значения параметров: $\beta_1 = 0{,}9$, $\beta_2 = 0{,}999$, $\epsilon = 10^{-8}$.

3.2. Основные метрики бинарной классификации

В результате бинарной классификации могут быть получены четыре различных типа результатов: 1) истинно положительные результат получается в случае, когда классификатор правильно определил метку объекта положительного класса; 2) истинно отрицательный результат (классификатор правильно определил метку объекта отрицательного класса); 3) ложноположительный результат (классификатор определил метку объекта отрицательного класса как положительную); 4) ложноотрицательный результат (классификатор определил метку объекта положительного класса как отрицательную). Далее рассмотрим некоторые метрики качества бинарной классификации, которые применяются на практике.

Правильность (англ. *accuracy*) – доля правильно классифицированных тестовых примеров.

$$Acc = \frac{TP + TN}{TP + FP + TN + FN},$$

где TP – число истинно положительных результатов классификации; TN – число истинно отрицательных результатов классификации; FP – число ложноположительных результатов классификации; FN – число ложноотрицательных результатов классификации.

Правильность характеризует частоту правильных решений, выносимых классификатором. Однако данный показатель не дает полного представлении о качестве работы классификатора, особенно если в исходном наборе данных классы не сбалансированы.

Частота ошибок (англ. *error rate*) – доля неправильно классифицированных тестовых примеров:

$$ER = \frac{FP + FN}{TP + FP + TN + FN} = 1 - Acc.$$

Точность (англ. *precision*) – отношение истинно положительных прогнозов к общему количеству положительных прогнозов:

$$Prec = \frac{TP}{TP + FP}.$$

Полнота (англ. *recall*) – отношение истинно положительных прогнозов к общему количеству положительных примеров:

$$Recall = \frac{TP}{TP + FN}.$$

Перечисленные метрики являются базовыми и используются наиболее широко. Рассмотрим еще несколько метрик, которые часто используются в контексте медицинских исследований, где классификатор рассматривается как метод постановки правильного диагноза, т. е. для выявления болезни (патологии).

Чувствительность – доля правильных диагнозов болезни к общему количеству больных:

$$Sens = \frac{TP}{TP + FN} = Recall.$$

Параметр чувствительности показывает способность классификатора детектировать патологию, если она есть.

Специфичность – доля правильно диагностированных здоровых к общему количеству здоровых:

$$Spec = \frac{TN}{TN + FP}.$$

Специфичность характеризует способность классификатора определять отсутствие патологии, когда она действительно отсутствует.

Средняя полнота

$$R_{ave} = \frac{1}{2}(Sens + Spec).$$

Средняя полнота показывает общую способность классификатора относить тестируемого к правильному классу.

3.3. Перекрестная проверка

Часто для оценки качества классификаторов использовался метод перекрестной проверки по *K*-блокам (англ. *K-fold cross-validation*). Согласно этому методу исходный набор данных перемешивается случайным образом и разбивается на *K* блоков. Далее выполняется обучение классификатора, причем один из блоков выступает как тестовый набор, а оставшиеся $K - 1$ в совокупности составляют обучающий набор. Эта процедура повторяется *K* раз так, чтобы каждый блок один раз выступил в роли тестового набора. Метки, присвоенные классификаторами, для тестовых наборов сохраняются и по ним выполняется оценка производительности классификатора.

Чаще всего в перекрестной проверке полагают $K = 10$. Эвристическое правило гласит, что в каждом блоке должно быть не менее 30 экземпляров. Таким образом, если в выборке данных имеется меньше 300 объектов, то *K* нужно скорректировать. Один из хороших вариантов в этом случае принять $K = N$, где *N* – размер обучающей выборки. Таким образом, мы будем всегда обучаться на всех данных, кроме одного тестового. В этом случае нужно будет повторить обучение *N* раз. Это называется перекрестной проверкой с исключением по одному (англ. *LOO-CV – leave-one-out cross-validation*).

3.4. Порядок выполнения работы

1) Загрузите набор данных согласно варианту (табл. 3.1).

Таблица 3.1. Набор данных

Номер варианта	Набор данных	Целевая переменная
1	ENT_data (https://www.kaggle.com/datasets/daniilkrasnoproshin/healthy-vs-laryngeal-disorder-classification/data)	Диагноз (healthy/ disphony)
2	ENT_data	Диагноз (healthy/ Paresis)
3	Breast Cancer Wisconsin Prognostic (https://archive.ics.uci.edu/dataset/16/breast+cancer+wisconsin+prognostic)	Outcome (R = recur, N = nonrecur)
4	iris	Сорта (Setosa/ Versicolor)
5	iris	Сорта (Versicolor / Virginica)
6	diabetes_data (https://www.kaggle.com/code/sakshisatre/diabetes-classification-using-logistic-regression/input)	Диагноз (Diagnosis)
7	Seeds (https://archive.ics.uci.edu/dataset/236/seeds)	Сорт семян (класс 1/ класс 2)
8	Seeds	Сорт семян (класс 1/ класс 3)

2) Самостоятельно напишите программу для обучения классификатора на основе логистической регрессии на языке Python. В процессе обучения используйте метод градиентного спуска согласно варианту (табл. 3.2).

Таблица 3.2. Варианты градиентного спуска

Номер варианта	1, 5	2, 6	3, 7	4, 8
Вариант градиентного спуска	Метод моментов	Adadelta	RMSprop	Adam

3) Примените запрограммированный классификатор к набору данных, рассчитайте полученную **правильность**, точность и полноту классификации.

4) Повторите задание 3, но используя схему перекрестной проверки по пяти блокам.

5) Оформите отчет в соответствии с СТП 01-2017.

ЛАБОРАТОРНАЯ РАБОТА № 4. ПОЭЛЕМЕНТНАЯ ОБРАБОТКА ИЗОБРАЖЕНИЙ

ЦЕЛЬ РАБОТЫ – изучение основных методов обработки изображений при помощи точечных операций в Python; освоение методов корректировки гистограммы изображения в Python.

4.1. Теоретические сведения

4.1.1. Преобразования яркости полутоновых изображений

Динамический диапазон изображения

Динамический диапазон изображения определяется как разница между наименьшим и наибольшим значениями пикселей в изображении. Мы можем определять функциональные преобразования (или отображения), которые позволят более эффективно использовать динамический диапазон. Эти преобразования в первую очередь применяются для улучшения контрастности (яркости) изображения.

В общем случае мы будем предполагать 8-битный (от 0 до 255) диапазон градации серого как для входных, так и для выходных изображений, но рассматриваемые методы могут быть обобщены на другие входные диапазоны и отдельные каналы из цветных изображений.

4.1.2. Преобразование яркости: общие сведения

Яркостными преобразованиями изображения $I_{input}(i,j)$ называются преобразования, описываемые формулой

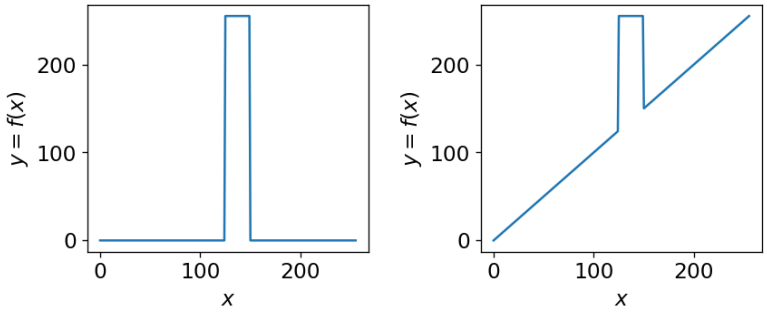
$$I_{output}(i,j) = f\left(I_{input}(i,j)\right).$$

Преобразования яркости изображений относятся к точечным операциям (или к поэлементной обработке), если значение яркости пиксела после преобразования зависит от яркости одной точки (пиксела) исходного изображения и не зависит от ее местоположения. Пусть $x = I_{input}(i,j)$ – это значение яркости пиксела в позиции (i,j) на исходном изображении, а $y = I_{output}(i,j)$ – это значение яркости пиксела в позиции (i,j) преобразованного изображения. Поэлементная обработка означает, что изменение яркости можно описать функцией $y = f(x)$ независимо от координат пиксела. Ниже показаны несколько примеров построения функции f для различных практических целей.

4.1.3. Яркостный срез

С помощью яркостного среза изображения можно выделить области изображения с яркостью из определенного интервала. При этом остальным областям можно присвоить черный цвет и получить бинарное изображение (рис. 4.1, *а*) или оставить неизменными (рис. 4.1, *б*). Перемещая выделенный

интервал по шкале яркости и изменяя его ширину, можно детально исследовать содержание изображения.



а *б*

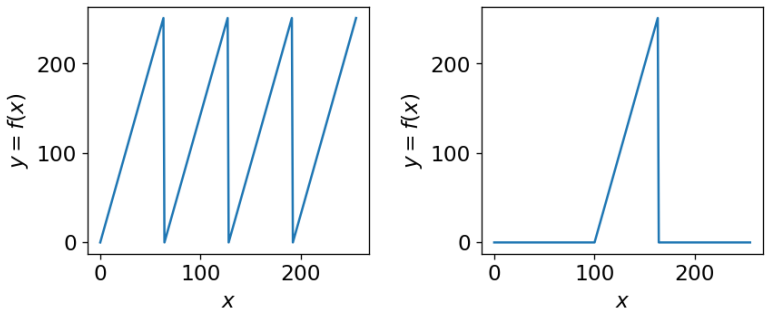
Рис. 4.1. Варианты функции яркостного среза:
а – выделение заданного диапазона; *б* – подчеркивание заданного диапазона с сохранением прочих уровней яркости

Аналитически функция среза для первого варианта (рис. 4.1, *а*) задается следующим образом:

$$y = f(x) = \begin{cases} 255, & \text{если } V_{\min} < x < V_{\max} \\ 0, & \text{иначе.} \end{cases} \quad (4.1)$$

4.1.4. Пилообразное контрастирование

Пилообразное контрастирование – увеличение контрастности в одном или нескольких диапазонах яркости (рис. 4.2). Оно также позволяет повысить детальность изображения в выбранном интервале яркости.



а *б*

Рис. 4.2. Варианты пилообразного контрастирования:
а – многодиапазонное; *б* – однодиапазонное

4.1.5. Бинаризация изображения

Простейшим методом препарирования изображений является бинаризация. Она преобразует полутоновое изображение в бинарное (черно-белое). Преобразование имеет единственный параметр – порог V_{thr} ,

относительно которого яркость пикселов меняется на черную или белую (рис. 4.3). Функция бинаризации с глобальным (т. е. единым для всех пикселов) порогом имеет вид

$$y = f(x) = \begin{cases} 255, & \text{если } x > V_{thr} \\ 0 & \text{иначе.} \end{cases} \quad (4.2)$$

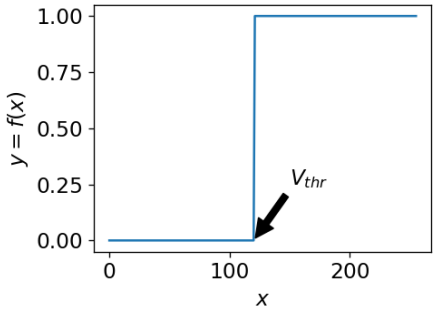
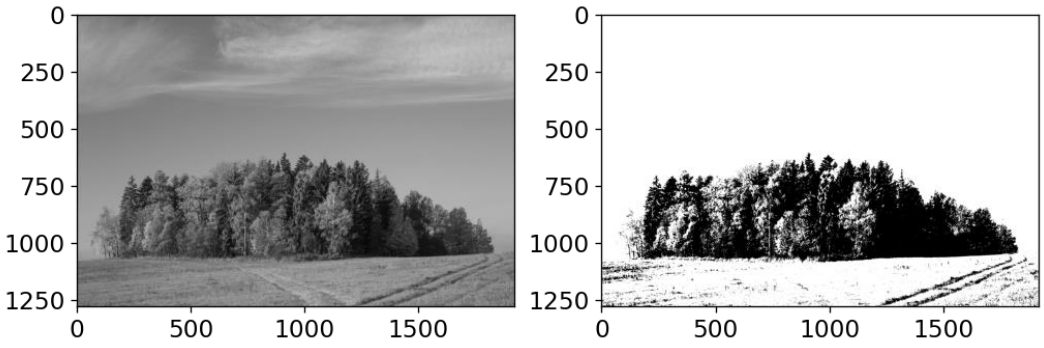


Рис. 4.3. Функция бинаризации с глобальным порогом

На рис. 4.4 показан пример бинаризации полутонового изображения с порогом $V_{thr} = 100$.



а *б*

Рис. 4.4. Пример бинаризации с глобальным порогом:
а – исходное изображение; *б* – бинаризированное изображение

Наиболее простой вариант выбора порога заключается в нахождении минимального $V_{\min}$ и максимального $V_{\max}$ значений яркости изображения и вычисления среднего между ними:

$$V_{thr} = \frac{V_{\max} - V_{\min}}{2}. \quad (4.3)$$

4.1.6. Растяжение контрастности

Чтобы выполнить растяжение контрастности, мы должны сначала знать верхний $V_{\max}$ и нижний $V_{\min}$ пределы значений пикселов, по которым изображение должно быть нормализовано. Обычно это верхний и нижний пределы используемого диапазона квантования пикселов (т. е. для 8-битного изображения $V_{\max} = 255$ и $V_{\min} = 0$). На начальном этапе операции

контрастного растяжения входное изображение сканируется для определения максимального ($I_{\max}$) и минимального значений пикселов ($I_{\min}$), присутствующих на изображении. На основе этих четырех значений ($V_{\max}$, $V_{\min}$, $I_{\max}$ и $I_{\min}$) диапазон пикселов изображения растягивается в соответствии со следующей формулой:

$$y = f(x) = (x - I_{\max})\left(\frac{V_{\max} - V_{\min}}{I_{\max} - I_{\min}}\right) + V_{\max}. \quad (4.4)$$

Обратите внимание, что в соответствии с выражением (4.4) значение $f(I_{\max}) = V_{\max}$, а $f(I_{\min}) = V_{\min}$.

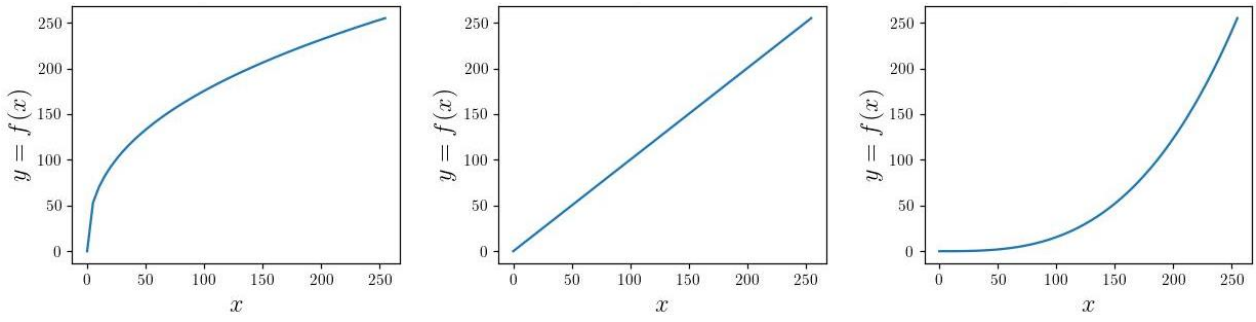
4.1.7. Гамма-преобразование (степенное преобразование)

Если изобразить график функции (4.4), он будет иметь линейный вид. Иногда для растяжения контрастности функция отображения $y = f(x)$ должна иметь нелинейный вид. Одним из вариантов построения такой функции является использование следующей степенной функции:

$$y = f(x) = \left(\frac{x - I_{\min}}{I_{\max} - I_{\min}}\right)^{\gamma} (V_{\max} - V_{\min}) + V_{\min}, \quad (4.5)$$

где γ – задает форму кривой отображения яркости.

Если $\gamma > 1$, то яркость отображения смещается вниз в сторону менее ярких значений. Если $\gamma < 1$, то яркость отображения смещается вверх в сторону более ярких значений. При $\gamma = 1$ функция отображения яркости имеет линейный вид. На рис. 4.5 показаны примеры графиков степенных отображений для различных параметров γ.



а *б* *в*

Рис. 4.5. Степенное преобразование для различных значений γ:
а – $\gamma = 0{,}4$; *б* – $\gamma = 1$; *в* – $\gamma = 3$

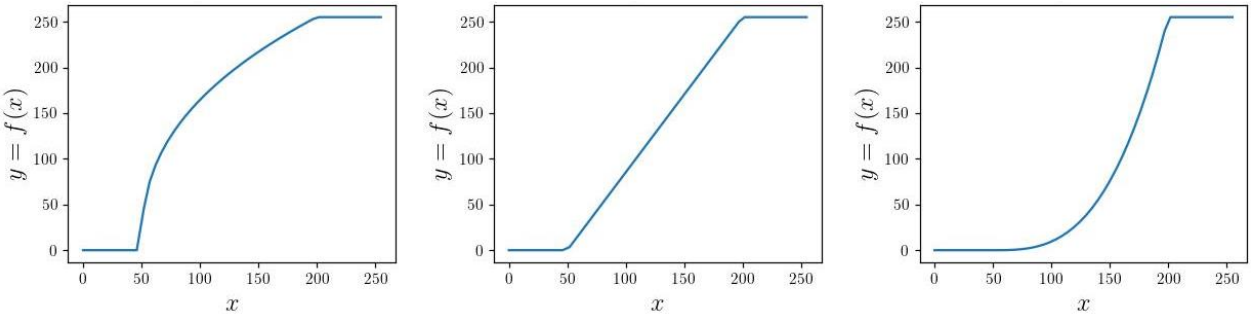
4.1.8. Степенное преобразование с ограничением входного диапазона

Иногда на практике требуется выполнить степенное преобразование в определенном диапазоне яркости $[I_L, I_H]$. В этом случае все пикселы изображения, которые имеют яркость $x < I_L$ отображаются в значение $y = V_{\min}$, а пикселы со значением $x > I_H$ отображаются в значение $y = V_{\max}$.

Математически степенное преобразование с ограничением входного диапазона можно записать следующим образом:

$$y = f(x) = \left(\frac{\min(\max(x, I_L), I_H) - I_L}{I_H - I_L}\right)^{\gamma} (V_{\max} - V_{\min}) + V_{\min}. \qquad (4.6)$$

На рис. 4.6 показаны примеры графиков степенных отображений с ограничением входного диапазона для различных параметров γ.



а *б* *в*

Рис. 4.6. Функции степенных преобразований для различных значений γ с ограничением входного диапазона: $I_L = 50$, $I_H = 200$:
а – $\gamma = 0{,}4$; *б* – $\gamma = 1$; *в* – $\gamma = 3$

4.2. Обработка гистограммы изображения

4.2.1. Получение гистограммы

Гистограмма изображения – это функция, показывающая, какое число раз на изображении появлялось каждое из допустимых значений пикселов. Например, если пиксел со значением 128 появлялся на изображении 5 раз, то можно сказать, что функция гистограммы для этого изображения в точке 128 будет иметь значение, равное пяти: $h(128) = 5$. Если нормализовать функцию гистограммы так, чтобы общая сумма всех ее значений в допустимом диапазоне аргумента была равна единице, мы можем рассматривать гистограмму изображения как ***дискретную функцию вероятности***, которая определяет вероятность появления данного значения пиксела в изображении.

Формально гистограммой цифрового изображения, число уровней яркости которого лежит в диапазоне $V = [V_{\min},\ V_{\max}]$, называют функцию вида

$$h(x_k) = n_k,$$

где $x_k \in V$ – это k-й уровень яркости, а n_k – число пикселов изображения, уровень яркости которых равен v_k.

Для изображений, которые хранятся в формате `uint8`, значение $V_{\max} = 255$, а значение $V_{\min} = 0$. Общее число уровней обозначают буквой $L = V_{\max} - -V_{\min} + 1$. Для формата данных `uint8` значение $L = 256$.

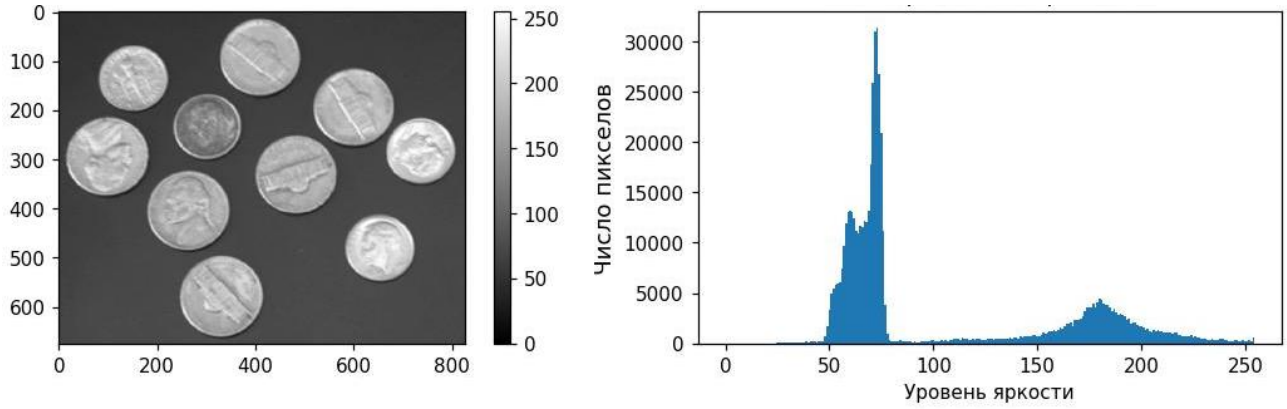
Как говорилось ранее, если нормировать гистограмму (т. е. поделить каждое значение $h(x_k)$ на число пикселов изображения), то получим величину

$$p(x_k) = \frac{h(x_k)}{M \times N} = \frac{n_k}{M \times N}, \tag{4.7}$$

где M и N – число строк и столбцов изображения.

Функция $p(x_k)$ – это вероятность появления уровня интенсивности x_k на данном изображении. Иными словами, $p(x_k)$ представляет собой дискретную функцию плотности вероятности. Визуальный осмотр гистограммы изображения может выявить основной контраст, присутствующий в изображении, и любые потенциальные различия в распределении цветов компонентов сцены переднего плана и фона.

На рис. 4.7 приведен пример изображения и его гистограммы.



а *б*

Рис. 4.7. Пример расчета гистограммы изображения:
а – исходное изображение; *б* – гистограмма изображения

В данном примере мы видим график гистограммы с двумя отличительными пиками: высокий пик в нижнем диапазоне значений соответствует распределению яркости (интенсивности) фона изображения, а более низкий пик в более высоком диапазоне значений (т. е. яркие пикселы) соответствует объектам переднего плана (монетам).

4.2.2. Эквализация гистограммы

Под эквализацией гистограммы изображения понимают выполнения такого преобразования, в результате которого гистограмма изображения становится равномерной (приблизительно можно считать, что каждый пиксел появляется на изображении с равной вероятностью).

Рассмотрим вопрос получения такого преобразования T, после применения которого гистограмма изображения становится равномерной. Для простоты вначале рассмотрим случай, в котором интенсивность изображения I распределена с непрерывной функцией распределения $p_I(x)$. После этого мы расширим наше описание на случай дискретных переменных.

Очевидно, что функция $T(x)$ выполняет отображение

$$T: [0, 1] \to [0, 1].$$

В данном случае имеется в виду, что каждому пикселу x, значение которого лежит в диапазоне от 0 до 1, будет ставится пиксел со значением также в диапазоне от 0 до 1. Мы считаем, что яркость нормирована, 0 соответствует черному цвету, а 1 – белому.

Пусть $F_I(x)$ – функция распределения (англ. cdf – cumulative distribution function) уровней яркости на изображении I. Предположим, что мы применили преобразование $T(\cdot)$ к изображению I, в этом случае мы получим новое изображение D:

$$D = T(I).$$

Функция распределения полученного изображения D будет задаваться как

$$F_D(x) = P(D[i,j] < x) = P(T(I[i,j]) < x) = P\big(I[i,j] < T^{-1}(x)\big) = \\ = F_I\big(T^{-1}(x)\big).$$

Если T такое преобразование, что $T(I)$ имеет равномерное распределение, то это значит, что

$$P(T(I[i,j]) < x) = x, \tag{4.8}$$

и, следовательно,

$$F_I\big(T^{-1}(x)\big) = x. \tag{4.9}$$

Если $T = F_I$, то уравнение (4.9) будет удовлетворено. Таким образом, преобразование эквализирующее гистограмму, это просто функция распределения данного изображения:

$$F_I(x) = \int_0^x p_I(z)dz. \tag{4.10}$$

Уравнение (4.10) записано исходя из того, что яркость нормирована и лежит в диапазоне от 0 до 1.

Рассмотрим теперь дискретный случай. Число уровней яркости обозначим через L (как правило, $L = 256$). Дискретные значения яркости $x_i \in [0,\ L-1]$, где $i = 0,1,\dots,L-1$. Далее пусть $p_I(x_i)$ – это значения нормированной гистограммы исходного изображения.

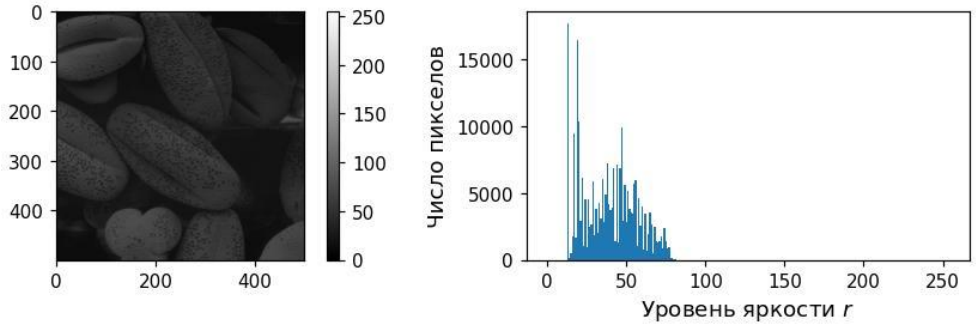
Для дискретного случая дискретная функция распределения равна

$$F_I(x_i) = \sum_{k=0}^{i} p_I(x_i), \qquad i = 0,1,\dots L. \tag{4.11}$$

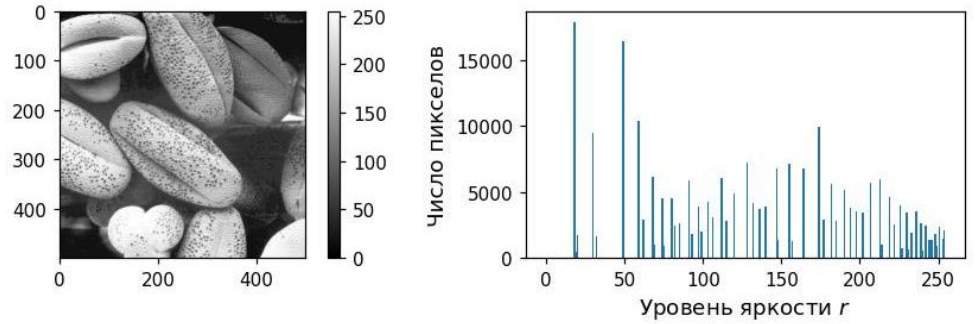
Чтобы из данной функции получить преобразование $T(x)$, которое выполняет эквализацию гистограммы, необходимо домножить (4.11) на L, чтобы максимальное значение соответствовало L. И также требуется применить оператор округления в меньшую сторону $\lfloor a \rfloor$, что соответствует операции `floor`. Таким образом, окончательное выражение для $T(x)$ будет иметь следующий вид:

$$T(x_i) = \lfloor (L-1) \cdot F_I(x_i) \rfloor = \left\lfloor (L-1) \cdot \sum_{k=0}^{i} p_I(x_i) \right\rfloor. \quad (4.12)$$

Результат применения эквализации гистограммы показан на рис. 4.8.



а



б

Рис. 4.8. Пример эквализации гистограммы:
а – исходное изображение и его гистограмма; *б* – обработанное изображение и его гистограмма

Рассмотрим пример эквализации простого изображения 4 × 4, который позволит показать вычислительные аспекты данного алгоритма. Предположим, что необходимо выполнить эквализацию гистограммы монохромного изображения, представленного на рис. 4.9. Общее значение уровней яркости равно $L = 8$.

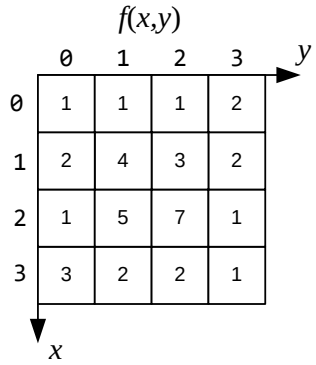


Рис. 4.9. Изображения для эквализации

Вначале построим гистограмму данного изображения $h_I(x_i)$. Результат показан на рис. 4.10.

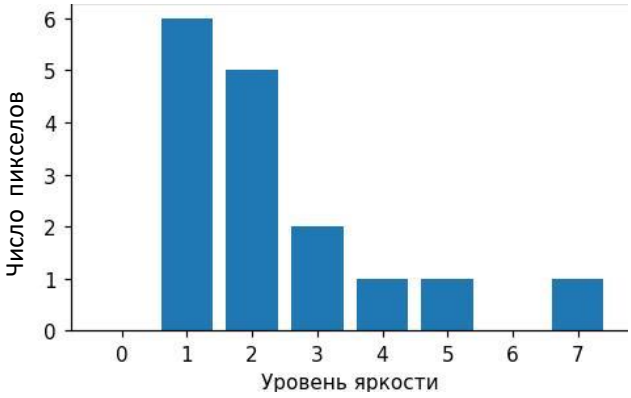


Рис. 4.10. Гистограмма изображения

Затем получим функцию распределения вероятности $p_I(x_i)$ по гистограмме $h_I(x_i)$ в соответствии с выражением (4.7):

$$p_I(x_i) = \frac{h_I(x_i)}{4 \times 4} = \frac{h_I(x_i)}{16}.$$

Результат показан на рис. 4.11.

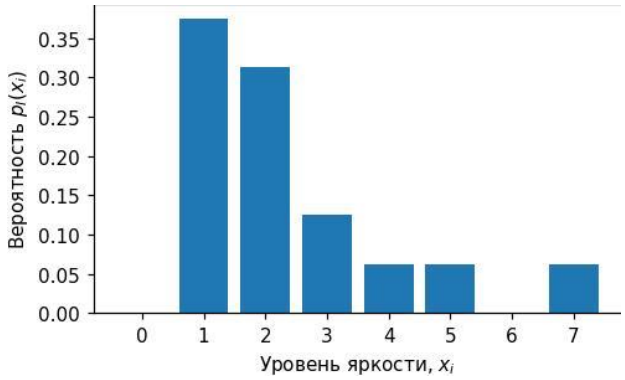


Рис. 4.11. Нормированная плотность вероятности

Численно значения $p_I(x_i)$ равны:

```
[0.0  0.375  0.3125  0.125  0.0625  0.0625 0.0  0.0625]
```

Построим функцию преобразования в соответствии с (4.12). Результат показан на рис. 4.12.

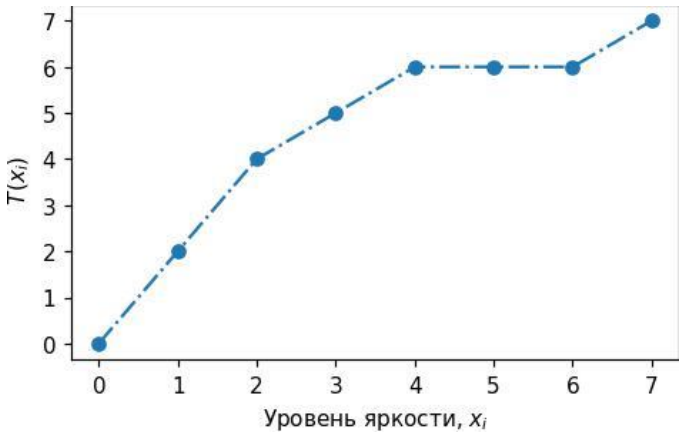


Рис. 4.12. Функция преобразования

Воспользуемся функцией $T(x_i)$ для преобразования изображения I (см. рис. 4.9):

$$D[i,j] = T(I[i,j]).$$

В результате получим изображение D:

```
[[2 2 2 4]
 [4 6 5 4]
 [2 6 7 2]
 [5 4 4 2]]
```

Построим теперь гистограмму изображения D (рис. 4.13).

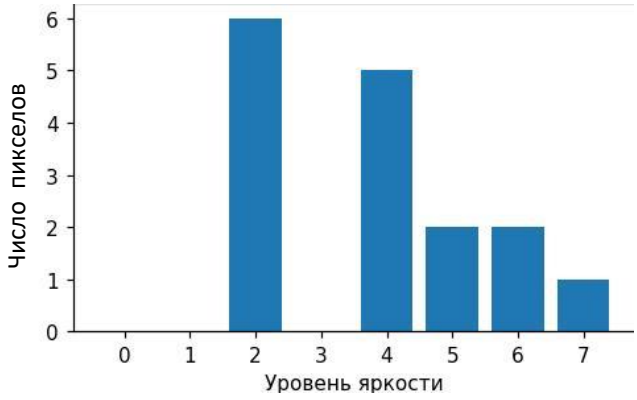


Рис. 4.13. Гистограмма изображения

4.2.3. Локальная эквализация гистограммы

Рассмотренный выше метод гистограммной обработки являлся глобальным, что означало построение функции преобразования на основе анализа распределения яркостей по всему изображению. Хотя такой глобальный подход и пригоден для улучшения в целом, существуют случаи, когда приходится улучшать детали на основе анализа малых областей изображения (рис. 4.14). Связано это с тем, что число пикселов в таких областях мало и не может оказывать замет-

ного влияния на глобальную гистограмму. Решение состоит в разработке функции преобразования, основанной на распределении яркостей по окрестности каждого элемента изображения.

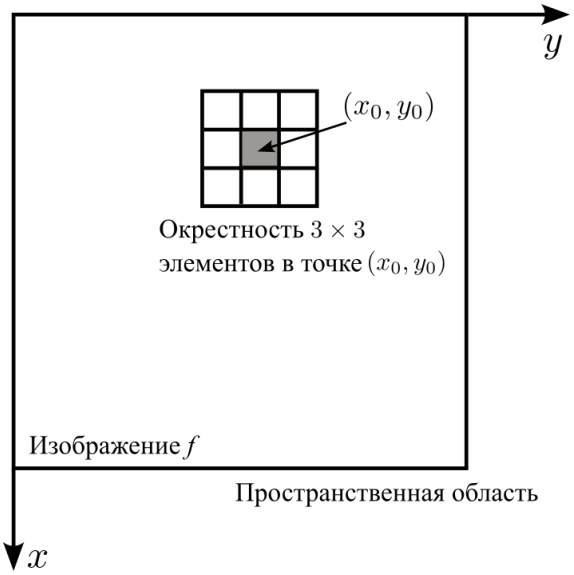


Рис. 4.14. Выделение области на изображении для метода локальной эквализации гистограммы

Локальная гистограммная обработка состоит из шагов:

1. Задается окрестность с центром в обрабатываемом элементе и центр этой окрестности передвигается от точки к точке.

2. Для каждого нового положения окрестности подсчитывается гистограмма по входящим в нее точкам и находится функция преобразования эквализации гистограммы (4.12).

3. Выполняется преобразование только центрального элемента окрестности.

4. Центр окрестности перемещается на соседний пиксел, и процедура повторяется.

4.2.4. Гистограммная подгонка

Процедура гистограммной подгонки описана в [6, с. 98–102] и является по сути развитием метода эквализации гистограммы.

Вернемся опять к рассмотрению непрерывных интенсивностей. В этом случае вместо нормированных гистограмм мы можем рассматривать плотность распределения вероятности исходного изображения $p_I(x)$ и обработанного изображения $p_D(x)$. В случае эквализации гистограммы мы добивались того, чтобы все яркости на изображении были равновероятными, что означает, что $p_D(x) = \text{const}$.

Теперь нам предстоит рассмотреть метод, в котором плотность распределения вероятностей описывается некоторой функцией

$$p_D(x) = p_t(x).$$

Мы используем букву t от английского слова *target*, т. е. *целевой*.

Вернемся к уравнению (4.8). Оно получено нами для случая, когда целевое распределение $p_t(x) = 1$ (при условии, что $0 \le x \le 1$). Для плотности распределения $p_t(x) = 1$ (кумулятивная) функция распределения равна

$$F_t(x) = \int_0^x p_t(x)dx = \int_0^x 1dx = x.$$

По этой причине и записывалось выражение (4.8).

В методе гистограммной подгонки предполагается, что $p_t(x)$ может иметь произвольную форму при соблюдении условия нормировки:

$$\int_0^1 p_t(x)dx = 1. \tag{4.13}$$

Пришло время рассмотреть вопрос, когда функция плотности распределения $p_t(x)$ имеет произвольный вид (естественно, при условии, что (4.13) выполняется).

Перепишем уравнение (4.8) в следующем виде:

$$P(T(I[i,j]) < x) = F_t(x) = \int_0^x p_t(x)dx.$$

Мы можем переписать $F_t(x)$, используя функцию распределения исходного изображения $F_I(x)$:

$$P(T(I[i,j]) < x) = F_I\left(T^{-1}(x)\right) = F_t(x). \tag{4.14}$$

Левую часть (4.14) можно переписать как $T^{-1}(x) = {F_I}^{-1}\left(F_t(x)\right)$.

Далее от обратной функции T^{-1} перейдем к прямой:

$$T(x) = F_t^{-1}\left(F_I(x)\right). \tag{4.15}$$

Рассмотрим задачу гистограммной подгонки монохромного изображения, показанного на рис. 4.15.

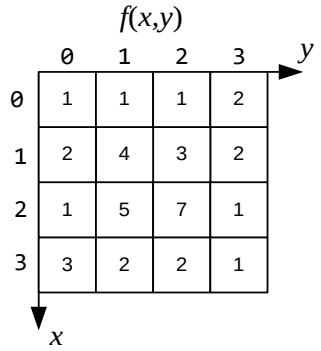


Рис. 4.15. Пример изображения (общее значение уровней яркости $L = 8$)

Целевая форма гистограммы приблизительно описывается гауссовой функцией (рис. 4.16)

$$\hat{p}_t(x) \sim e^{-\frac{(x-\mu)^2}{\sigma^2}}, \tag{4.16}$$

где $\mu = 0{,}625$, $\sigma = 0{,}125$.

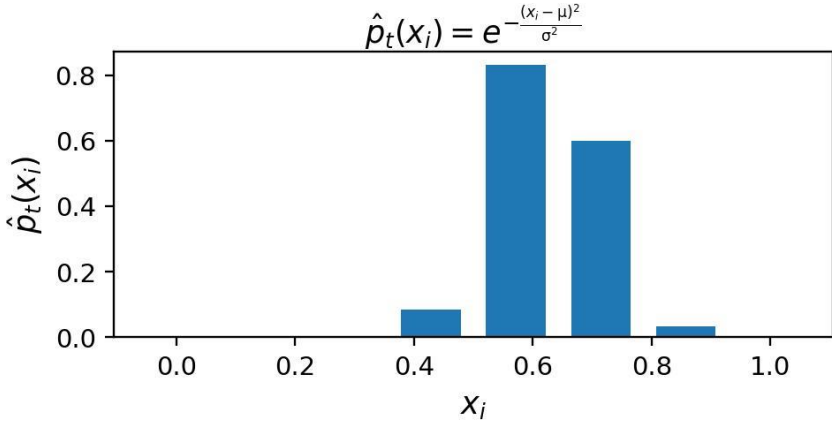


Рис. 4.16. Целевая форма плотности распределения вероятности

Обратимся к выражению (4.15), искомая нами функция преобразования $T(x)$ является композицией двух функций F_t^{-1} и F_I. Причем функция $F_I(x)$ уже была найдена в предыдущем примере.

x_i	0	0,142	0,285	0,428	0,571	0,714	0,857	1
$F_I(x_i)$	0	0,375	0,6875	0,8125	0,875	0,9375	0,9375	1
$T(x_i)$	0	2	4	5	6	6	6	7

Поэтому ближайшей целью является построение функции F_t и нахождение обратной функции F_t^{-1}.

Для дискретных нормированных значений яркости x_i рассчитаем целевую форму плотности распределения $\hat{p}_t(x_i)$:

x_i	0	0,142	0,285	0,428	0,571	0,714	0,857	1
$\hat{p}_t(x_i)$	≈0	≈0	≈0	0,08	0,83	0,6	0,03	≈0

Получим нормированную функцию распределения:

$$p_t(x_i) = \frac{\hat{p}_t(x_i)}{\sum_{i=0}^{L-1} \hat{p}_t(x_i)}. \tag{4.17}$$

На рис. 4.17 показана нормированная функция распределения $p_t(x_i)$. Численно значения $p_t(x_i)$ равны:

x_i	0	0,142	0,285	0,428	0,571	0,714	0,857	1
$p_t(x_i)$	≈0	≈0	≈0	0,05	0,53	0,38	0,02	≈0

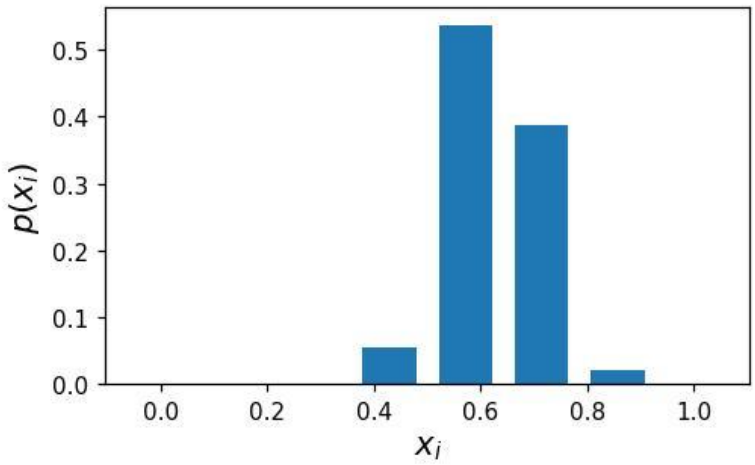


Рис. 4.17. Нормированная плотность вероятности

Построим функцию распределения вероятности $F_t(x_i)$:

x_i	0	0,142	0,285	0,428	0,571	0,714	0,857	1
$F_t(x_i)$	≈0	≈0	≈0	0,05	0,59	0,98	≈1	1

Теперь перейдем к денормированным переменным x_i и отмасштабируем $F_t(x_i)$:

$$T_t(x_i) = \lfloor (L-1)F_t(x_i) \rfloor.$$

В результате получим следующую функцию:

x_i	0	1	2	3	4	5	6	7
$T_t(x_i)$	0	0	0	0	4	6	6	7

Функция $T_t(x_i)$ графически показана на рис. 4.18.

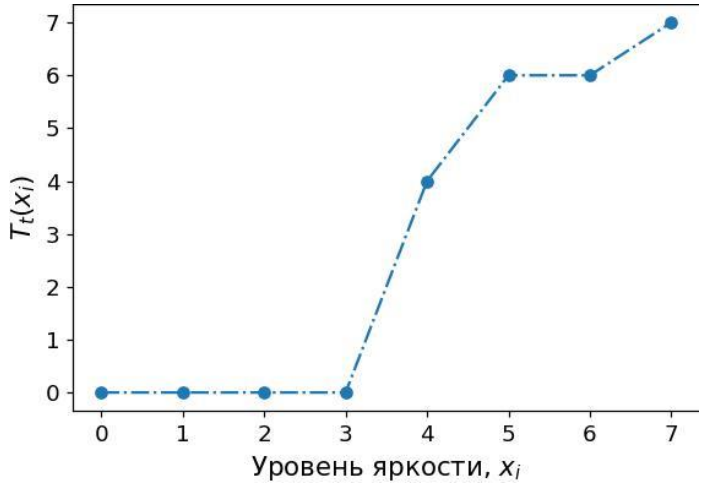


Рис. 4.18. Функция $T_t(x_i)$

Запишем теперь обратную функцию $F_t^{-1}(x_i)$, для денормированных переменных это будет означать T_t^{-1}:

x_i	0	1	2	3	4	5	6	7
$T_t^{-1}(x_i)$	0	3	3	3	4	4	5	7

И наконец, составим композицию функций $F(x_i)$ и $T_t^{-1}(x_i)$:

$$T(x_i) = T_t^{-1}\big(F_I(x_i)\big).$$

Функция $T(x_i)$ может быть записана в табличном виде:

x_i	0	1	2	3	4	5	6	7
$F_I(x_i)$	0	2	4	5	6	6	6	7
$T_t^{-1}\big(F(x_i)\big)$	0	3	4	4	5	5	5	7

Воспользуемся функцией $T(x_i)$ для преобразования изображения I:

$$D[i,j] = T(I[i,j]).$$

В результате получим изображение D:

```
[[3 3 3 5]
 [4 6 5 4]
 [2 6 7 2]
 [5 4 4 2]]
```

4.3. Порядок выполнения работы

1) Напишите функцию на языке Python для выполнения преобразования яркости в соответствии с вариантом (табл. 4.1). Постройте график функции преобразования для своего варианта.

Таблица 4.1. Варианты для задания 1

Номер варианта	Вид операции
1	Яркостный срез, (см. рис. 4.1, *а*)
2	Яркостный срез, (см. рис. 4.1, *б*)
3	Пилообразное контрастирование (см. рис. 4.2, *а* – четыре зубца)
4	Пилообразное контрастирование (см. рис. 4.2, *б* – один смещенный зубец)
5	Бинаризация изображения (формула (4.2))
6	Растяжение контрастности (формула (4.4))
7	Степенное преобразование яркости (формула (4.5))
8	Степенное преобразование яркости с ограничением входного диапазона (формула (4.6))

2) Загрузите изображение из папки `LR3\test-1` в соответствии с номером варианта и обработайте функцией из задания 1. Параметры преобразования подберите самостоятельно для достижения наилучшего визуального эффекта.

3) Разработайте функцию на языке Python для расчета гистограммы полутонового изображения (без использования сторонних библиотек, таких как `numpy`, `cv2` и проч.). Используя разработанную функцию, постройте гистограммы исходного и обработанного изображения из задания 2.

4) Разработайте функцию, реализующую метод обработки изображения на основе его гистограммы, в соответствии с вариантом (табл. 4.2).

Таблица 4.2. Варианты для задания 4

Номер варианта	Вид операции
1, 2	Эквализация гистограммы
3, 4	Локальная эквализация гистограммы (размер окрестности 75 × 75)
5, 6	Локальная эквализация гистограммы (размер окрестности 50 × 50)
7, 8	Гистограммная подгонка

5) Загрузите изображение из папки `LR3\test-2` в соответствии с номером варианта и обработайте функцией из задания 4. Параметры преобразования подберите самостоятельно для достижения наилучшего визуального эффекта.

6) Оформите отчет в соответствии с СТП 01-2017.

4.4. Дополнительные задания

Возьмите в качестве вида целевой гистограммы функцию

$$h(x_k) = (x_k - 128)^2, \tag{4.18}$$

где $x_k = 0, 1, \ldots, 255$.

Выполните подгонку гистограммы, чтобы результирующее изображение имело гистограмму вида $h(x_k)$. На результирующем графике приведите: 1) исходное изображение; 2) изображение после подгонки гистограммы; 3) гистограмму исходного изображения; 4) гистограмму изображения после подгонки.

ЛАБОРАТОРНАЯ РАБОТА № 5. ПРОСТРАНСТВЕННАЯ ФИЛЬТРАЦИЯ ИЗОБРАЖЕНИЙ

ЦЕЛЬ РАБОТЫ – изучение принципов пространственной обработки и фильтрации изображений; реализация методов пространственной обработки изображений на языке Python.

5.1. Теоретические сведения

5.1.1. Техника обработки изображений в пространственной области

Обработку изображения $s(i,j)$ в пространственной области можно описать выражением

$$d(i,j) = T\big(s(i,j)\big), \tag{5.1}$$

где T – некоторый оператор (преобразование) над s, который определен в некоторой окрестности точки (i,j).

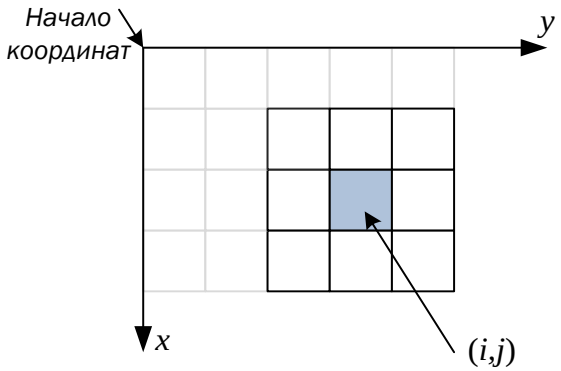


Рис. 5.1. Окрестность размером 3 × 3 вокруг точки (i,j)

Окрестность вокруг точки (i,j), как правило, выбирается в виде квадратной или прямоугольной области с центром в точке (i,j), как показано на рис. 5.1. Центр заданной шаблонной подобласти перемещается от пиксела к пикселу, начиная, как правило, из верхнего левого угла, и на своем пути он покрывает все изображение. Преобразование T применяется в каждой точке (i,j), давая в результате выходное (обработанное) значение $d(i,j)$ для данной точки.

5.1.2. Свертка

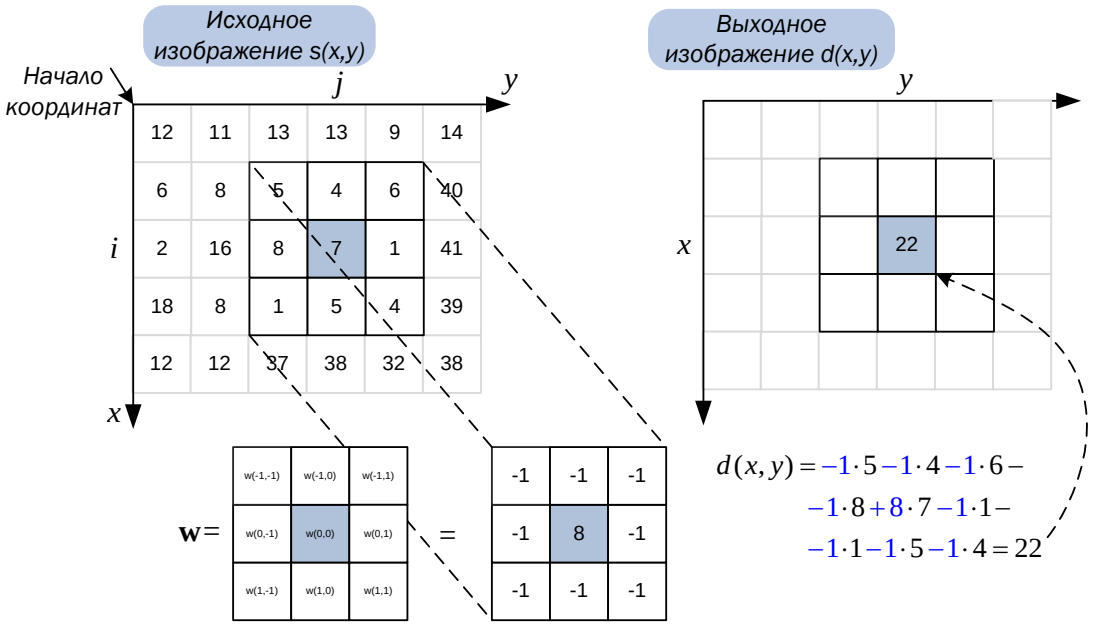
Исходя из описанной выше концепции, значение пиксела (i,j) на обработанном изображении является линейной комбинацией значений пикселов, находящихся в окрестности точки (i,j) на исходном изображении. Значения, на которые умножаются пикселы в окрестности точки (i,j), образуют матрицу w размером $n \times m$, которую называют ***фильтром*** или ***ядром*** (реже используют термины ***шаблон*** и ***маска***). Обработка изображения при помощи пространственного фильтра $\mathrm{w} = [w(i,j)]$ описывается следующим уравнением свертки:

$$d(x,y) = \mathbf{s} * \mathbf{w} = \sum_{i=-a}^{a} \sum_{j=-b}^{b} w(i,j) \cdot s(x+i, y+j), \qquad (5.2)$$

где $*$ – обозначение операции свертки; m – ширина ядра; n – высота ядра, $a = \frac{m-1}{2}$; $b = \frac{n-1}{2}$.

В выражении (5.2) значения индексов $i = 0,\ j = 0$ соответствуют центру фильтра, ширина и высота ядра всегда выбираются нечетными.

На рис. 5.2 поясняется процесс вычисления свертки изображения с ядром 3×3.



$$d(x,y) = w(-1,-1)s(x-1,y-1) + w(-1,0)s(x-1,y) + w(-1,1)s(x-1,y+1) +$$
$$+ w(0,-1)s(x,y-1) + w(0,0)s(x,y) + w(0,1)s(x,y+1) +$$
$$+ w(1,-1)s(x+1,y-1) + w(1,0)s(x+1,y) + w(1,1)s(x+1,y+1)$$

Рис. 5.2. Процесс вычисления свертки изображения с ядром 3×3

5.1.3. Сглаживающий фильтр

Из предыдущего раздела нам известно, что для формирования линейного пространственного фильтра размерами $m \times n$ требуется задать $m \cdot n$ коэффициентов маски $w(i,j)$. Выбор этих коэффициентов основан на том, какие действия ожидаются от фильтра.

Выход простейшего линейного сглаживающего (усредняющего) пространственного фильтра есть среднее значение элементов по окрестности, покрытой маской фильтра. Идея применения сглаживающих фильтров заключается в уменьшение «резких» переходов уровней яркости, поскольку случайный шум как раз характеризуется резкими скачками яркости.

Ниже приведен пример сглаживающего фильтра по окрестности 3×3:

$$\mathrm{w} = \frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}. \tag{5.3}$$

Данный вариант дает обычное среднее значение яркости по окрестности 3×3. Заметим, что коэффициенты указаны как единицы, место $1/9$. Такой вариант является более эффективным с вычислительной точки зрения. В общем случае маска размерами $m \times n$ будет иметь нормировочный коэффициент $1/mn$. Такой пространственный фильтр, все коэффициенты которого одинаковы, иногда называют ***однородным усредняющим фильтром***.

Рассмотрим еще один сглаживающий фильтр, называемый ***взвешенным средним***:

$$\mathrm{w} = \frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}. \tag{5.4}$$

В данном случае значения элементов умножаются на разные коэффициенты, что позволяет им присвоить как бы разные «важности» (веса) по сравнению с другими.

5.1.4. Фильтры, основанные на порядковых статистиках

Базовые понятия

Фильтры, основанные на порядковых статистиках, относятся к методам нелинейной обработки изображений. Процесс обработки заключается в том, что пикселы, располагающиеся в окрестности точки с координатами (i, j), рассматриваются как (одномерный) массив данных, на основе которого производится вычисление некоторой порядковой статистики. Наиболее распространенными являются медианный фильтр, min- и max-фильтры.

Медианная фильтрация

Вначале вспомним определение того, что именно в статистике называют медианой. ***Медиана*** набора чисел есть такое число ξ, что половина чисел из набора меньше или равны ξ, а другая половина – больше или равны ξ. Чтобы выполнить медианную фильтрацию для элемента изображения, необходимо:

1. Упорядочить по возрастанию значения пикселов внутри окрестности.

2. Найти значение медианы. Для окрестности 3×3 элементов медианой будет пятое значение по величине, для окрестности 5×5 – тринадцатое значение и т. д.

3. Присвоить полученное значение обрабатываемому элементу.

Указанная последовательность действий схематично представлена на рис. 5.3.

Рис. 5.3. Идея медианной фильтрации для области в окне 3 × 3

Ранговая фильтрация

Медианный фильтр является частным случаем класса фильтров, называемых ранговыми или порядковыми. Ранговый фильтр порядка r ($1 \leq r \leq N$, где N – число элементов в окрестности) выбирает из полученного ряда элемент с номером r и присваивает его значение как результат фильтрации пиксела исходного изображения. Если число N нечетное и $r = (N + 1)/2$, фильтр становится медианным. Если $r = 1$, фильтр выбирает минимальное значение яркости в окне и называется min-фильтром. Если $r = N$, фильтр выбирает максимальное значение яркости в окне и называется max-фильтром.

5.1.5. Фильтр повышения резкости изображения

Главная цель повышения резкости заключается в том, чтобы усилить перепады и другие разрывы (например, шумы) и не подчеркивать области с медленными изменениями уровней яркостей. Простейшим оператором повышения резкости, основанным на производных, является лапласиан (оператор Лапласа), который в случае функции двух переменных $f(x, y)$ определяется как

$$\nabla^2 f(x,y) = f(x+1,y) + f(x-1,y) +$$
$$+ f(x,y+1) + f(x,y-1) - 4f(x,y).$$

Данное выражение можно описать в терминах свертки (5.2). В этом случае ядро $w(i, j)$ будет иметь следующий вид:

$$\mathrm{w}_{\text{Lapl-1}} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

Диагональные элементы также могут быть включены в формулу дискретного лапласиана:

$$\mathrm{w}_{\text{Lapl-2}} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

Поскольку оператор Лапласа по сути является второй производной, его применение подчеркивает разрывы уровней яркости на изображении и подавляет области со слабыми уровнями яркости.

Метод использования лапласиана для улучшения изображения сводится к следующему:

$$d(x,y) = s(x,y) - \nabla^2 s(x,y) = \\ = s(x,y) - \mathrm{w}_{\mathrm{Lapl}} * s(x,y). \tag{5.5}$$

5.1.6. Выделение границ на изображении

Выделение границ на изображении выполняется при помощи фильтров, которые аппроксимируют операцию вычисления производной. Ядра таких фильтров устроены таким образом, чтобы возвращать большие значения в тех местах, где изображение имеет резкие переходы (разрывы). Таким образом, выполняется детектирование границ (англ. *edge detection*).

Оператор выделения границ на изображении обычно имеет следующее описание:

$$G = \sqrt{{G_x}^2 + {G_y}^2}, \\ \theta = \mathrm{arctg}\left(\frac{G_y}{G_x}\right). \tag{5.6}$$

где G – это оценка градиента изображения, который подчеркивает имеющиеся границы; θ – ориентация градиента изображения.

x- и y-компоненты градиента вычисляются путем свертки изображения с ядром, аппроксимирующим вычисление производной вдоль соответствующей оси:

$$G_x = \mathbf{S} * \mathrm{D}_x, \qquad G_y = \mathbf{S} * \mathbf{D}_y. \tag{5.7}$$

На практике чаще всего для дифференцирования используются ядра, приведенные в табл. 5.1.

Таблица 5.1. Фильтры для детектирования границ

Название оператора	$\mathbf{D}_x$	$\mathbf{D}_y$
Робертса	$\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$
Прюитта	$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$
Собеля	$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$

Перекрестный оператор Робертса быстро вычисляется (из-за минимального размера ядер), но он очень чувствителен к шуму. Детекторы Прюитта и Собеля не так чувствительны к шуму, но используют более сложные ядра.

Ядра Прюитта/Собеля, как правило, предпочтительнее подхода Робертса, поскольку градиент не сдвигается на половину пиксела в обоих направлениях. Ключевое различие между операторами Собеля и Прюитта заключается в том, что ядро Собеля реализует дифференцирование в одном направлении и (приблизительное) усреднение по Гауссу в другом. Преимущество этого заключается в том, что он сглаживает краевую область, снижая вероятность того, что зашумленные или изолированные пикселы будут доминировать в отклике фильтра.

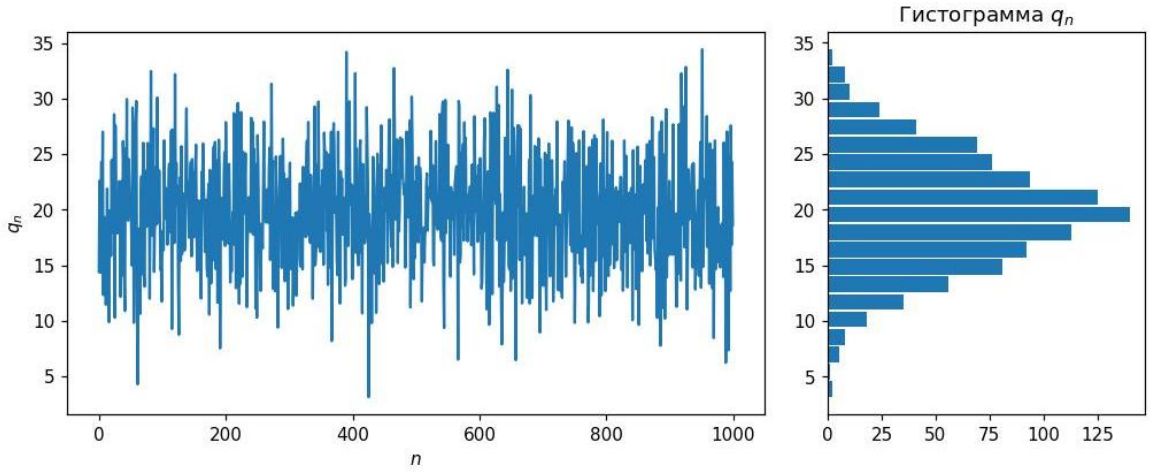
5.1.7. Модели шума

Гауссов шум

Наиболее часто возникающий тип шума – аддитивный гауссов шум. Он широко используется для моделирования тепловых эффектов и многого другого. Функция плотности вероятности гауссова шума q со средним значением μ и дисперсией σ^2 описывается выражением

$$p_q(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}. \quad (5.8)$$

На рис. 5.4 показан пример сгенерированного гауссова шума, который имеет среднее значение $\mu = 20$, а СКО $\sigma = 5$.



а *б*

Рис. 5.4. Гауссов шум: $\mu = 20$, $\sigma = 5$:
а – шумовой сигнал; *б* – гистограмма шумового сигнала

На рис. 5.4, *а* приведена гистограмма сгенерированного шума, которая по форме напоминает функцию плотности вероятности (5.8).

Гауссов шум используется в качестве естественного приближения в тех случаях, когда сенсорные детекторы изображения работают на пороге чувствительности.

В Python для генерации приведенного выше примера гауссова шума использовался следующий код:

```
[1]: mean = 20
[2]: sigma = 5
[3]: q = np.random.normal(mean, sigma, 1000)
```

Для добавления шума к изображению необходимо вначале сгенерировать число отсчетов шума, равное числу пикселов на изображении, а затем применить к полученному массиву метод `reshape`, после чего сложить изображение с шумом.

Шум «соль и перец» (импульсный шум)

Шум типа «соль и перец» возникает в устройствах с ошибочной коммутацией. Этот шум характеризуется тем, что на изображении возникает множество случайных белых («соль») и черных («перец») пикселов.

Функция плотности распределения вероятностей импульсного шума задается выражением

$$p(x) = \begin{cases} P_p \text{ при } x = 0, \\ P_s \text{ при } x = 255, \\ 0 \text{ в остальных случаях,} \end{cases} \tag{5.9}$$

где P_s – вероятность появления светлых пикселов («соли»); P_p – вероятность появления темных пикселов («перца»).

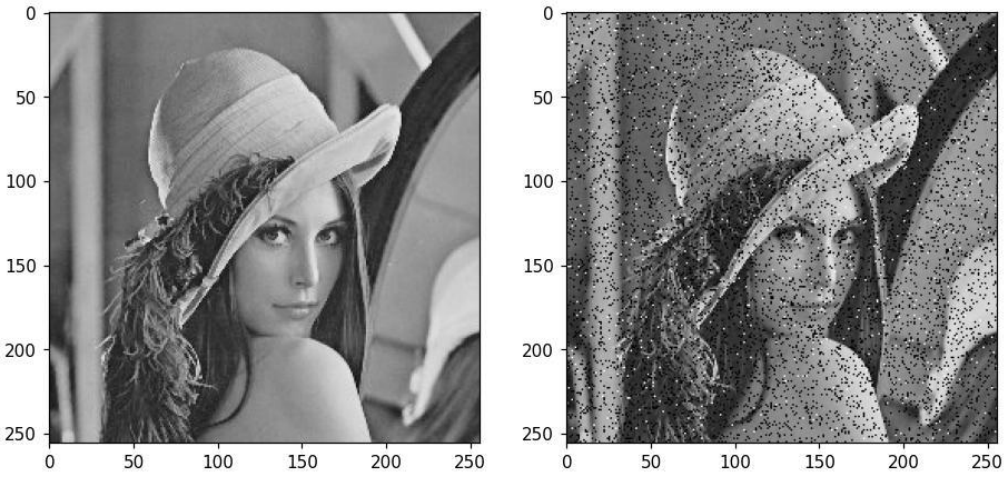
Пиксел с яркостью $x = 255$ выглядит как светлая точка на изображении. Пиксел с яркостью $x = 0$ выглядит, наоборот, как темная точка.

Следующая функция Python иллюстрирует, как можно добавить импульсный шум на изображение:

```
# Adding salt & pepper noise to an image
def salt_pepper(img, prob_s, prob_p):
    # prob_s – “salt” probability
    # prob_p – “pepper” probability

    output = np.copy(img).reshape((img.size))

    # Apply salt noise on each pixel individually
    num_salt = np.ceil(prob_s * img.size)
    coords = np.random.randint(0, img.size - 1, int(num_salt))
    output[coords] = 255
```

```
    # Apply pepper noise on each pixel individually
    num_pepper = np.ceil(prob_p * img.size)
    coords = np.random.randint(0, img.size - 1, int(num_pepper))
    output[coords] = 0

    return output.reshape(img.shape)
```

На рис. 5.5 показан пример добавления шума типа «соль и перец» на изображение. В данном случае вероятность появления ярких точек («соли») в 10 раз меньше, чем темных («перца»), поэтому визуально на изображении больше черных точек.



а *б*

Рис. 5.5. Добавление к изображению шума «соль и перец» ($P_s = 0{,}01$, $P_p = 0{,}1$):
а – исходное изображение; *б* – зашумленное изображение

5.2. Порядок выполнения работы

1) Возьмите монохромное изображение размером не более 400 × 400 пикселов и добавьте к нему гауссов шум с параметрами, приведенными в табл. 5.2.

Таблица 5.2. Варианты для задания 1

Номер варианта	1	2	3	4	5	6	7	8
μ	20	40	60	70	75	82	91	80
σ	15	25	32	41	22	36	40	45

На экране отобразить: 1) исходное изображение; 2) сгенерированный шум и 3) изображение с добавленным шумом.

2) Напишите функцию на языке Python для сглаживающей фильтрации в соответствии с вариантом (табл. 5.3).

Таблица 5.3. Варианты для задания 2

Номер варианта	Размер окрестности	Тип фильтра
1	3×3	Однородный усредняющий
2	5×5	Взвешенное среднее
3	7×3	Однородный усредняющий
4	5×7	Взвешенное среднее
5	7×7	Однородный усредняющий
6	9×5	Взвешенное среднее
7	11×9	Однородный усредняющий
8	13×13	Взвешенное среднее

3) К изображению из задания 1 добавьте шум типа «соль и перец» с параметрами, приведенными в табл. 5.4.

Таблица 5.4. Варианты для задания 3

Номер варианта	P_s	P_p
1	0,05	0,05
2	0,01	0,09
3	0,12	0,00
4	0,03	0,08
5	0,00	0,15
6	0,10	0,08
7	0,08	0,10
8	0,02	0,12

4) Напишите функцию на языке Python для ранговой фильтрации в соответствии с вариантом (табл. 5.5).

Таблица 5.5. Варианты для задания 3

Номер варианта	Размер окрестности	Порядок фильтра r
1	5×5	10
2	7×7	25
3	9×9	41
4	11×11	121
5	17×17	501
6	19×19	201
7	15×15	111
8	13×13	89

Примените разработанную функцию фильтрации к зашумленному изображению из задания 3. Отобразите на экране результат фильтрации.

5) Разработайте функцию, реализующую метод повышения резкости (5.5) на изображении, в соответствии с вариантом (табл. 5.6).

Таблица 5.6. Варианты для задания 5

Номер варианта	Тип лапласиана
1	$\mathbf{w}_{\text{Lapl-2}}$
2	$\mathbf{w}_{\text{Lapl-2}}$
3	$\mathbf{w}_{\text{Lapl-2}}$
4	$\mathrm{w}_{\text{Lapl-1}}$
5	$\mathbf{w}_{\text{Lapl-2}}$
6	$\mathbf{w}_{\text{Lapl-1}}$
7	$\mathbf{w}_{\text{Lapl-1}}$
8	$\mathbf{w}_{\text{Lapl-1}}$

Примените разработанную функцию к изображению. Выведите на экран исходное изображение, изображение после обработки его лапласианом и изображение с повышенной резкостью.

6) Разработайте функцию, реализующую метод выделения краев на изображении, в соответствии с вариантом (табл. 5.7).

Таблица 5.7. Варианты для задания по детектированию границ

Номер варианта	Тип оператора
1	Робертса
2	Прюитта
3	Собеля
4	Робертса
5	Прюитта
6	Собеля
7	Прюитта
8	Собеля

Примените разработанную функцию к изображению. Выведите на экран исходный градиент изображения G и ориентацию градиента θ.

7) Оформите отчет в соответствии с СТП 01-2017.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Флах, П. Машинное обучение. Наука и искусство построения алгоритмов, которые извлекают знания из данных / П. Флах. – М. : ДМК Пресс, 2015. – 400 с.

2 Келлехер, Дж. Основы машинного обучения для аналитического прогнозирования: алгоритмы, рабочие примеры и тематические исследования / Дж. Келлехер, Б. Мак-Нейми, А. Д'Арси – СПб. : Диалектика, 2019. – 656 с.

3 Николенко, С. Глубокое обучение / С. Николенко, А. Кадурин, Е. Архангельская. – СПб. : Питер, 2019. – 480 с.

4 Гудфеллоу, Я. Глубокое обучение / Я. Гудфеллоу, И. Бенджио, А. Курвилль. – 2-е изд. – М. : ДМК Пресс, 2018. – 652 с.

5 Стивенс, Э. PyTorch. Освещая глубокое обучение : справ. пособие / Э. Стивенс, Л. Антига, Т. Виман. – СПб. : Питер, 2022. – 576 с.

6 Гонсалес, Р. Цифровая обработка изображений в среде MATLAB / Р. Гонсалес, Р. Вудс, С. Эддинс. – М. : Техносфера, 2006. – 616 с.

7 Старовойтов, В. В. Цифровые изображения: от получения до обработки / В. В. Старовойтов, Ю. И. Голуб. – Минск : ОИПИ НАН Беларуси, 2014. – 202 с.

Св. план 2024, поз. 35

Учебное издание

Вашкевич Максим Иосифович
Петровский Николай Александрович

ВСТРАИВАЕМЫЕ СИСТЕМЫ ОБРАБОТКИ МЕДИАДАННЫХ. ЛАБОРАТОРНЫЙ ПРАКТИКУМ

ПОСОБИЕ

Редактор *Ю. В. Граховская*
Корректор *Е. Н. Батурчик*
Компьютерная правка, оригинал-макет *Е. Г. Бабичева*

Подписано в печать 04.06.2025. Формат 60×84 1/16. Бумага офсетная. Гарнитура «Таймс».
Отпечатано на ризографе. Усл. печ. л. 4,53. Уч.-изд. л. 4,5. Тираж 60 экз. Заказ 140.

Издатель и полиграфическое исполнение: учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники».
Свидетельство о государственной регистрации издателя, изготовителя,
распространителя печатных изданий №1/238 от 24.03.2014,
№2/113 от 07.04.2014, №3/615 от 07.04.2014.
Ул. П. Бровки, 6, 220013, г. Минск