УДК 62-712
doi: 10.21685/2587-7704-2024-9-2-5

Open Access | RESEARCH ARTICLE

# Применение линейной регрессии в обработке данных

***Всеволод Николаевич Зорин***
Пензенский государственный университет, Россия, г. Пенза, ул. Красная, 40
vsevolod.zorin@internet.ru

***Алексей Викторович Дубравин***
Пензенский государственный университет, Россия, г. Пенза, ул. Красная, 40
radamsa@yandex.ru

**Аннотация.** Результатом проведенной работы стало создание модели линейной регрессии, позволяющей определить наличие заболеваний сердца у пациента на основе трех признаков. Исследована библиотека Sklearn, предназначенная для обработки и анализа данных. Была проверена точность модели предсказания заболевания с помощью метрик Sklearn.

**Ключевые слова**: линейная регрессия, метод опорных векторов, Python, предсказание на основе данных

**Для цитирования**: Зорин В. Н., Дубравин А. В. Применение линейной регрессии в обработке данных // Инжиниринг и технологии. 2024. Т. 9 (2). С. 1–5. doi: 10.21685/2587-7704-2024-9-2-5

# Application of linear regression in data processing

***Vsevolod N. Zorin***
Penza State University, 40 Krasnaya Street, Penza, Russia
vsevolod.zorin@internet.ru

***Alexey V. Dubravin***
Penza State University, 40 Krasnaya Street, Penza, Russia
radamsa@yandex.ru

**Abstract.** The result of this work has been the development of a linear regression model that makes it possible to detect a heart disease in a patient by three signs. The Sklearn library designed for data processing and analysis has been investigated. The accuracy of the disease prediction model was also tested by using Sklearn metrics.

**Keywords**: linear regression, support vector machine, Python, data-based prediction

**For citation**: Zorin V.N., Dubravin A.V. Application of linear regression in data processing. *Inzhiniring i tekhnologii = Engineering and Technology*. 2024;9(2):1–5. (In Russ.). doi: 10.21685/2587-7704-2024-9-2-5

## Введение

Компьютеризация человечества является основной сферой в автоматизации разных видов работ, где нужна обработка большого количества данных. Автоматизирование на персональных компьютерах изменяет стандарты переработки данных, придавая слаженную работу промышленности и организаций на базе новейших технологий [1].

Грамотная обработка и анализ большого количества информации увеличивает эффективность принимаемых решений.

## Линейная регрессия как инструмент анализа данных

Для предсказания значения переменной, если известны значения другой или других переменных, применяется линейная регрессия.

Линейная регрессия представляет собой математическую формулу, которая позволяет строить прогнозы [2]. Построение регрессии означает подбор функции, которая будет описывать данные наиболее точно.

Существует два вида линейной регрессии:
– парная линейная регрессия;
– множественная линейная регрессия.

## Парная линейная регрессия

Парная линейная регрессия является методом анализа, который используется для определения зависимости между двумя переменными. Такую регрессию также называют однофакторной. Ее особенность состоит в том, что фактор, выраженный в виде независимой переменной *x*, может оказывать влияние на зависимую переменную *y*.

Данный вид регрессии описывает следующее уравнение:

$$y_i = \beta_0 + \beta_1 + \varepsilon_i \left(i = 1 \ldots N\right),$$

где $\beta_0$ – свободный член прямой парной линейной регрессии, $\beta_1$ – коэффициент направления прямой парной линейной регрессии, $\varepsilon_i$ – случайная погрешность, $N$ – число элементов.

## Множественная линейная регрессия

Множественной линейной регрессией называют прямую зависимость среднего значения переменной *y* от двух или более переменных $x_1, x_2, \ldots, x_m$. Переменную *y* называют зависимой, а переменные $x_1, x_2, \ldots, x_m$ – независимыми.

Она позволяет создавать модели, которые учитывают множество факторов, а также определять зависимость каждого из них.

При множественной линейной регрессии зависимость итоговой переменной одновременно от нескольких объясняющих переменных описывается следующим уравнением:

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \ldots + \beta_m x_{mi} + \varepsilon_i,$$

где $\beta_0, \beta_1, \beta_2, \ldots \beta_m$ — коэффициенты функции линейной регрессии генеральной совокупности, $\varepsilon_i$ – случайная ошибка.

Функция множественной линейной регрессии для выборки описывается с помощью уравнения:

$$y_i = b_0 + b_1 x_{1i} + \cdots + b_m x_{mi} + e_i,$$

где $b_0, b_1, \ldots, b_m$ — коэффициенты модели регрессии выборки, $e_i = \hat{y}_i - y_i$ — ошибка.

## Линейная регрессия с применением инструментов языка программирования Python

Проведем простейшую линейную регрессию с использованием таблицы формата csv, содержащей данные о сердечно-сосудистых заболеваниях.

NumPy – фундаментальный пакет для работы над различными видами массивов данных. Упрощает работу с математическими данными, а также обладает открытым исходным кодом, что может быть полезным в разработке.

Scikit-learn – программная библиотека для решения задач машинного обучения. Обрабатывает входные данные для выполнения над ними различных процедур: уменьшение размерности данных, реализация регрессии, проведение классификации и т.д.

Приступим к проведению регрессии. Для этого импортируем вышеперечисленные библиотеки:

```
import numpy as np
from sklearn.linear_model import LinearRegression
```

Далее укажем таблицу данных в формате csv с помощью библиотеки Pandas:

```
data = pd.read_csv('heart.csv')
```

После этого выберем столбцы таблицы для проведения регрессии. В качестве входных данных возьмем следующие показатели: возраст пациента, давление в покое и максимальное давление. В качестве выходных – показатель наличия диагноза у пациента (1 – у пациента имеется заболевание, 0 – заболевание отсутствует):

```
X = data[['age', 'trestbps', 'thalach']].values
y = data['target'].values
```

Далее необходимо провести разделение данных на обучающую и тестовую выборки. Оно необходимо для улучшения качества прогнозов. В sklearn предусмотрена функция train_test_split для выполнения данной операции:

```
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
```

После разделения проведем линейную регрессию с применением тестовых данных с помощью класса LinearRegression:

```
modelOfRegression = LinearRegression()
```

Для данного класса можно задать следующие параметры:
– fit_intercept;
– normalize;
– copy_X;
– n_jobs.

Fit_intercept – логический (true по умолчанию) параметр, решающий, вычислять отрезок $b_0$ (true) или рассматривать его как равный нулю (false) [3].

Normalize также является логическим параметром, который принимает значение true или false, определяющий необходимость нормализации входной переменной. По умолчанию используется значение false.

Copy_X – еще один логический параметр, который решает, будут ли входные переменные перезаписаны или скопированы. По умолчанию используется значение true.

N_jobs – параметр, фиксирующий число процессов в параллельных вычислениях. Может принимать как целое значение, так и значение none. Он показывает отсутствие каких-либо процессов. При использовании параметра – 1 используются все доступные процессы.

Теперь вызовем функцию fit, которая вычисляет оптимальные значения весов $b_0$ и $b_1$ с использованием переменных входа и выхода ($x$ и $y$):

```
modelOfRegression.fit(x_train, y_train)
```

Стоит отметить, что данная функция совмещает нашу модель.

После ее выполнения необходимо проверить, можем ли мы использовать полученные результаты для интерпретации. Для решения такой задачи существует метод опорных векторов.

Метод опорных векторов (англ. Support Vector Machines – SVM) – это набор контролируемых методов обучения, используемых для классификации, регрессии и обнаружения выбросов [4].

Он эффективен для работы с большим объемом данных. Он также применяется в ситуациях, когда количество измерений превышает количество данных.

В Python его можно реализовать с помощью модуля SVC библиотеки sklearn:

```
from sklearn.svm import SVC
```

Далее вызываем функцию принятия решений SVC, затем после преобразования модели вызовем оператор score:

```
modelOfRegression = SVC(kernel='rbf', C=1).fit(x_train, y_train)
score = modelOfRegression.score(x_test,y_test)
print(score)
```

В результате выполнения оператора score получен коэффициент детерминации. Он равен 0,72, что говорит о наличии возможности предсказать диагноз благодаря данным показателям.

Проверим полученную модель на тестовом наборе данных:

```
y_pred = modelOfRegression.predict(x_test)
```

В ходе выполнения команды отображается таблица результатов предсказаний на основе данных, представленная на рис. 1.

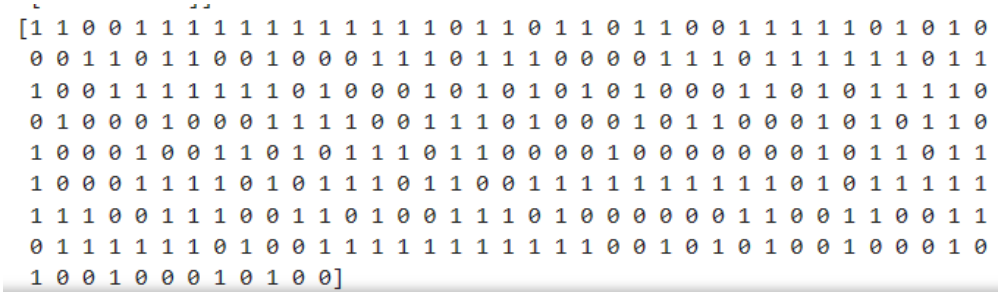


Рис. 1. Фрагмент таблицы результатов

## Проверка точности модели с помощью метрик sklearn

Теперь проверим точность данной модели с помощью метрик sklearn:

```
from sklearn import metrics
print(metrics.classification_report(y_test, y_pred))
```

После выполнения операции будет создан отчет с показателями оценки модели в виде таблицы. Её пример представлен на рис. 2.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.73 | 0.66 | 0.70 | 146 |
| 1 | 0.72 | 0.78 | 0.75 | 162 |
| accuracy | | | 0.73 | 308 |
| macro avg | 0.73 | 0.72 | 0.72 | 308 |
| weighted avg | 0.73 | 0.73 | 0.73 | 308 |

Рис. 2. Фрагмент таблицы точности

Показатели могут быть интерпретированы следующим образом:

– точность (Precision). Данный показатель определяется как отношение положительных срабатываний к сумме всех срабатываний. В нашем случае показатель равен 0,72, это означает, что из всех пациентов, которых предсказывала модель, с наличием заболевания 72 %;

– отзыв (Recall). Данный показатель определяется как отношение положительных результатов к сумме положительных и ложноотрицательных результатов. В нашем случае показатель равен 0,78, это означает, что из всех пациентов модель правильно предсказала правильный результат только для 78 % этих пациентов;

– оценка F1 (F1-score). Данный показатель обозначает взвешенное среднее гармоническое значение показателей точности и отзыва. Рассчитывается по следующей формуле:

$$2 * (Precision * Recall) / (Precision + Recall) =$$
$$2 * (0,72 * 0,78) / (0,72 + 0,78) = 0,75.$$

Поскольку результат близок к 1, это говорит нам о том, что модель хорошо предсказывает наличие болезни у пациента;

– поддержка (Support). Данный показатель обозначает, сколько пациентов принадлежало к каждому классу в тестовом наборе данных [5]. Среди пациентов в тестовом наборе данных у 146 отсутствовало заболевание, а у 162 – присутствовало.

## Заключение

Результатом проведенной работы стало создание модели линейной регрессии, позволяющей выявить наличие сердечного заболевания на основе трех признаков: уровень холестерина, максимальное давление пациента и показатель наследственного заболевания крови у пациента. В результате тестирования также было выявлено, что модель может определить наличие заболевания у пациента.

## Список литературы

1. Флоренция Нами. Применение баз данных в современном мире // Портал «Ида Тен». URL: https://idaten.ru/technology/primenenie-baz-dannih-v-sovremennom-mire (дата обращения: 11.03.2024).
2. Основные понятия линейной регрессии. Парная и множественная регрессия // Портал «BIRDYX». URL: https://birdyx.ru/blog/show/linear-regression (дата обращения: 09.03.2024).
3. Линейная регрессия на Python: объясняем на пальцах // Портал «BIRDYX». URL: https://proglib.io/p/linear-regression (дата обращения: 09.03.2024).
4. Метод опорных векторов SVM // Портал «Skit-Learn». URL: https://scikit-learn.ru/1-4-support-vector-machines/ (дата обращения: 09.03.2024).
5. Как интерпретировать отчет о классификации в sklearn (с примером) // Портал «Кодкамп». URL: https:// www.codecamp.ru/blog/sklearn-classification-report/ (дата обращения: 09.03.2024).

## References

1. Florence Nami. The use of databases in the modern world. *Portal «Ida Ten»*. (In Russ.). Available at: https://idaten.ru/technology/primenenie-baz-dannih-v-sovremennom-mire (accessed 11.03.2024).
2. The basic concepts of linear regression. Paired and multiple regression. *Portal «BIRDYX»*. (In Russ.). Available at: https://birdyx.ru/blog/show/linear-regression (accessed 09.03.2024).
3. Linear regression in Python: we explain it on our fingers. *Portal «BIRDYX»*. (In Russ.). Available at: https://proglib.io/p/linear-regression (accessed 09.03.2024).
4. SVM Support Vector Machine. *Portal «Skit-Learn»*. (In Russ.). Available at: https://scikit-learn.ru/1-4-support-vector-machines/ (accessed 09.03.2024).
5. How to interpret the classification report in sklearn (with an example). *Portal «Kodkamp»*. (In Russ.). Available at: https:// www.codecamp.ru/blog/sklearn-classification-report/ (accessed 09.03.2024).

**Поступила в редакцию / Received** 13.03.2024

**Принята к публикации / Accepted** 13.04.2024