УДК *519.6*

В. С. Попова [1], Н. С. Попова [1], Г. Д. Гефан [1]

[1] Иркутский государственный университет путей сообщения, г. Иркутск, Российская Федерация

ЛИНЕЙНЫЙ КЛАССИФИКАТОР В ЗАДАЧАХ АУТЕНТИФИКАЦИИ ЛИЧНОСТИ ПО РУКОПИСНЫМ СИМВОЛАМ

Аннотация. *В настоящей статье рассматривается метод простого линейного классификатора для решения задачи установления исполнителей рукописных символов. Актуальной проблемой является создание быстрой и надежной модели классификации, позволяющей оптимально разделить различные почерки: эталонную подпись от поддельной. В статье представлены результаты построенной модели, обеспечивающей анализ образцов почерка двух лиц. Моделирование проведено с помощью языка программирования Python.*

Ключевые слова: *аутентификация, линейное программирование, машинное обучение, методы классификации, различение рукописных символов, язык программирования Python.*

V. S. Popova [1], N. S. Popova [1], G. D. Gefan [1]

[1] Irkutsk State Transport University, Irkutsk, the Russian Federation

LINEAR CLASSIFIER IN THE PROBLEMS OF AUTHENTICATION OF THE PERSON FROM HAND-WRITTEN CHARACTERS

Abstract. *This article discusses the method of a simple linear classifier for solving the problem of identifying the performers of handwritten characters. An urgent problem is the creation of a fast and reliable classification model that allows you to optimally separate different handwriting: a reference signature from a fake one. The article presents the results of a constructed model that provides an analysis of handwriting samples of two persons. The simulation was carried out using the Python programming language.*

Keywords: *authentication, linear programming, machine learning, methods of classification, recognition of hand-written characters, Python programming language.*

Введение

Различение рукописных текстов играет важную роль во многих сферах. Часто мошенники подделывают подписи при заключении договоров займа или оформлении доверенности. Поэтому возникает потребность различения подлинных почерков от поддельных. Такую задачу решает почерковед. Любопытно узнать, как с подобной задачей справится алгоритм машинного обучения. Для решения подобных задач существует множество методов классификации. Например, искусственные нейронные сети, случайный лес (Random Forests), линейный дискриминантный анализ (Linear Discriminant Analysis, LDA). В этой работе речь пойдет о двух подходах к классификации: методе опорных векторов (SVM) и простом линейном классификаторе (ПЛК).

Теоретическая часть

Метод SVM направлен на создание линии или гиперплоскости, которая оптимально разделяет данные на два класса. Основная цель – сделать разделяющую полосу максимально широкой, одновременно минимизируя ошибки классификации. Векторы именуются опорными, если они располагаются на границах разделительной полосы [1-4]. Рассматриваемый метод опорных векторов, как и любой другой метод, имеет свои недостатки. Перечислим их:

1. Чувствителен к «выбросам» в данных [5];
2. Необходима настройка параметра C [6];
3. Скорость обучения зависит от количества обучающих векторов.

Метод простого линейного классификатора описанных ранее недостатков не имеет [7]. Его принцип работы заключается в следующем. Предположим, есть два класса обучающих векторов $\mathbf{x_i}$, каждому из которых присваивается метка z_i. Для одного класса +1 и для другого -1. Затем задается условие оптимизации (1):

$$\sum_{i=1}^{n}(\mathbf{x_i}\cdot\mathbf{w}-b)z_i \to \max, \tag{1}$$

где b – неизвестное расстояние от начала координат до границы, вычисляемое по формуле (2):

$$b=\frac{1}{2}\left[\overline{\mathbf{x}\cdot\mathbf{w}}\Big|_{(1)}+\overline{\mathbf{x}\cdot\mathbf{w}}\Big|_{(2)}\right]. \tag{2}$$

Слагаемые в скобках во второй формуле представляют собой усредненные скалярные произведения векторов двух классов на неизвестный нормальный вектор $\mathbf{w}$ разделительной гиперплоскости. Задается нормирующее ограничение $|\mathbf{w}|=1$. Решая задачу оптимизации, определяем оптимальные b^* и $\mathbf{w}^*$. Для того, чтобы классифицировать новый вектор $\mathbf{x}$, вычисляем величину $\mathbf{x}\cdot\mathbf{w}^*-b^*$. Если она меньше нуля, то вектор относится к классу $z_i=-1$. В противном случае он принадлежит классу с меткой $z_i=1$.

Практическая часть

Чтобы опробовать простой линейный классификатор, создадим набор случайных двумерных векторов. Их координаты будут иметь нормальное распределение (табл. 1).

Таблица 1

Параметры нормального распределения

Координаты векторов		Математическое ожидание	Среднеквадратическое отклонение
Класс 1	$x_{(1)}$	3	1
	$x_{(2)}$	5	
Класс 2	$x_{(1)}$	5	
	$x_{(2)}$	7	

Из соображений симметрии теоретические параметры границы в этом случае равны: $x_{(1)}+x_{(2)}=10$ или $\mathbf{w}^*=\left(\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right)$, $b^*=\frac{10}{\sqrt{2}}$.

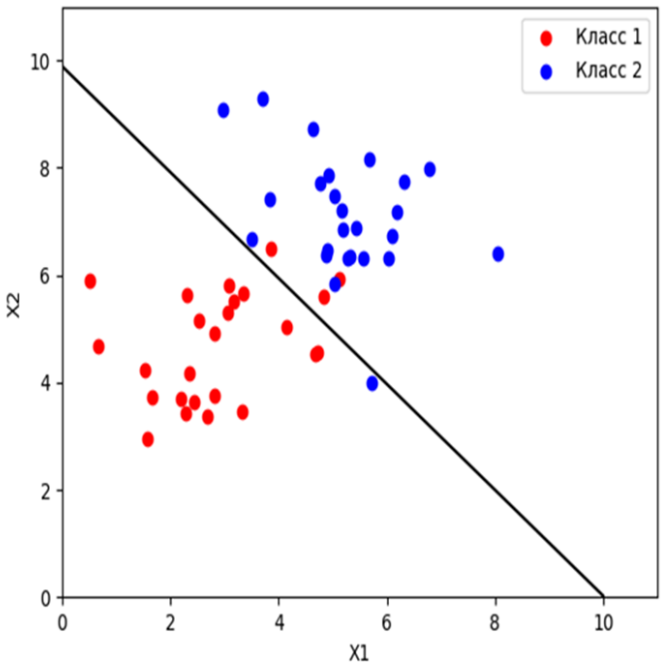
В нашей работе модель будет реализовываться на языке программирования Python. Параметры разделяющей границы определяются в процессе решения оптимизационной задачи. После этого анализируются «ошибки» классификации.

С помощью заранее установленной библиотеки NumPy (Numerical Python) [8], которая очень часто используется во всех областях науки и техники для работы с числовыми данными, сгенерируем двадцать пять обучающих векторов для каждого из двух классов.

Результаты классификации векторов относительно линии приведены в соответствии с рисунком 1. Таким образом, четыре вектора были опознаны неверно.

Теперь приступим к решению задачи, часто встречающейся в системах распознавания рукописных символов.

Пусть требуется установить, кому из двух лиц принадлежит каждая из имеющихся в нашем распоряжении N росписей.




```
Параметры границы:
w* = [0.70227967 0.71190117]
b* = 7.039779989494047
Классификация векторов:
Вектор 1: -1                    Вектор 27: 1
Вектор 2: -1                    Вектор 28: 1
Вектор 3: -1                    Вектор 29: 1
Вектор 4: -1                    Вектор 30: 1
Вектор 5: 1                     Вектор 31: 1
Вектор 6: -1                    Вектор 32: 1
Вектор 7: -1                    Вектор 33: 1
Вектор 8: 1                     Вектор 34: 1
Вектор 9: -1                    Вектор 35: 1
Вектор 10: 1                    Вектор 36: 1
Вектор 11: -1                   Вектор 37: 1
Вектор 12: -1                   Вектор 38: 1
Вектор 13: -1                   Вектор 39: 1
Вектор 14: -1                   Вектор 40: 1
Вектор 15: -1                   Вектор 41: -1
Вектор 16: -1                   Вектор 42: 1
Вектор 17: -1                   Вектор 43: 1
Вектор 18: -1                   Вектор 44: 1
Вектор 19: -1                   Вектор 45: 1
Вектор 20: -1                   Вектор 46: 1
Вектор 21: -1                   Вектор 47: 1
Вектор 22: -1                   Вектор 48: 1
Вектор 23: -1                   Вектор 49: 1
Вектор 24: -1                   Вектор 50: 1
Вектор 25: -1                   Кол-во ошибок: 4
Вектор 26: 1
```

Рис. 1. Пример работы классификатора

Для выполнения поставленной задачи были выбраны 2 индивидуума, чей почерк имеет значительное сходство. Оба лица написали шестьдесят раз букву «з» ($N = 120$), так как существуют внутриавторские вариации (некоторый разброс) в почерке. Каждая написанная буква была сфотографирована и масштабирована одинаковым образом.

Алгоритм действий начинается с создания обучающего набора данных. Этот процесс предполагает извлечение признаков и предварительную обработку для представления образцов почерка в виде числовых данных. В нашем случае в качестве признаков выступали расстояния между точками 1-2, 2-3 и 3-4 (рис. 2).

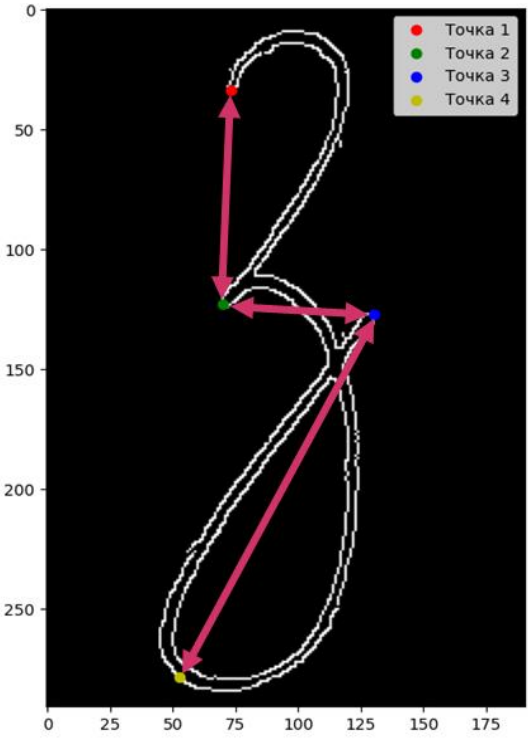
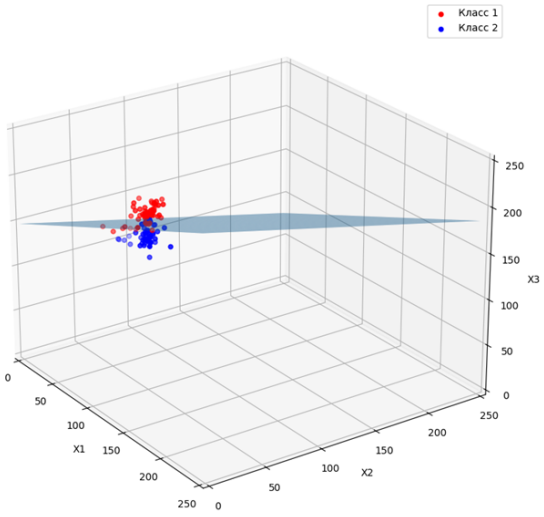


Рис. 2. Масштабирование и отображение характерных точек

В начертании букв «з» эти точки выбирались одинаково. Таким образом, имеем дело с трехмерными обучающими векторами.

Исследование с $N = 120$ письменными знаками показало впечатляющий результат. В сумме количество ошибок классификации для первого и второго классов составило двадцать штук (рис. 3). Правильное опознание векторов имело место в восьмидесяти трех процентах случаев.




```
w* = [ 0.37934994 -0.24578255 -0.89201152]
b* = -131.76423122442833
Классификация векторов:
Вектор 1: 1
Вектор 2: 1
Вектор 3: 1
Вектор 4: 1
Вектор 5: 1
Вектор 6: 1
Вектор 7: 1
Вектор 8: -1
Вектор 9: -1
Вектор 10: -1
Вектор 11: -1
Вектор 12: -1
Вектор 13: -1
Вектор 14: -1
Вектор 15: -1
Вектор 16: -1
Вектор 17: 1
Вектор 18: 1
Вектор 19: 1
Вектор 20: 1
Вектор 21: 1
Вектор 22: 1
Вектор 23: 1
Вектор 24: 1
Вектор 25: 1
Вектор 26: 1
Вектор 27: 1
Вектор 28: 1
Вектор 29: 1
Вектор 30: 1
Вектор 31: 1
Вектор 32: 1
Вектор 33: 1
Вектор 34: 1
Вектор 35: 1
Вектор 36: 1
Вектор 37: 1
Вектор 38: 1
Вектор 39: 1
Вектор 40: 1
Вектор 41: 1
Вектор 42: 1
Вектор 43: 1
Вектор 44: 1
Вектор 45: 1
Вектор 46: 1
Вектор 47: 1
Вектор 48: 1
Вектор 49: 1
Вектор 50: 1
Вектор 51: 1
Вектор 52: 1
Вектор 53: 1
Вектор 54: 1
Вектор 55: 1
Вектор 56: 1
Вектор 57: 1
Вектор 58: 1
Вектор 59: 1
Вектор 60: 1
Вектор 61: -1
Вектор 62: -1
Вектор 63: -1
Вектор 64: -1
Вектор 65: -1
Вектор 66: -1
Вектор 67: 1
Вектор 68: -1
Вектор 69: -1
Вектор 70: 1
Вектор 71: -1
Вектор 72: -1
Вектор 73: -1
Вектор 74: 1
Вектор 75: 1
Вектор 76: -1
Вектор 77: -1
Вектор 78: -1
Вектор 79: -1
Вектор 80: -1
Вектор 81: -1
Вектор 82: -1
Вектор 83: -1
Вектор 84: -1
Вектор 85: -1
Вектор 86: -1
Вектор 87: -1
Вектор 88: 1
Вектор 89: -1
Вектор 90: 1
Вектор 91: -1
Вектор 92: -1
Вектор 93: -1
Вектор 94: 1
Вектор 95: -1
Вектор 96: -1
Вектор 97: 1
Вектор 98: -1
Вектор 99: -1
Вектор 100: -1
Вектор 101: -1
Вектор 102: -1
Вектор 103: -1
Вектор 104: -1
Вектор 105: -1
Вектор 106: -1
Вектор 107: -1
Вектор 108: -1
Вектор 109: -1
Вектор 110: 1
Вектор 111: -1
Вектор 112: -1
Вектор 113: -1
Вектор 114: -1
Вектор 115: 1
Вектор 116: -1
Вектор 117: 1
Вектор 118: -1
Вектор 119: -1
Вектор 120: -1
Количество ошибок: 20
```

Рис. 3. Сконструированная модель

Чтобы удостовериться в оценке, используем тестовый набор из ста росписей (то есть семьдесят векторов от одного лица и тридцать – от другого). Прогоним его через сконструированную ранее модель. В итоге получили, что для каждого класса было обнаружено по восемь ошибочно классифицированных векторов. Таким образом, надежность распознавания достигла восьмидесяти четырех процентов.

Заключение

Основные итоги проведенного исследования можно выделить следующим образом. В рамках данной работы был разработан и протестирован алгоритм простого линейного классификатора. Он был применен при решении задачи различения похожих почерков двух индивидуумах. Классификация тестового набора дала результат даже лучше, чем классификация обучающего набора. Следовательно, созданная модель успешно прошла кросс-проверку.

В будущем данный алгоритм может применяться в качестве инструмента для аутентификации личности по рукописному тексту. Он может использоваться при проведении почерковедческой экспертизы и в системах безопасности государственных учреждений.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Griva I., Nash S.G., Sofer A. Linear and Nonlinear Optimization. SIAM. – 2009. – 764 p.
2. Cristianini N., Shawe-Taylor J. An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press. – 2000. – 204 p.
3. Shalev-Shwartz S., Ben-David S. Understanding Machine Learning: From Theory to Algorithms. Cambridge University Press. – 2014. – 410 p.
4. Schölkopf B., Smola A.J. Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. The MIT Press. – 2001. – 644 p.
5. Harrington P. Machine Learning in Action. Manning. – 2012. – 384 p.
6. Nefedov A. Support Vector Machines: A Simple Tutorial, 2016. – 35 p.

7. Гефан Г.Д., Иванов В.Б. Метод опорных векторов и альтернативный ему простой линейный классификатор // Информационные технологии и проблемы математического моделирования сложных систем. – Иркутск : ИрГУПС, 2012. – Вып. 10. – С. 84-94.

8. Hill C. Learning Scientific Programming with Python. Cambridge University Press. – 2020. – 204 p.

REFERENCES

1. Griva I., Nash S.G., Sofer A. Linear and Nonlinear Optimization. SIAM. – 2009. – 764 p.

2. Cristianini N., Shawe-Taylor J., An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press. – 2000. – 204 p.

3. Shalev-Shwartz S., Ben-David S. Understanding Machine Learning: From Theory to Algorithms. Cambridge University Press. – 2014. – 410 p.

4. Schölkopf B., Smola A.J. Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. The MIT Press. – 2001. – 644 p.

5. Harrington P. Machine Learning in Action. Manning. – 2012. – 384 p.

6. Nefedov A. Support Vector Machines: A Simple Tutorial, 2016. – 35 p.

7. Gefan G.D., Ivanov V.B. Metod opornykh vektorov i al'ternativnyy yemu prostoy lineynyy klassifikator [Support vector machine and its alternative simple linear classifier]. Informacionnye tekhnologii i problemy matematicheskogo modelirovaniya slozhnyh system [Information technologies and problems of mathematical modeling of complex systems]. Irkutsk, IrGUPS, 2012, no. 10, pp. 84-94.

8. Hill C. Learning Scientific Programming with Python. Cambridge University Press. – 2020. – 204 p.

Информация об авторах

Попова Виктория Сергеевна – студентка гр. БАС.5-22-1 факультета «Управление на транспорте и информационные технологии», Иркутский государственный университет путей сообщения, г. Иркутск, e-mail: popovavika2017@yandex.ru

Попова Надежда Сергеевна – студентка гр. БАС.5-22-1 факультета «Управление на транспорте и информационные технологии», Иркутский государственный университет путей сообщения, г. Иркутск, e-mail: Nm2nadia@yandex.ru

Гефан Григорий Давыдович – к. ф.-м. н., доцент кафедры «Математика», Иркутский государственный университет путей сообщения, г. Иркутск, e-mail: grigef@rambler.ru

Information about the authors

Victoria Sergeevna Popova – student of gr. BAS.5-22-1 of the Faculty of "Transport Management and Information Technology", Irkutsk State Transport University, Irkutsk, e-mail: popovavika2017@yandex.ru

Nadezhda Sergeevna Popova – student of gr. BAS.5-22-1 of the Faculty of "Transport Management and Information Technology", Irkutsk State Transport University, Irkutsk, e-mail: Nm2nadia@yandex.ru

Grigory Davydovich Gefan – Candidate of Physical and Mathematical Sciences, Associate Professor of the Department of Mathematics, Irkutsk State Transport University, Irkutsk, e-mail: grigef@rambler.ru