Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

|  |  |
| --- | --- |
| Институт | математики и компьютерных наук |
| Кафедра | компьютерной безопасности |

ОТЧЕТ

о Лабораторная работа №3

по теме «Метрические методы классификации»

Выполнил: Окунев Николай Александрович,

студент 2 курса

группы КМБ-с-о-23-1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2025 г.

Проверено с оценкой:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2025 г.

Ставрополь, 2025

1. **Цели и задачи**

Цель лабораторной работы: изучение принципов построения информационных систем с использованием метрических методов классификации.

Основные задачи: – изучение инструментария Python для реализации алгоритмов метрической классификации; – изучение методов оптимизации параметров метрической классификации; – освоение модификаций kNN-метода.

**2. Теоретическое обоснование**

Перед выполнением лабораторной работы необходимо ознакомиться с теорией построения метрических классификаторов, используя следующие источники: [1-5]. Особое внимание необходимо уделить репозитарию [5] с исходными кодами.

**3. Методика и порядок выполнения работы**

Перед выполнением индивидуального задания рекомендуется выполнить все пункты учебной задачи.

**3.1 Учебная задача**

В рамках данной задачи рассматривается построение классификатора с использованием метода ближайших соседей. В качестве набора данных используются данные об ирисах Фишера. 33 В рамках данной лабораторной работы рекомендуется использование библиотеки pandas. Pandas – это библиотека Python, предоставляющая широкие возможности для анализа данных. Данные, с которыми работают специалисты Data Science, часто хранятся в табличном формате (.csv, .tsv, .xlsx, …). С помощью библиотеки Pandas данные удобно загружать, обрабатывать и анализировать с помощью SQL-подобных запросов. Pandas предоставляет широкие возможности визуального анализа табличных данных в связке с библиотеками Matplotlib и Seaborn. Основными структурами данных в Pandas являются классы Series и DataFrame. Первый из них представляет собой одномерный индексированный массив данных некоторого фиксированного типа. Второй – это двухмерная структура данных, представляющая собой таблицу, каждый столбец которой содержит данные одного типа. Можно представлять её как словарь объектов типа Series. Структура DataFrame отлично подходит для представления реальных данных: строки соответствуют признаковым описаниям отдельных объектов, а столбцы соответствуют признакам. 1. На рис. 3.1 представлен код в Python Notebook для загрузки исходного набора данных.

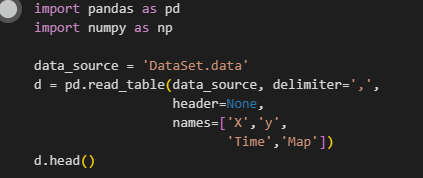


Рисунок 3.1 – Использование pandas для загрузки данных

Слудует обратить внимание, что первая строка набора данных интерпретировалась как шапка таблицы (название столбцов).

Исследователь также может дать символьные имена столбцам при загрузке (рис. 3.2).

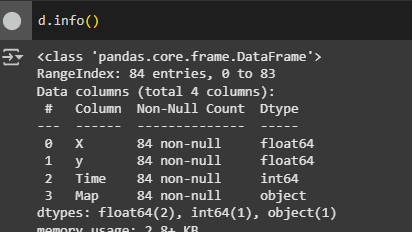


Рисунок 3.2 – Добавление шапки DataFrame с символьными именами столбцов 2.

После загрузки данных можно визуализировать полученныйы набор данных. Для визуализации будем использовать библиотеку seaborn (рис. 3.3).



Рисунок 3.3 – Добавление шапки DataFrame по умолчанию

В результате будет выведен график, отображающий распределение объектов попарно по различным признакам (рис. 3.4).

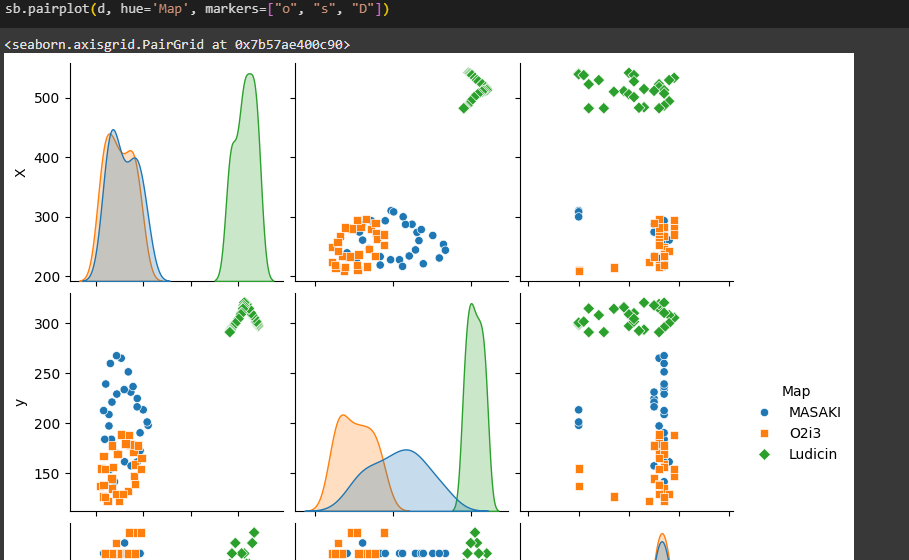


Рисунок 3.4 – Попарное признаковое распределение ирисов

На графике попарного распределения видно преимущество символьного обозначения столбцов – график легче интерпретировать. Отдельные классы не отмечаются различными цветами, но видно, что на отдельных подграфиках множества точек разделены. Следует обратить внимание на подграфики, расположенные по диаганали. Подумайте, что они отображают? 3. Для придания отдельным классам своих цветов необходимо указать, по какому признаку разделяются точки. Можно изменить маркеры каждого класса. Для этого необходимо использовать код: sb.pairplot(d, hue='answer', markers=["o", "s", "D"]). 4. Перейдем к построению модели. Модель метрической классификации должна обеспечивать следующий алгоритм работы: пользователь вводит новое признаковое описание объекта (объект отсутствует 37 в обучающей выборке), а алгоритм классификации относит новый объект к одному из классов ирисов. 5. Существует несколько вариаций метода ближайших соседей. Каждая модель предполагает наличие различных параметров для оптимизации. Воспользуемся библиотекой scikit для построения классификатора (рис. 3.5).

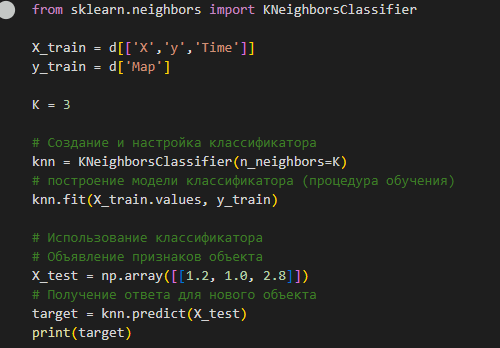


Рисунок 3.5 – Основные этапы решения задачи классификации методом ближайших соседей с использованием библиотеки scikit

Для прелставленного объекта X\_test попробуйте поменять значение признаков и проследите за изменением значения выходного класса. 6. Модель построена и выдает ответ для новых (отсутствующих в исходной выборке) объектов. Но, анализируя код на рис. 3.5, следует отметить следующие недостатки такого подхода: – в качестве количества ближайших соседей выбрано значение K=3, выбор данного значения не обосновывается, но в данном методе именно данный параметр должен оптимизироваться; отсутствует какое-либо графическое представление модели, нет визуализации процесса принятия решения. Исправим данные недостатки. Займемся обоснованием выбора оптимального значения количества ближайших соседей. Для этого будем использовать простейшую оценку качества hold-out (рис. 3.6).

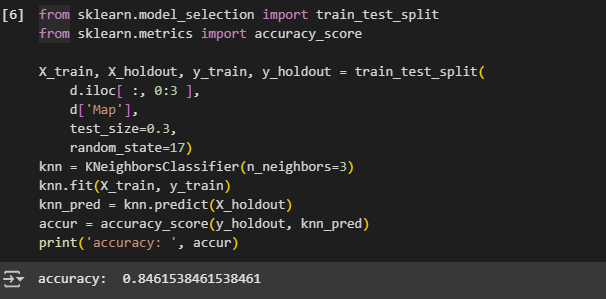


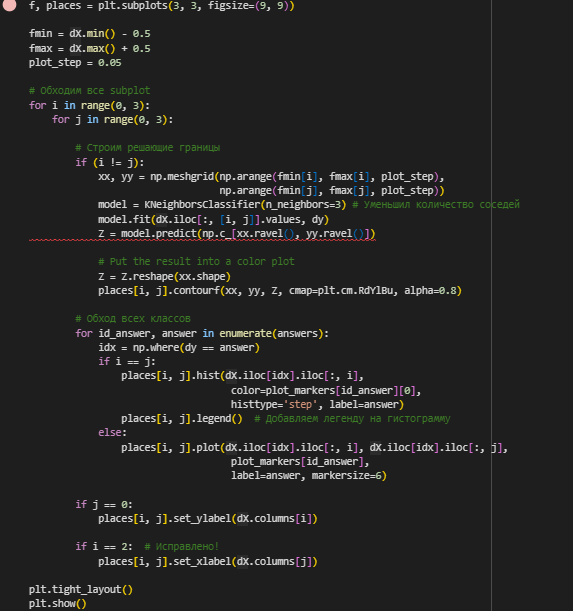
Рисунок 3.6 – Оценка точности классификатора с использованием методики hold-out

В качестве эксперимента попробуйте поменять значение количества соседей и расмотрите изменение точности класификатора. 8. Еще одна оценка качества – cross validation (CV) error. На рис. 3.7 показан алгоритм получения оценки точности классификации CV и процедура выбора оптимального значения количества соседей в алгоритме kNN на основе данной оценки.



Рисунок 3.7 – Реализация процедуры выбора оптимального параметра на основе cross validation error

Вывод для данного кода представлен на рис. 3.8.



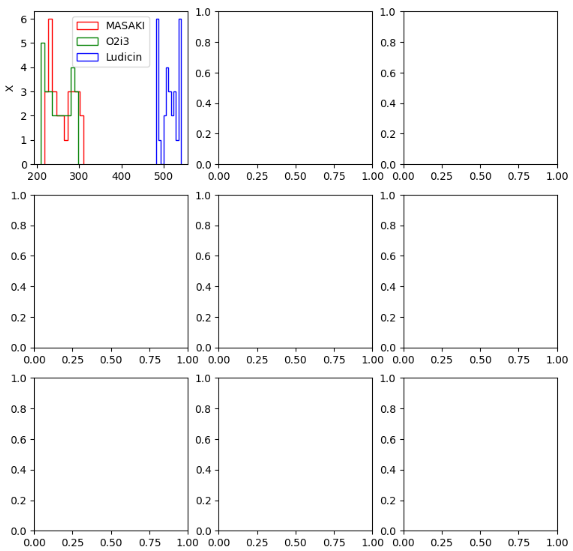


Рисунок 3.8 – Визуализация выбора оптимального параметра на основе cross validation error

**Контрольный вопросы**

1)Особенности методов ближайшего соседа (NN) и k ближайших соседей (kNN):

NN:

* Простой и интуитивно понятный.
* Для классификации объекта ищется только один ближайший сосед.
* Очень чувствителен к шуму и выбросам.
* Может давать плохие результаты если классы сильно пересекаются.

kNN:

* Более устойчив к шуму чем NN.
* Для классификации объекта ищется k ближайших соседей и класс определяется голосованием.
* Нужно выбирать оптимальное значение k.
* Может быть вычислительно затратным если данных очень много.

Оба метода относятся к "ленивым" алгоритмам (lazy learning) потому что не строят явную модель на этапе обучения а просто запоминают обучающие данные.

2)Основные принципы и этапы реализации метода kNN:

Принципы:

* Объект относится к тому классу к которому принадлежат большинство его k ближайших соседей в пространстве признаков.
* Расстояние между объектами измеряется с помощью метрики расстояния (например Евклидово расстояние, Манхэттенское расстояние).

Этапы:

* Выбор метрики расстояния.
* Выбор значения k (количество соседей).
* Для каждого объекта который нужно классифицировать:
* Вычисляем расстояние до всех объектов в обучающей выборке.
* Выбираем k ближайших соседей.
* Определяем класс объекта голосованием (или взвешенным голосованием если учитываем расстояние до соседей).

3)Принцип выбора количества соседних объектов (k):

* Выбор k влияет на качество классификации.
* Слишком маленькое k: Модель чувствительна к шуму и выбросам (как NN).
* Слишком большое k: Модель может "размыть" границы классов и плохо классифицировать объекты на границе.
* Оптимальное k обычно выбирают с помощью кросс-валидации.
* Часто используют нечетные значения k чтобы избежать ничьей при голосовании.
* Правило большого пальца: k = sqrt(n), где n - количество объектов в обучающей выборке.

4)Метод парзеновского окна:

Это метод оценки плотности вероятности.

Используется для классификации и оценки распределения данных.

Суть: В каждой точке пространства признаков строится "окно" (например с помощью Гауссовского ядра) и считается количество объектов попавших в это окно. Чем больше объектов тем выше плотность вероятности в этой точке.

Размер окна (ширина ядра) - важный параметр который влияет на результат.

5)Принцип метода потенциальных функций:

Каждому объекту обучающей выборки присваивается "потенциал" (функция).

Потенциал убывает с увеличением расстояния от объекта.

Для классификации нового объекта вычисляется суммарный потенциал создаваемый всеми объектами обучающей выборки.

Объект относится к тому классу который создает наибольший потенциал в точке нахождения объекта.

Метод похож на kNN но вместо расстояний используются значения потенциальных функций.

6)Параметры которые оптимизируют в методах kNN:

k: Количество ближайших соседей.

Метрика расстояния: Евклидово расстояние, Манхэттенское расстояние, расстояние Минковского и другие.

Веса соседей: Можно давать разным соседям разные веса в зависимости от расстояния (например чем ближе сосед тем больше вес).

1. **Вывод**

Изучили принципы построения информационных систем с использованием метрических методов классификации.