**ФГБОУ ВО   
Уфимский университет науки и технологий**

**Кафедра ВМиК**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 100 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 90 |  |  |  |  |  |  |  |  |  |  |  |  |
| 80 |  |  |  |  |  |  |  |  |  |  |  |  |
| 70 |  |  |  |  |  |  |  |  |  |  |  |  |
| 60 |  |  |  |  |  |  |  |  |  |  |  |  |
| 50 |  |  |  |  |  |  |  |  |  |  |  |  |
| 40 |  |  |  |  |  |  |  |  |  |  |  |  |
| 30 |  |  |  |  |  |  |  |  |  |  |  |  |
| 20 |  |  |  |  |  |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

Метод ближайших соседей

**ОТЧЕТ**

**по лабораторной работе**

**по** Основам машинного обучения

(*наименование дисциплины*)

|  |
| --- |
| Лабораторная работа 1 |
| (обозначение документа) |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Группа |  |  | Фамилия, И., О. | Подпись | Дата | Оценка |
| МО-325Б |  |
|  |  |
| Студент | | | Шарыгин М.С.,  Агафонов Р.В.,  Лепоринский Г.А. |  |  |  |
| Преподаватель | | | Миронов К.В. |  |  |  |
| Принял | | |  |  |  |  |

**Уфа 2025 г****.**

Содержание

[1 Цель работы 3](#_Toc191002415)

[2 Практическая часть 4](#_Toc191002416)

[2.1 Импорт датасета 4](#_Toc191002417)

[2.2 Метод главных компонент 5](#_Toc191002418)

[2.3 Метод ближайших соседей 10](#_Toc191002419)

[2.4 Итог 13](#_Toc191002420)

[3 Вывод 16](#_Toc191002421)

# Цель работы

Целью работы является изучения метода ближайших соседей на конкретном примере (датасете).

# Практическая часть

## Импорт датасета

Для начала необходимо выбрать и найти датасет: пусть это будет «Wholesale customers». Он [датасет] относится к области бизнеса ( клиенты оптового дистрибьютора), а задача состоит в определении региона поставки продукта («Region») по следующим признакам, характеризующим датасет:

* «Channel» – канал сбыта клиентов: гостиница/ресторан/кафе или розничный канал;
* «Fresh» – годовые расходы на свежие продукты;
* «Milk» – годовые расходы на молочные продукты;
* «Grocery» – годовые расходы на бакалейные товары;
* «Frozen» – годовые расходы на замороженные продукты;
* «Detergents\_Paper» – годовые расходы на моющие средства и бумажные изделия;
* «Delicassen» – годовые расходы на продукты и деликатесы.

После того, как датасет был выбран, необходимо его импортировать в «Python», для чего можно воспользоваться способом, показанным на рисунке 2.1.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.1 – Импорт датасета

Затем необходимо импортированный датасет нормализовать функцией «preprocessing.StandardScaler().fit\_transform()» из библиотеки «sklearn», как показано на рисунке 2.2.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.2 – Нормализация датасета

## Метод главных компонент

При помощи метода главных компонент «PCA» заполним таблицу с собственными векторами и собственными значениями (рисунок 2.3). На рисунке 2.4 представлена заполненная таблица.

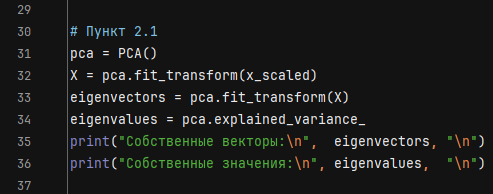


Рисунок 2.3 – Заполнение таблицы с собственными векторами и значениями

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.4 – Полученная таблица

Далее снизим размерность датасета до двух (рисунок 2.5) и трех (рисунок 2.6) параметров методом главных компонент «PCA», где установим количество параметров «n\_components» равным двум и трем соответственно. На рисунке 2.7 и рисунке 2.8 показаны полученные двухмерная и трехмерная карты массивов соответственно.

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.5 – Снижение размерности до 2 параметров и вывод двухмерной карты

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.6 – Снижение размерности до 3 параметров и вывод трехмерной карты

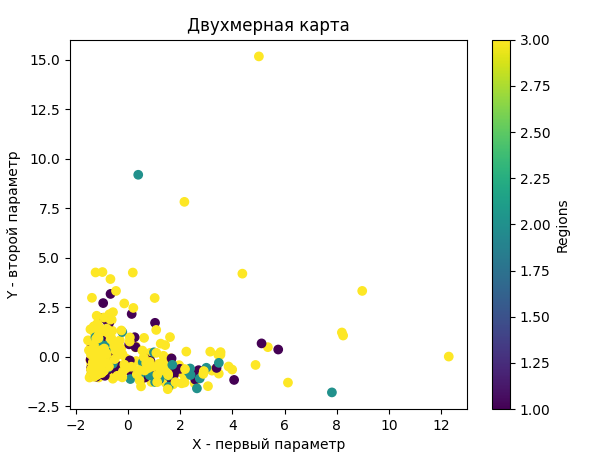


Рисунок 2.7 – Полученная двухмерная карта

Изображение выглядит как текст, снимок экрана, диаграмма, линия

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.8 - Полученная трехмерная карта

Наконец найдем оптимальное количество параметров, которое может иметь потерю информации до 20%. Для этого циклом «for» будем суммировать собственные значения от первого до последнего и делить их на всю сумму [собственных значений], пока потеря информации не станет меньше 20%. Реализация поиска приведена на рисунке 2.9, а оптимальное количество параметров (3) – на рисунке 2.10.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.9 – Поиск оптимального количества параметров

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.10 – Найденное оптимальное количество параметров

## Метод ближайших соседей

Импортируем метод k-ближайших соседей и способ «Leave-One-Out», при котором один из экземпляров датасета классифицируется на основе остальных. Затем разделим выборку на обучающую и тестовую, после чего по каждой тестовой выборке определим количество правильных и неправильных ответов. Также определим время работы метода и его точность (рисунок 2.11).

Аналогичным образом реализуем «Radius Neighbors» и «Nearest Centroid», как показано на рисунке 2.12 и рисунке 2.13.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.11 – Реализация метода k-ближайших соседей

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.12 – Реализация «Radius Neighbors»

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.13 – Реализация «Nearest Centroid»

В конце циклами «for» получим результаты (точность и производительность) с помощью трех методов при изменении количества соседей «k» в методе k-ближайших соседей и радиуса в «Radius Neighbors» (рисунок 2.14).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.14 – Применение методов с изменением параметров

## Итог

По полученным в пункте 2.2 картам можно предположить, что выборка плохо разделима, так как:

1. примеры из разных классов располагаются между собой очень близко;
2. примеров 3 класса значительно больше;

из-за чего полученная точность будет достаточно низкой.

При использовании метода k-ближайших соседей максимальная точность (~72%) достигается:

* при «uniform[[1]](#footnote-1)»: с 15 соседями за 0.6 – 0.7 секунд при разном количестве параметров;
* при «distance[[2]](#footnote-2)»: с 17 – 77 соседями за 0.4 – 0.5 секунд при разном количестве параметров.

При использовании метода «Radius Neighbors» максимальная точность (~72%) достигается:

* при «uniform»: с 15 соседями за 0.4 – 0.5 секунд при разном количестве параметров;
* при «distance»: с 15 соседями за 0.4 – 0.5 секунд при разном количестве параметров.

При использовании метода «Nearest Centroid» максимальная точность (~72%) не достигается.

На рисунок 2.15 приведены подробные результаты применения разных методов с различными параметрами.

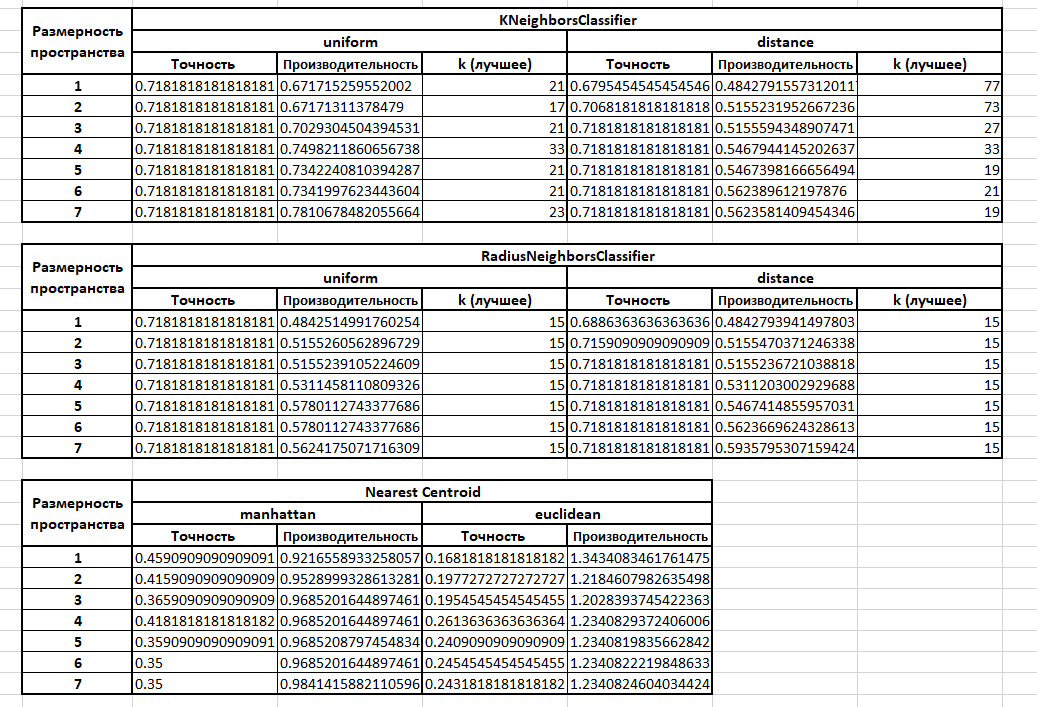


Рисунок 2.15 – Полученные результаты

Таким образом, можно отметить, что из-за особенности выборки данных в данном датасете большой разницы в результатах применения между методами k-ближайших соседей и «Radius Neighbors» нет, но при этом «Nearest Centroid» показал худшие результаты. Причины таких результатов заключаются в следующем:

* неравномерная выборка: объектов класса «Other» значительно больше остальных (316 против 77 и 47);
* объекты выборки плохо разделимы: при одинаковых характеристиках имеются примеры объектов разных классов;
* неравномерные расстояния между объектами разных классов.

# Вывод

В ходе работы мы изучили метода ближайших соседей на конкретном примере (датасете).

1. «uniform» – равномерные веса, то есть все точки в каждом районе весят одинаково [↑](#footnote-ref-1)
2. «distance» – вес точек определяется обратной величиной их расстояния, то есть более близкие соседи точки будут иметь большее влияние, чем более удаленные соседи. [↑](#footnote-ref-2)