АЛГОРИТМЫ И СТРУКТУРЫ ДАННЫХ

ЛЕКЦИЯ 10

Алгоритм k-ближайших соседей. Построение рекомендательной системы

Рекомендация зависит от признаков, масштаба и выбранного расстояния.

10

ИЗ 15 ЛЕКЦИЙ



МОДУЛЬ 3 Динамическое программирование

1 КУРС / 2 АКАДЕМИЧЕСКИХ ЧАСА / PYTHON 3

Цель и содержание лекции

ЦЕЛЬ И РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

представлять объекты векторами признаков, находить k ближайших соседей и строить рекомендации фильмов на основе оценок похожих пользователей.

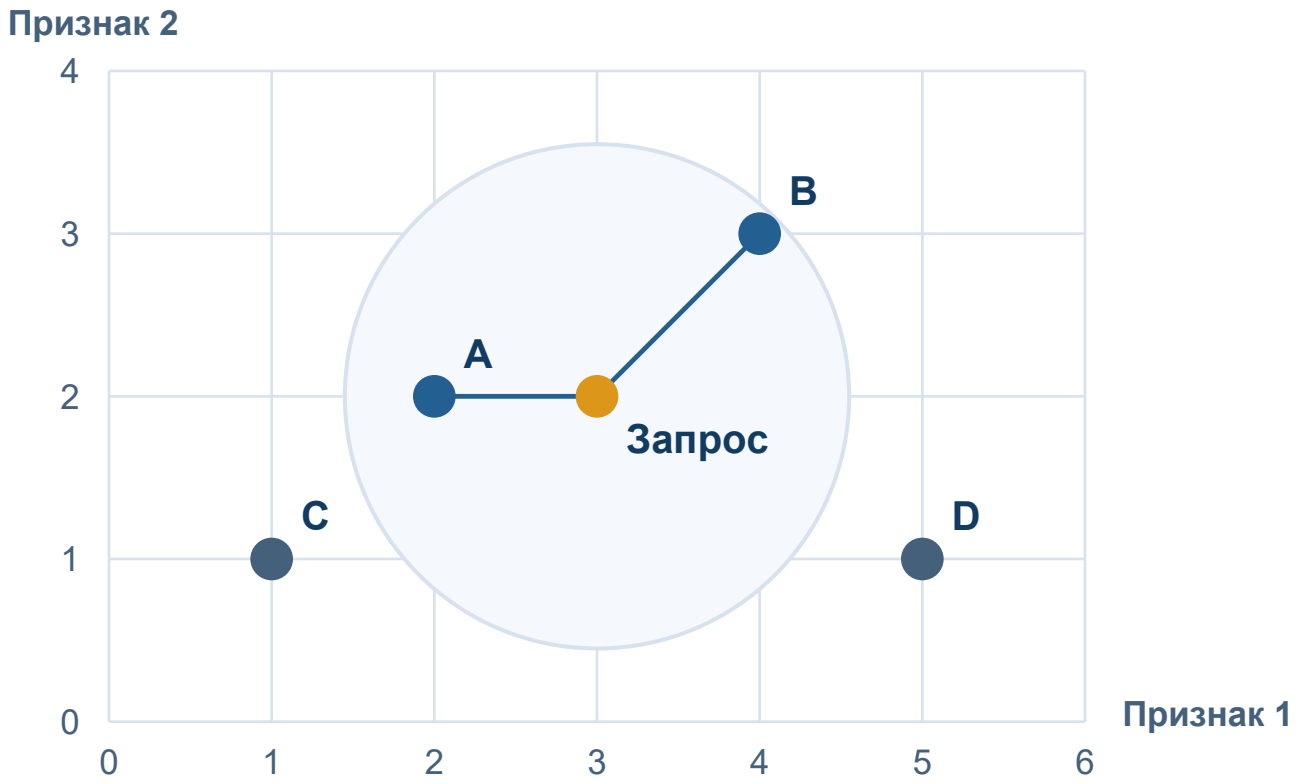
- Сравнивать объекты по числовым признакам.
- Различать классификацию и регрессию по соседям.
- Строить и проверять простой рекомендательный алгоритм.

ПЛАН ЛЕКЦИИ

1. **Признаки, расстояние и выбор соседей**
2. **Классификация по голосам ближайших соседей**
3. **Представление вкусов пользователя и расчёт расстояния**
4. **Прогноз оценки и ранжирование фильмов на Python**
5. **Выбор k, оценка качества и сложность вычислений**

Два ближайших соседа в пространстве признаков

СХЕМА



k = 2

Сравниваем точки по евклидову расстоянию.

Для запроса (3, 2) расстояния до A и B равны 1 и $\sqrt{2}$. До остальных точек они больше.

Признаки, расстояние и выбор соседей

01 ТЕМА

Алгоритм k-ближайших соседей, или kNN, опирается на предположение, что похожие объекты часто имеют похожий ответ. Например, фрукт можно описать числом, выражающим размер, и числом, выражающим оттенок кожуры. Тогда он становится точкой с двумя координатами. Чтобы сравнить пользователей по вкусам, вместо характеристик фрукта можно использовать оценки нескольких жанров. Получится вектор из стольких чисел, сколько жанров мы выбрали.

Извлечение признаков переводит исходное описание объекта в сопоставимый набор чисел. Порядок и смысл координат должны совпадать у всех объектов. Для числовых признаков удобно использовать евклидово расстояние. Чтобы найти его, по каждой координате вычитаем одно значение из другого, возводим разность в квадрат, складываем квадраты и извлекаем корень. Если нужно только отсортировать объекты по близости, достаточно сравнивать суммы квадратов. Квадратный корень сохраняет их порядок.

Признаки, расстояние и выбор соседей

01 ПРОДОЛЖЕНИЕ

Параметр k задаёт число соседей, участвующих в прогнозе. Для каждого нового объекта сравниваем его признаки с признаками известных объектов, сортируем их по расстоянию и берём первые k. При равных расстояниях нужно заранее установить правило разрешения ничьей, чтобы повторный запуск на тех же данных дал тот же результат.

Классификация по голосам ближайших соседей

02 ТЕМА

В главе 10 классификация объясняется на примере апельсинов и грейпфрутов. Введём учебные координаты, где первое число отражает размер, а второе — интенсивность красного оттенка кожуры. Пусть новый фрукт имеет координаты `(4, 4)`. Для апельсинов `(4, 3)` и `(3, 3)` квадраты расстояний до него равны 1 и 2. Для грейпфрутов `(4, 6)` и `(7, 6)` получаем 4 и 13. При `k = 3` среди ближайших фруктов два апельсина и один грейпфрут. Алгоритм предсказывает класс «апельсин» большинством голосов.

При классификации результат — метка из заданного набора. При **регрессии** результат — число. Тот же поиск соседей можно использовать для обоих вариантов. Если известные объекты имеют числовые ответы, например оценки фильма, простейший прогноз — среднее этих ответов у выбранных соседей. В обоих случаях прогноз зависит от признаков, меры близости и состава известных примеров.

Представление вкусов пользователя и расчёт расстояния

03 ТЕМА

В примере из книги Приянка, Джастин и Морфеус оценивают пять жанров на общей шкале от 1 до 5. Эти оценки используются как признаки. Оценки конкретных фильмов хранятся отдельно и служат для прогноза.

Пользователь	Комедия	Боевик	Драма	Ужасы	Мелодрама
Приянка	3	4	4	1	4
Джастин	4	3	5	1	5
Морфеус	2	5	1	3	1

Представление вкусов пользователя и расчёт расстояния

03 ПРОДОЛЖЕНИЕ

Для Приянки и Джастина квадраты разностей равны `1, 1, 1, 0, 1`. Сумма равна 4, значит, евклидово расстояние равно 2. Для Приянки и Морфеуса сумма квадратов равна `1 + 1 + 9 + 4 + 9 = 24`. Расстояние равно корню из 24, примерно 4,90. На выбранных признаках Джастин ближе к Приянке. Из такой близости делаем прогноз о вкусе в конкретном фильме и затем можем сравнить прогноз с её будущей оценкой.

Прогноз оценки и ранжирование фильмов на Python

04 ТЕМА

Сначала находим k ближайших пользователей и исключаем саму Приянку из поиска соседей. Затем для каждого фильма, который она ещё не оценила, усредняем имеющиеся оценки выбранных соседей. Пропуски игнорируются. Фильмы без оценок от соседей исключаем из выдачи. Сортируем результат по убыванию прогнозной оценки, при равенстве — по названию. Дополнительно показываем число оценок, чтобы читатель мог отличить среднее по одному соседу от среднего по нескольким.

В коде жанровые оценки трёх пользователей и пять оценок фильма «Идеальный голос» взяты из главы 10. Жанровые оценки остальных четырёх соседей и оценки других фильмов добавлены для демонстрации. На входе все профили имеют одинаковый порядок и число координат. Параметр k лежит от 1 до числа других пользователей.

Прогноз оценки и ранжирование фильмов на Python

10_01.py / фрагмент 1 из 2

```
def squared_distance(left, right):
    return sum((x - y) ** 2 for x, y in zip(left, right))


def recommend(profiles, ratings, user, k):
    ranked = sorted(
        (squared_distance(profiles[user], vector), name)
        for name, vector in profiles.items()
        if name != user
    )
    neighbors = [name for _, name in ranked[:k]]
    results = []
    for title, scores in ratings.items():
        if user in scores:
            continue
        available = [scores[name] for name in neighbors if name in scores]
        if available:
            average = sum(available) / len(available)
            results.append((title, average, len(available)))
    return sorted(results, key=lambda item: (-item[1], item[0]))
```

Прогноз оценки и ранжирование фильмов на Python

10_01.py / фрагмент 2 из 2

```
profiles = {
    'Приянка': (3, 4, 4, 1, 4), 'Джастин': (4, 3, 5, 1, 5),
    'Морфеус': (2, 5, 1, 3, 1), 'Джей-Си': (3, 2, 4, 1, 5),
    'Джозеф': (4, 2, 4, 1, 5), 'Ланс': (3, 2, 4, 1, 2),
    'Крис': (2, 4, 2, 1, 2),
}
ratings = {
    'Идеальный голос': {'Джастин': 5, 'Джей-Си': 4, 'Джозеф': 4,
                        'Ланс': 5, 'Крис': 3},
    'Городские огни': {'Джастин': 4, 'Морфеус': 5, 'Крис': 3},
    'Окно во двор': {'Морфеус': 5},
    'Амели': {'Джастин': 5, 'Приянка': 5},
}

for title, average, count in recommend(profiles, ratings, 'Приянка', 5):
    print(f'{title}: {average:.2f} ({count})')
```

РЕЗУЛЬТАТ ЗАПУСКА

```
Идеальный голос: 4.20 (5)
Городские огни: 3.50 (2)
```

Прогноз оценки и ранжирование фильмов на Python

04 ПРОДОЛЖЕНИЕ

Для первого фильма среднее равно `(5 + 4 + 4 + 5 + 3) / 5 = 4,2`. Для второго используются только оценки Джастина и Криса. Морфеус шестой по близости и в группу из пяти не попал. «Окно во двор» оценил только Морфеус, а «Амели» Приянка уже оценила.

Выбор k, оценка качества и сложность вычислений

При k = 1 прогноз полностью зависит от одного примера, поэтому случайная или ошибочная оценка сильно влияет на результат. При слишком большом k в прогноз начинают попадать пользователи с далёкими вкусами. Значение k можно выбирать по проверочным данным. Например, скрыть некоторые известные оценки, построить прогнозы без них и сравнить среднюю абсолютную ошибку для нескольких значений k. Вместе с ошибкой полезно учитывать долю скрытых оценок, для которых алгоритм вообще смог построить прогноз. При проверке скрытые оценки не должны участвовать ни в построении признаков, ни в усреднении ответов.

Попробуем такой расчёт на оценке фильма «Амели». Временно уберём пятёрку Приянки из словаря. Джастин оценил фильм на 5, поэтому прогноз по одному соседу тоже равен 5. После возврата скрытого значения получаем абсолютную ошибку 0. Для сравнения, если бы Приянка поставила 3, ошибка была бы 2. Один пример показывает способ вычисления. Чтобы сравнить значения k, повторяем процедуру на нескольких фильмах и пользователях, используя одинаковый набор скрытых оценок.

Выбор k, оценка качества и сложность вычислений

05 ПРОДОЛЖЕНИЕ

Если один признак задан на шкале от 1 до 5, а другой — в сотнях минут, второй может определять почти всё расстояние. Для сравнения признаков приводят их к сопоставимому масштабу или задают обоснованные веса. Отсутствие оценки также имеет смысл. Если поставить ноль вместо пропуска, алгоритм воспримет незнакомый жанр как сильную неприязнь. В коде профили заполнены целиком, а пропуски оценок фильмов представлены отсутствием ключа.

Даже в пределах одной шкалы люди пользуются баллами по-разному. Один пользователь обычно ставит любимым фильмам 5, другой выражает то же предпочтение оценкой 4. Если похожие профили отличаются систематическим сдвигом, при сравнении можно вычитать из каждого профиля его среднюю оценку. Тогда координаты показывают, какие жанры пользователь предпочитает относительно собственного среднего.

Выбор k, оценка качества и сложность вычислений

05 ПРОДОЛЖЕНИЕ

Пусть сравниваем N других пользователей по d признакам и просматриваем M фильмов. Расчёт расстояний занимает $O(Nd)$, сортировка пользователей — $O(N \log N)$. Для каждого фильма проверяется до k оценок, что даёт $O(Mk)$, а сортировка рекомендаций занимает до $O(M \log M)$. На маленьком наборе такой прямой поиск позволяет легко проверить каждый шаг по исходным данным.

Что нужно уметь после лекции

ИТОГИ

kNN использует одну и ту же процедуру поиска соседей для классификации и регрессии. В рекомендациях жанровые признаки определяют сходство пользователей, оценки отдельных фильмов образуют прогноз, а значение k влияет на состав учитываемой группы.

> Рекомендация зависит от признаков, масштаба и выбранного расстояния.

Проверьте понимание

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Зачем у всех пользователей должен совпадать порядок признаков?
2. Как рассчитать квадрат расстояния между двумя профилями?
3. Чем голосование в классификации отличается от прогноза оценки фильма?
4. Почему «Окно во двор» не попало в результат программы?
5. Как проверить, улучшает ли увеличение k качество прогноза?

Самостоятельная работа

ПРАКТИКА

1. Установите `k = 1`, затем `k = 6`. Запишите новые прогнозы и объясните, какой сосед изменил выдачу.
2. Для фруктов из лекции реализуйте голосование по классам. Выберите правило для равного числа голосов и проверьте его при `k = 4`.
3. Добавьте пять оценок к одному или нескольким фильмам. По очереди скрывайте каждую оценку и измеряйте абсолютную ошибку прогноза для двух значений `k`. Отдельно отметьте случаи, когда прогноз построить не удалось.

Источники и документация

ЧТЕНИЕ

ОСНОВНОЕ ЧТЕНИЕ

А. Бхаргава. Грокаем алгоритмы

СПб.: Питер, 2017. Глава 10. Алгоритм k ближайших соседей. С. 236–253.

> При чтении сначала вручную пройдите один пример, затем запустите код и сравните результат.