Карта задач ИИ

Что послушать дальше? Соберём личную рекомендацию, заглянем в её устройство и проверим прогноз.

В плейлисте могут соседствовать Кино, Земфира, Noize MC, Linkin Park и Gorillaz. Но что включить дальше? Любимый жанр не решает вопрос: две рок-группы могут нравиться вам совершенно по-разному. Друг, который знает ваши вкусы, предложит что-то конкретное. Попробуем дать такую же задачу программе.

Найдите в каталоге хотя бы трёх знакомых исполнителей. Поставьте каждому оценку от 1 до 5: единица — совсем не нравится, пятёрка — очень нравится. Незнакомых оставьте без оценки. Посмотрите, кого программа предложит, затем измените одну оценку. Как изменится список?

Что послушать дальше?

Пример: оценки выбраны автором. Результаты рассчитаны тем же правилом.

Что нравится вам?

1 — не нравится · 5 — очень нравится

Gorillaz	4
Lady Gaga	2
Radiohead	5
Linkin Park	5
Eminem	4

Что послушать дальше?

Личный профиль обучен в браузере по 5 оценкам.

Первые три среди ещё не оценённых — по прогнозу вашей оценки.

▶ Как обучился профиль?

1 Massive Attack 4,41 / 5

Наибольший вклад: ваша оценка Radiohead — 5/5 (+0,62 к прогнозу).

2 OneRepublic 4,34 / 5

Наибольший вклад: ваша оценка Linkin Park — 5/5 (+0,83 к прогнозу).

3 Sigur Rós 4,32 / 5

Наибольший вклад: ваша оценка Radiohead — 5/5 (+1,47 к прогнозу).

Даже первое место не обещает, что музыка вам понравится. Поменяйте одну оценку и сравните список.

Ваши оценки меняют рекомендации. Модель обучена на реальных записях Last.fm о том, каких исполнителей слушали одни и те же люди. Здесь подбираются исполнители, а не отдельные песни.

Интерактивный эксперимент — в электронной версии урока.

Похож ли результат на совет друга? Пока это впечатление. Чтобы понять, чему доверять, разберём, откуда берётся список и как устроить проверку.

1. Что программа знает о вашем вкусе

Вход нашей модели — ваши оценки знакомых исполнителей и сведения об их общей аудитории. **Ответ** — предположение о том, какую оценку вы поставите другому исполнителю. Отсортировав такие прогнозы, получим рекомендации.

В основе опыта — открытый исследовательский набор Last.fm, выпущенный в мае 2011 года: 1 892 слушателя, 17 632 исполнителя и 92 834 записи о прослушиваниях. Для опыта выбраны 200 самых представленных исполнителей и ещё 19 русскоязычных из того же набора — например, Кино, Земфира и Сплин. Это старый срез: многих современных артистов в нём нет. У добавленных исполнителей здесь всего от 4 до 21 слушателя, поэтому данных для рекомендаций мало.

В исходной записи указано, сколько раз человек слушал исполнителя. Звёздных оценок там нет — оценки в нашем опыте ставите вы.

Представьте большую таблицу: строки — слушатели, столбцы — исполнители, отметки — известные прослушивания. Программа ищет повторяющиеся сочетания и сжимает эту таблицу: каждому исполнителю достаётся всего 12 чисел. По вашим оценкам она подбирает ещё 12 чисел — ваш личный профиль. Сопоставляя его с профилями исполнителей, можно получить новые прогнозы. Такой способ сжатия называют **матричной факторизацией**; вычисления разберём в Профи.

История

Набор подготовили исследователи Universidad Autónoma de Madrid для семинара HetRec 2011. В нём сохранены числовые идентификаторы слушателей, без их имён. Мы используем совместные прослушивания, а не сведения о дружбе.

Откуда взялся этот совет?

Пример: оценки выбраны автором. Результаты рассчитаны тем же правилом.

Разобрать рекомендацию Massive Attack

Каждое место — одна из 12 скрытых координат. Это не названия жанров: числа получены обучением и показаны с округлением.

Ваш профиль

1	2	3	4	5	6	7	8	9	10	11	12
0,14	0,06	1,01	-0,06	0,24	-1,08	0,05	0,67	0,15	-0,38	0,49	-0,93

Massive Attack

1	2	3	4	5	6	7	8	9	10	11	12
0,30	-0,12	0,50	-0,31	0,08	0,21	-0,03	0,71	-0,11	-0,34	0,59	-0,20

Перемножаем по местам

1	2	3	4	5	6	7	8	9	10	11	12
0,04	-0,01	0,51	0,02	0,02	-0,22	-0,00	0,48	-0,02	0,13	0,29	0,18

Складываем произведения: 1,41. Прибавляем 3 и ограничиваем шкалой 1–5 → 4,41.

Ваша оценка	Вес	Вклад
Lady Gaga: 2/5	-0,15	+0,15
Radiohead: 5/5	0,31	+0,62
Linkin Park: 5/5	0,14	+0,27
Eminem: 4/5	0,32	+0,32
Gorillaz: 4/5	0,05	+0,05

Начальные 3 балла + вклады всех оценок = 4,41. Прогноз: 4,41. Вклад = вес × (оценка − 3). При отрицательном весе низкая оценка поднимает прогноз другого исполнителя. Это зависимость в модели, а не объяснение причины вашего вкуса.

В цветных ячейках — настоящие числа обученной модели, в таблице — вклад ваших оценок в прогноз. Поменяйте оценку в начале главы: ваш профиль и рекомендация пересчитаются. Скрытые координаты не подписаны жанрами: таких названий данные нам не дают.

Интерактивный эксперимент — в электронной версии урока.

Это простой вариант **рекомендательной системы**. Она использует закономерности в чужих прослушиваниях и ваши примеры. Здесь нет понимания музыки: программа не слышит мелодию, не читает тексты и не знает, что вы хотите включить именно сегодня.

Заметка

Отсутствие записи не означает «не нравится». Человек мог не знать исполнителя; кроме того, в набор попали не все его прослушивания. Нельзя превращать пустое место в плохую оценку.

Музыкальное приложение использует набор Last.fm с количествами прослушиваний. Перед прогнозом оно просит вас оценить знакомых исполнителей от 1 до 5. Откуда берутся эти оценки?

- ○ Авторы учебника заранее назначили всем исполнителям правильные оценки
- ○ Это средние оценки всех слушателей Last.fm
- ○ Их ставит читатель; исходный набор содержит количества прослушиваний
- ○ Программа услышала песни и оценила их качество

2. Предсказать оценку

Можно спросить: «Какую оценку я поставлю этому исполнителю?» Программа отвечает числом, например 3,9 из 5. Предсказание числовой величины называют **регрессией**. Тем же способом ставят задачи о времени доставки, температуре или цене подержанного велосипеда.

Для проверки нужна ваша настоящая оценка. Если вы поставили 4, прогноз 3,9 ошибся на 0,1, а прогноз 2 — на 2. Чем меньше такой промах, тем точнее числовой ответ. Но одна угаданная оценка ещё не делает модель надёжной: нужно проверить несколько примеров.

В нашей модели 3 — отправная точка. Программа настраивает личный профиль так, чтобы приблизиться к вашим оценкам, но не делать слишком резких выводов из нескольких примеров. Затем использует тот же профиль для незнакомых исполнителей. Если расчёт выходит за пределы 1–5, ограничиваем его этой шкалой. Это предполагаемая оценка, а не измеренная вероятность вашей реакции.

Пример

В числовом прогнозе 3,9 означает предполагаемую оценку на шкале 1–5. Это не «вероятность понравиться 78 %»: деление на пять не превращает оценку в вероятность.

3. Выбрать ответ из двух вариантов

Теперь спросим иначе: «Поставлю ли я не ниже четырёх?» Возможны два ответа: «да» или «нет». Выбор из заранее известных категорий называют **классификацией**. Для нашего опыта договоримся: прогноз от 4 означает «да», ниже 4 — «нет».

Пусть ваша настоящая оценка — 4. Прогноз А равен 3,9, прогноз Б — 4,7. А ближе по числу, но только Б правильно отвечает на вопрос «не ниже четырёх?». Мы не поменяли ваш вкус — поменяли требуемый ответ и способ проверки.

Вы оценили исполнителя на 4. Модель А предсказала 3,9, модель Б — 4,7. Проверяют и близость к оценке, и ответ «оценка не ниже 4». Как верно сравнить прогнозы?

- ○ А лучше при любой проверке
- ○ Б ближе по числу; А верно отвечает «оценка не ниже 4»
- ○ А ближе по числу; Б верно отвечает «оценка не ниже 4»
- ○ Оба прогноза означают «оценка не ниже 4»

Рекомендация задаёт ещё один вопрос: «Кого показать первым?» Расположение кандидатов по предполагаемому интересу называется ранжированием. Программа может правильно упорядочить исполнителей, хотя и неточно предскажет каждую оценку. Поэтому качество списка и точность чисел не одно и то же.

Подумай

Если всем поставить 3, личный профиль получится нулевым, а все прогнозы — равными. Первое место в таком списке определяется порядком каталога, а не вашим вкусом.

4. Проверить без подсказки

Вы уже оценили нескольких исполнителей, значит правильные ответы у нас есть. Спрячем одну оценку, построим прогноз только по оставшимся и лишь затем раскроем спрятанное число. Иначе программа получила бы ответ вместе с условием.

Угадаёт ли модель скрытую оценку?

Пример: оценки выбраны автором. Результаты рассчитаны тем же правилом.

Скрываем оценку Gorillaz

Для прогноза оставлены 4 оценки: Lady Gaga — 2; Radiohead — 5; Linkin Park — 5; Eminem — 4. Скрытая оценка в расчёт не входит.

Модель предполагает **3,74 / 5**

Ваша оценка — 4. Промах — 0,26 балла. Модель занизила оценку.

Если по очереди скрыть каждую из 5 оценок, средний промах модели — 1,03. У простого правила «всегда 3» — 1,40.

Это небольшая проверка на ваших оценках. Если менять их, глядя на результат, получится подгонка под проверку; для новой проверки понадобятся другие примеры.

Для проверки нужны хотя бы три ваши оценки. Проверяемая оценка исключается из расчёта прогноза. Для каждой проверки личный профиль обучается заново без спрятанного ответа. Профили исполнителей остаются известными: они получены из исторического набора, независимо от ваших оценок.

Интерактивный эксперимент — в электронной версии урока.

Примеры, по которым модель настраивается, называют **обучающей выборкой**, или **train**. Отложенные для проверки — **тестовой выборкой**, или **test**. В нашем маленьком опыте оставшиеся оценки задают личный прогноз, а спрятанная служит тестом. Числа, описывающие исполнителей, мы при этом не переучиваем: они получены до ваших оценок.

Повторим опыт, по очереди пряча каждую оценку. Сложим величины промахов и разделим на число проверок — получим среднюю ошибку. Рядом посчитаем ошибку простого соперника: он всегда предсказывает 3. Сложная модель должна доказывать свою пользу сравнением, а не названием.

Программа по вашим оценкам знакомых исполнителей прогнозирует оценку другого. Вы уже оценили Gorillaz и хотите проверить прогноз для этой группы. Что нужно сделать до получения прогноза?

- ○ Заменить настоящую оценку прогнозом, чтобы убрать ошибку
- ○ Оставить оценку во входе и проверить, повторила ли её программа
- ○ Убрать эту оценку из входа, получить прогноз и затем сравнить с ней
- ○ Проверять только исполнителей, на которых модель уже ответила верно

Ссылки

Курс Google о рекомендательных системах различает явную обратную связь, например оценку, и косвенную — например просмотр или прослушивание.

Три личные оценки — маленькая проверка, а не доказательство качества для всех слушателей. Если после каждого результата менять правило, вы начнёте подгонять его под тест. Для окончательного сравнения понадобится ещё один набор ответов, который не участвовал в выборе правила.

5. Какие ещё ответы бывают

Подбор исполнителя использует готовый каталог. А просьба «сочини песню для нашего класса» требует создать новый материал. Это **генерация**. У неё может быть много подходящих ответов; проверять нужно выполнение просьбы, факты и качество результата, а не совпадение с одной заранее записанной песней.

Другой пример — афиша школьного концерта. Чтобы вырезать человека из фотографии, недостаточно ответа «на снимке человек». Нужна **маска**: для каждого пикселя, маленькой клетки изображения, метка «человек» или «фон». Такую разметку изображения называют сегментацией. Если рука ошибочно отмечена как фон, редактор удалит её вместе с ним.

Для афиши нужно вырезать человека из фотографии, сохранив все части его силуэта. Какой ответ должна выдать модель, чтобы редактор мог отделить человека от фона?

- ○ Только число людей на снимке
- ○ Метка «человек» или «фон» для каждого пикселя
- ○ Только одна метка «есть человек» для всего снимка
- ○ Название места съёмки

Так карта задач расширяется: число, категория, упорядоченный список, разметка изображения, новый текст или звук. Название приложения не определяет задачу. В одном музыкальном сервисе могут работать сразу несколько моделей с разными ответами.

6. От маленького опыта к сервису

В учебном каталоге можно пересчитать все варианты после каждого клика. В огромном сначала стоит быстро отобрать кандидатов, затем точнее оценить короткий список. Короткие числовые профили исполнителей уже вычислены заранее: для нового прогноза не нужно снова читать всю историю прослушиваний. Подробно к скорости вернёмся после знакомства с алгоритмами поиска.

Заметка

Быстрый отбор тоже может ошибаться: если подходящий исполнитель не попал в короткий список, точная модель на следующем этапе уже не вернёт его. Ускорение проверяют вместе с качеством результата.

Для рабочего сервиса придётся уточнить цель. Хотим, чтобы человек нажал на исполнителя, дослушал трек или нашёл новую любимую музыку? Это разные признаки успеха. Много прослушиваний может означать привычку; высокий интерес к обложке ещё не доказывает, что музыка понравилась.

Мы заменили вопрос «какую оценку поставит слушатель?» на «будет ли оценка не ниже 4?». Что изменилось?

- ○ Теперь программа должна сочинить новую песню
- ○ Теперь прошлые оценки нельзя использовать
- ○ Ошибок больше не будет, ведь ответов только два
- ○ Нужны два варианта ответа и проверка совпадения категории

Профи. Формула прогноза и цена вычислений

Начнём с исторической таблицы P: $P_{ui} = 1$, если слушатель u встречается у исполнителя i, иначе 0. Здесь 1 830 строк и 219 столбцов. Ноль — отсутствие записи, не оценка «не нравится».

Подберём для слушателей векторы a_u, для исполнителей — q_i, по 12 координат в каждом. Их скалярное произведение должно приближать таблицу:

$$P \approx AQ^\top, \qquad \min_{A,Q} \sum_{u,i} c_{ui}(P_{ui} - a_u^\top q_i)^2 + 0{,}1(\|A\|_F^2 + \|Q\|_F^2)$$

Вес $c_{ui} = 20$ для известного прослушивания и 1 для пустого места. Так пустые места служат слабым нулевым ориентиром, а не известными отрицательными отзывами. Для обучения заранее сделано 20 проходов ALS: фиксируем одну матрицу, подбираем другую и меняемся ролями. Уменьшение этой обучающей ошибки ещё не доказывает качество рекомендаций.

Для браузера все векторы исполнителей поделены на общий коэффициент: средний квадрат их длины стал равен 1. Обратное масштабирование векторов слушателей сохраняет произведение матриц. Координаты не имеют заданных названий вроде «рок» или «поп».

Теперь учим новый личный профиль p по вашим оценкам r_j исполнителей из множества R. Это отдельная задача: в исходном Last.fm звёздных оценок нет.

$$p = \arg\min_{p} \sum_{j \in R} (r_j - 3 - p^\top q_j)^2 + 0{,}5 \|p\|^2, \qquad \hat{r}_i = \min(5, \max$$

Браузер решает небольшую систему линейных уравнений при каждом изменении оценки. Штраф на длину профиля удерживает прогнозы ближе к тройке, когда примеров мало. В таблице объяснения показано точное разложение прогноза на вклады оценок до ограничения шкалой; это не причинное объяснение музыкального вкуса.

Для проверки оценки r_i сначала исключаем её из R и заново находим p. Иначе личный профиль уже знал бы проверяемый ответ. Ошибка по тем же оценкам, на которых он учился, не заменяет эту проверку.

При N исполнителях и $d = 12$ координатах хранение профилей и полный пересчёт их числовых прогнозов требуют порядка Nd чисел и операций. Обучение личного профиля по m оценкам требует порядка $md^2 + d^3$ операций. Для большого каталога можно ускорить поиск кандидатов, но точность такого отбора надо проверять отдельно.

Условимся обозначать настоящий ответ через y, прогноз — $\hat{y}$. Для регрессии считаем $|\hat{y} - y|$, для обычной классификации — совпадение категории. Эти две проверки не взаимозаменяемы.

В генерации текста ответ собирается последовательно. Вероятность последовательности раскладывается по правилу цепочки:

$$P(w_1, \ldots, w_n) = \prod_{t=1}^{n} P(w_t \mid w_1, \ldots, w_{t-1}).$$

Это точное равенство без предположения о независимости слов. Токен — слово, часть слова или знак. В следующем упражнении токены совпадают со словами, а вероятности заданы вручную: это имитация последовательного выбора, не обученная языковая модель.

> В словаре двадцать токенов. Сколько разных последовательностей ровно из трёх токенов можно составить? Повторы разрешены; порядок важен.

Генерация — классификация в цикле

Цель: Сделайте три шага генерации, выбирая на каждом самый вероятный токен, и получите фразу «буду через десять минут»

```
vocab_probs = {    # что мини-модель знает: условные вероятности пяти
кандидатов для набранного текста
    "буду": {"через": 0.46, "дома": 0.21, "готов": 0.15, "рад": 0.10,
"поздно": 0.08},
    "буду через": {"десять": 0.42, "пять": 0.18, "час": 0.15, "минуту":
0.13, "чуть": 0.12},
    "буду через десять": {"минут": 0.55, "часов": 0.15, "секунд": 0.10,
"дней": 0.10, "утра": 0.10},
}
K = 5                       # показываем только пять кандидатов на каждом
шаге
text = "буду"               # что уже набрано
steps = 0                   # сколько раз модель выбрала токен

n_steps = 1                 # TODO: сколько шагов генерации сделать (в
фразе их три)
for _ in range(n_steps):
    dist = vocab_probs.get(text)       # распределение по показанным
кандидатам — для текущего текста
    if dist is None:                   # мини-модель знает продолжение не
для всякой фразы
        print(f"для «{text}» продолжения нет: значит, на прошлом шаге
выбран не тот токен")
        break
    token = min(dist, key=dist.get)    # TODO: сейчас берём самый
маловероятный вариант, нужен самый вероятный
    text = f"{text} {token}"           # выбранный токен приписали — и
тот же вопрос задаём заново
    steps += 1
    print(f"шаг {steps}: выбрали «{token}» с p = {dist[token]:.2f} →
«{text}»")

print(f"итог: «{text}», выборов сделано: {steps}")
print(f"шагов сделано: {steps}")
```

Эпик. Точнее оценки — лучше рекомендации?

У трёх исполнителей настоящие оценки слушателя равны 5, 4 и 1. Модель А предсказывает 4, 5 и 1; модель Б — 3, 2 и 1. Какая модель лучше по средней абсолютной ошибке? Какая верно выбирает любимого исполнителя? Докажите на этом примере, что улучшение числовой ошибки не гарантирует улучшения первой рекомендации.

▾*Решение*

У А средняя ошибка равна $(1+1+0)/3 = 2/3$, у Б — $(2+2+0)/3 = 4/3$: по числам А точнее. Но А ставит второго исполнителя выше первого, а Б правильно выбирает первого. Числовая ошибка

учитывает расстояния до оценок, первая рекомендация — победителя по порядку. Оба показателя полезны, но отвечают на разные вопросы.

Следующий сюжет

Разложим данные по строкам и столбцам. На примере сообщений разберём, какие сведения получает модель и чего из таблицы узнать нельзя.

Рекомендации используют известные предпочтения; чужие прослушивания и ваши оценки — разные сведения.

Регрессия предсказывает число, классификация выбирает категорию, ранжирование упорядочивает кандидатов.

Тестовый ответ скрывают до прогноза; качество сравнивают с простым правилом.

Маска отмечает части изображения, генерация создаёт новый материал; каждому ответу нужна своя проверка.

Домашняя работа

1. В учебном примере слушатель поставил трём исполнителям оценки 5, 4 и 1. Модель, каждый раз не видя проверяемой оценки, предсказала соответственно 4, 3 и 2. Посчитайте три абсолютные ошибки и их среднее. Сравните с правилом «всегда отвечать 3». Какая модель точнее на этих трёх примерах?

2. Слушатель оценил трёх исполнителей на 5, 4 и 1; прогнозы модели — соответственно 4, 3 и 2. Теперь нужен ответ «оценка не ниже 4?». Запишите настоящие категории и категории по прогнозам 4, 3 и 2. Сколько ошибок получилось? Объясните, почему число неверных категорий нельзя называть средней ошибкой в звёздах.

Источники

- Matrix factorization, Google (https://developers.google.com/machine-learning/recommendation/collaborative/matrix) — представление слушателей и объектов короткими векторами, взвешенная функция ошибки и ALS.
- HetRec 2011, GroupLens (https://grouplens.org/datasets/hetrec-2011/) — Last.fm и набор Universidad Autónoma de Madrid: Cantador, Brusilovsky, Kuflik, HetRec 2011. Записи о прослушиваниях, не звёздные оценки. Условия набора предусматривают некоммерческое использование.
- README набора Last.fm (https://files.grouplens.org/datasets/hetrec2011/hetrec2011-lastfm-readme.txt) — состав файлов, статистика, значение listening count и условия использования.
- Google: Collaborative filtering (https://developers.google.com/machine-learning/recommendation/collaborative/basics) — рекомендации по взаимодействиям пользователей, явная и косвенная обратная связь.
- Google: Recommendation systems overview (https://developers.google.com/machine-learning/recommendation/overview/types) — отбор кандидатов, оценка и окончательное упорядочивание.
- Google: What is Machine Learning? (https://developers.google.com/machine-learning/intro-to-ml/what-is-ml) — обучение по примерам, классификация и регрессия.
- Vaswani et al.: Attention Is All You Need (https://arxiv.org/abs/1706.03762) — архитектура трансформера; дополнительное чтение о генерации.

- Bike Sharing, UCI (https://archive.ics.uci.edu/dataset/275/bike+sharing+dataset) — источник отдельного упражнения тренажёра о выдачах велосипедов; Hadi Fanaee-T, CC BY 4.0.