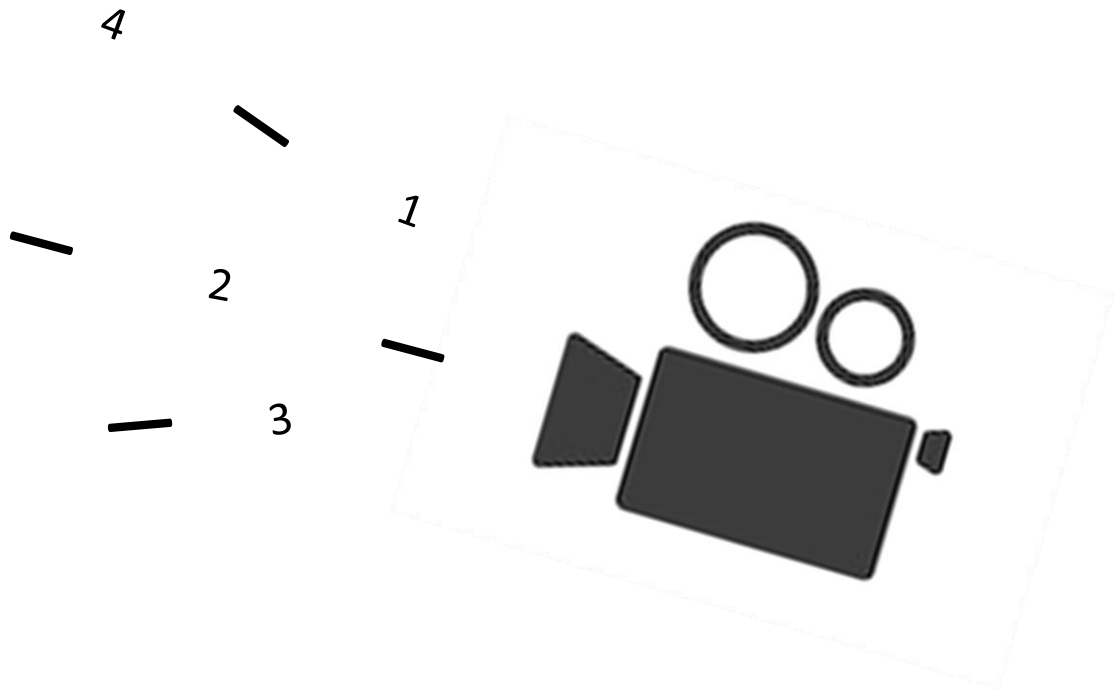
Рекомендация фильмов по известным данным



Задание второго этапа отбора

Ка-фу Софья
Калинина Вероника

Алгоритм решения задачи

Для наибольшей точности работы рекомендательной системы фильмов искусственному интеллекту необходимо пройти по алгоритму, состоящему из характерных параметров:

1. Разделим данный список фильмов на категории по их жанру. Некоторые из фильмов, совмещающие в себе несколько жанров, определим повторно в каждую из категорий;
2. У каждого фильма имеются оценки пользователей, исходя из них необходимо найти общую оценку кино-картины - среднее арифметическое между всеми оценками пользователей этого фильма (сложить все имеющиеся значения и разделить на их количество).
   Таким образом, в каждом жанре мы сможем составить рейтинг фильмов от наиболее понравившемуся потребителям до менее;
3. Предположим, что у клиента уже есть n-ое количество просмотренных фильмов на сервере. Проводим анализ жанров этих фильмов и делаем вывод о предпочтениях клиента. Сортируем релевантность жанров для определённого клиента;

Алгоритм решения задачи

4. Проанализировав клиентов платформы, формируем периодически изменяемые группы пользователей со схожими вкусами (совпадающая по значению и последовательности релевантность жанров различных пользователей);
5. Среди этого списка находим людей, которые смотрели те же фильмы, что и клиент, и которым они тоже понравились;
6. Смотрим, какие фильмы понравились тем пользователям;
7. Рекомендуем данные фильмы нашему клиенту с объяснением «Нравится тем, кто смотрел *название фильма, понравившегося клиенту, по которому мы нашли сходство с другими пользователями*».

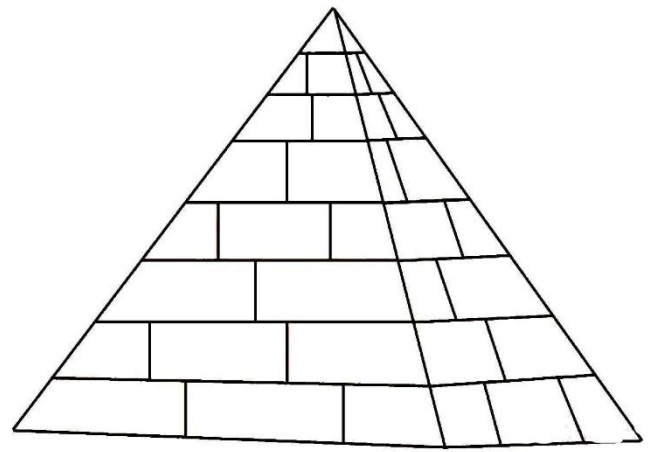
Дополнительное задание 1

Релевантность вывода фильмов напрямую зависит от нескольких параметров:

~ совпадение приоритетных жанров пользователя с жанрами рекомендуемого фильма;

~ среднее арифметическое поставленных оценок людьми группы;

~ количество человек из группы положительно оценившие фильм.



Дополнительное задание 2

Количество рекомендуемых фильмов (от 1 до 20) зависит от формулы:

p – процент от 20 (например, 90% - 18 фильмов)

k – количество просмотренных фильмов

r – поставленная оценка фильму пользователем в процентах от 5

z – доля пользователей из группы, которым понравился данный фильм

$$p = \frac{z * r}{k}$$

Дополнительное задание 2

Формула была выведена самостоятельно исходя из следующих факторов:

- Чем больше фильмов просмотрено пользователем, тем точнее рекомендации, соответственно, тем больше уменьшается количество фильмов, которые с большей вероятностью понравятся пользователю.
- Чем больше доля людей из группы, которым понравился данный фильм, тем с большей вероятностью данному пользователю будут интересны другие фильмы из будущей рекомендации. Условно, если фильм понравился каждому четвертому, то вероятность меньше, чем если фильм понравился каждому второму из группы.
- Чем выше оценка данного фильма (больше в процентном соотношении), тем с большей вероятностью человеку понравятся фильмы, которые высоко оценили пользователи группы.

Дополнительное задание 3

Объяснения, почему пользователю рекомендуют именно этот фильм, могут быть следующими:

- ✓ «Недавно вы смотрели фильм жанра *жанр рекомендуемого фильма*»;
- ✓ «Люди со схожими интересами часто смотрят»;
- ✓ «Люди со схожими интересами нравится»;
- ✓ «Фанаты *название фильма, просмотренного пользователем* рекомендуют».

Практическое решение задачи

1.Чтобы разделить список фильмов по категориям по жанру, необходимо использовать функцию switch (item.Type), которая поможет классифицировать фильмы для дальнейшей работы. После распределения к какому-либо жанру цикл не заканчивается, а проходит все варианты. Тем самым, мы получаем фильмы, распределенные на 21 группу.

2. Из таблицы ratings.cvs необходимо взять значения movield и ratings, используя также функцию switch (item.Type), разделить данные списки по номеру фильма, например, (список_фильм1). Переходим в списки по жанрам, в них для каждого фильма высчитываем среднее арифметическое его оценки. В списке нужно сложить все значения через sum (список_фильм1), посчитать количество данных через len(список_фильм1) и разделить первое значение на второе, получив среднюю оценку этого фильма. Далее в каждой категории фильмов составляем рейтинг через команду sorted(), используя аргумент key, который указывает, по какому ключу будет производиться сортировка и аргумент reverse=True, чтобы получить список в порядке убывания

Практическое решение задачи

3. Из аккаунта клиента мы берем список просмотренных им фильмов и проводим анализ жанров этих фильмов через цикл for, в котором будут содержаться несколько последовательных команд elif «фильм принадлежит к жанру *a*»; x - количество фильмов, принадлежащих к жанру a (изначально x=0), следовательно elif=Tru, тогда x=x+1. После получения популяризации жанров пользователя выставляем их по релевантности, то есть составляем рейтинг так же, как это производилось в пункте 2 (через команду sorted(), используя аргумент key, который указывает, по какому ключу будет производиться сортировка и аргумент reverse=True, чтобы получить список в порядке убывания).

4. Анализируя таким образом всех пользователей платформы, составляем список наиболее совпадающих вкусов пользователей с нашим клиентом через switch (item.Type).

Практическое решение задачи

5. Для обработки берем списки, сформированные во втором пункте. Приведем пример работы, взяв один из списков (список1) и сравним пользователей, проставивших оценки из него, с пользователями из списка list(set(список1).intersection(список2)). Полученную выборку пользователей мы выставляем по рейтингу (рейтинг3) оценок фильмов так же, как это производилось в пункте 2 и 3.

6. Выбираем первого человека из полученного рейтинга рейтинг 3[0] и из списка фильмов, которым он ставил оценки, выявляем фильм с наивысшей его оценкой max() (если таковых несколько, выбираем по релевантности жанров клиента).

7. Рекомендуем данный фильм нашему клиенту, выводя через print () объяснение нашей рекомендации.