Системы рекомендаций

Катышев Алексей
Школа анализа данных

Каков план?

- Рекомендательные системы
  - Что это?
  - Зачем это?
- Основные проблемы и методы их решения
- Типы рекомендательных систем
  - Плюсы и минусы каждого
- Оценка качества системы

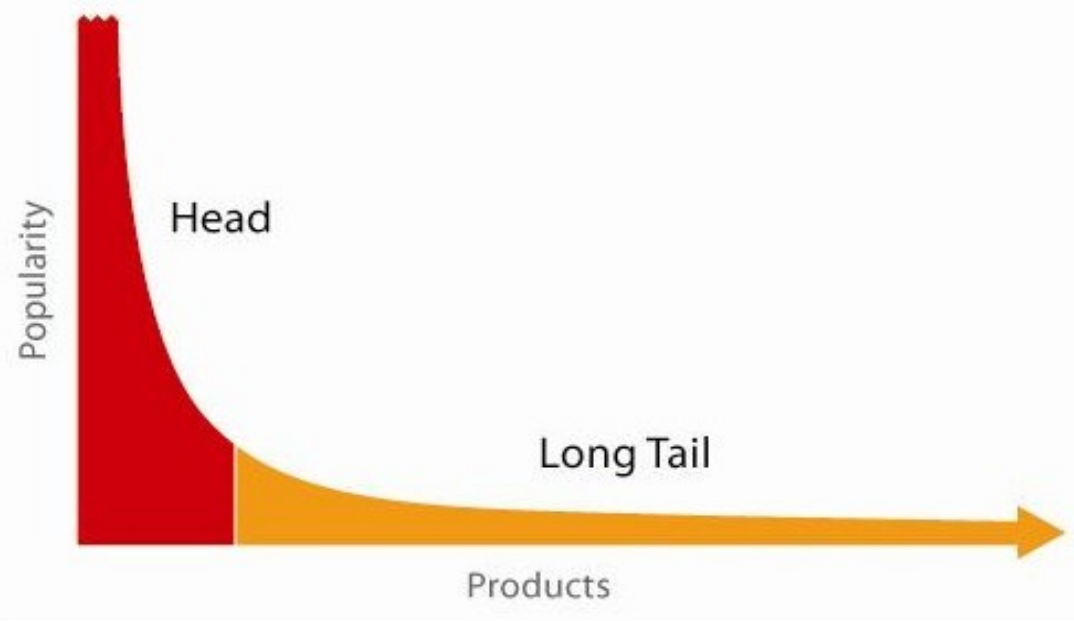
Пример

- Клиент А
  - Купил конфеты
  - Нравится шоколад
- Клиент В
  - Купил шоколад
  - Рекомендуем конфеты

Механизм

- Пользователь может
  - Искать объекты (новости, музыка, видео, …)
  - Покупать, скачивать, слушать, смотреть, …
  - Оценивать (0-5, 1-10, «Мне нравится ♥»)
- Система
  - Предлагает список рекомендованных объектов

Длинный хвост (Long tail)



- О большинстве товаров пользователь даже не знает
- С появление on-line рекомендации эффект длинного хвоста в некоторых областях значительно снизился

Виды рекомендаций

- Общие для всех клиентов (physical)
  - Выбор эксперта/директора
  - Популярное, Топ10, Топ50, …
  - Новое
  - «Смотрели только что» и т. п.
- Уникальным пользователям – уникальные рекомендации (on-line)
  - Рассмотрим далее

Модель

- Множество клиентов – C (customer, client)
- Множество объектов – I (item)
- Множество возможных рейтингов – R (rating)
  - [0, 1]
  - 0-5 (обычно звёзды)
  - 0-1 (мне нравится)

- Функция пользы u (utility) : $C \times I \rightarrow R$
  - насколько данный объект полезен данному клиенту.

Матрица пользы

	12 Chairs	Operation “Y”	Lock, Stock and Two …	Snatch
User A	9		4	5
User B			9	9
User C	10	9		
User D		10	9	

Проблемы

- Заполнение матрицы данными
  - Оценки от пользователей
  - Оценка действий пользователя
- На основании заполненных ячеек попытаться предугадать (высокие) значения в незаполненных ячейках (экстраполяция)
- Измерить качество экстраполяции

Заполнение матрицы

- Оценки от пользователей
  - В действительности работает не очень хорошо
  - Приходится мотивировать пользователей или
  - Упрощать рейтинговую систему
- Оценка действий пользователя
  - Покупка – однозначно высокий рейтинг
  - В остальном довольно сложно выявлять точные рейтинги

Вычисление пользы

- Проблемы:
  - Холодный старт в целом и для новых объектов
  - Матрица всё время очень разреженна
- Обычно метод вычисления пользы и определяет тип рекомендательной системы

Основные типы систем

- Content-based
  - Основывается на свойствах предлагаемых объектов
  - Похожесть объектов ≈ похожесть отдельных свойств
- Collaborative-Filtering
  - Рассматривает отношения пользователь-объект
  - Похожесть объектов ≈ насколько похожий рейтинг им дали одни и те же пользователи

Content-based, профиль объекта

- Профиль объекта – набор свойств
- Фильмы – главные актёры, режиссер, год, жанр, сценарист, …
- Новости, статьи, книги – набор важных или ключевых слов
  - Здесь часто применяется TF.IDF
- Картинки – набор тегов

Похожесть объектов

- На самом деле профиль объекта - бесконечный вектор из числовых значений (конечное количество которых ≠ 0)
- Как вариант - косинусное расстояние

Рейтинг	Звёзда 1	Звезда 2	Звёзда 3	Режиссёр 1	Режиссёр 2
2.5	1	1	0	0	1
4	0	1	1	1	0

- Похожесть ≈ 0.83

Профиль пользователя

- Как же вычислять полезность для определенного пользователя
- Ответ – профиль пользователя
  - Это некое среднее профилей тех объектов, которые оценил пользователь
  - Может быть взвешенное среднее
  - Но чаще из весов вычитают средний рейтинг
- Полезность – $u(c, i) = \cos(c, i)$

Content-based, выводы

Плюсы

- Нет проблем с холодным стартом всей системы
- Рекомендует новые и непопулярные объекты
- Каждому пользователю – свои рекомендации
- Может явно показать причину рекомендации

Минусы

- Холодный старт для новых пользователей
- Выявление свойств трудоёмко
- Интересы пользователей усредняются
- Не используются рейтинги других пользователей

Collaborative-Filtering, user-user

- Для пользователя с
  - Имеется некоторое количество его оценок
  - Хочется заполнить строчку матрицы пользы
- Находим множество пользователей D, похожих на с по выставленным оценкам
- Тогда оценки пользователя с по неоценённым объектам ≈ средняя оценка пользователей из D по этим объектам

Похожесть пользователей

- r_c – вектор рейтингов пользователя c
- Косинусное расстояние
  - $sim(x, y) = cos(r_x, r_y)$
- Можно нормализовать рейтинги
  - Вычесть из всех рейтингов средний рейтинг
- Есть разные варианты вычисления

Collaborative-Filtering, item-item

- Для объекта i аналогично ищутся похожие объекты
  - Заполняем матрицу пользы по вертикали
- Аналогичные метрики

- На практике доказано, что такой подход работает лучше

Пример

	12 Chairs	Operation "Y"	Lock, Stock and Two ...	Snatch
User A	9		4	5
User B			9	9
User C	10	9		
User D		10	9	

Что порекомендуем по "12 Chairs"?
cos(12 Chairs, Operation Y) ≈ 0.5
cos(12 Chairs, LockStock) ≈ 0.2
cos(12 Chairs, Snatch) ≈ 0.3

Collaborative-Filtering, выводы

Плюсы

- Не нужно выявлять свойства объектов
- Объекты могут быть любые

Минусы

- Холодный старт по всем фронтам
- Интересы пользователей усредняются

Вывод: Гибридные системы!

Оценка качества

3		5		8		6	
	4			3	7	4	
4	6	7			9		2
		3	5			5	
		8			2		
6			3	4			8

Оценка качества

3		5		8		6	
	4			3	7	4	
4	6	7			9		2
		3	5	?	?	?	?
		8		?	?	?	?
6			3	?	?	?	?

Оценка качества

- Берётся таблица с реальными данными
- Часть таблицы вырезается как тестовое множество
- Система пытается предугадать рейтинги в вырезанной части таблицы
- Вычисляется отклонение от реальных данных
  - Обычно это среднеквадратичное отклонение (RMSE)
  - Обычно вычисляется на топ10% рейтингов
  - Можно также считать разные дополнительные характеристики