Рекомендательные системы (часть 2)

Практикум на ЭВМ, весна 2018

Попов Артём Сергеевич

МГУ имени М. В. Ломоносова, факультет ВМК, кафедра ММП

29 марта 2018 г.

Введение

Латентные модели

Практические аспекты

Постановка задачи

Дано:

- U — множество субъектов (users/пользователи)
- I — множество объектов (items/товары/ресурсы)
- Y — множество возможных действий
- T — множество транзакций

$$T = \{(u_j, i_j, y_j) \mid u \in U, i \in I, y \in Y\}_{j=1}^{N}$$

- Работать со списком транзакций неудобно, заведём матрицу users-items (bag of items) $X \in \mathbb{R}^{|U| \times |I|}$:

$$X_{ui} = \sum_{j=1}^{N} [u_j = u][i_j = i] y_j, \text{ если } (u, i) \text{ встречалась в X}$$

Не все ячейки X заполнены, но не значит, что они нулевые!

Постановка задачи

Необходимо:

- Предсказать незаполненные ячейки X
- Посчитать близости $\rho(u, u')$, $\rho(i, i')$, $\rho(u, i)$
- Сформировать рекомендации для всех u (по всем i)

На прошлом занятии рассмотрели:

- Корреляционные методы
- Сведение задачи к задачам классификации/регрессии
- Факторизационные машины

Латентные модели

Идея: для каждого $u \in U$ построить вектор $p_u \in \mathbb{R}^g$, $g \ll |U|$
для каждого $i \in I$ построить вектор $q_i \in \mathbb{R}^h$, $h \ll |I|$
$\hat{X}_{ui} = F(p_u, q_i)$

Способы построения моделей:

- Жёсткая кластеризация ($p_{uc} = \mathbb{I}[u$ в кластере $c]$)
- Мягкая кластеризации (тематические модели)
  (p_{uc} — оценка принадлежности u кластеру c)
- Матричные разложения
  (p_u — столбцы/строки каких-то матриц после разложения)
- Специальные методы обучения представлений
  (p_u — вектор из модели skip-gram)
- End-to-end построение представлений

Матричные разложения: SVD

Хотим найти разложение матрицы X:

$$X = PQ^T \quad \text{или} \quad X = P\Sigma Q^T$$

p_u — строки матрицы P, q_i — строки матрицы Q

Использование сингулярного разложения (SVD):

$$\|X - P\Sigma Q^T\|^2 \to \min_{P,Q,\Sigma}$$

$$PP^T = I \qquad QQ^T = I$$

$$\Sigma = diag(\sigma_1, \ldots, \sigma_d), \quad \sigma_1 \geqslant \sigma_2 \geqslant \ldots \geqslant \sigma_d \geqslant 0$$

Можно записать так:

$$\sum_{u \in U} \sum_{i \in I} \left(x_{ui} - p_u^T \Sigma q_i\right)^2 \to \min_{P,Q,\Sigma}$$

SVD: недостатки и преимущества

- – Если X_{ui} неизвестно, мы будем считать его нулём
- – Все вектора одной сущности ортогональны между собой, сложно искать похожие
- – Неинтерпретируемые
- \+ Можно использовать Truncated SVD для уменьшения размерности
- \+ Много готовых реализаций

В качестве представлений можно также использовать:

$$p_u = P_u \Sigma \qquad q_i = Q_i$$

$$p_u = P_u \sqrt{\Sigma} \qquad q_i = \sqrt{\Sigma} Q_i$$

LFM, Latent Fator Model

Не будем учитывать неизвестные элементы как нулевые[1]:

$$\sum_{(u,i)\in T} \left(x_{ui} - p_u^T q_i\right)^2 \to \min_{P,Q}$$

Оптимизация модели с помощью метода SGD

Можно учитывать регуляризацию:

$$\sum_{(u,i)\in T} \left(x_{ui} - p_u^T q_i\right)^2 + \lambda \sum_{u\in U} \|p_u\|^2 + \mu \sum_{i\in I} \|q_i\|^2 \to \min_{P,Q}$$

Можно учитывать средний вклад пользователя и товара:

$$\sum_{(u,i)\in T} \left(x_{ui} - \hat{x}_u - \hat{x}_i - p_u^T q_i\right)^2 \to \min_{P,Q}$$

[1] Tacaks G., Pilaszy I., Nemeth B., Tikk D. Salable collaborative filtering approahes for large reommendation systems

LFM, Latent Fator Model

Можно делать неотрицательные компоненты:

$$\sum_{(u,i)\in T} \left(x_{ui} - p_u^T q_i\right)^2 \to \min_{p\geqslant 0,\ q\geqslant 0}$$

Обучение с помощью метода проекции градиента

Можно использовать функцию β (пример: σ, если $x_{ui} \in [0,1]$):

$$\sum_{(u,i)\in T} \left(x_{ui} - \beta(p_u^T q_i)\right)^2 \to \min_{P,Q}$$

Можно использовать вместо квадратичной ошибки любую другую, например hinge loss

ALS для LFMs

Можно использовать метод ALS для обучения

Идея: в точке оптимума L должно выполняться:

$$\frac{\partial L}{\partial p_u} = 0, \quad \frac{\partial L}{\partial q_i} = 0$$

Зафиксируем переменные Q:

$$\sum_{u \in U} \left(\|X_u - Q p_u\|^2 + \frac{\lambda}{2} \|p_u\|^2 \right) \to \min_P$$

ALS для LFMs

Можно использовать метод ALS для обучения

Идея: в точке оптимума L должно выполняться:

$$\frac{\partial L}{\partial p_u} = 0, \quad \frac{\partial L}{\partial q_i} = 0$$

Зафиксируем переменные Q:

$$\sum_{u \in U} \left(\|X_u - Q p_u\|^2 + \frac{\lambda}{2} \|p_u\|^2 \right) \to \min_P$$

Задача минимизации решается аналитически:

$$p_u = (Q^T Q + \lambda I)^{-1} Q^T X_u$$

Аналогично, можно решить задачу, зафиксировав P:

$$q_u = (P^T P + \mu I)^{-1} P^T X_i$$

ALS для LFMs

Будем решать итерационно:

- Зафиксировав Q, пересчитываем P
- Зафиксировав P, пересчитываем Q

Используем разложение Холецкого вместо обращения

- Хорошо и быстро работает
- Можно обобщить на случай неотрицательных разложения (положительная срезка $x \to \max(x, 0)$)
- Легко обновлять профили пользователей после прихода новых оценок

Интерпретация ALS

Распишем формулу принятия решения:

$$\hat{X}_{ui} = q_i^T p_u = q_i^T (Q^T Q + \lambda I)^{-1} Q^T X_u = \sum_{j \in I} q_i^T W q_j X_{uj}$$

$$W = (Q^T Q + \lambda I)^{-1} = LL^T$$

$$\hat{X}_{ui} = \sum_{j \in I} q_i^T L^T L q_j X_{uj} = \sum_{j \in I} (Lq_I)^T L q_j X_{uj}$$

На что похоже?

Интерпретация ALS

Распишем формулу принятия решения:

$$\hat{X}_{ui} = q_i^T p_u = q_i^T (Q^T Q + \lambda I)^{-1} Q^T X_u = \sum_{j \in I} q_i^T W q_j X_{uj}$$

$$W = (Q^T Q + \lambda I)^{-1} = LL^T$$

$$\hat{X}_{ui} = \sum_{j \in I} q_i^T L^T L q_j X_{uj} = \sum_{j \in I} (Lq_I)^T L q_j X_{uj}$$

На что похоже?

Корреляционные методы:

$$\hat{x}_{ui} = \bar{x}_i + \frac{\sum_{i' \in I_\alpha} \mathrm{sim}(i, i')(x_{ui'} - \bar{x}_{i'})}{\sum_{i' \in I_\alpha} \mathrm{sim}(i, i')}$$

Неявные и явные предпочтения

Явные (explicit):

- Проставил рейтинг фильму
- Лайкнул запись
- Написал рецензию на товар

Пользователь явно сообщает своё отношение к объекту

Неявные (implicit):

- Просмотрел страницу фильма
- Посетил страницу пользователя
- Купил товар в интернет-магазине

Если есть доступ к неявным предпочтениям, как их учитывать?
Можно ли строить латентные модели по бинарным данным?

Implicit ALS

Пусть x_{ui} — неявный фидбек

Пусть s_{ui} — показатель неявного интереса

$$s_{ui} = \begin{cases} 1, & x_{ui} \geqslant 0 \\ 0, & x_{ui} = 0 \end{cases}$$

Пусть c_{ui} — уровень доверия показателю s_{ui}

$$c_{ui} = 1 + \alpha x_{ui}$$

Модель Implicit ALS (оптимизация с помощью ALS):

$$\sum_{(u,i) \in T} c_{ui} \left(s_{ui} - p_u^T q_i \right)^2 \to \min_{P,Q}$$

Модели cbow и skip-gram

В модель cbow по словам контекста предсказывается слово:

$$\mathcal{L}(U, V) = \sum_{i=1}^{N} \log p(w_i | w_{i-k}^{i+k}) \to \max_{U,V}$$

$$p(w_i | d, w_{i-k}^{i+k}) = \underset{w_i \in W}{\operatorname{softmax}} \left\langle v_{w_i}, \sum_{\substack{j=-k \\ j \neq 0}}^{k} u_{w_{i+j}} \right\rangle$$

В модели skip-gram по слову предсказывается его контекст:

$$\mathcal{L}(U, V) = \sum_{i=1}^{N} \sum_{\substack{j=-k \\ j \neq 0}}^{k} \log p(w_{i+j} | w_i) \to \max_{V,U}$$

$$p(c|w) = \underset{c \in W}{\operatorname{softmax}} \langle v_c, u_w \rangle = \frac{\exp(\langle v_c, v_w \rangle)}{\sum_{c'} \exp(\langle v_{c'}, u_w \rangle)}$$

Модель paragraph2vec

paragraph2vec (PV-DBOW) — расширение моделей word2vec на представления документов

По словам из контекста и текущему документу предсказываем слово:

$$\mathcal{L}(U, V) = \sum_{d \in D} \sum_{i=1}^{N_d} \log p(w_i | d, w_{i-k}^{i+k}) \to \max_{U,V}$$

$$p(w_i | d, w_{i-k}^{i+k}) = \underset{w_i \in W}{\mathrm{softmax}} \left\langle v_{w_i}, \sum_{\substack{j=-k \\ j \neq 0}}^{k} u_{w_{i+j}} + u_d \right\rangle$$

Адаптация модели под рекомендации

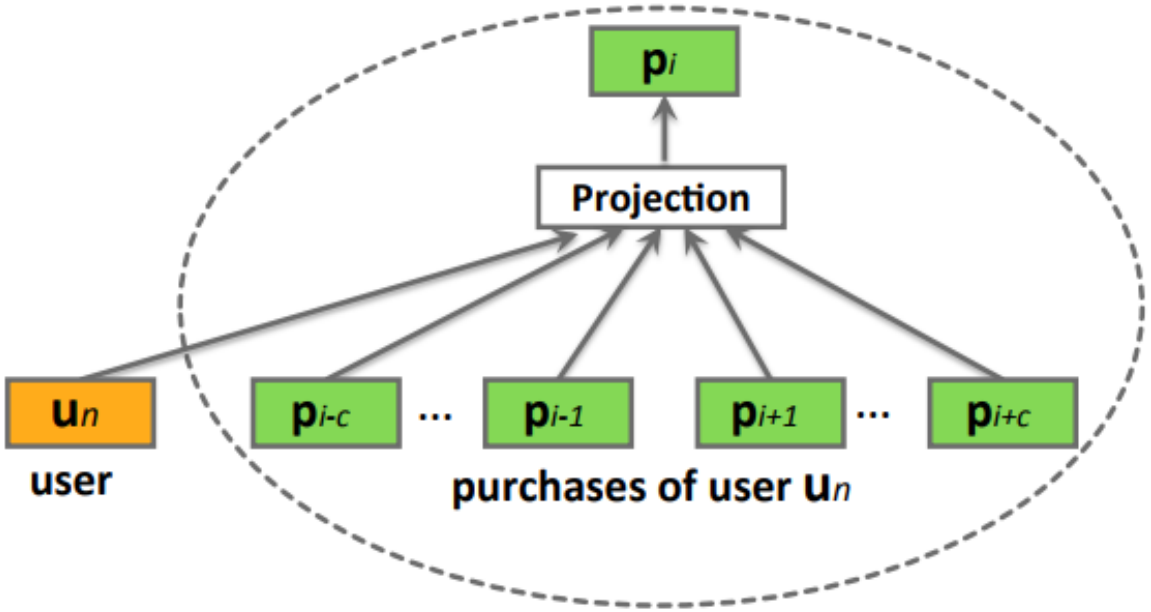
По пользователю предсказываем товары:

$$\mathcal{L}(U, V) = \sum_{(u,i)\in T} \log p(i|u) \to \max_{V,U}$$

Может хорошо работать в задачах, где у пользователя есть константные предпочтения:

- музыка
- фильмы

user2vec



По товарам пользователя предсказываем другие его товары (user2vec):

$I(u)$ — товары пользователя

$$\mathcal{L}(U, V) = \sum_{u \in U} \sum_{i=1}^{I(u)} \log p(i|u, sample \sim I(u) \setminus i) \to \max_{U,V}$$

Адаптация модели под рекомендации

Товары, которые покупаются одновременно, похожи (product2vec):

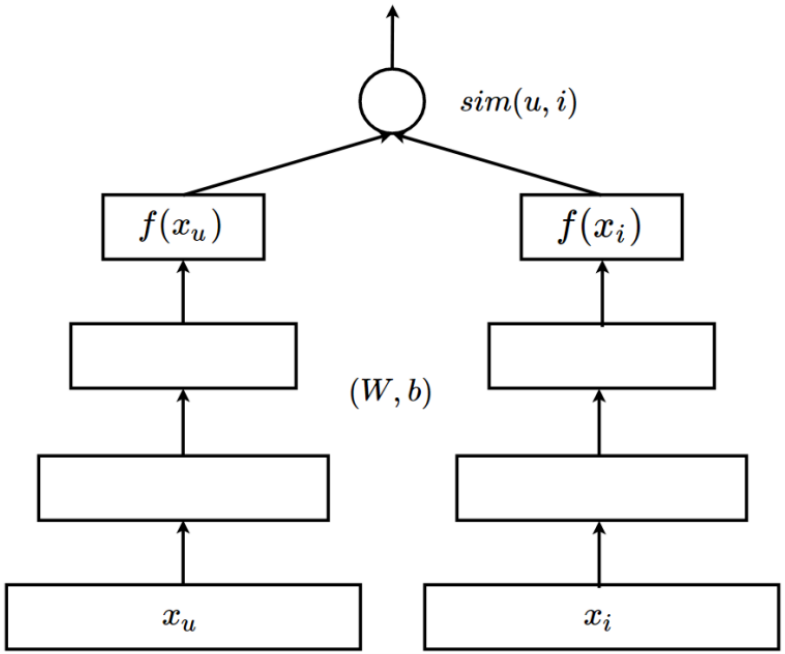
$I(i)$ — товары, которые покупались в связке

$$\mathcal{L}(U, V) = \sum_{i \in I} \sum_{j \in I(i)} \log p(j|i) \to \max_{V,U}$$

Может хорошо работать в задачах, где нет константных предпочтений:

- покупки в интернет-магазине

Deep semantic similarity based personalized recommendation

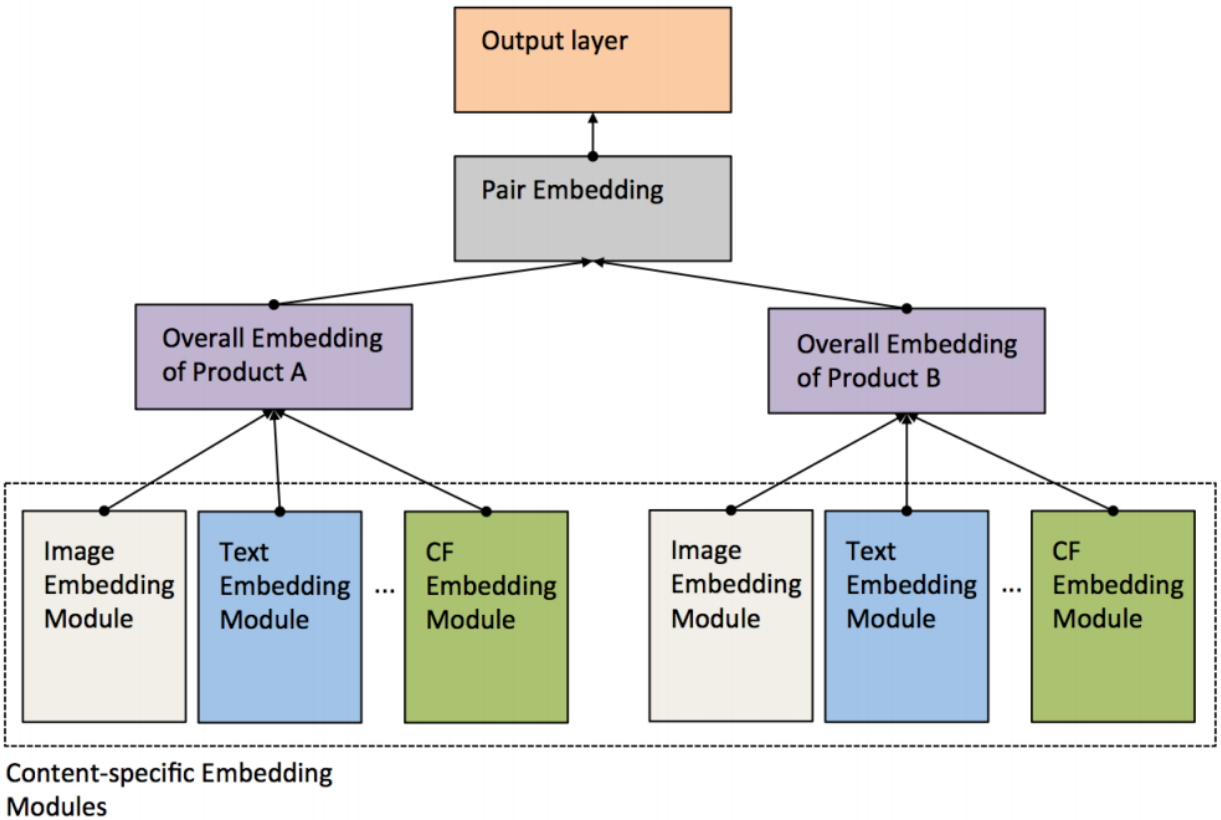


Учимся предсказывать близость пользователя и товара[2]

Обучаем с negative sampling

[2]Deep Learning based Recommender System: A Survey and New Perspectives

Комбинация разных факторов: content2vec



Обучаем с negative sampling

Ещё раз о метриках...

Метрика RMSE — не всегда хороша, т.к. задача точно предсказать оценку обычно не стоит

- подобрать рекомендации для пользователя
- отранжировать их по релевантности
- точные оценки не важны, важен порядок

Какие метрики лучше?
L_u — истинные предпочтения u
$R_u(k)$ — лучшие k рекомендаций

$$\text{precision@k} = \frac{|L_u \cap R_u(k)|}{|R_u(k)|} \qquad \text{recall@k} = \frac{|L_u \cap R_u(k)|}{|L_u|}$$

$$\text{hitrate@k} = [L_u \cap R_u(k) \neq \varnothing]$$

Другие метрики

- Разнообразие (diversity): например, число рекомендаций из разных категорий, или степень различия рекомендаций между сессиями пользователей
- Новизна (novelty): сколько среди рекомендаций объектов, новых для пользователей?
- Покрытие (coverage): доля объектов, которые хоть раз побывали в числе рекомендованных
- Прозорливость (serendipity): способность угадывать непопулярные предпочтения

Можно оптимизировать сумму функционалов

Ещё несколько фактов

- История действий пользователя построена с учётом существующих методов рекомендации
- Можно смотреть на результаты онлайн-метрик (полученная прибыль, полученное количество кликов)
- Хотелось бы, чтобы пользователю рекомендовалось то, что он не купил бы без рекомендаций
- A/Б тестирование для тестирования качества рекомендаций на практике