华东师范大学
EAST CHINA NORMAL UNIVERSITY

数据科学与工程算法基础

Algorithm Foundations of Data Science and Engineering

第十章 矩阵分解

课程提纲

Content

课程提纲

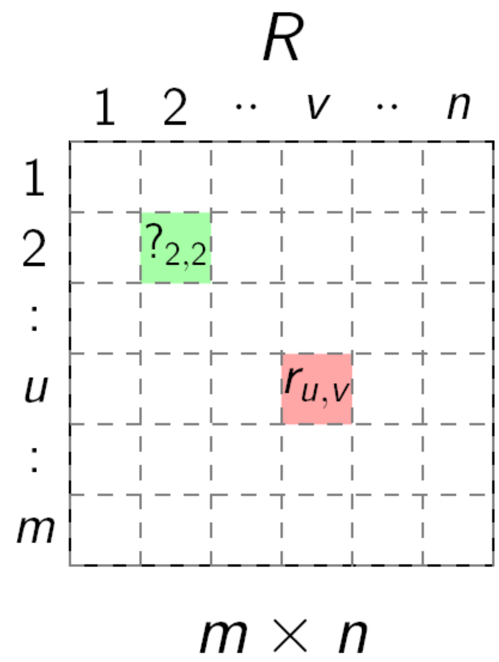
Content

带有缺失值的矩阵

User	Item	Rating
1	5	100
1	10	80
1	13	30
2	10	50
…	…	…
U	V	R
…	…	…

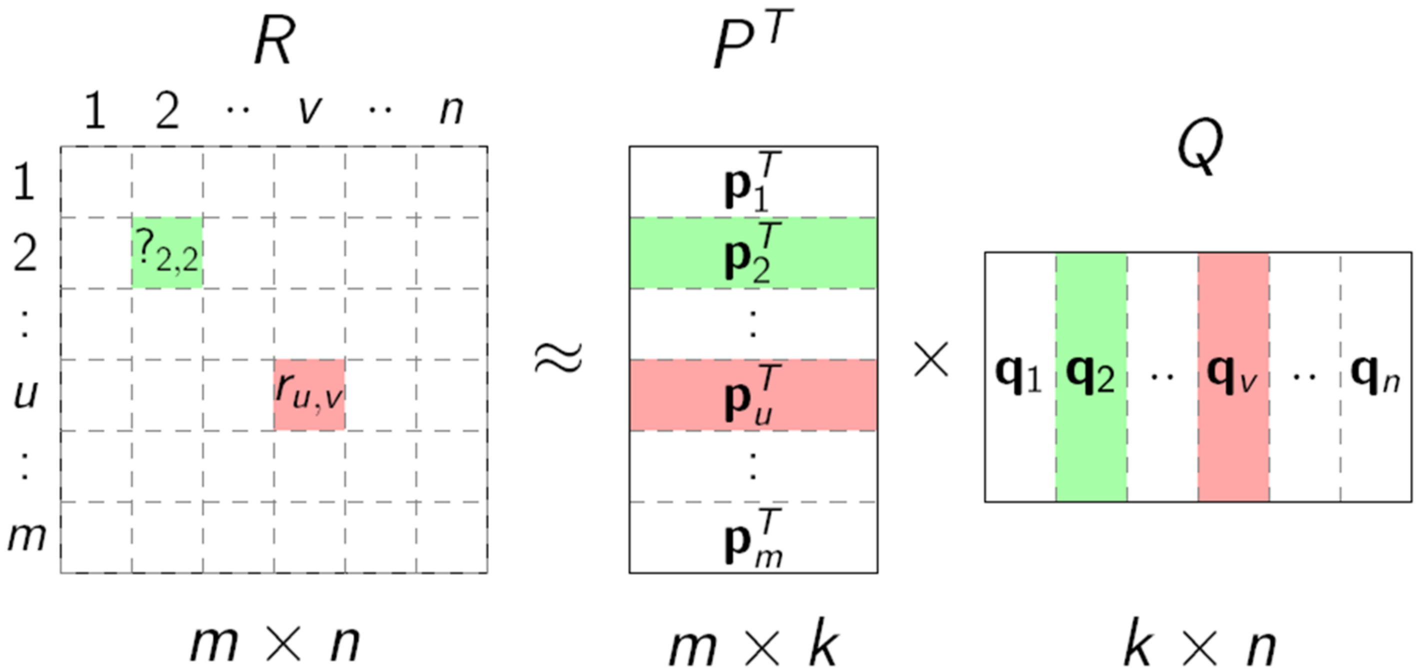
二分图

- 二分图可以对用户行为进行建模
- 二分图可以建模成带有缺失值的矩阵
  - 购买产品
  - 下载 APP
  - 听音乐
  - 看电影
  - 访问 POI
  - 回答问题
  - 贡献代码
  - …



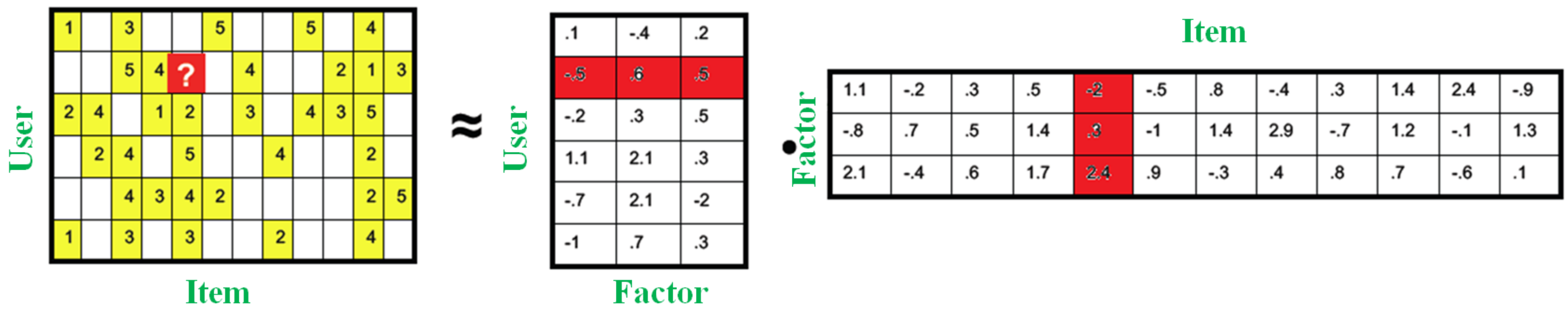
- m, n：用户数和项目数
- u, i：第 u 个用户和第 i 个项目
- r_{ui}：第 u 个用户对第 i 个项目的评分

缺失值预测



- k：潜在维数
- $r_{ui} = \mathbf{p}_u^\top \mathbf{q}_i$
- $r_{22} = \mathbf{p}_2^\top \mathbf{q}_2$
- 该方法称为矩阵分解，即 $R \approx P^\top \cdot Q$
- 应用
  - 推荐，相似性查询，数据可视化

问题建模



- 给定矩阵 P 和 Q，重构用户 u 对项目 i 的评分

$$\widehat{r_{ui}} = \mathbf{p}_u^\top \mathbf{q}_i = \sum_{j=1}^{k} p_{uj} q_{ji}$$

- 重构误差为 $e_{ui} = r_{ui} - \widehat{r_{ui}}$

重构误差最小

- 给定矩阵 P 和 Q，基于F–范数的重构误差可以表示为
  $$J = \frac{1}{2}\|R - P^{\top}Q\|_F^2$$
- 为实现高精度的矩阵分解，希望重构误差最小，即
  Minimize $J = \frac{1}{2}\|R - P^{\top}Q\|_F^2$
- 求解一个连续优化的问题
  - 凸函数优化
    - 任何局部最优解即为全局最优解
    - 梯度法、拟牛顿法、共轭梯度法等算法都可以收敛到全局最优解
  - 非凸优化：大多数情形下都是局部最优解，而求解全局最优的算法复杂度是指数级的（NP–Hard）

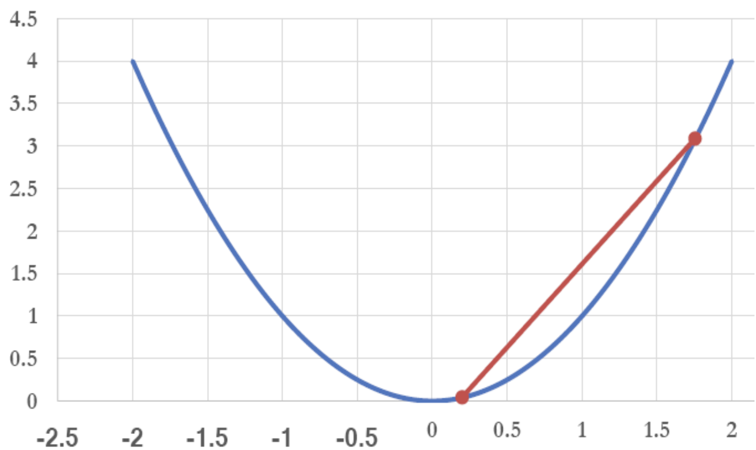
课程提纲

Content

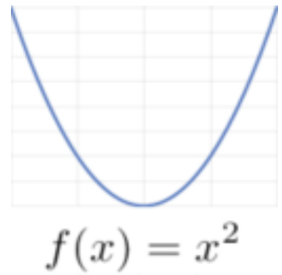
凸函数的定义

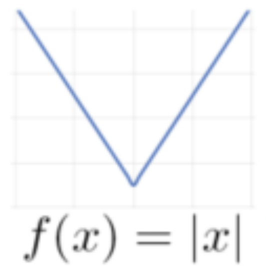
- 对于任意 $\alpha \in [0,1]$，$f(x)$ 是一个凸函数当且仅当

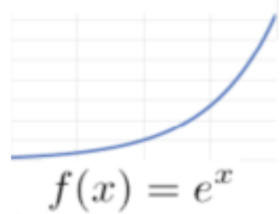
$$f(\alpha x + (1-\alpha)y) \le \alpha f(x) + (1-\alpha)f(y)$$



凸函数图像上任意两点间的连接段一定位于图像的上方，它的局部极小值就是全局最小值





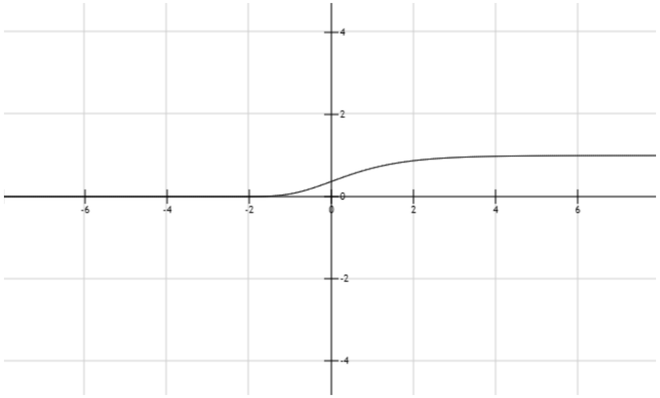


- 凸函数的例子
  - $f(x) = x^2$
  - $f(x) = |x|$
  - $f(x) = e^x$

凸函数的性质

- 凸函数的非负线性组合 $h(x) = af(x) + bg(x)$ 还是凸函数
- 凸函数的仿射标量 $h(\mathbf{x}) = f(A\mathbf{x} + b)$ 仍然是凸函数
- 凸函数的复合不一定是凸函数，除非是单调非减函数 $h(x) = f(g(x))$（类似神经网络）

- 例如 $f(x) = e^{-x}, x \in [0,1]$，$f$ 是凸函数，但 $f \circ f = f(f(x))$ 是凹函数



梯度下降法

- 考虑无约束的平滑凸优化问题 $\min_x f(x)$
  - 即 f 是凸函数且可微
  - 定义 $x^* = \arg\min_x f(x)$，其中 x^* 表示最优解

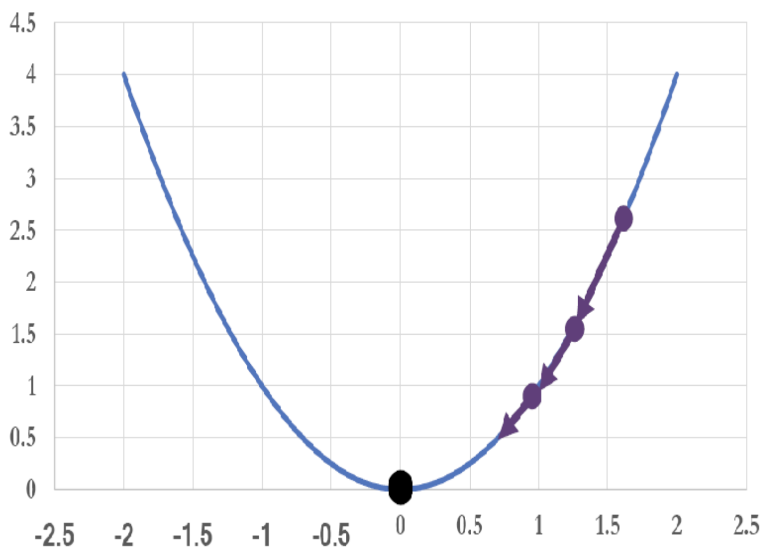
梯度下降算法

1. 选择初始点 $\mathbf{x}^{(0)} \in \mathbb{R}^n$

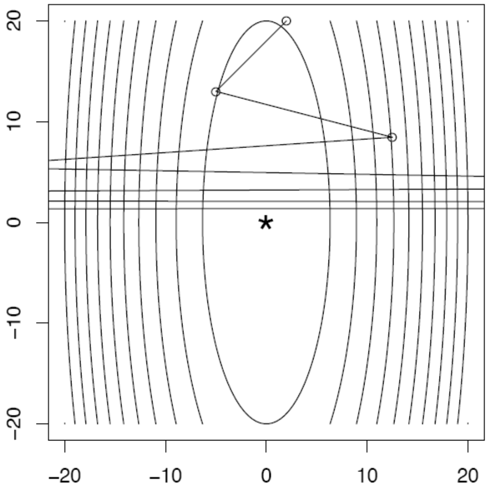
2. 对于 $k = 1,2,\ldots$

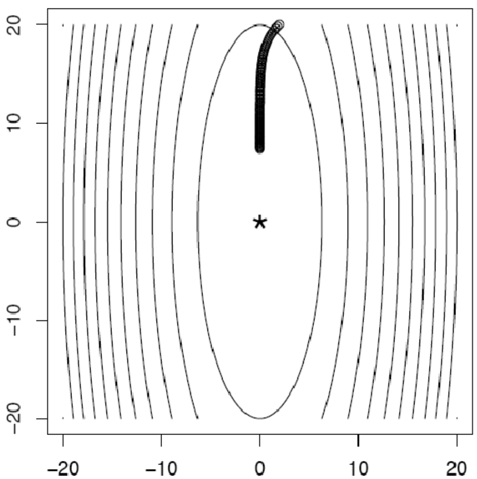
3. $\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} - \lambda \cdot \nabla f(\mathbf{x}^{(k-1)})$

其中 λ 是步长



步长（学习率）选择





- 考虑函数 $f(\mathbf{x}) = (10x_1^2 + x_2^2)/2$
- 取较大的 λ 值，算法可能收敛很慢
- 取较小的 λ 值，算法也可能缓慢收敛
- 可能需要自适应地调整步长
  - AdaGrad：每轮迭代根据历史梯度调整
  - RMSProp：衰减的历史梯度方法
  - Adam：结合蒙特卡洛和 RMSProp

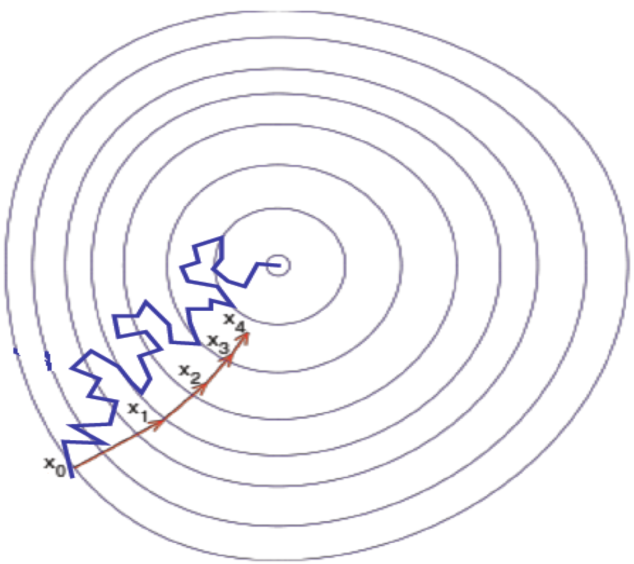
梯度下降问题

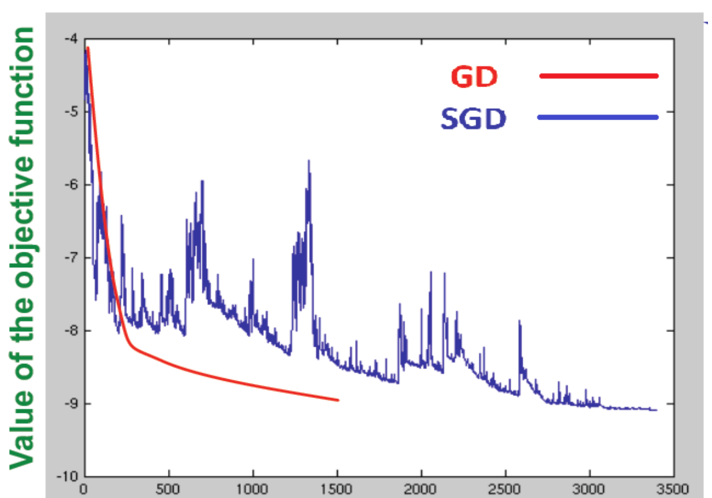
- 在许多机器学习任务中，损失函数可以写成
  $h(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}; y_i)$
- 其中 f 为损失函数，y_i 是训练数据，$\mathbf{x}$ 为待估参数
- 对大规模优化，计算梯度的时间复杂度为 $O(N)$
- $\nabla h(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} \nabla f(\mathbf{x}; y_i)$
  - 当训练数据集特别大时，梯度计算消耗很大
  - 如何能够降低梯度更新的计算开销?
    - 小批量梯度下降
    - 随机梯度下降

随机梯度下降

- 是否可以仅使用少量样本去更新梯度呢?
- 随机梯度下降：每次迭代仅随机选择一个样本计算梯度，而不用计算所有样本梯度的平均值
  - $\mathbf{x}_{t+1} = \mathbf{x}_t - \lambda \nabla f(\mathbf{x}; y_{\tilde{t}_i})$，其中 $y_{\tilde{t}_i}$ 是一个从数据集中随机选择的样本
  - 计算速度非常快
- 为什么随机梯度下降是可行的?
  - 根据弱大数据定律
  - $E(\mathbf{x}_{t+1}) = E(\mathbf{x}_t) - \lambda E(\nabla f(\mathbf{x}; y_{\tilde{t}_i})) = E(\mathbf{x}_t) - \lambda \frac{1}{N} E(\sum_{i=1}^{N} \nabla f(\mathbf{x}; y_i))$

SGD vs. GD





- GD目标函数的值在每一步都有提升
- SGD目标函数同样在提升，但收敛曲线震荡很大。
- GD收敛所需的步骤更少，但计算所需的时间却更长。
- 实际上，SGD快得多！

课程提纲

Content

1 算法引入

2 梯度下降法

3 矩阵分解

矩阵分解的定义

- 给定用户集合 U 和项目集合 D，以及由用户对项目的评分所构成的评分矩阵 $R \in \mathbb{R}^{|U| \times |D|}$，矩阵分解旨在找到两个矩阵 $P \in \mathbb{R}^{K \times |U|}$ 和 $Q \in \mathbb{R}^{K \times |D|}$，使得 $R \approx P^\top Q = \widehat{R}$，其中 K 表示潜在特征的维数
  - P 的每一行表示一个用户在低维空间中的投影向量
  - Q 的每一行表示一个项目在低维空间中的投影向量
  - R 是一个缺失矩阵
  - 通过最小化已观测评分的重构误差，可以得到 P 和 Q

矩阵分解问题定义

$$\min_{P,Q} J(R;P;Q) = \frac{1}{2} \sum_{(u,i)\in\mathscr{R}} (r_{ui} - \mathbf{p}_u^\top \mathbf{q}_i)^2$$

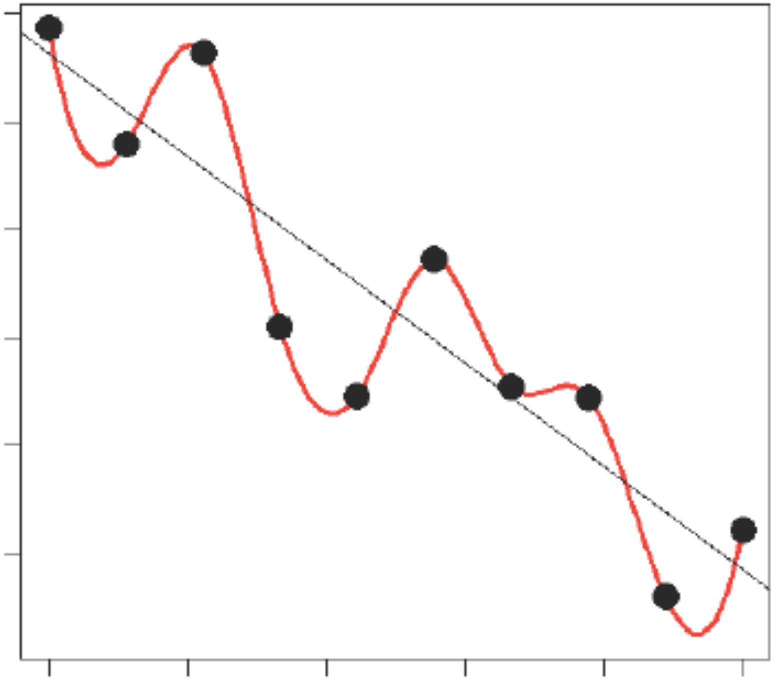
- 其中 r_{ui} 表示用户 u 对项目 i 的真实评分，$\widehat{r_{ui}} = \mathbf{p}_u^\top \mathbf{q}_i$ 是用户 u 对项目 i 的预测评分，$\mathscr{R}$ 为评分矩阵 R 中观测到的评分，即 $\mathscr{R} = \{(u,i) \mid r_{u,i} \neq 0\}$
- 对每个评分项，通过以下公式计算预测评分与实际评分之间的误差：$e_{ui} = r_{ui} - \widehat{r_{ui}} = r_{ui} - \sum_{j=1}^{k} p_{uj} q_{ji}$
- 因此，$J = \frac{1}{2} \sum_{(u,i)\in\mathscr{R}} e_{ui}^2$

梯度下降算法

- 首先初始化矩阵 $P^{(0)}$ 和 $Q^{(0)}$
- 对参数求偏导，得 $\dfrac{\partial e_{ui}^2}{\partial p_{uj}} = -2e_{ui}q_{ji}$ 和 $\dfrac{\partial e_{ui}^2}{\partial q_{ji}} = -2e_{ui}p_{uj}$
- 更新规则
  - $p_{uj}^{(t+1)} \leftarrow p_{uj}^{(t)} + \epsilon \sum_{i:(u,i)\in\mathscr{R}} e_{ui}^{(t)} q_{ji}^{(t)}$ 和 $q_{ji}^{(t+1)} \leftarrow q_{ji}^{(t)} + \epsilon \sum_{u:(u,i)\in\mathscr{R}} e_{ui}^{(t)} p_{uj}^{(t)}$
  - 其中 $e_{ui}^{(t)} = r_{ui} - \mathbf{p}_u^{(t)\top}\mathbf{q}_i^{(t)}$，$\epsilon$ 为学习率
- 以矩阵方式表达
  - $P^{(t+1)} \leftarrow P^{(t)} + \epsilon E^{(t)} Q^{(t)}$
  - $Q^{(t+1)} \leftarrow Q^{(t)} + \epsilon E^{(t)\top} P^{(t)}$

只要算不缺失的就可以了

正则化



- 已观察到的真实评分集合 $\mathcal{R}$ 很小，可能导致过拟合
- 正则化是解决该问题的常用方法

$$\min_{P,Q} \frac{1}{2}\Big[\sum_{(u,i)\in\mathcal{R}} (r_{ui} - \mathbf{p}_u^\top \mathbf{q}_i)^2 + \lambda(\|P\|_F^2 + \|Q\|_F^2)\Big]$$

带正则化项的矩阵分解

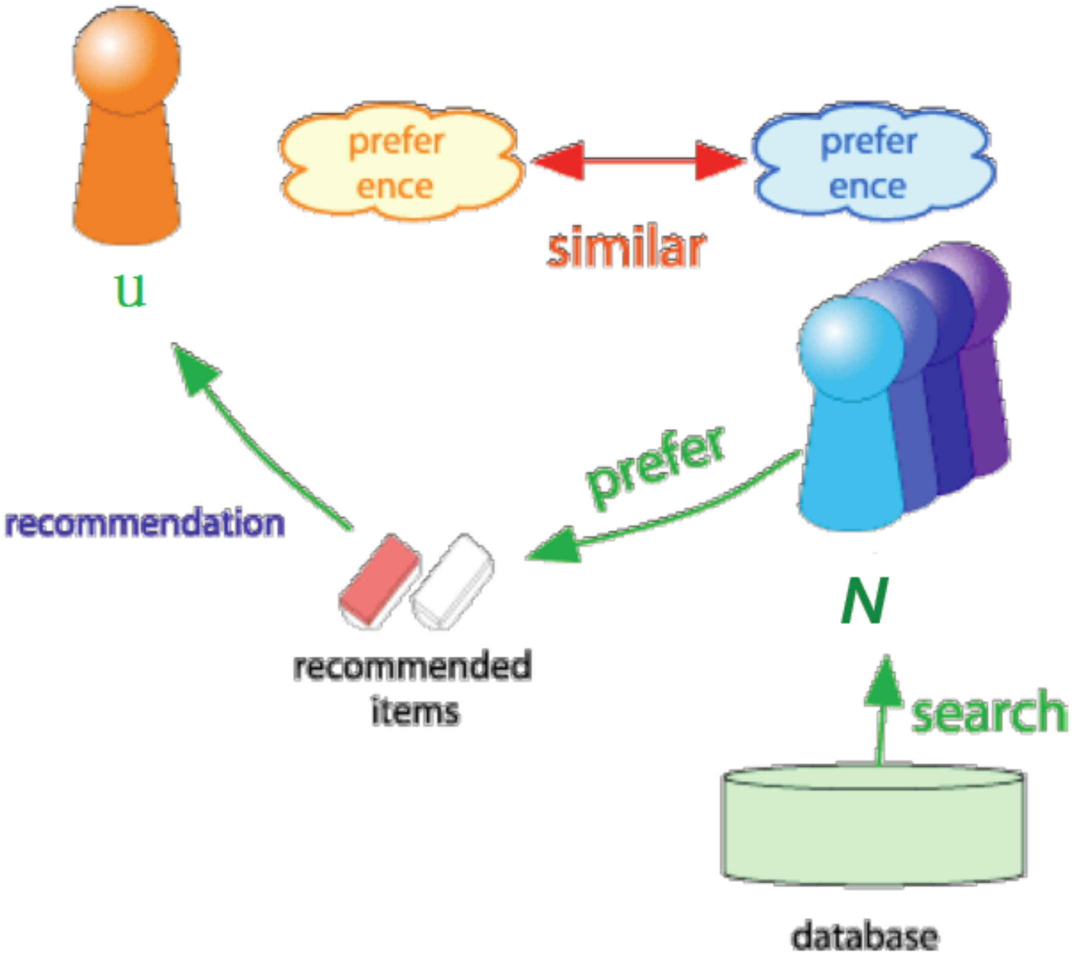
- 使用梯度下降算法的更新规则
  - $p_{uj}^{(t+1)} \leftarrow p_{uj}^{(t)} + \epsilon\left(\sum_{i:(u,i)\in\mathscr{R}} e_{ui}^{(t)} q_{ji}^{(t)} - \lambda p_{uj}^{(t)}\right)$
  - $q_{ji}^{(t+1)} \leftarrow q_{ji}^{(t)} + \epsilon\left(\sum_{u:(u,i)\in\mathscr{R}} e_{ui}^{(t)} p_{uj}^{(t)} - \lambda q_{ji}^{(t)}\right)$
  - 其中 $e_{ui}^{(t)} = r_{ui} - \mathbf{p}_u^{(t)\top} \mathbf{q}_i^{(t)}$
- 使用随机梯度下降算法的更新规则
  - $p_{uj}^{(t+1)} \leftarrow p_{uj}^{(t)} + \epsilon\left(e_{ui}^{(t)} q_{ji}^{(t)} - \lambda p_{uj}^{(t)}\right)$
  - $q_{ji}^{(t+1)} \leftarrow q_{ji}^{(t)} + \epsilon\left(e_{ui}^{(t)} p_{uj}^{(t)} - \lambda q_{ji}^{(t)}\right)$
  - 其中 $e_{ui}^{(t)} = r_{ui} - \mathbf{p}_u^{(t)\top} \mathbf{q}_i^{(t)}$

考虑用户和项目的偏置信息

$$\min_{P,Q,\mathbf{b},\mathbf{d}} \frac{1}{2}\Big[\sum_{(u,i)\in\mathscr{R}} (r_{ui} - \mu - b_u - d_i - \mathbf{p}_u^\top \mathbf{q}_i)^2 + \lambda(\|P\|_F^2 + \|Q\|_F^2 + \|\mathbf{b}\|_2^2 + \|\mathbf{d}\|_2^2)\Big]$$

- 可以增加因子矩阵的大小来合并这些偏差变量，而不用为用户和项目设置单独的偏置变量
    - $p_{u(k+1)} = b_u$，$p_{u(k+2)} = 1$
    - $q_{i(k+1)} = 1$，$q_{i(k+2)} = d_i$
- $\min_{\tilde{P},\tilde{Q}} \frac{1}{2}\Big[\sum_{(u,i)\in\mathscr{R}} (r_{ui} - \mu - \tilde{\mathbf{p}}_u^\top \tilde{\mathbf{q}}_i)^2 + \lambda(\|\tilde{P}\|_F^2 + \|\tilde{Q}\|_F^2)\Big]$，其中 $\tilde{P}$ 矩阵的 (k+2) 列和 $\tilde{Q}$ 矩阵的 (k+1) 列都仅包含 1

协同过滤



- 协同过滤：物以类聚、人以群分
- 考虑用户 u
  - 查找评分与 u “相似”的用户的集合 S
  - 基于集合 S 中的用户评分估计 u 对项目的评分
  - 如何评价“相似”用户呢?

协同过滤的矩阵分解法

$$\min_{P,Q} \frac{1}{2}\Big[\sum_{(u,i)\in\mathscr{R}} (r_{ui} - \mathbf{p}_u^\top \mathbf{q}_i)^2 + \gamma \sum_{u,v\in\mathscr{U}} s_{uv}\|\mathbf{p}_u - \mathbf{p}_v\|_2^2\Big]$$

- r_{ui} 表示用户 u 对项目 i 的真实评分，s_{uv} 表示用户 u 和用户v 的偏好相似度
- 随机梯度下降
  - $p_{uj}^{(t+1)} \leftarrow p_{uj}^{(t)} + \epsilon\Big(e_{ui}^{(t)} q_{ji}^{(t)} - \gamma s_{uv}(p_{uj}^{(t)} - p_{vj}^{(t)})\Big)$
  - $q_{ji}^{(t+1)} \leftarrow q_{ji}^{(t)} + \epsilon e_{ui}^{(t)} p_{uj}^{(t)}$
  - 其中 $e_{ui}^{(t)} = r_{ui} - \mathbf{p}_u^{(t)\top} \mathbf{q}_i^{(t)}$

非负矩阵分解定义

- 定义优化问题 $\min\limits_{W,H} \frac{1}{2}\|V - WH^\top\|_2^2$

  s.t. $W \geq 0, H \geq 0$
- 有一些现实场景中，用户可能表达出“喜欢”一个商品，但没有表达“不喜欢”，例如浏览或购买行为，网页点击和 Facebook 喜欢等
- 其他一些应用中，实体的许多属性都是非负的
  - 图像像素
  - 文档中单词出现的频率
  - 股票价格

非负矩阵分解算法

- 非负矩阵分解满足存在性和唯一性
- 课本上的基于梯度下降的非负矩阵分解算法，有兴趣的同学可以参照课本和参考文献课后自学
- 非负矩阵分解问题是NP-hard的
- 因此，基于梯度下降的算法只能保证收敛到局部最优解

本章小结

- 矩阵分解
  - 梯度下降算法
  - 正则化
  - 协同过滤
  - 非负矩阵分解
- 矩阵分解算法能够保证全局最优解吗?
- 应用
  - 个性化推荐
  - 相似性查询
  - 数据可视化