



# BPR: Bayesian Personalized Ranking from Implicit Feedback

`/* { for beginners } */`

**Steffen Rendle, Christoph Freudenthaler, Zeno Gantner and Lars Schmidt-Thieme**  
{srendle, freudenthaler, gantner, schmidt-thieme}@ismll.de  
Machine Learning Lab, University of Hildesheim  
Marienburger Platz 22, 31141 Hildesheim, Germany

2009 UAI

王润生  
July 11, 2019

# Matrix Factorization

	$i_1$	$i_2$	$i_3$	$i_4$	$i_5$	$i_6$	$i_7$	$i_8$
$u_1$	5	2		3		4		
$u_2$	4	3			5			
$u_3$	4		2				2	4
$u_4$								
$u_5$	5	1	2		4	3		
$u_6$	4	3		2	4		3	5

User-Item Matrix

$$R = U^T V$$

- 致力于单个项目的得分  $\hat{r}_{ui}$
- 没有对排名进行直接的优化

$$U = \begin{bmatrix} 1.55 & 1.22 & 0.37 & 0.81 & 0.62 & -0.01 \\ 0.36 & 0.91 & 1.21 & 0.39 & 1.10 & 0.25 \\ 0.59 & 0.20 & 0.14 & 0.83 & 0.27 & 1.51 \\ 0.39 & 1.33 & -0.43 & 0.70 & -0.90 & 0.68 \\ 1.05 & 0.11 & 0.17 & 1.18 & 1.81 & 0.40 \end{bmatrix} \quad V = \begin{bmatrix} 1.00 & -0.05 & -0.24 & 0.26 & 1.28 & 0.54 & -0.31 & 0.52 \\ 0.19 & -0.86 & -0.72 & 0.05 & 0.68 & 0.02 & -0.61 & 0.70 \\ 0.49 & 0.09 & -0.05 & -0.62 & 0.12 & 0.08 & 0.02 & 1.60 \\ -0.40 & 0.70 & 0.27 & -0.27 & 0.99 & 0.44 & 0.39 & 0.74 \\ 1.49 & -1.00 & 0.06 & 0.05 & 0.23 & 0.01 & -0.36 & 0.80 \end{bmatrix}$$

$$\begin{bmatrix} 1.55 \\ 0.36 \\ 0.59 \\ 0.39 \\ 1.05 \end{bmatrix} \begin{bmatrix} 1.00 \\ 0.19 \\ 0.49 \\ -0.40 \\ 1.49 \end{bmatrix}$$

$$= 1.55*1 + 0.36*0.19 + 0.59*0.49 - 0.39*0.4 + 1.05*1.49 = 3.316$$

# BPR-Opt

- 这篇论文，从贝叶斯分析入手，提出了一个通用的优化标准 BPR-Opt，用于个性化排序
- 隐反馈（在线商城的购买记录、音乐视频网站的播放记录等） $S \subseteq U \times I$  如图 1 所示
- 仅 positive observation(+) 可用
- non-observed user-item pairs(?)
  - negative feedback  
(the user is not interested in buying the item)
  - missing values  
(the user might want to buy the item in the future)

	$i_1$	$i_2$	$i_3$	$i_4$
$u_1$	?	+	+	?
$u_2$	+	?	?	+
$u_3$	+	+	?	?
$u_4$	?	?	+	+
$u_5$	?	?	+	?

图1

# BPR-Opt

- 同时包含 positive pairs、negative pairs、missing values pairs

- $(u, i_2, j_1)$  为 positive pair(+)
- $(u, i_1, j_2)$  为 negative pair(-)
- $(u, i_4, j_1)$  为 missing value pair(?)

	$i_1$	$i_2$	$i_3$	$i_4$
$u_1$	?	+	+	?
$u_2$	+	?	?	+
$u_3$	+	+	?	?
$u_4$	?	?	+	+
$u_5$	?	?	+	?

图1

- Item Recommendation  $\succ_u \subset I^2$

- totality  $\forall i, j \in I: i \neq j \Rightarrow i \succ_u j \vee j \succ_u i$
- antisymmetry  $\forall i, j \in I: i \succ_u j \wedge j \succ_u i \Rightarrow i = j$
- transitivity  $\forall i, j, k \in I: i \succ_u j \wedge j \succ_u k \Rightarrow i \succ_u k$

$u_1: i \succ_{u_1} j$	$i_1$	$i_2$	$i_3$	$i_4$
$j_1$		+	+	?
$j_2$	-		?	-
$j_3$	-	?		-
$j_4$	?	+	+	

图2

- 常规方法

# BPR-Opt

- 在这篇论文中，为了更好反映问题的本身（项目排序），作者使用 item pairs(项目对)作为训练数据并且致力于优化正确的 item pairs 的排名，而不是单个项目的得分
- 即，利用隐反馈 $S$ ，作者致力于为每个用户重构序列  $>_u$ ，如果用户 $u$ 和项目 $i$ 有交互  $(u,i) \in S$ ，那么假设该用户更喜欢这个项目相对于其他的 non-observed items  $j$ ，有  $i >_u j$ ，表示为三元组  $(u,i,j)$

$$D_S := \{(u, i, j) | i \in I_u^+ \wedge j \in I \setminus I_u^+\}$$

$$\text{where } I_u^+ := \{i \in I : (u, i) \in S\}$$

# BPR-Opt

- 具体地，贝叶斯个性化排序就是要最大化下面的后验概率  $p(\Theta | >_u) \propto p(>_u | \Theta) p(\Theta)$
- 假设用户间行为独立，且用户对每个项目对的排序也是独立的，因此似然函数  $p(>_u | \Theta)$  可以被重写为

$$\prod_{u \in U} p(>_u | \Theta) = \prod_{(u,i,j) \in D_S} p(i >_u j | \Theta)$$

- 而用户相对项目  $j$  更喜欢项目  $i$  的个体概率可以定义为

$$p(i >_u j | \Theta) := \sigma(\hat{x}_{uij}(\Theta))$$

其中  $\sigma$  是 logistic sigmoid function:

$$\sigma(x) := \frac{1}{1+e^{-x}}$$

这里的  $\hat{x}_{uij}(\Theta)$  可以是任意一个关于模型参数  $\Theta$ ，可以捕获用户  $u$  和项目  $i, j$  之间的关系的实值函数。比如我们可以定义为：

$$\hat{x}_{uij} := \hat{x}_{ui} - \hat{x}_{uj}$$

# BPR-Opt

- 关于先验分布，作者引入了一个比较通用的均值为 0，方差为协方差矩阵  $\Sigma_{\Theta}$  的正态分布  $p(\Theta) \sim N(0, \Sigma_{\Theta})$
- 这个时候，我们的贝叶斯个性化排序框架 BPR-OPT 就已经完成了

$$\begin{aligned}\text{BPR-OPT} &:= \ln p(\Theta | \succ_u) \\ &= \ln p(\succ_u | \Theta) p(\Theta) \\ &= \ln \prod_{(u,i,j) \in D_S} \sigma(\hat{x}_{uij}) p(\Theta) \\ &= \sum_{(u,i,j) \in D_S} \ln \sigma(\hat{x}_{uij}) + \ln p(\Theta) \\ &= \sum_{(u,i,j) \in D_S} \ln \sigma(\hat{x}_{uij}) - \lambda_{\Theta} \|\Theta\|^2\end{aligned}$$

可以看出，当  $\Theta$  的先验分布是上面说的正态分布的时候，有着类似正则化项的效果

$$\begin{aligned}\ln p(\Theta) &\propto -\lambda_{\Theta} \|\Theta\|^2 \\ \ln p(\Theta) &:= \ln N(\Theta | 0, \lambda_{\Theta} I) \\ &= \ln \frac{1}{\sqrt{2\pi\lambda_{\Theta} I}} \exp\left(-\frac{\|\Theta\|^2}{2\lambda_{\Theta} I}\right) \\ &= -\frac{1}{2} \ln(2\pi\lambda_{\Theta} I) - \frac{\|\Theta\|^2}{2\lambda_{\Theta} I} \\ &= -\frac{1}{2} \ln(\lambda_{\Theta} I) - \frac{\|\Theta\|^2}{2\lambda_{\Theta} I} + C \\ &\propto -\lambda_{\Theta} \|\Theta\|^2\end{aligned}$$

# BPR-Learn

- 贝叶斯个性化排序需要最大化后验概率，为此我们可以用梯度上升的方法求解

$$\begin{aligned}\frac{\partial \text{BPR} - \text{OPT}}{\partial \Theta} &= \sum_{(u,i,j) \in D_S} \frac{\partial}{\partial \Theta} \ln \sigma(\hat{x}_{uij}) - \lambda_{\Theta} \frac{\partial}{\partial \Theta} \|\Theta\|^2 \\ &\propto \sum_{(u,i,j) \in D_S} \frac{1}{\sigma(\hat{x}_{uij})} \cdot \frac{0 - (-e^{-\hat{x}_{uij}})}{(1 + e^{-\hat{x}_{uij}})^2} \cdot \frac{\partial}{\partial \Theta} \hat{x}_{uij} - \lambda_{\Theta} \Theta \\ &\propto \sum_{(u,i,j) \in D_S} \frac{e^{-\hat{x}_{uij}}}{1 + e^{-\hat{x}_{uij}}} \cdot \frac{\partial}{\partial \Theta} \hat{x}_{uij} - \lambda_{\Theta} \Theta \\ &\propto \sum_{(u,i,j) \in D_S} (1 - \sigma(\hat{x}_{uij})) \cdot \frac{\partial}{\partial \Theta} \hat{x}_{uij} - \lambda_{\Theta} \Theta\end{aligned}$$

具体优化流程为：

```
1: procedure LearnBPR( $D_S, \Theta$ )
2:   initialize  $\Theta$ 
3:   repeat
4:     draw  $(u, i, j)$  from  $D_S$ 
5:      $\Theta \leftarrow \Theta + \alpha((1 - \sigma(\hat{x}_{uij})) \cdot \frac{\partial}{\partial \Theta} \hat{x}_{uij} - \lambda_{\Theta} \Theta)$ 
6:   until convergence
7:   return  $\hat{\Theta}$ 
8: end procedure
```



# BPR-MF

我们知道在矩阵分解中，我们想把评分矩阵R分解为用户隐特征因子矩阵P和项目隐特征因子矩阵Q相乘的形式：

$$R = P^T Q$$

即用户u对项目i的评分为 $\hat{x}_{ui} = \sum_{f=1}^k p_{uf} \cdot q_{if}$

所以这里 $\hat{x}_{uij}$ 可以定义为：

$$\hat{x}_{uij} := \hat{x}_{ui} - \hat{x}_{uj} = \sum_{f=1}^k p_{uf} \cdot q_{if} - \sum_{f=1}^k p_{uf} \cdot q_{jf}$$

所以：

$$\frac{\partial}{\partial \theta} \hat{x}_{uij} = \begin{cases} (q_{if} - q_{jf}) & \text{if } \theta = p_{uf} \\ p_{uf} & \text{if } \theta = q_{if} \\ -p_{uf} & \text{if } \theta = q_{jf} \end{cases}$$

$$\frac{\partial \text{BPR} - \text{OPT}}{\partial \Theta}$$

$$\propto \sum_{(u,i,j) \in D_S} (1 - \sigma(\hat{x}_{uij})) \cdot \frac{\partial}{\partial \Theta} \hat{x}_{uij} - \lambda_{\Theta} \Theta$$

```
def optimization(self, u, i, j):
    s = sigmoid(self.P[u].dot(self.Q[i]) - self.P[u].dot(self.Q[j]))
    # update latent features of user u
    self.P[u] += self.lRate * ((1 - s) * (self.Q[i] - self.Q[j]) - self.regU * self.P[u])
    # update latent features of item i
    self.Q[i] += self.lRate * ((1 - s) * self.P[u] - self.regI * self.Q[i])
    # update latent features of item j
    self.Q[j] -= self.lRate * ((1 - s) * self.P[u] - self.regI * self.Q[j])
    self.loss += -log(s)
```



# The End

# Thank You