

第 1 章

推荐算法



1.0.1 NMF

NMF(Non-negative matrix factorization) [1], 直译为非负矩阵分解, 主要用于分解矩阵, 缺失数据的预测等。矩阵分解有很多相关的研究, 可以用于加快计算结果, 以及表示缺失的数据等。在推荐系统中, 缺失值处理是获取推荐物品较为关键的一步, 比如电影推荐, 往往推荐的都是未看过/评分过的电影。如何计算缺失值, 有很多方法。



Note: *Todo:* 缺失值计算方法学习

NMF 在分解矩阵的同时, 对缺失值也进行了处理, 考虑一个 $n * m$ 的矩阵 V , 我们想要将这个矩阵使用 NMF 的方法进行分解, 获取缺失值等操作, 可以利用如下公式:

$$V_{i\mu} \approx (WH)_{i\mu} = \sum_{a=1}^r W_{ia} H_{a\mu}$$

其中包含 r 列的 W 表示基 (原文中为 basis images) 矩阵。 H 表示编码 (encoding) 矩阵, H 中每一列与原矩阵 W 中的每一列对应。组合在一起的含义就是 H 中的一列应用 W 中的一个编码方式可以获得原矩阵中对应位置的数值 (近似值)。NMF 矩阵分解满足的约束为 $(n + m) * r < nm$ (很明显, 分解后矩阵数据量变少了)。

原文描述如下:



Ref: The r columns of W are called basis images. Each column of H is called an encoding and is in one-to-one correspondence with a face in V . An encoding consists of the coefficients by which a face is represented with a linear combination of basis images. The dimensions of the matrix factors W and H are $n * r$ and $r * m$, respectively. \square

但是这个公式并没有体现出如何去计算, 我们只知道 V , 其他一无所知, 唯一知道的是可以输入一个叫 $rank$ 的东西来把 V 分解为包含 $n * r$ 的 W 以及 $r * m$ 的 H 。通过公式还能猜出来, 优化的目标一定是 WH 无限的接近 V , 也就是两者尽可能的相似。根据这个目标, 大致需要一个 cost function 来评估分解的好坏。

计算方式 [2]

判断两个矩阵间是否近似的方法有很多种，但是在判断的过程中我们需要加入一个衡量的标准，在这里称为 cost function。我们的目标是 $V \approx WH$ ，也就是最小化两个矩阵之间的差异，最小化两个矩阵之间的差异有很多方式，比如 \square Euclidean distance:

$$\text{Minimize}(\|V - WH\|^2)$$

亦或是最小化 (Kullback-Leibler divergence):

$$D(A||B) = \sum_{ij} (A_{ij} \log \frac{A_{ij}}{B_{ij}} - A_{ij} + B_{ij})$$

这里考虑 Euclidean distance。

观察上面的公式，当只有一个变量时是很好求的，因为是凸函数 ($f(x) = ax^2$ 这种)，必然会找到最优解。但是公式中只有 V 是已知的，因此需要其他的方式去求解，比如梯度下降，由于多个变量的存在，我们往往找到的是局部最优解。虽然梯度下降的解法简单，但是存在一个诟病就是对步长 (step size) 非常的敏感，换言之就是步子太大有可能直接跳过去了。因此，论文的作者给了一个迭代更新的方法 (Euclidean distance):

$$H_{a\mu} \leftarrow H_{a\mu} \frac{(W^T V)_{a\mu}}{(W^T W H)_{a\mu}}$$

$$W_{ia} \leftarrow W_{ia} \frac{(V H^T)_{ia}}{(W H H^T)_{ia}}$$

我们看到，其实 $H_{a\mu}$ 的下一代迭代值是乘上了一个系数 $\frac{(W^T V)_{a\mu}}{(W^T W H)_{a\mu}}$ ，这也就是原文中倍增的意思 (Multiplicative)。使用乘法来算是不明智的，因为很难复用上一次计算的结果，既浪费性能又浪费时间，那么有没有渐进式的算法呢？让人兴奋的是作者给出了渐进式的更新计算公式：

$$H_{a\mu} \leftarrow H_{a\mu} + \eta_{a\mu} [(W^T V)_{a\mu} - (W^T W H)_{a\mu}]$$

这个公式很漂亮， $H_{a\mu}$ 每次加上一点就能不断的迭代下去，并且利用上了上一次计算的结果。同时，如果令 $\eta_{a\mu} = \frac{H_{a\mu}}{(W^T W H)_{a\mu}}$ ，我们就得到了倍增更新公式。



参考文献



- [1] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788, 1999.
- [2] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 556–562. MIT Press, 2001.