《数据挖掘技术》

⭐ CH15 奇异值分解

➡️ Created by *Wang JingHui*

➡️ Version: *4.0*

主要内容

简介

- SVD是线性代数的概念，但在统计学中有广泛应用，PCA和LSA中都有应用，定义为基础学习方法。
- SVD是矩阵分解方法，特点是分解的矩阵正交。还有另外一种矩阵分解方法叫做NMF，其特点是分解的矩阵非负。
- 奇异值分解是在平方损失意义下对矩阵的最优近似，即**数据压缩**。图像存储是矩阵，那么图像也可以用SVD实现压缩。
- 任意给定一个实矩阵，其奇异值分解一定存在，但并不唯一。Σ是唯一的，U和V^{T}是可变的。
- 奇异值分解有明确的几何意义，事实上，整个线性代数都有明确的几何意义。
- 奇异值分解可以扩展到Tensor。

线性代数基础

为了把向量和点分开，向量通常竖着写，用方括号包围

$$\mathbf{A} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

而点用$(1, 2)$

在现在理论中，向量的形式并不重要， 箭头，一组数，函数等都可以是向量。只要向量相加和数乘的概念遵守以下规则即可，这些规则叫做公理：

$$\overrightarrow{u}+(\overrightarrow{v}+\overrightarrow{w})=(\overrightarrow{u}+\overrightarrow{v})+\overrightarrow{w}$$

$$\overrightarrow{v}+\overrightarrow{w}=\overrightarrow{w}+\overrightarrow{v}$$

$$There\ is\ a\ vector\ 0\ such\ that\ 0+\overrightarrow{v}=\overrightarrow{v} for\ all\ \overrightarrow{v}$$

$$For\ every\ vector\ \overrightarrow{v} there\ is\ a\ vector\ -\overrightarrow{v} so\ that\ \overrightarrow{v}+(-\overrightarrow{v})=0$$

$$a(b\overrightarrow{v})=(ab)\overrightarrow{v}$$

$$1\overrightarrow{v}=\overrightarrow{v}$$

$$a(\overrightarrow{v}+\overrightarrow{w})=a\overrightarrow{v}+a\overrightarrow{w}$$

$$(a+b)\overrightarrow{v}=a\overrightarrow{v}+b\overrightarrow{v}$$

线性代数的每个主题都围绕着向量加法和向量数乘。

向量加法

$$\left[\begin{bmatrix} x_1 \\ y_1 \end{bmatrix}\right] + \left[\begin{bmatrix} x_2 \\ y_2 \end{bmatrix}\right] = \left[\begin{bmatrix} x_1 + x_2 \\ y_1 + y_2 \end{bmatrix}\right]$$

向量数乘

Scaling，缩放的过程。用于缩放的数字，叫做标量，Scalar。

在线性代数中，数字的作用就是缩放向量。

$$2 \cdot \left[\begin{bmatrix} x \\ y \end{bmatrix}\right] = \left[\begin{bmatrix} 2x \\ 2y \end{bmatrix}\right]$$

函数实际上是另一种向量。

- 函数的线性变换，比如微积分中的导数，有时候会用**算子**来表示**变换**的意思。
- 求导具有可加性和成比例性。
- 函数空间趋近于无限维
- 多项式空间，求导
- 矩阵向量乘法和矩阵求导看起来是不相关的，但实际上是一家人。

线性代数	函数
线性变换	线性操作
点乘	内积
特征向量	特征函数

只要处理的对象有合理的数乘和相加的概念，只要定义满足公理，就能应用线性代数中的结论。

奇异值分解定义与性质

定义

矩阵的奇异值分解是指将$m \times n$实矩阵A表示为以下三个实矩阵乘积形式的运算

$$A = U\Sigma V^{\mathrm{T}}$$

其中

- U是$m \times m$阶酉矩阵；Σ是半正定$m \times n$阶对角矩阵；
- V^T是V的共轭转置，是$n \times n$阶酉矩阵；
- Σ对角线上的元素Σ_i，即为M的奇异值。

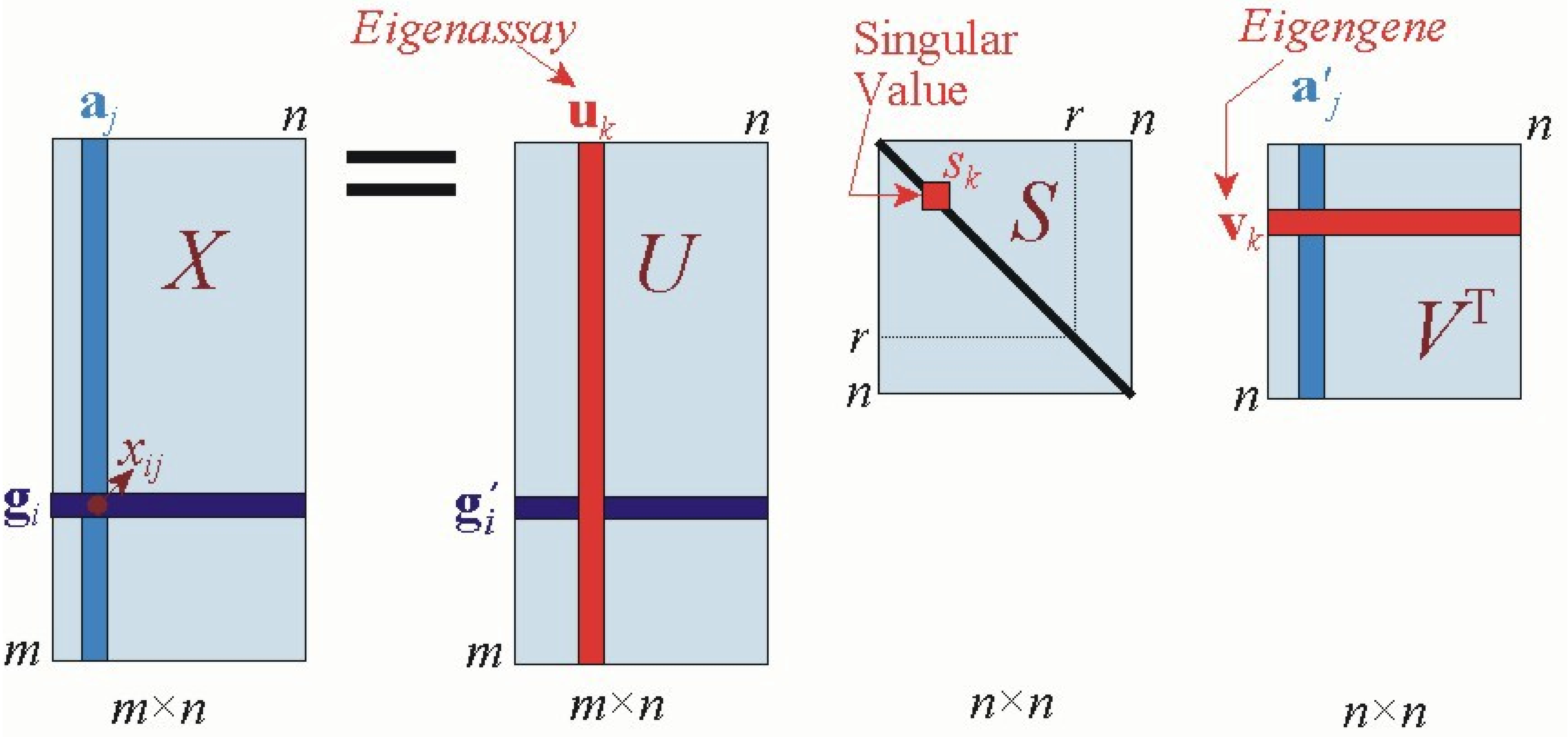
矩阵的奇异值分解是指将$m \times n$实矩阵A表示为以下三个实矩阵乘积形式的运算

$$A = U\Sigma V^{\mathrm{T}}$$

- 完全奇异值分解：$A = U\Sigma V^{\mathrm{T}}$
- 紧奇异值分解：$A = U_r\Sigma_r V_r^{\mathrm{T}}$
- 截断奇异值分解：$A = U_k\Sigma_k V_k^{\mathrm{T}}$

奇异值分解（Singular Value Decomposition）是线性代数中一种重要的矩阵分解，可以在任意矩阵上的推广。

$$X = USV^{\mathrm{T}}$$



几何解释

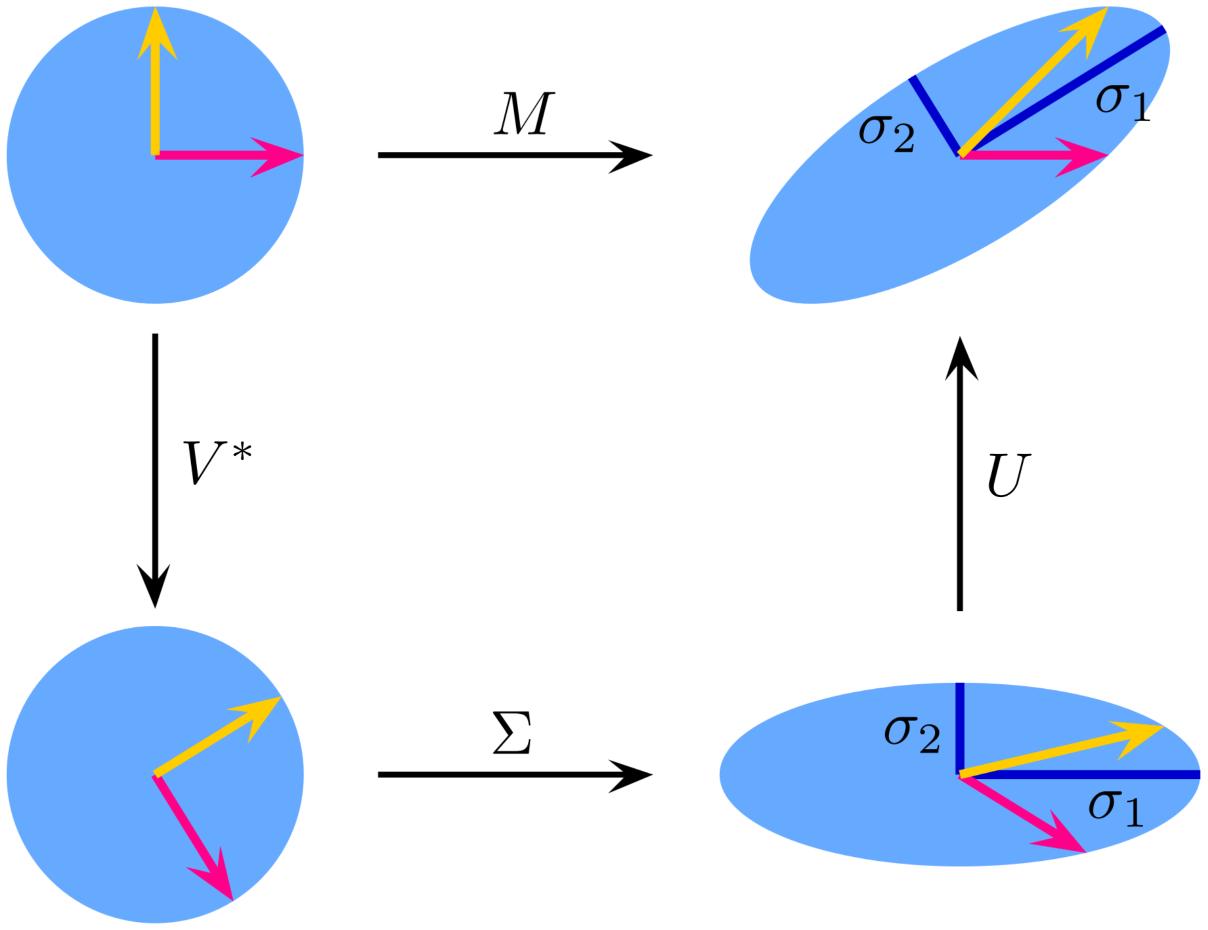
$A_{m\times n}$表示了一个从n维空间$\mathbf{R}^n$到m维空间$\mathbf{R}^m$的一个**线性变换**

$$T: x \rightarrow Ax$$
$$x \in \mathbf{R}^n$$
$$Ax \in \mathbf{R}^m$$

线性变换可以分解为三个简单的变换：

1. 坐标系的旋转或反射变换，V^{T}
2. 坐标轴的缩放变换，Σ
3. 坐标系的旋转或反射变换，U

这里面注意，A其实就是**线性变换**。



$$M = U \cdot \Sigma \cdot V^*$$

主要性质

1. AA^{T}和$A^{\mathrm{T}}A$的特征分解存在，且可由矩阵A的奇异值分解的矩阵表示；
2. 奇异值，左奇异向量，右奇异向量之间的关系
3. 矩阵A的奇异值分解中，奇异值是唯一的，但是矩阵U和V不是唯一的，所以numpy.linalg.svd中有参数控制是否输出U和V

奇异值分解的算法

1. 计算 AA^H 和 $A^H A$
2. 分别计算 AA^H 和 $A^{\mathrm{H}} A$ 的特征向量及其特征值；
3. AA^{H} 的特征向量组成 U;$A^{\mathrm{H}} A$ 的特征向量组成 V；
4. 对 AA^{H} 和 $A^{\mathrm{H}} A$ 的非零特征值求平方根，对应上述特征向量的位置，填入 Σ

例： 计算以下矩阵奇异值分解：

$$\mathbf{A}=\begin{bmatrix}2 & 4\\ 1 & 3\\ 0 & 0\\ 0 & 0\end{bmatrix}$$

需要计算 A 与其共轭转置的左右积；这里主要以 AA^{H} 为例。

首先，我们需要计算 AA^{H}

$$\mathbf{W}=\mathbf{A}\mathbf{A}^{\mathsf{H}}=\begin{bmatrix}2 & 4\\ 1 & 3\\ 0 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}2 & 1 & 0 & 0\\ 4 & 3 & 0 & 0\end{bmatrix}=\begin{bmatrix}20 & 14 & 0 & 0\\ 14 & 10 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{bmatrix}.$$

,

求W的特征值与特征向量，定义$\mathbf{W}\vec{x}=\lambda\vec{x}$，即

$$\begin{bmatrix} 20-\lambda & 14 & 0 & 0 \\ 14 & 10-\lambda & 0 & 0 \\ 0 & 0 & -\lambda & 0 \\ 0 & 0 & 0 & -\lambda \end{bmatrix}\vec{x}=\vec{0}.$$

根据线性方程组的理论，要求系数矩阵的行列式为0，即

$$\begin{vmatrix} 20-\lambda & 14 & 0 & 0 \\ 14 & 10-\lambda & 0 & 0 \\ 0 & 0 & -\lambda & 0 \\ 0 & 0 & 0 & -\lambda \end{vmatrix}=\begin{vmatrix} 20-\lambda & 14 \\ 14 & 10-\lambda \end{vmatrix}\begin{vmatrix} -\lambda & 0 \\ 0 & -\lambda \end{vmatrix}=0$$

即

$$\big((20-\lambda)(10-\lambda)-196\big)\lambda^2=0$$

解得:

$$\lambda_1=\lambda_2=0$$

$$\lambda_3=15+\sqrt{221}\approx 29.866$$

$$\lambda_4=15-\sqrt{221}\approx 0.134$$

将特征值带入原非常，可解得对应的特征向量，形成矩阵：

$$\mathbf{U}=\begin{bmatrix}-0.82 & -0.58 & 0 & 0\\ -0.58 & 0.82 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}.$$

同理可解得，AA^H和 $A^H A$ 的特征值相同。

$$\mathbf{V}=\begin{bmatrix}-0.40 & -0.91\\ -0.91 & 0.40\end{bmatrix}.$$

Σ上的对角线元素由W的特征值的算术平方根组成：

$$\mathbf{\Sigma}=\begin{bmatrix}5.46 & 0\\ 0 & 0.37\\ 0 & 0\\ 0 & 0\end{bmatrix}.$$

因此得到矩阵M的SVD分解

$$\begin{bmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \approx \begin{bmatrix} -0.82 & -0.58 & 0 & 0 \\ -0.58 & 0.82 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 5.46 & 0 \\ 0 & 0.37 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -0.40 & -0.91 \\ -0.91 & 0.40 \end{bmatrix}$$

考虑一个问题：

- 假设样本集 $X=(x_i, x_2, ..., x_m$ 中每个 x_i 是一个图片，并且是一个 100×100 的图片，如果以像素值作为特征，那么每张图片的特征维度是10000。
- 当进行PCA降维时，难点在于我们构造协方差矩阵时，维度达到 10000*10000 。
- 在这样的协方差矩阵上求解特征值，耗费的计算量程平方级增长。
- 面对这样一个难点，从而引出奇异值分解(SVD)，利用SVD不仅可以解出PCA的解，而且无需大的计算量。

奇异值分解与矩阵近似

奇异值分解也是一种矩阵近似的方法，这个近似是在**弗罗贝尼斯范数**意义下的近似。

$$\|A\|_F = \left(\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}^2\right)^{\frac{1}{2}}$$
$$A \in \mathbf{R}^{m\times n}, A = [a_{ij}]_{m\times n}$$

矩阵的弗罗贝尼斯范数是向量的L_2范数的直接推广，对应着机器学习里面的平方损失函数。矩阵范数(matrix norm)也是一个很大的概念，详细内容可以扩展下^1。

$$A \in \mathbf{R}^{m\times n}$$
$$A = U\Sigma V^{\mathrm{T}}$$
$$\Sigma = \operatorname{diag}(\sigma_1, \sigma_2, \cdots, \sigma_{\mathrm{n}}) \|A\|_F = (\sigma_1^2 + \sigma_2^2 + \cdots + \sigma_{\mathrm{n}}^2)^{\frac{1}{2}}$$

矩阵的最优近似

$$\|A - X\|_F = \min_{S \in \mathcal{M}} \|A - S\|_F$$

$$\|A - X\|_F = (\sigma_{k+1}^2 + \sigma_{k+2}^2 + \cdots + \sigma_n^2)^{\frac{1}{2}}$$

矩阵的外积展开式

$$\begin{aligned} A &= \sigma_1 u_1 v_1^{\mathrm{T}} + \sigma_2 u_2 v_2^{\mathrm{T}} + \cdots + \sigma_n u_n v_n^{\mathrm{T}} \\ &= \sum_{k=1}^{n} A_k \\ &= \sum_{k=1}^{n} \sigma_k u_k v_k^{\mathrm{T}} \end{aligned}$$

其中，$u_k v_k^{\mathrm{T}}$为$m \times n$矩阵

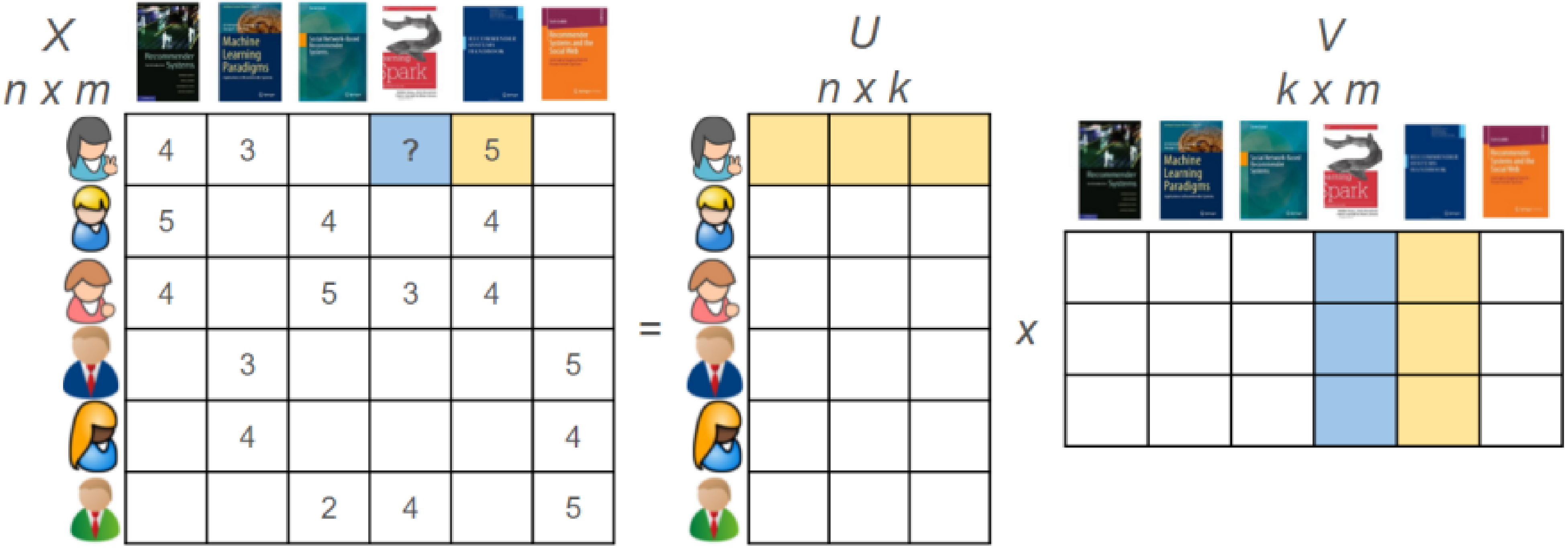
应用

数据压缩

- 对于RGB图像来说，对应三色层，对每一个色层进行SVD分解，得到 U , S , Σ 设利用前K大奇异值进行图像压缩。

推荐算法

在推荐系统中，根据用户行为通常可以得到user-item的评分矩阵。



SVD 推荐[1]

- 由于每个用户可能只在少部分商品上有历史行为，因此评分矩阵往往是稀疏的；
- 希望可以预测出用户对未评分商品的评分，从而将评分高的商品推荐给用户；
- 如果对user-item评分矩阵进行SVD分解，就可以得到：
  $M_{m*n} = U^T_{m_k} \Sigma_{k*k} V_{k*n}$
  其中k是矩阵M中较大的部分奇异值的个数，一般会远远的小于用户数和物品数。如果我们要预测第i个用户对第j个物品的评分M_{ij},则只需要计算$U_i^T \Sigma V_j$即可。
- 通过这种方法，可以将评分表里面所有没有评分的位置得到一个预测评分。通过找到最高的若干个评分对应的物品推荐给用户。到这里似乎很完美了。
- 新的问题：
  - 矩阵太稠密计算复杂度高
  - 如何对缺失值进行填充

FunkSVD

为了解决上述传统SVD的问题，FunkSVD被提出，这种改进算法称为隐语义模型或潜在因素模型。该方法只将矩阵分解成2个矩阵——用户隐含特征组成的矩阵和项目隐含特征组成的矩阵：

$$R \approx P_{m*k}^{T} Q_{k*n}$$

其中k表示隐特征的数量，P为k×m矩阵，表示用户特征向量；Q为k×n矩阵，表示物品特征向量。那么用户u对用户i的预测评分为：

$\hat{r}_{ui} = p_u^T q_i$

- 求解

求解方法是将问题转化为最小化误差函数，目标函数为：
$C = \sum_{(u,i)\in R}(r_{ui} - p_u^T q_i)^2 + \lambda(||p_u||^2 + ||q_i||^2)$
目标
$\min_{p_u,q_i} C$

- 优化方法

最常用的两种优化方法：随机梯度下降（SGD)和最小二乘（ALS）。通常SGD比ALS(Alternating Least Squares)简单而且快速；但是ALS的并行性能比较好，而且可以较好地处理稀疏数据，spark的实现方式就是ALS。

基于SVD的其他改进方法

- Bias-SVD
- SVD++

习题

15.5

$$\begin{vmatrix} & u_1 & u_2 & u_3 & u_4 & u_5 \\ q_1 & 0 & 20 & 5 & 0 & 0 \\ q_2 & 10 & 0 & 0 & 3 & 0 \\ q_3 & 0 & 0 & 0 & 0 & 1 \\ q_4 & 0 & 0 & 1 & 0 & 0 \end{vmatrix}$$

这个矩阵恢复出来就是这样的，SVD分解之后，U应该是quary的相似度

参考

[1] https://www.sohu.com/a/190681269_470008

[2] https://www.jianshu.com/p/a2d564e88379

↘️ Enjoy your machine learning!

https://github.com/wjssx/Statistical-Learning-Slides-Code

E-mail: csr_dsp@sina.com