SVD（奇异值分解）小结

> **注：** 奇异值分解在数据降维中有较多的应用，这里把它的原理简单总结一下，并且举一个图片压缩的例子，最后做一个简单的分析，希望能够给大家带来帮助。

1、特征值分解（EVD）

实对称矩阵

在理角奇异值分解之前，需要先回顾一下特征值分解，如果矩阵A是一个$m \times m$的`实对称矩阵`（即$A = A^T$），那么它可以被分解成如下的形式

$$A = Q\Sigma Q^T = Q\begin{bmatrix} \lambda_1 & \cdots & \cdots & \cdots \\ \cdots & \lambda_2 & \cdots & \cdots \\ \cdots & \cdots & \ddots & \cdots \\ \cdots & \cdots & \cdots & \lambda_m \end{bmatrix} Q^T \tag{1-1}$$

其中Q为标准正交阵，即有$QQ^T = I$，Σ为对角矩阵，且上面的矩阵的维度均为$m \times m$。λ_i称为`特征值`，q_i是Q（特征矩阵）中的列向量，称为`特征向量`。

> **注：** I在这里表示单位阵，有时候也用E表示单位阵。式（1–1）的具体求解过程就不多叙述了，可以回忆一下大学时的线性数。简单地有如下关系：$Aq_i = \lambda_i q_i, \quad q_i^T q_j = 0(i \neq j)$。

一般矩阵

上面的特征值分解，对矩阵有着较高的要求，它需要被分解的矩阵A为实对称矩阵，但是现实中，我们所遇到的问题一般不是实对称矩阵。那么当我们碰到一般性的矩阵，即有一个$m \times n$的矩阵A，它是否能被分解成上面的式（1-1）的形式呢？当然是可以的，这就是我们下面要讨论的内容。

2、奇异值分解（SVD）

2.1 奇异值分解定义

有一个$m \times n$的实数矩阵A，我们想要把它分解成如下的形式

$$A = U\Sigma V^T \tag{2-1}$$

其中U和V均为单位正交阵，即有$UU^T = I$和$VV^T = I$，U称为`左奇异矩阵`，V称为`右奇异矩阵`，Σ仅在主对角线上有值，我们称它为`奇异值`，其它元素均为0。上面矩阵的维度分别为$U \in R^{m\times m}$，$\Sigma \in R^{m\times n}$，$V \in R^{n\times n}$。

一般地Σ有如下形式

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & 0 & \ddots & 0 \end{bmatrix}_{m\times n}$$

图1–1 奇异值分解

对于奇异值分解，我们可以利用上面的图形象表示，图中方块的颜色表示值的大小，颜色越浅，值越大。对于奇异值矩阵Σ，只有其主对角线有奇异值，其余均为0。

2.2 奇异值求解

正常求上面的U,V,Σ不便于求，我们可以利用如下性质

$$AA^T = U\Sigma V^T V\Sigma^T U^T = U\Sigma\Sigma^T U^T \tag{2-2}$$

$$A^T A = V\Sigma^T U^T U\Sigma V^T = V\Sigma^T\Sigma V^T \tag{2-3}$$

> **注：** 需要指出的是，这里$\Sigma\Sigma^T$与$\Sigma^T\Sigma$在矩阵的角度上来讲，它们是不相等的，因为它们的维数不同$\Sigma\Sigma^T \in R^{m\times m}$，$\Sigma^T\Sigma \in R^{n\times n}$，但是它们在主对角线的奇异值是相等的，即有

$$\Sigma\Sigma^T = \begin{bmatrix} \sigma_1^2 & 0 & 0 & 0 \\ 0 & \sigma_2^2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \ddots \end{bmatrix}_{m\times m} \quad \Sigma^T\Sigma = \begin{bmatrix} \sigma_1^2 & 0 & 0 & 0 \\ 0 & \sigma_2^2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \ddots \end{bmatrix}_{n\times n}$$

可以看到式（2-2）与式（1-1）的形式非常相同，进一步分析，我们可以发现AA^T和A^TA也是对称矩阵，那么可以利用式（1-1），做特征值分解。利用式（2-2）特征值分解，得到的特征矩阵即为U；利用式（2-3）特征值分解，得到的特征矩阵即为V；对$\Sigma\Sigma^T$或$\Sigma^T\Sigma$中的特征值开方，可以得到所有的奇异值。

3、奇异值分解应用

3.1 纯数学例子

假设我们现在有矩阵A，需要对其做奇异值分解，已知

$$A = \begin{bmatrix} 1 & 5 & 7 & 6 & 1 \\ 2 & 1 & 10 & 4 & 4 \\ 3 & 6 & 7 & 5 & 2 \end{bmatrix}$$

那么可以求出AA^T和A^TA，如下

$$AA^T = \begin{bmatrix} 112 & 105 & 114 \\ 105 & 137 & 110 \\ 114 & 110 & 123 \end{bmatrix} \quad A^TA = \begin{bmatrix} 14 & 25 & 48 & 29 & 15 \\ 25 & 62 & 87 & 64 & 21 \\ 48 & 87 & 198 & 117 & 61 \\ 29 & 64 & 117 & 77 & 32 \\ 15 & 21 & 61 & 32 & 21 \end{bmatrix}$$

分别对上面做特征值分解，得到如下结果

```
U =
```

```
[[-0.55572489, -0.72577856,  0.40548161],
 [-0.59283199,  0.00401031, -0.80531618],
 [-0.58285511,  0.68791671,  0.43249337]]

V =
[[-0.18828164, -0.01844501,  0.73354812,  0.65257661,  0.06782815],
 [-0.37055755, -0.76254787,  0.27392013, -0.43299171, -0.17061957],
 [-0.74981208,  0.4369731 , -0.12258381, -0.05435401, -0.48119142],
 [-0.46504304, -0.27450785, -0.48996859,  0.39500307,  0.58837805],
 [-0.22080294,  0.38971845,  0.36301365, -0.47715843,  0.62334131]]
```

奇异值$\Sigma = \mathrm{Diag}(18.54, 1.83, 5.01)$

3.2 在图像压缩中的应用

准备工具

下面的代码运行环境为 `python3.6` + `jupyter5.4`

SVD（Python）

这里暂时用numpy自带的svd函数做图像压缩。

①读取图片

```
%matplotlib inline
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import numpy as np

img_eg = mpimg.imread("../img/beauty.jpg")
print(img_eg.shape)
```

图片的大小是600 × 400 × 3

②奇异值分解

```
img_temp = img_eg.reshape(600, 400 * 3)
U,Sigma,VT = np.linalg.svd(img_temp)
```

我们先将图片变成600×1200，再做奇异值分解。从 `svd` 函数中得到的奇异值 `sigma` 它是从大到小排列的。

③取前部分奇异值重构图片

```
# 取前60个奇异值
sval_nums = 60
img_restruct1 = (U[:,0:sval_nums]).dot(np.diag(Sigma[0:sval_nums])).dot(VT[0:sval_nums,:])
img_restruct1 = img_restruct1.reshape(600,400,3)

# 取前120个奇异值
sval_nums = 120
img_restruct2 = (U[:,0:sval_nums]).dot(np.diag(Sigma[0:sval_nums])).dot(VT[0:sval_nums,:])
img_restruct2 = img_restruct2.reshape(600,400,3)
```

将图片显示出来看一下，对比下效果

```
fig, ax = plt.subplots(1,3,figsize = (24,32))

ax[0].imshow(img_eg)
ax[0].set(title = "src")
ax[1].imshow(img_restruct1.astype(np.uint8))
ax[1].set(title = "nums of sigma = 60")
ax[2].imshow(img_restruct2.astype(np.uint8))
ax[2].set(title = "nums of sigma = 120")
```

图3-1 奇异值重构图片

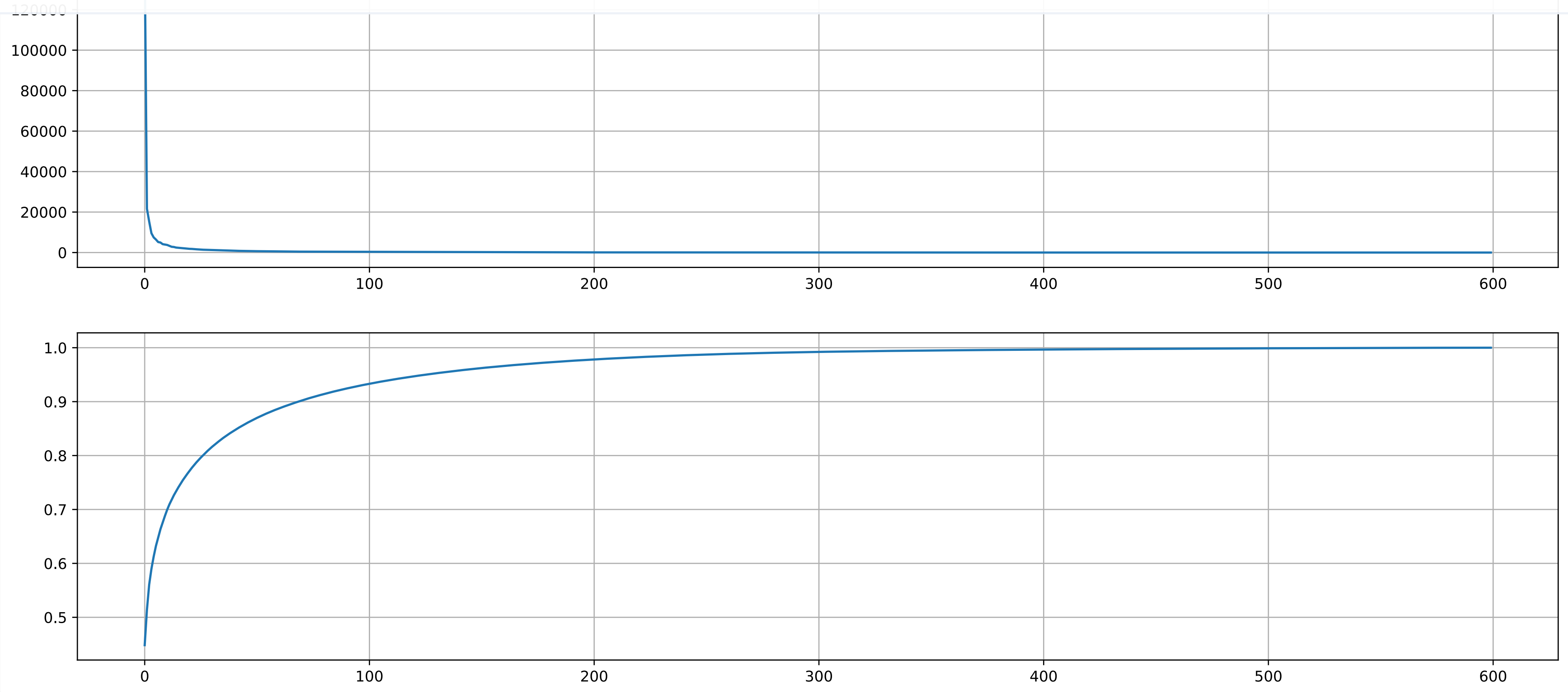
可以看到，当我们取到前面120个奇异值来重构图片时，基本上已经看不出与原图片有多大的差别。

> **注：** 上面的美女图片源于网络，侵删。

总结

从上面的图片的压缩结果中可以看出来，奇异值可以被看作成一个矩阵的代表值，或者说，奇异值能够代表这个矩阵的信息。**当奇异值越大时，它代表的信息越多。**因此，我们取前面若干个最大的奇异值，就可以基本上还原出数据本身。

如下，可以作出奇异值数值变化和前部分奇异值和的曲线图，如下图所示



奇异值变化图

从上面的第1个图，可以看出，奇异值下降是非常快的，因此可以只取前面几个奇异值，便可基本表达出原矩阵的信息。从第2个图，可以看出，当取到前100个奇异值时，这100个奇异值的和已经占总和的95%左右。

最后，还有一点需要提到的是，如果自己想不调用 `np.linalg.svd` 函数，手动实现奇异值分解的话，单纯利用第2小节的内容实现，有点不够，有个问题需要注意。这里暂时不多做讨论了，大家有兴趣可以看我下面分享的**《SVD（奇异值分解）Python实现》**，重可以看看其中**SVD算法实现**。

分类：人工智能，数据科学