斯坦福 ML 公开课笔记 15

我们在上一篇笔记中讲到了 PCA(主成分分析)。PCA 是一种直接的降维方法，通过求解特征值与特征向量，并选取特征值较大的一些特征向量来达到降维的效果。

本文继续 PCA 的话题，包括 PCA 的一个应用——LSI(Latent Semantic Indexing, 隐含语义索引)和 PCA 的一个实现——SVD(Singular Value Decomposition,奇异值分解)。在 SVD 和 LSI 结束之后，关于 PCA 的内容就告一段落。视频的后半段开始讲无监督学习的一种——ICA(Independent Component Analysis, 独立成分分析)。

隐含语义索引(LSI)

视频只是在概念上对 LSI 进行了介绍，并举了一个文本相似度计算的例子，其实网上关于 SVD 和 LSI 的文章有很多，可以参考我之前转载的一篇文章[1]。

假设我们使用多元伯努利事件模型(NB-MBEM, 见笔记 6)来表示文本：

$$\mathrm{x}=\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{4321} \\ \vdots \\ x_{50000} \end{bmatrix}=\begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix}\begin{matrix} a \\ aardvark \\ aardwolf \\ \vdots \\ buy \\ \vdots \\ zygmurgy \end{matrix} \tag{1}$$

整个文本被表示成为一个由 0,1 组成的向量。对于这样的向量来说，Ng 的视频中讲了它的两个特性。第一是这样的向量在 PCA 的时候一般不做标准化，注意到上一篇笔记中的预处理步骤，在计算均值的时候是在样本 x 的每个分量上做平均，这样对于不常出现的词来说，会使其权重增大。比如对于公式 1 中的 aardvark，这单词不管你见没见过，我反正是没见过，假如它只在一个样本$x^{(i)}$中出现了，那么它的权重在$x^{(i)}$中为 1，在其他地方都为 0，生僻词的权重增大对于计算相似度来说是不合理的，因为生僻词的使用过度依赖于个人喜好（这是我个人想的，不是视频中，欢迎拍砖）。第二个特性是表达成这样的向量之后，就可以用向量之间的距离来衡量文本之间的相似度了。比如经典的余弦相似度计算方法：

$$\mathrm{Sim}\left(x^{(i)},x^{(j)}\right)=cos\theta=\frac{{x^{(i)}}^T x^{(j)}}{||x^{(i)}||*||x^{(j)}||} \tag{2}$$

那么啥是隐含语义索引呢？在主成分分析中隐含语义索引的意思就是通过降维的手段，将意义相同的词映射到低维空间中的同一个维度上去。这样一可以降低计算复杂度，二可以减少噪声。其中，减少噪声是指两个在高维上完全不相似的文本通过降维以后可能变得相似了。比如，假如有一篇文章，只包含一个单词 learn，另一篇文章只包含 study，在高维上计算相似度为 0，通过隐含语义索引，将 learn 和 study 映射到同一维上，再计算相似度就更精确了。

个人觉得关于 LSI 视频中讲的太粗糙，欲知详情，请查看参考文章或自行谷歌百度之。

奇异值分解(SVD)

奇异值分解是 PCA 的一种实现，在上一篇文章中，我们介绍了 PCA 的基本实现手段，即首先计算协方差矩阵$\Sigma=\frac{1}{m}\sum_{i=1}^{m}x^{(i)}{x^{(i)}}^T$，然后对其特征值与特征向量进行求解。这样做的不好的地方在于，协方差矩阵的维度是样本维度*样本维度。比如，对于 100*100 的图片来说，如果以像素值作为特征，那么每张图片的特征维度是 10000，那么，协方差矩阵的维度就是 10000*10000。在这样的协方差矩阵上求解特征值，耗费的计算量呈平方级增长。利

用 SVD 可以求解出 PCA 的解但无需耗费大计算量。下面就介绍 SVD。

SVD 的基本公式如下：

$$A = UDV^T$$

其中，$A \in \mathbb{R}^{m*n}$, $U \in \mathbb{R}^{m*n}$, $D \in \mathbb{R}^{n*n}$且为对角矩阵，D 对角线上的每个值都是特征值，且已按大小排好序，$V \in \mathbb{R}^{n*n}$。其中，U 的列向量即是AA^T的特征向量，V 的列向量是A^TA的特征向量。令：

$$A = X = \begin{bmatrix} | & | & \cdots & | \\ x^{(1)} & x^{(2)} & \cdots & x^{(m)} \\ | & | & \cdots & | \end{bmatrix}$$

则协方差矩阵$\Sigma = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}x^{(i)^T} = XX^T$,那么 U 恰好为 PCA 的解。将 PCA 转化为 SVD 求解问题后，就可以进行加速了，因为 SVD 的求解有其特定的加速方法。本节不涉及。

SVD 可以理解为 PCA 的一种求解方法。SVD 也可以用于降维，一般情况下，D 对角线中前 10%或 20%的特征值已占全部特征值之和的 90%以上。因而可以对UDV^T三个矩阵各自进行裁剪，比如将特征由 n 维将为 k 维，那么$U \in \mathbb{R}^{m*k}$, $D \in \mathbb{R}^{k*k}$, $V^T \in \mathbb{R}^{k*n}$即可。

关于 SVD 的应用，比如在推荐系统中的应用，可以参考我的博文《概率矩阵分解模型 PMF》[2]。

在 SVD 的最后，Ng 总结出一张表。

	密度估计法（概率方法）	非概率方法
降维到子空间	因子分析	PCA
假设数据位于区块中	混合高斯模型	K-Means

表格中的内容很好理解，Ng 特地强调的是这样的思考方式，寻找算法中的相同点和不同点有利于更好的理解算法。

独立成分分析(ICA)

本节主要参考博客[3]。

问题

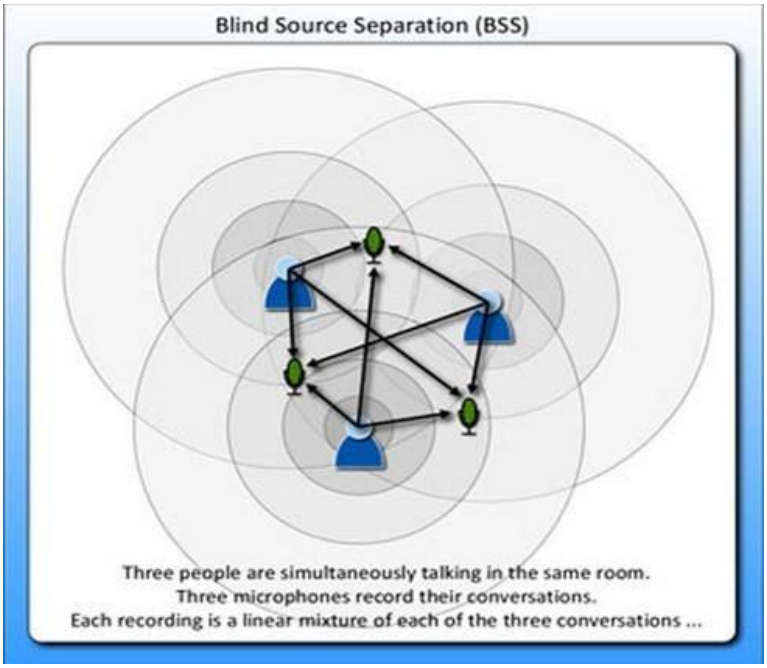
经典的鸡尾酒宴会问题（cocktail party problem）。假设在 party 中有 n 个人，他们可以同时说话，我们也在房间中一些角落里共放置了 n 个声音接收器（Microphone）用来记录声音。宴会过后，我们从 n 个麦克风中得到了一组数据$\{x^{(i)}\left(x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\right), i = 1,2, \ldots m\}$，i 表示采样的时间顺序，也就是说共得到了 m 组采样，每一组采样都是 n 维的。我们的目标是单单从这 m 组采样数据中分辨出每个人说话的信号。

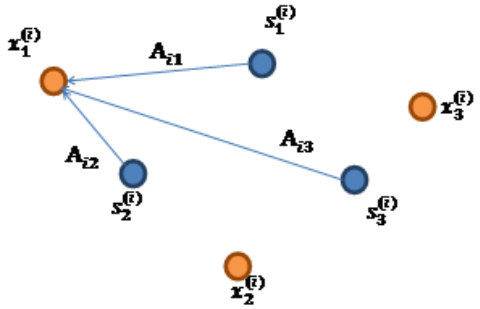
将问题细化一下，有 n 个信号源$s(s_1, s_2, \ldots, s_n)^T, s \in \mathbb{R}^n$，每一维都是一个人的声音信号，每个人发出的声音信号独立。A 是一个未知的混合矩阵（mixing matrix），用来组合叠加信号 s，那么

$$x = As$$

x 的意义在上文解释过，这里的 x 不是一个向量，是一个矩阵。其中每个列向量是$x^{(i)}$，$x^{(i)} = As^{(i)}$。

表示成图就是





$x^{(i)}$的每个分量都由$s^{(i)}$的分量线性表示。A 和 s 都是未知的，x 是已知的，我们要想办法根据 x 来推出 s。这个过程也称作为盲信号分离。

令$W = A^{-1}$,那么$s^{(i)} = A^{-1}x^{(i)} = Wx^{(i)}$,则可将 W 表示成

$$W = \begin{bmatrix} -w_1^T - \\ \vdots \\ -w_n^T - \end{bmatrix}$$

其中$w_i \in \mathbb{R}^n$，其实就是将w_i写成行向量形式。那么得到：

$$s_j^{(i)} = w_j^T x^{(i)}$$

ICA 的不确定性

由于 w 和 s 都不确定，那么在没有先验知识的情况下，无法同时确定这两个相关参数。比如上面的公式$s = wx$。当 w 扩大两倍时，s 只需要同时扩大两倍即可，等式仍然满足，因此无法得到唯一的 s。同时如果将人的编号打乱，变成另外一个顺序，如上图的蓝色节点的编号变为 3,2,1，那么只需要调换 A 的列向量顺序即可，因此也无法单独确定 s。这两种情况称为原信号不确定。

还有一种 ICA 不适用的情况，那就是信号不能是高斯分布的。假设只有两个人发出的声音信号符合多值正态分布，$s \sim N(0, I)$，I 是 2*2 的单位矩阵，s 的概率密度函数就不用说了吧，以均值 0 为中心，投影面是椭圆的山峰状（参见多值高斯分布）。因为$x = As$，因此，

x 也是高斯分布的，均值为 0，协方差为$\mathrm{E}[xx^T] = \mathrm{E}[Ass^TA^T] = AA^T$。

令 R 是正交阵($RR^T = R^TR = I$)，$A' = AR$。如果将 A 替换成A'。那么$x' = A's$。s 分布没变，因此x'仍然是均值为 0，协方差

$$\mathrm{E}[x'(x')^T] = \mathrm{E}[A'ss^T(A')^T] = \mathrm{E}[ARss^T(AR)^T] = ARR^TA^T = AA^T$$

因此，不管混合矩阵是 A 还是A'，x 的分布情况是一样的，那么就无法确定混合矩阵，也就无法确定原信号。

密度函数与线性变换

在讨论 ICA 具体算法之前，我们先来回顾一下概率和线性代数里的知识。

假设我们的随机变量 s 有概率密度函数$p_s(s)$(连续值是概率密度函数，离散值是概率)。为了简单，我们再假设 s 是实数，还有一个随机变量 x=As，A 和 x 都是实数。令p_x是 x 的概率密度，那么怎么求p_x？

令$W = A^{-1}$，首先将式子变换成$s = Wx$，然后得到$p_x(x) = p_s(Ws)$，求解完毕。可惜这种方法是错误的。比如 s 符合均匀分布的话（s~Uniform[0,1]），那么 s 的概率密度是$p_s(s) = 1\{0 \le s \le 1\}$，现在令 A=2，即 x=2s，也就是说 x 在[0,2]上均匀分布，可知$p_x(x) = 0.5$。然而，前面的推导会得到$p_x(x) = p_s(0.5s) = 1$。正确的公式应该是

$$p_x(x) = p_s(Wx)|W|$$

推导方法如下：

$$F_X(a) = P(X \le a) = P(AS \le a) = P(S \le Wa) = F_S(Wa)$$

$$p_x(a) = F_X'(a) = F_S'(Wa) = p_s(Wa)|W|$$

更一般地，如果 s 是向量，A 可逆的方阵，那么上式子仍然成立。

ICA 算法

ICA 算法归功于 Bell 和 Sejnowski，这里使用最大似然估计来解释算法，原始的论文中使用的是一个复杂的方法 Infomax principal。

我们假定每个s_i有概率密度P_s，那么给定时刻原信号的联合分布就是

$$\mathrm{p(s)} = \prod_{i=1}^{n} p_s(s_i)$$

这个公式代表一个假设前提：每个人发出的声音信号各自独立。有了 p(s)，我们可以求得 p(x):

$$\mathrm{p(x)} = p_s(Wx)|W| = |W|\prod_{i=1}^{n} p_s(w_i^Tx_i)$$

左边是每个采样信号 x（n 维向量）的概率，右边是每个原信号概率的乘积的|W|倍。

前面提到过，如果没有先验知识，我们无法求得 W 和 s。因此我们需要知道$p_s(s_i)$，我们打算选取一个概率密度函数赋给 s，但是我们不能选取高斯分布的密度函数。在概率论里我们知道密度函数 p(x)由累计分布函数（CDF）F(x)求导得到。F(x)要满足两个性质是：单调递增和在[0,1]。我们发现 sigmoid 函数很适合，定义域负无穷到正无穷，值域 0 到 1，缓慢递增。我们假定 s 的累积分布函数符合 sigmoid 函数

$$\mathrm{g(s)} = \frac{1}{1+e^{-s}}$$

求导后

$$p_s(s) = g'(s) = \frac{e^s}{(1+e^s)^2}$$

这就是 s 的密度函数。这里 s 是实数。

如果我们预先知道 s 的分布函数，那就不用假设了，但是在缺失的情况下，sigmoid 函

数能够在大多数问题上取得不错的效果。由于上式中$p_s(s)$是个对称函数，因此 E[s]=0（s 的均值为 0），那么 E[x]=E[As]=0，x 的均值也是 0。

知道了$p_s(s)$，就剩下 W 了。给定采样后的训练样本$\{x^{(i)}\left(x_1^{(i)}, x_2^{(i)}, \dots, x_n^{(i)}\right), \mathrm{i} = 1,2, \dots \mathrm{m}\}$，样本对数似然估计如下，使用前面得到的 x 的概率密度函数，得

$$\ell(\mathrm{W}) = \sum_{i=1}^{m}\left(\sum_{j=1}^{n} log g'\left(w_j^T x^{(i)}\right) + \log|W|\right)$$

大括号里面是$\mathrm{p}(x^{(i)})$。

接下来就是对 W 求导了，这里牵涉一个问题是对行列式|W|进行求导的方法，属于矩阵微积分。这里先给出结果，在文章最后再给出推导公式。

$$\nabla_W|W| = |W|(W^{-1})^T$$

最终得到的求导后公式如下，$log g'(s)$的导数为$1 - 2\mathrm{g}(\mathrm{s})$（可以自己验证）：

$$\mathrm{W} \coloneqq \mathrm{W} + \alpha\left(\begin{bmatrix} 1 - 2g(w_1^T x^{(i)}) \\ 1 - 2g(w_2^T x^{(i)}) \\ \vdots \\ 1 - 2g(w_n^T x^{(i)}) \end{bmatrix} x^{(i)^T} + (W^T)^{-1}\right)$$

其中α是梯度上升速率，人为指定。

当迭代求出 W 后，便可得到$s^{(i)} = Wx^{(i)}$来还原出原始信号。

注意：我们计算最大似然估计时，假设了$x^{(i)}$与$x^{(j)}$之间是独立的，然而对于语音信号或者其他具有时间连续依赖特性（比如温度）上，这个假设不能成立。但是在数据足够多时，假设独立对效果影响不大，同时如果事先打乱样例，并运行随机梯度上升算法，那么能够加快收敛速度。

回顾一下鸡尾酒宴会问题，s 是人发出的信号，是连续值，不同时间点的 s 不同，每个人发出的信号之间独立（s_i和s_j之间独立）。s 的累计概率分布函数是 sigmoid 函数，但是所有人发出声音信号都符合这个分布。A（W 的逆阵）代表了 s 相对于 x 的位置变化，x 是 s 和 A 变化后的结果。

行列式的梯度

对行列式求导，设矩阵 A 是 n×n 的，我们知道行列式与代数余子式有关，

$$|\mathrm{A}| = \sum_{i=1}^{n}(-1)^{i+j} a_{ij}|A_{\backslash i,\backslash j}| \ (for\ any\ j \in 1, \dots, n)$$

$A_{\backslash i,\backslash j}$ 是去掉第 i 行第 j 列后的余子式，那么对$a_{k,l}$求导得

$$\frac{\partial}{\partial a_{k,l}}|A| = \frac{\partial}{\partial a_{k,l}}\sum_{i=1}^{n}(-1)^{i+j} a_{ij}\left|A_{\backslash i,\backslash j}\right| = (-1)^{k+l}\left|A_{\backslash k,\backslash l}\right| = (A^*)_{lk}$$

A^*是伴随矩阵，因此

$$\nabla_A|A| = (A^*)^T = |A|(A^{-1})^T$$

参考文章

[1]. http://blog.csdn.net/stdcoutzyx/article/details/8495948
[2]. http://blog.csdn.net/stdcoutzyx/article/details/21347157
[3]. http://blog.csdn.net/u012409883/article/details/17091299