Department of computer science and technology

Introduction to Data Mining Assignment *I* LaTeX

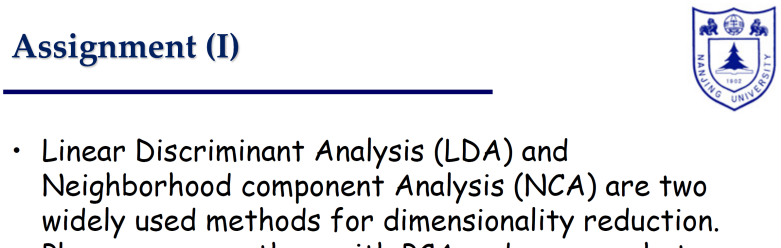
第*1*次作业

2019 年 4 月 8 日

姓名: 张逸凯
学号: 171840708
年级: 大二
指导老师: 黎铭
邮箱: zykhelloha@gmail.com
联系电话: 18051988316

目录

1 本次作业综述



在每小节的前半部分是对三种降维方法的数学推导

后半部分剖析了具体是如何降维的(*Assignment* (*a*))

在第6节分析对比了LDA与PCA, 第7分析对比了NCA和PCA(*Assignment* (*b*)).

2 预备知识

这次作业围绕的核心就是高维数据的降维, 在数据挖掘中, 属性可以达到成百上千, 首先我就需要理解高维空间(超空间-hyperspace)

引入超立方体的概念:

其中每个属性X_j的最小值和最大值:

$$\min\left(X_j\right)=\min\left\{x_{ij}\right\}\quad \max\left(X_j\right)=\max_i\left\{x_{ij}\right\}$$

定义超矩形:

$$\begin{aligned}R_d&=\prod_{j=1}^{d}\left[\min\left(X_j\right),\max\left(X_j\right)\right]\\&=\left\{x=(x_1,x_2,\cdots,x_d)^{\mathrm{T}}\,|x_j\in\left[\min\left(X_j\right),\max\left(X_j\right)|,j=1,\cdots,d\right\}\right.\end{aligned}$$

把这里的每一层都拼起来, 每一面的长度$l=2\times\max_{j=1}\max_{i=1}\left\{\left|x_{ij}\right|\right\}$, 我们可以得到如下超立方体:

$$H_d(l)=\left\{x=(x_1,x_2,\cdots,x_d)^T\,|\forall i,x_i\in[-l/2,l/2]\right\}$$

这就是高维数据的一种表达, 我们需要尽可能多地保留高维数据的关键特性, 然后进行降维.

3 主成分分析(Principal components analysis, PCA)

3.1 概述

主成分分析法(Principal component analysis) 是一种数据降维算法, 主要思想是将n维特征映射到全新的正交特征k维上. PCA从原始的空间中顺序地找一组相互正交的坐标轴, 新的坐标轴的选择与数据是密切相关的. 其中, 第一个新坐标轴是原始数据中方差最大的方向, 第二个新坐标轴是与第一个坐标轴正交的平面中使得方差最大的, 第三个轴是与第一, 二个轴正交的平面中方差最大的, 依次类推...可以得到n个这样的坐标轴. 保留包含绝大部分方差的特征维度, 而忽略包含方差几乎为0的特征维度, 实现对数据特征的降维处理.

3.2 数学推导

其实说简单一点, 就是寻找低维度数据最大方差方向的算法. 最大限度地提高数据集的方差

也就是最小化如下函数:

$$J(e) = \sum_{i=1}^{n} \left\| x'_k - x_k \right\|^2$$

不妨把投影到低维的那条线的方向向量设为e, 设数据点x_k对应的投影长度为a_k, (如图1)

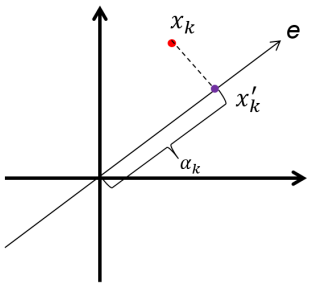


图 1: 投影示意图

有:

$$\begin{aligned} J(e) &= \sum_{i=1}^{n} \|\alpha_k e - x_k\|^2 \\ &= \sum_{i=1}^{n} \alpha_k^2 \|e\|^2 - 2\sum_{i=1}^{n} \alpha_k e^T x_k + \sum_{i=1}^{n} \|x_k\|^2 \\ &= -\sum_{i=1}^{n} \alpha_k^2 + \sum_{i=1}^{n} \|x_k\|^2 \\ &= -\sum_{i=1}^{n} e^T x_k x_k^{\mathrm{t}} e + \sum_{i=1}^{n} \|x_k\|^2 \end{aligned}$$

不妨令协方差矩阵:

$$S = \sum_{i=1}^{n} x_k x_k^T$$

所以我们最后的目标就是最大化(注意前面的负号)$e^T S e$, 且使得 $\|e\| = 1$.

这是带条件的约束问题, 就像老师上课讲的那样, 用拉格朗日乘数法:

$$\begin{aligned} u &= e^T S e - \lambda\left(e^T e - 1\right) \\ \tfrac{\partial u}{\partial e} &= 2Se - 2\lambda e \end{aligned}$$

然后就是解特征值, 做一次特征值分解, 然后找第二个, 这时候就要强制每一个特征值对应一个特征向量, 按照特征值大小排序, 找前 k 个就可以了, 他们本身就是正交的.

求解最大特征值对应特征向量, 就是答案了.

给出主成分分析的算法伪代码:

PCA$(\boldsymbol{D}, \alpha)$

$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$ // 计算均值

$Z = \boldsymbol{D} - \mathbf{1} \cdot \boldsymbol{\mu}^{\mathrm{T}}$ // 数据中心化, **这是我刚刚在上面没有提到的**

$S = \frac{1}{n}\left(\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{Z}\right)$ // 计算协方差矩阵

$(\lambda_1, \lambda_2, \cdots, \lambda_d)$ = 特征值 // 计算出所有特征值

$f(r) = \frac{\sum_{i=1}^{r}\lambda_i}{\sum_{i=1}^{d}\lambda_i}(r = 1, 2, \cdots, d)$ // 在总方差的比例

然后选择最小的r, 使得$f(r) \geqslant \alpha$ // 这里加了一步选择更合适的维数

$A = \left\{a_i | a_i = U_r^{\mathrm{T}} x_i, i = 1, \cdots, n\right\}$ // 最后结果, **其中U_r 是降维后的基向量**

3.3 如何降维-从数学角度

注意到高等代数里的内容(捂脸), 主成分分析其实就是一个正交基变换, **使总方差是每个特征向量上的方差的和组成, 且所有协方差为零, 下面就来说说是为什么**.

很容易看出, 所有特征向量拼起来(不妨设k个k维特征向量), 构成一个正交阵, 令为$\mathscr{A}$, 由正交阵性质有$\mathscr{A}^{-1} = \mathscr{A}^T$.

上面我定义的协方差矩阵(令特征向量α_i):

$$S\alpha_i = \lambda_i\alpha_i$$

来一波数学变换有: $S\mathscr{A} = \mathscr{A}\Lambda$**, 其中**$\Lambda$ **是所有特征值构成的对角阵(前面已经假设了特征值矩阵可逆).**

$$\Leftrightarrow \mathscr{A}^T S\mathscr{A} = \mathscr{A}^T\mathscr{A}\Lambda = \Lambda$$

也就是说, $\mathscr{A}$的基变换使协方差矩阵变成一个对角阵, 这就是前面说的协方差都消失了, 只有在主方向(就是特征向量)上面有方差, 而且注意我们取新的基U_i, 有

$$U^T\mathscr{A}^{-1}U = \mathscr{E} \qquad \text{其中}\mathscr{E}\text{ 是单位阵.}$$

对比三维空间椭球的表达式, 可以发现这是一个"广义"的椭球, 在 k 维空间上的.

其中所有特征向量的方向就是主轴的方向, 其中 $\sqrt{\lambda_i}$(标准差) 是半轴的长度.

所以有一个很鲜明的几何特征了, 在超椭球主轴上的一些个投影.

再有更精确的如何降维的数学分析, 首先定义**投影矩阵:**

$$\boldsymbol{P}_r = \boldsymbol{U}_r\boldsymbol{U}_r^{\mathrm{T}} = \sum_{i=1}^{r}\boldsymbol{u}_i\boldsymbol{u}_i^{\mathrm{T}}$$

其中$\boldsymbol{U}$ 是我们选取的投影的向量构成的矩阵, r 就是上面说的r个特征向量.

求出投影方差:

$$\mathrm{var}(\boldsymbol{A}) = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{P}_r\boldsymbol{x}_i = \sum_{i=1}^{r}\boldsymbol{u}_i^{\mathrm{T}}S\boldsymbol{u}_i = \sum_{i=1}^{r}\lambda_i$$

其中S 是刚刚定义的协方差矩阵, 这就很明显了, 总的投影方差就是S的前 r 个最大特征值之和.

对于均方误差(MSE):

$$\begin{aligned}\mathrm{MSE} &= \frac{1}{n}\sum_{i=1}^{n}\left\|\boldsymbol{x}_i - \boldsymbol{x}_i'\right\|^2 \\ &= \mathrm{var}(\boldsymbol{D}) - \mathrm{var}(\boldsymbol{A})\end{aligned}$$

让var 最大, 就是让MSE 最小.

4 近邻成分分析(Neighbourhood components analysis, NCA)

4.1 概述

基本定义 近邻成分分析(Neighbourhood Components Analysis)是一种 k-nearest neighbours(KNN)下 learning a Mahalnobis distance 算法, 距离学习测度算法. 最大限度地提高了KNN随机变量训练集上的leave one out (LOO)误差. NCA还可以学习标记数据的低维线性嵌入, 以提高KNN分类速度. NCA分类模型是非参数的, 没有对类分布的形状或它们之间的边界做任何假设.

4.2 预备知识-KNN算法

那么就先简单说一下k-Nearest Neighbours Algorithm: KNN是通过测量不同特征值之间的距离进行分类. 它的思路是：如果一个样本在特征空间中的k个最相似(即特征空间中最邻近)的样本中的大多数属于某一个类别, 则该样本也属于这个类别.

KNN算法中, 所选择的邻居都是已经正确分类的对象. 在定类决策上只依据最邻近的一个或者几个样本的类别来决定待分样本所属的类别.

思想其实很简单, 有一堆已经分类好的数据, 现在加入一个未分类的数据, 只需要根据它的邻居大多数是哪一类的来确定这个数据被分类在哪里.

但是KNN存在两个问题:

- 首先, 对点进行分类可能在计算上比较昂贵, 因为它们必须存储在整个训练集中并进行比较.
- 其次, 如何确定距离度量来定义”最近”点, 这影响分类情况.

4.3 数学推导

NCA就是来优化上面两个问题的, 首先我们引入为了降维, NCA定义的一个二次距离度量:

定义 the quadratic distance metric $d(x,y) = (x-y)^T Q(x-y) = (Ax-Ay)^T(Ax-Ay)$

并且NCA使用stochastic neighbour assignment, 也就是训练集中的每一个数据都是邻居, 但是概率不同:

定义 Stochastic Nearest Neighbours' p_{ij} **&** p_i :

$$p_{ij} = \begin{cases} \frac{e^{-\left\|Ax_i - Ax_j\right\|^2}}{\sum_k e^{-\left\|Ax_i - Ax_k\right\|^2}}, & \text{if } j \neq i \\ 0, & \text{if } j = i \end{cases}$$

$$p_i = \sum_{j \in c_i} p_{ij} \text{ where } c_i = \left\{ j | c_i = c_j \right\}$$

为了让在同一个类里面的正确概率最大化, 定义了一个类似Cost Function:

$$f(A) = \sum_i \sum_{j \in C_i} p_{ij} = \sum_i p_i$$

运用梯度下降算法(Gradient descent), 其实就是又变成了一个优化问题, 有一个学习的step, 往更优的地方走, 这里的Cost Function 不是凸函数, 因此优化时必须处理局部最优点, 我们处理$f(A)$:

$$\frac{\partial f}{\partial A} = 2A \sum_i \left(p_i \sum_k p_{ik} x_{ik} x_{ik}^T - \sum_{j \in C_i} p_{ij} x_{ij} x_{ij}^T \right)$$

我们可以发现优化的点在哪里: NCA允许线性降维输入数据. 这就是计算上的优势, 并且相对不受过度拟合的影响.

总结一下, NCA在功能上其和k近邻算法的目的相同, 直接利用随即近邻的概念确定与测试样本临近的有标签的训练样本, 根据一种给定的距离度量算法对样本数据进行度量, 然后对多元变量数据进行分类.

4.4 如何降维

NCA根据一种给定的距离度量算法对样本数据进行分类, 就是一种特征选取, 降低时间的复杂度和空间复杂度, 本质就是找到一个分类映射, 把高维度的数据映射到低纬度.

5 线性判别分析(Linear discriminant analysis, LDA)

5.1 概述

线性判别分析(Linear Discriminant Analysis)是将带上标签的数据, 通过投影的方法, 投影到维度更低的空间中, 使得投影后的点会形成按类别区分, 一簇一簇的情况, 相同类别的点, 将会在投影后的空间中更接近. LDA试图找到一个特征子空间, 使类的可分性最大化, 我们引入如下定义:

定义 Fisher判别准则

$$J = \frac{|\mu_1 - \mu_2|^2}{S_1^2 + S_2^2}$$

J 可以衡量数据投影后分开的程度

其中 $|\mu_1 - \mu_2|^2$ 表示不同class之间的距离, $S_1^2 + S_2^2$ 表示每个class里的scatter.

5.2 数学推导-运用Fisher判别准则

考虑简单的两类数据投影问题:
设投影前的坐标(都是用矩阵处理):

$$\mu_{\mathrm{i}} = \frac{1}{\mathrm{N_i}} \sum_{\mathrm{x} \in \omega_{\mathrm{i}}} \mathrm{x}$$

首先写出投影过后的坐标表达, 这里$w^T x$ 就是做了一个投影.

$$\widetilde{\mu}_{\mathrm{i}} = \frac{1}{\mathrm{N_i}} \sum_{\mathrm{y} \in \omega_{\mathrm{i}}} \mathrm{y} = \frac{1}{N_i} \sum_{x \in \omega_i} w^T x = w^T \mu_i$$

代入 J 表达式中:

$$(\widetilde{\mu}_1 - \widetilde{\mu}_2)^2 = \mathrm{w}^T \underbrace{(\mu_1 - \mu_2)(\mu_1 - \mu_2)^{\mathrm{T}}}_{\mathrm{S_{Between}}} \mathrm{w} = \mathrm{W^T S_{Between} W}$$

这样分子就表达出来了, 接下来写分母, (同样是处理投影过后的 $\widetilde{\mathrm{s}_i}$)

$$\widetilde{\mathrm{s}_i^2} = \sum_{\mathrm{y} \in \omega_{\mathrm{i}}} (\mathrm{y} - \widetilde{\mu}_{\mathrm{i}})^2 = \sum_{\mathrm{x} \in \omega_{\mathrm{i}}} \mathrm{w}^T (\mathrm{x} - \mu_{\mathrm{i}})(\mathrm{x} - \mu_{\mathrm{i}})^{\mathrm{T}} \mathrm{W} = \mathrm{W}^T \mathrm{S_i W}$$

为了更加精简的形式, 不妨让

$$\mathrm{S_1 + S_2 = S_w}$$

所以有:

$$\widetilde{\mathbf{s}_1^2} + \tilde{\mathbf{s}}_2^2 = \mathrm{W^T S_{Within} W}$$

最后得到:

$$J(W) = \frac{W^T S_{Between} W}{W^T S_{Within} W}$$

这样就好办了, 把 J 表达成了一个关于W(矩阵)的函数, 接下来可以用矩阵求导去优化.
(掏出*Matrix Cookbook*...)
最后得到:

$$S_{Within}^{-1} S_{Between} w = Jw$$

这样就变成了特征值问题

但是注意下面的标量:

$$R = (\mu_1 - \mu_2)^T w$$

所以:

$$J_W = S_{Within}^{-1} (\mu_1 - \mu_2) R$$

所以我们不断优化$S_W^{-1} (\mu_1 - \mu_2)$ 就可以了.

当需要推广的超过两类的数据情况, 显然S_{Within} 的计算方法是一样的, 分别算出每一类里面的散度矩阵即可, 而对于$S_{Between}$:

$$S_{Between} = \sum_{i=1}^{C} N_i (\mu_i - \mu) (\mu_i - \mu)^T$$

$$\text{其中 } \mu = \frac{1}{N} \sum_{\forall x} x = \frac{1}{N} \sum_{x \in \omega_i} N_i \mu_i$$

5.3 如何降维

LDA使得投影后的样本尽可能可分. 它通过在k维空间选择一个投影超平面, 使得不同类别在该超平面上的投影之间的距离尽可能近, 同时不同类别的投影之间的距离尽可能远. 从而试图明确地模拟数据类之间的差异.

这里完全可以像上节PCA一样写一个类似的数学角度分析, 抓住两个矩阵: S_{Within} 和$S_{Between}$, 类内和类间散度矩阵, 类似上面PCA的分析, 这里选取的最优向量是 $S_{Within}^{-1} S_{Between}$ 的主特征向量. 其他还有不同的就是这里是一个泛化的特征值问题, 就是不断保证$J(W) = \frac{W^T S_{Between} W}{W^T S_{Within} W}$的最大化目标的特征值. J 衡量数据投影后分开的程度就很好地体现了降维的思想.

6 LDA与PCA的分析对比

- 1. 从motivation上看, LDA处理的是**有标签**的数据, 是监督学习[1], 目的是让数据在降维之后尽量的按类别区分开, 而PCA处理**无标签**的数据, 是无监督学习[2], 把高维空间中的数据降维并寻找投影方向使方差最大.

  如图2, 我们可以看到LDA做出的投影轴上1和2两类分得很开, 但是在PCA的眼睛里是没有1和2类这两个标签的, PCA做的投影轴是全部数据分散得最开的那个方向, 在这个方向上保留了最多的数据信息, 数据损失最少.

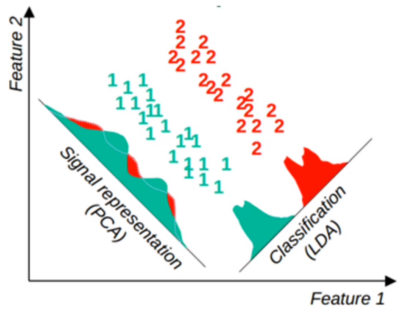


图 2: Difference between LDA & PCA

  再从数学层面来理解这里的不同, 首先看看LDA和PCA都是在处理什么函数:

$$J(\omega)_{LDA} = \frac{|\mu_1 - \mu_2|^2}{S_1^2 + S_2^2}, \quad J(e)_{PCA} = \sum_{i=1}^{n} \left\| x_k' - x_k \right\|^2$$

  $J(\omega)_{LDA}$ 考虑的是两类之间的关系, $|\mu_1 - \mu_2|^2$ 是不同类中心的距离, 要做的是尽可能让这个值大, 就是每一类分尽量开, 但是$S_1^2 + S_2^2$ 指的是每一类里的散度, 我们又希望这个值尽量小, 也就是每一类尽量集中.

  $J(e)_{PCA}$ 很明显就是考虑所有数据之间的关系了, 最大化这个值⇒ 让所有数据分得尽量开.

- 2. 从处理效果上看, LDA将每个类的中心之间的距离最大化, 也就是区分每一个类别的特性, 并且在每个类别中聚集程度最大化.

  而PCA不会选择一组特性并丢弃其他特性, 但是它会推断出一些新特性, 这些新特性能够从现有特性中最好地描述类的类型[3].

[1]监督学习在斯坦福吴恩达老师网课上分为回归问题和分类问题, 这里的分类指试图在离散输出中预测结果, 也就是我们正在尝试将输入变量映射到离散类别.

[2]无监督学习是指在的目标是不给任何额外提示的情况下(无标签), 能对观察值进行分类或者区分等.

[3]这里是指对所有的数据处理, 不区分类别

- 3. LDA 不能保证投影到的坐标系是正交的（根据类别的标注, 关注的是分类能力）, LDA可以用于降维, 还可用于分类.
- 4. LDA中 $S_{Between}$ 的 $rank <= C - 1$ 不能像PCA一样想降到几维就降到几维. 而且S_{Within} 的逆可能不存在: 当样本点个数低于维度, 生成的散读矩阵不是满秩的.

  总的来说: PCA为非监督降维, LDA为有监督降维, PCA希望投影后的数据方差尽可能的大（最大可分性）, 因为其假设方差越多, 则所包含的信息越多; 而LDA则希望投影后相同类别的组内方差小, 而组间方差大. LDA能合理运用标签信息, 使得投影后的维度具有判别性, 不同类别的数据尽可能的分开.

7 NCA和PCA的分析对比

NCA的核心就在学习一种距离测度$d(x, y)$来表征x和y之间的某种相似度, 其本质是通过对样本进行线性或非线性变换(变换过程保留样本的流形)获取另一种更有类别区分度的表示形式, 更好地反映样本的空间特性, 使邻近分类的效果尽可能最优.

其实就是另一种方式的度量学习, 低维空间对应了在样本属性上定义的距离度量, **本质上就是寻找一个合适的距离度量(合适的低维空间).**

NCA和PCA, LDA的差别还是比较大的, 宽泛地说一个是选择合适的度量, 另外两个是找到合适的投影向量.

参考文献

https://stats.stackexchange.com/questions/164621/limitation-of-lda-latent-dirichlet-alloca

https://sebastianraschka.com/faq/docs/lda-vs-pca.html

https://www.youtube.com/watch?v=M4HpyJHPYBY

https://www.cnblogs.com/zuoshoushizi/p/9242050.html

https://blog.csdn.net/fjssharpsword/article/details/68059285

https://blog.csdn.net/tdj8866/article/details/78539024

https://medium.com/@adi.bronshtein/a-quick-introduction-to-k-nearest-neighbors-algorithm-6

https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clusterin

https://www.youtube.com/watch?v=UqYde-LULfs

https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

http://www.cs.toronto.edu/~fritz/absps/nca.pdf

Matrix cookbook

数据挖掘概念与技术