作者：凯鲁嘎吉 - 博客园 http://www.cnblogs.com/kailugaji/

1. 主成成分分析

- 最大可分性：样本点在这个超平面上的投影能尽可能分开。
  - PCA可以被定义为数据在低维线性空间上的正交投影，这个线性空间被称为主子空间(principal subspace)，使得投影数据的方差被最大化。
- 最近重构性：样本点到这个超平面的距离都足够近。
  - 它也可以被定义为使得平均投影代价最小的线性投影。平均投影代价是指数据点和它们的投影之间的平均平方距离 。

假定数据样本进行了中心化, 即 $\sum_i \boldsymbol{x}_i = \mathbf{0}$; 再假定投影变换后得到的新坐标系为 $\{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_d\}$, 其中 $\boldsymbol{w}_i$ 是标准正交基向量, $||\boldsymbol{w}_i||_2 = 1$, $\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{w}_j = 0$ $(i \neq j)$. 若丢弃新坐标系中的部分坐标, 即将维度降低到 $d' < d$, 则样本点 $\boldsymbol{x}_i$ 在低维坐标系中的投影是 $\boldsymbol{z}_i = (z_{i1}; z_{i2}; \ldots; z_{id'})$, 其中 $z_{ij} = \boldsymbol{w}_j^{\mathrm{T}}\boldsymbol{x}_i$ 是 $\boldsymbol{x}_i$ 在低维坐标系下第 j 维的坐标. 若基于 $\boldsymbol{z}_i$ 来重构 $\boldsymbol{x}_i$, 则会得到 $\hat{\boldsymbol{x}}_i = \sum_{j=1}^{d'} z_{ij}\boldsymbol{w}_j$.

考虑整个训练集, 原样本点 $\boldsymbol{x}_i$ 与基于投影重构的样本点 $\hat{\boldsymbol{x}}_i$ 之间的距离为

$$\begin{aligned}\sum_{i=1}^{m}\left\|\sum_{j=1}^{d'} z_{ij}\boldsymbol{w}_j - \boldsymbol{x}_i\right\|_2^2 &= \sum_{i=1}^{m}\boldsymbol{z}_i{}^{\mathrm{T}}\boldsymbol{z}_i - 2\sum_{i=1}^{m}\boldsymbol{z}_i{}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}\boldsymbol{x}_i + \text{const} \\ &\propto -\mathrm{tr}\left(\mathbf{W}^{\mathrm{T}}\left(\sum_{i=1}^{m}\boldsymbol{x}_i\boldsymbol{x}_i^{\mathrm{T}}\right)\mathbf{W}\right). \qquad (10.14)\end{aligned}$$

根据最近重构性, 式(10.14)应被最小化, 考虑到 $\boldsymbol{w}_j$ 是标准正交基, $\sum_i \boldsymbol{x}_i\boldsymbol{x}_i^{\mathrm{T}}$ 是协方差矩阵, 有

$$\begin{aligned}\min_{\mathbf{W}} \quad & -\mathrm{tr}\left(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W}\right) \qquad (10.15)\\ \text{s.t.} \quad & \mathbf{W}^{\mathrm{T}}\mathbf{W} = \mathbf{I}.\end{aligned}$$

这就是主成分分析的优化目标.

从最大可分性出发, 能得到主成分分析的另一种解释. 我们知道, 样本点 $\boldsymbol{x}_i$ 在新空间中超平面上的投影是 $\mathbf{W}^{\mathrm{T}}\boldsymbol{x}_i$, 若所有样本点的投影能尽可能分开, 则应该使投影后样本点的方差最大化, 如图 10.4 所示.

投影后样本点的方差是 $\sum_i \mathbf{W}^{\mathrm{T}}\boldsymbol{x}_i\boldsymbol{x}_i^{\mathrm{T}}\mathbf{W}$, 于是优化目标可写为

$$\begin{aligned} \max_{\mathbf{W}} \quad & \operatorname{tr}\left(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W}\right) \\ \text{s.t.} \quad & \mathbf{W}^{\mathrm{T}}\mathbf{W}=\mathbf{I}\,, \end{aligned} \tag{10.16}$$

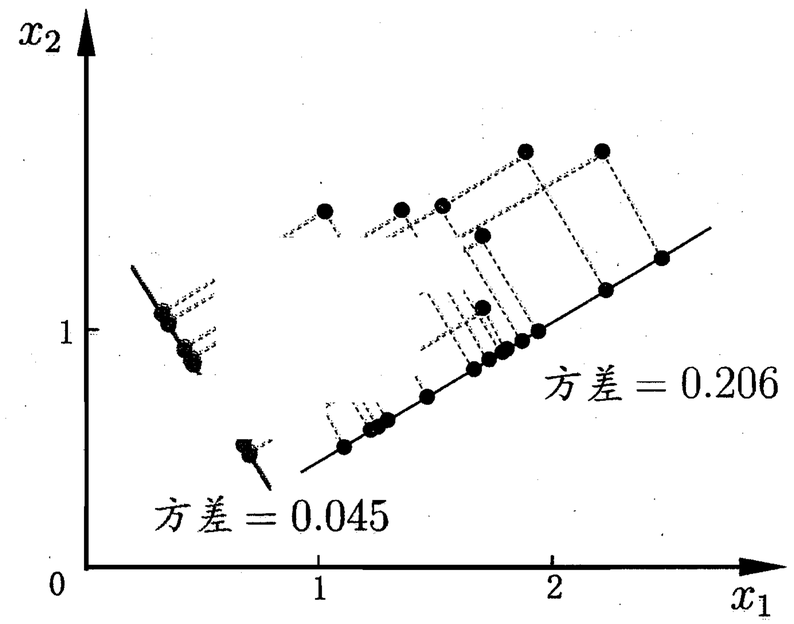


图 10.4 使所有样本的投影尽可能分开(如图中红线所示), 则需最大化投影点的方差

显然, 式(10.16)与(10.15)等价.

对式(10.15)或(10.16)使用拉格朗日乘子法可得

$$\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W} = \lambda\mathbf{W} \ , \tag{10.17}$$

于是, 只需对协方差矩阵 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$ 进行特征值分解, 将求得的特征值排序: $\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_d$, 再取前 d' 个特征值对应的特征向量构成 $\mathbf{W} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d'})$. 这就是主成分分析的解. PCA 算法描述如图 10.5 所示.

输入: 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
低维空间维数 d'.

过程:

1: 对所有样本进行中心化: $\boldsymbol{x}_i \leftarrow \boldsymbol{x}_i - \frac{1}{m}\sum_{i=1}^{m}\boldsymbol{x}_i$;
2: 计算样本的协方差矩阵 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$;
3: 对协方差矩阵 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$ 做特征值分解;
4: 取最大的 d' 个特征值所对应的特征向量 $\boldsymbol{w}_1,\boldsymbol{w}_2,\ldots,\boldsymbol{w}_{d'}$.

输出: 投影矩阵 $\mathbf{W}=(\boldsymbol{w}_1,\boldsymbol{w}_2,\ldots,\boldsymbol{w}_{d'})$.

降维后低维空间的维数 d' 通常是由用户事先指定, 或通过在 d' 值不同的低维空间中对 k 近邻分类器(或其他开销较小的学习器) 进行交叉验证来选取较好的 d' 值. 对 PCA, 还可从重构的角度设置一个重构阈值, 例如 $t=95\%$, 然后选取使下式成立的最小 d' 值:

$$\frac{\sum_{i=1}^{d'}\lambda_i}{\sum_{i=1}^{d}\lambda_i} \geqslant t \ . \tag{10.18}$$

2. MATLAB解释

详细信息请看：Principal component analysis of raw data - mathworks

[coeff,score,latent,tsquared,explained,mu] = pca(X)

coeff = pca(X) returns the principal component coefficients, also known as loadings, for the n-by-p data matrix X. Rows of X correspond to observations and columns correspond to variables.

The coefficient matrix is p-by-p.

Each column of coeff contains coefficients for one principal component, and the columns are in descending order of component variance.

By default, pca centers the data and uses the singular value decomposition (SVD) algorithm.

coeff = pca(X,Name,Value) returns any of the output arguments in the previous syntaxes using additional options for computation and handling of special data types, specified by one or more Name,Value pair arguments.

For example, you can specify the number of principal components pca returns or an algorithm other than SVD to use.

[coeff,score,latent] = pca(___) also returns the principal component scores in score and the principal component variances in latent.

You can use any of the input arguments in the previous syntaxes.

Principal component scores are the representations of X in the principal component space. Rows of score correspond to observations, and columns correspond to components.

The principal component variances are the eigenvalues of the covariance matrix of X.

[coeff,score,latent,tsquared] = pca(___) also returns the Hotelling's T-squared statistic for each observation in X.

[coeff,score,latent,tsquared,explained,mu] = pca(___) also returns explained, the percentage of the total variance explained by each principal component and mu, the estimated mean of each variable in X.

coeff: X矩阵所对应的协方差矩阵的所有特征向量组成的矩阵，即变换矩阵或投影矩阵，coeff每列代表一个特征值所对应的特征向量，列的排列方式对应着特征值从大到小排序。

source: 表示原数据在各主成分向量上的投影。但注意：是原数据经过中心化后在主成分向量上的投影。

latent: 是一个列向量，主成分方差，也就是各特征向量对应的特征值，按照从大到小进行排列。

tsquared: X中每个观察值的Hotelling的T平方统计量。Hotelling的T平方统计量（T-Squared Statistic）是每个观察值的标准化分数的平方和，以列向量的形式返回。

explained: 由每个主成分解释的总方差的百分比，每一个主成分所贡献的比例。explained = 100*latent/sum(latent)。

mu: 每个变量X的估计平均值。

3. MATLAB程序

3.1 方法一：指定降维后低维空间的维度k

```
function [data_PCA, COEFF, sum_explained]=pca_demo_1(data,k)
% k:留k个主成分
data=zscore(data);  %归一化数据
[COEFF,SCORE,latent,tsquared,explained,mu]=pca(data);
latent1=100*latent/sum(latent);%将latent特征值总和统一为100，便于观察贡献率
data= bsxfun(@minus,data,mean(data,1));
data_PCA=data*COEFF(:,1:k);
pareto(latent1);%调用matlab画图 pareto仅绘制累积分布的前95%，因此y中的部分元素并未显示
xlabel('Principal Component');
ylabel('Variance Explained (%)');
% 图中的线表示的累积变量解释程度
print(gcf,'-dpng','PCA.png');
sum_explained=sum(explained(1:k));
```

3.2 方法二：指定贡献率percent_threshold

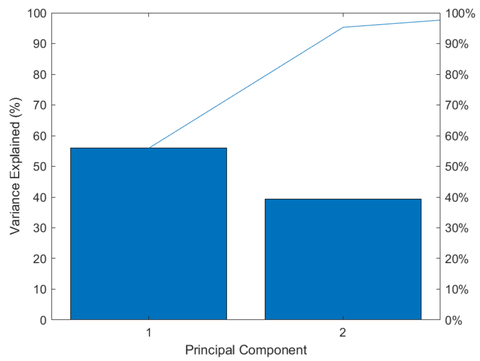
```
function [data_PCA, COEFF, sum_explained, n]=pca_demo_2(data)
%用percent_threshold决定保留xx%的贡献率
percent_threshold=95;   %百分比阈值，用于决定保留的主成分个数；
data=zscore(data);  %归一化数据
[COEFF,SCORE,latent,tsquared,explained,mu]=pca(data);
latent1=100*latent/sum(latent);%将latent特征值总和统一为100，便于观察贡献率
A=length(latent1);
percents=0;                         %累积百分比
for n=1:A
    percents=percents+latent1(n);
    if percents>percent_threshold
```

```
        break;
    end
end
data= bsxfun(@minus,data,mean(data,1));
data_PCA=data*COEFF(:,1:n);
pareto(latent);%调用matlab画图 pareto仅绘制累积分布的前95%，因此y中的部分元素并未显示
xlabel('Principal Component');
ylabel('Variance Explained (%)');
% 图中的线表示的累积变量解释程度
print(gcf,'-dpng','PCA.png');
sum_explained=sum(explained(1:n));
```

4. 结果

数据来源于MATLAB自带的数据集hald

```
>> load hald
>> [data_PCA, COEFF, sum_explained]=pca_demo_1(ingredients,2)

data_PCA =

   -1.4672378022580830   -1.9030357084255600
   -2.1358287463988750   -0.2383537022721984
    1.1298704738334220   -0.1838771541925830
   -0.6598958489750766   -1.5767742099657470
    0.3587645564703510   -0.4833787858589940
    0.9666396396922070   -0.1699440281036510
    0.9307051170772330    2.1348163511997477
   -2.2321379968848360    0.6916706828275924
   -0.3515155959755610    1.4322450694434040
    1.6025430141320206   -1.8280966432200180
   -1.6401799529266685    1.2951127514269280
    1.6925940911826333    0.3922488215304800
    1.7456786911649580    0.4373254879144250


COEFF =

    0.4759551727489700   -0.5089793848064100    0.6755001879642850    0.2410521840510940
    0.5638702421919940    0.4139314871269850   -0.3144204428192920    0.6417560744272130
   -0.3940665539093030    0.6049690784714390    0.6376910918065660    0.2848661102945330
   -0.5479311912608630   -0.4512351093300160   -0.1954209626117080    0.6767340194812840


sum_explained =

   95.2942526284391530

>> [data_PCA, COEFF, sum_explained, n]=pca_demo_2(ingredients)

data_PCA =

   -1.4672378022580830   -1.9030357084255600
   -2.1358287463988750   -0.2383537022721984
    1.1298704738334220   -0.1838771541925830
   -0.6598958489750766   -1.5767742099657470
    0.3587645564703510   -0.4833787858589940
    0.9666396396922070   -0.1699440281036510
    0.9307051170772330    2.1348163511997477
   -2.2321379968848360    0.6916706828275924
   -0.3515155959755610    1.4322450694434040
    1.6025430141320206   -1.8280966432200180
   -1.6401799529266685    1.2951127514269280
    1.6925940911826333    0.3922488215304800
    1.7456786911649580    0.4373254879144250


COEFF =

    0.4759551727489700   -0.5089793848064100    0.6755001879642850    0.2410521840510940
    0.5638702421919940    0.4139314871269850   -0.3144204428192920    0.6417560744272130
   -0.3940665539093030    0.6049690784714390    0.6376910918065660    0.2848661102945330
   -0.5479311912608630   -0.4512351093300160   -0.1954209626117080    0.6767340194812840


sum_explained =

   95.2942526284391530


n =

     2
```



5. 参考

[1] 周志华,《机器学习》.

[2] MATLAB实例：PCA降维