主成分分析

1 主成分分析

1.1 总体主成分

主成分分析使用正交变换将线性相关的观测数据转换为少数几个线性无关变量，同时使得少数几个线性无关的变量可以尽可能的提取出原始变量的信息。主成分分析使用方差来度量变量的信息量。

假设数据为 $x=(x_1,x_2,...,x_m)^T$，其均值为 $\mu=E(x)=(\mu_1,\mu_2,...,\mu_m)$，协方差矩阵为 $\Sigma=cov(x,x)=E[(x-\mu)(x-\mu)^T]$。主成分分析使用线性变换改变原始数据：$y_i=\alpha_i^T x, i=1,2,...,m$，其中 α_i 需要满足以下三个条件：

1. 系数向量 α_i 是单位向量，即 $\alpha_i^T\alpha_i=1, i=1,2,...,m$
2. 变量 y_i 和 y_j 互不相关，即 $cov(y_i,y_j)=0(i\neq j)$
3. 变量 y_1 是 x 所有线性变换中方差最大的；y_k 是与 $y_{k-1},y_{k-2},...,y_1$ 都不相关的 x 的所有线性变换中方差最大的。

我们将 $y_i,y_2,...,y_m$ 分别称为第一主成分，第二主成分直至第 m 主成分。主成分可以通过以下方式求得：

首先求第一主成分 $y_1=\alpha_1^T x$。由于 $Var(y_1)=\alpha_1^T\Sigma\alpha_1$，所以易知该问题和以下带约束优化问题相同：

$$\begin{aligned}\max_{\alpha_1} \quad & \alpha_1^T\Sigma\alpha_1 \\ s.t. \quad & \alpha_1^T\alpha_1=1\end{aligned}$$

构建拉格朗日函数 $L(\alpha_1,\lambda)=\alpha_1^T\Sigma\alpha_1-\lambda(\alpha_1^T\alpha_1-1)$。对 α_1 求导，并令其等于 0，可以得到：

$$\frac{\partial L}{\partial\alpha_1}=2\Sigma\alpha_1-2\lambda\alpha_1=0$$

因此，λ 是 Σ 的特征值，α_1 是其对应的特征向量。因此目标函数可以简化为 $\alpha_1^T\Sigma\alpha_1=\alpha_1^T\lambda\alpha_1=\lambda\alpha_1^T\alpha_1=\lambda$。因此，此处 λ 应为 Σ 的最大特征值 λ_1，α_1 为其对应的特征向量。

x 的第二主成分同样可以转化为一个带约束的优化问题。注意到 $0=Cov(y_1,y_2)=Cov(\alpha_1x,\alpha_2x)=\alpha_2^T\Sigma\alpha_1=\alpha_2^T\lambda\alpha_1$，因此该优化问题有约束条件 $\alpha_2^T\alpha_1=\alpha_1^T\alpha_2=0$：

$$\begin{aligned}\max_{\alpha_2} \quad & \alpha_2^T \Sigma \alpha_2 \\ s.t. \quad & \alpha_2^T \alpha_2 = 1 \\ & \alpha_2^T \alpha_1 = \alpha_1^T \alpha_2 = 0\end{aligned}$$

构建拉格朗日函数 $L(\alpha_2, \lambda, \phi) = \alpha_2^T \Sigma \alpha_2 - \lambda(\alpha_2^T \alpha_2 - 1) - \phi \alpha_2^T \alpha_1$。对 α_2 求导，并令其等于 0，有：

$$\frac{\partial L}{\partial \alpha_2} = 2\Sigma\alpha_2 - 2\lambda\alpha_2 - \phi\alpha_1 = 0$$

左乘 α_1^T，有 $0 = 2\alpha_1^T \Sigma \alpha_2 - 2\lambda \alpha_1^T \alpha_2 - \phi \alpha_1^T \alpha_1 = -\phi$。因此 $\phi = 0$，代入上式，有 $\Sigma\alpha_2 - \lambda\alpha_2 = 0$。代入目标函数，同样可以得到 α_2 是 Σ 的第二大特征值 λ_2 对应的单位特征向量。

以此类推，可以得到：x 的第 k 主成分是 $\alpha_k^T x$，且 $Var(\alpha_k^T x) = \lambda_k$。此处 λ_k 是 Σ 的第 k 大的特征值，α_k 是其对应的特征向量。

总体主成分具有以下性质：

1. 总体主成分 y 的协方差矩阵是对角矩阵：$Cov(y) = \Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_m)$
2. 总体主成分 y 的方差之和等于随机变量 x 的方差之和：$\sum_{i=1}^m \lambda_i = \sum_{i=1}^m \sigma_{ii}$
3. 第 k 个主成分 y_k 与变量 x_i 的相关系数（称为因子负荷量）为：$\rho(y_k, x_i) = \frac{\sqrt{\lambda_k}\alpha_{ik}}{\sqrt{\sigma_{ii}}}$

主成分分析是一种降维方法，其通过一个线性变换将高维数据压缩为低维数据。可以证明，如果使用线性变换将高维数据压缩为低维数据，主成分分析可以压缩后的数据的方差之和最大。

在使用主成分分析对数据进行降维时，还有以下几个值得注意的地方：

1. 对于主成分的个数的选择：往往使用累计方差贡献率来确定主成分个数。前 k 个主成分的累计贡献率为 $\eta_k = (\sum_{i=1}^k \lambda_i)/(\sum_{i=1}^m \lambda_i)$。一般选择 k 使得 η_k 达到 70% 至 80% 左右
2. 为了防止变量的量纲对求解主成分产生不利影响，往往需要对各个变量进行标准化：$x_i^* = \frac{x_i - E(x_i)}{\sqrt{var(x_i)}}$，并使用 x_i^* 继续进行主成分分析

1.2 样本主成分

在实际应用中，需要在实际观测数据上进行主成分分析，称为样本主成分。样本主成分作用于样本协方差阵 S 或样本相关矩阵 R。

$$S = \frac{1}{n-1}\sum_{i=1}^n (x_i - \bar{x})(x_i - \bar{x})^T$$

$$R = [r_{ij}]_{m\times m}, r_{ij} = \frac{s_{ij}}{\sqrt{s_{ii}s_{jj}}}$$

在得到 S 或 R 后，便可对矩阵进行分解以求得样本主成分，具体步骤如下：

输入：数据矩阵 X

1. 根据数据特征计算样本协方差阵 S 或样本相关矩阵 R
2. 对 S 或 R 进行矩阵分解得到其特征值 $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_m$ 和对应的单位特征向量 $\alpha_1, \alpha_2, ..., \alpha_m$
3. 根据方差贡献率选择主成分个数 k
4. 求出 k 个主成分 $y_i = \alpha_i^T x, i = 1, 2, ..., k$

输出：求出的 k 个主成分 $y_1, y_2, ..., y_k$。

该步骤是从数据的协方差阵或相关矩阵出发，对其进行方阵分解求得主成分。事实上，也可直接从数据矩阵出发，直接进行奇异值分解得到主成分。该做法是：对于矩阵 A，求其截断奇异值分解 $A \approx U_k \Sigma_k V_k^T$，并通过奇异值分解得到主成分。具体做法如下：

输入：$m \times n$ 样本矩阵 X，需要对其进行中心化

1. 若需要进行标准化，令 $X^* = \frac{1}{\sqrt{n-1}} X^T$；若不需要标准化，令 $X^* = X^T$
2. 根据主成分个数 k 对 X^* 进行奇异值分解 $X^* = U_k \Sigma_k V_k^T$
3. 计算前 k 个主成分矩阵 $Y = V^T X$

输出：主成分矩阵 Y。

2 代码实现

本次考虑的数据集为 iris 数据集，其可在 sklearn.datasets 中得到。

2.1 sklearn 实现

准备数据：

```
import numpy as np
from sklearn import datasets
X = datasets.load_iris()['data']
```

主成分分析的接口位于 sklearn.decomposition.PCA, 其文档可见**此处**。其中较为重要的参数有：

- n_components：主成分个数
- whiten：是否对数据标准化

此处数据集的维度是 4，使用主成分分析时，设置主成分的个数为 2.

```
from sklearn.decomposition import PCA
```

```
clf = PCA(n_components=2)
clf.fit(X)
```

2 个主成分的累计贡献率如下所示：

```
print(np.cumsum(clf.explained_variance_ratio_))
```

```
## [0.92461872 0.97768521]
```

可以使用以下语句得到变换后的主成分：

```
Y = clf.fit_transform(X)
```

2.2 主成分分析

```
class Pca:

    def __init__(self,n_components=1,standard=True):
        self.n_components = n_components
        self.standard = standard

    def fit(self,X):
        if self.standard:
            covx = np.corrcoef(X.T)
        else:
            covx = np.cov(X.T)
        u,v = np.linalg.eig(covx)
        self.variance = u
        self.variance_ratio = u/np.sum(u)
        self.trans = v

    def fit_transform(self,X):
        return X.dot(self.trans[:,:self.n_components])
```

结果如下：

```
clf = Pca(n_components=2,standard=False)
clf.fit(X)
print(np.cumsum(clf.variance_ratio))
```

```
## [0.92461872 0.97768521 0.99478782 1.        ]
```

同样，可如下得到变换后的主成分：

```
Y = clf.fit_transform(X)
```