Chapter 7

Principal Component Analysis

Contents

7.1 Introduction

- PCA is a linear dimensionality reduction technique that can be used to simplify a dataset by reducing the number of dimensions in the data.
- It is a linear transformation that chooses a new coordinate system for the dataset such that –
  - The greatest variance by any projection of the dataset lies on the first axis (this is called the first prinicipal component PC_1).
  - The second greatest variance is on the second axis PC_2 and so.
- PCA can be used for reducing dimensionality by eliminating the later principal components

7.2 Benefits

- Large datasets can be summarized into smaller ones that can be easier to analyze and visualize.
- Easy to calculate and compute.
- Identify correlations between data points.

- Can be used in exploratory data analysis,
- Prevents predictive algorithms from data overfitting issues.

7.3 PCA Algorithm

- Given the inputs $x_i \in \mathbb{R}^d$, normalize the data points.
- Compute the $d \times d$ covariance matrix S using the normalized-data matrix $\mathbf{X}_{n \times d}$
- ...
- From the $k \in K$ eigenvalues, pick $\lambda_1 > \lambda_2 > \cdots > \lambda_k$, and its associated eigenvectors $\{v_1, v_2, \ldots, v_k\}$. v_1 is PC_1, v_2 is PC_2, ..., v_k is PC_k,
- The k-dimensional projection of each input is $\mathbf{z}_i = \mathbf{v}_k^T \mathbf{x}_i$ where $\mathbf{v}_k$ are the principal components.
- Larger λ implies higher principal component.
- PC captures the greatest variance of the projection.
- Maximizing the greatest variance of the projection ...

7.4 PCA via Variance Maximization

- Consider projecting the inputs $\mathbf{x}_i \in \mathbb{R}^d$ along directions $\mathbf{v}_k$.
- Projection of $\mathbf{x}_i$ (red points) will be $\mathbf{v}_1^T \mathbf{x}_i$ (textcolorgreengreen points).
- Mean of the projections of all the inputs:

$$\frac{1}{n} \sum_{i=1}^{n} \mathbf{v}_k \ldots$$

- Construct a Lagrangian for this optimization problem:

$$\begin{aligned}
\mathcal{L} &= \mathbf{v}_k^T \mathbf{S} \mathbf{v}_k + \lambda_k (1 - \mathbf{v}_k^T \mathbf{v}_k) \\
\frac{\partial L}{\partial \mathbf{v}_k} &= 0 \\
&= 2 \left(\mathbf{S} \mathbf{v}_k - \lambda_k \mathbf{v}_k \right) \\
&= \mathbf{S} \mathbf{v}_k - \lambda_k \mathbf{v}_k \\
\mathbf{S} \mathbf{v}_k &= \lambda_k \mathbf{v}_k
\end{aligned}$$

- v_k are eigenvectors of the covariance matrix $\mathbf{S}$ with eigenvalues λ_k.

- Thus, variance $\mathbf{v}_k^T \mathbf{S} \mathbf{v}_k$ will be maximum for the largest value of λ_k, since:

$$\begin{aligned}\mathbf{v}_k^T \mathbf{S} \mathbf{v}_k &= \mathbf{v}_k^T \lambda_k \mathbf{v}_k \\ &= \lambda_k \mathbf{v}_k^T \mathbf{v}_k \\ &= \lambda_k\end{aligned}$$

- If λ_1 is the largest eigenvalue, then $\mathbf{v}_1$ is the corresponding eigenvector, also known as the first principal component.

7.5 PCA via Minimizing Reconstruction Error

$$\arg_{\mathbf{v}_k} \max \frac{1}{n} \tag{7.1}$$

7.6 PCA via Single Value Decomposition

Any matrix $\mathbf{X}_{n\times d}$ can have a SVD such that $\mathbf{X}_{n\times d} = \mathbf{U}_{n\times n} \Lambda_{n\times d} \mathbf{V}_{d\times d}^T$

- $\mathbf{U}$ is a matrix of left singular vectors i.e., columns of $\mathbf{U}$ are eigenvectors of $\mathbf{X}\mathbf{X}^T$.
- $\mathbf{V}$ is a matrix of right singular vectors i.e., columns of $\mathbf{V}$ are eigenvectors of $\mathbf{X}^T\mathbf{X}$.
- $\mathbf{\Lambda}$ is a diagonal matrix of singular values, where the squares of the diagonal elements are the eigenvalues of $\mathbf{X}\mathbf{X}^T$ and $\mathbf{X}^T\mathbf{X}$.
- $\mathbf{U}$ and $\mathbf{V}$ are orthonormal i.e., every vector (columns in matrix) have a magnitude of 1 and are mutually orthogonal i.e., their dot product is 0.

Recall, if $\mathbf{X}$ is the normalized-data matrix, then the covariance matrix is:

$$\begin{aligned}\mathbf{S} &= \frac{1}{n}\mathbf{X}^T\mathbf{X} \\ &= \frac{1}{n}\left(\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T\right)^T\left(\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T\right) \\ &= \frac{1}{n}\mathbf{V}\mathbf{\Lambda}\mathbf{U}^T\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T \\ &= \frac{1}{n}\mathbf{V}\mathbf{\Lambda}^2\mathbf{V}^T \\ \mathbf{S}\mathbf{V} &= \frac{1}{n}\mathbf{V}\mathbf{\Lambda}^2\mathbf{V}^T\mathbf{V} \\ \mathbf{S}\mathbf{V} &= \frac{1}{n}\mathbf{V}\mathbf{\Lambda}^2 \\ \mathbf{S}\mathbf{V} &= \frac{1}{n}\mathbf{\Lambda}^2\mathbf{V}\end{aligned}$$

where $\mathbf{V}$ is the eigenvector and $\frac{1}{n}\mathbf{\Lambda}^2$ is the eigenvalue.

7.7 PCA Example

Let a dataset of 5 samples with 3-dimensional data be

Table 7.1: PCA Example Data

A	B	C

Compute the covariance matrix S:

$$S = \begin{bmatrix} 1 & \frac{673}{1000} & \frac{433}{500} \\ \frac{673}{1000} & 1 & \frac{97}{250} \\ \frac{433}{500} & \frac{97}{250} & 1 \end{bmatrix}$$

Eigen decomposition of $\mathbf{S}$ generates eigenpairs:

$$\lambda_1 = 2.304 \qquad \lambda_2 = 0.628 \qquad \lambda_3 = 0.068$$

$$\mathbf{v}_1 = \begin{bmatrix} 1.115 \\ \\ \dots \end{bmatrix} \qquad \mathbf{v}_2 = \begin{bmatrix} \\ \\ \dots \end{bmatrix} \qquad \mathbf{v}_3 = \begin{bmatrix} \\ \\ \dots \end{bmatrix}$$

7.8 Linear PCA

- PCA excels in linear data transformations but can falter with complex, non-linear datasets.
- Non-linear PCA:
  - Kernel PCA
  - Autoencoder

7.9 Kernel PCA

- Replace $\mathbf{X}$ with $\Phi(\mathbf{X})$ where $\Phi(\cdot)$ is a kernel function.