Principle Component Analysis

Nicholas Ruozzi
University of Texas at Dallas
Eigenvalues

• $\lambda$ is an eigenvalue of a matrix $A \in \mathbb{R}^{n \times n}$ if the linear system $Ax = \lambda x$ has at least one non-zero solution

• If $Ax = \lambda x$ we say that $\lambda$ is an eigenvalue of $A$ with corresponding eigenvector $x$

• Could be multiple eigenvectors for the same $\lambda$
Eigenvalues of Symmetric Matrices

- If $A \in \mathbb{R}^{n \times n}$ is symmetric, then it has $n$ linearly independent eigenvectors $v_1, \ldots, v_n$ corresponding to $n$ real eigenvalues.

  - Moreover, it has $n$ linearly independent orthonormal eigenvectors
    - $v_i^T v_j = 0$ for all $i \neq j$
    - $v_i^T v_i = 1$ for all $i$
Eigenvalues of Symmetric Matrices

• If $A \in \mathbb{R}^{n \times n}$ is symmetric, then it has $n$ linearly independent eigenvectors $v_1, \ldots, v_n$ corresponding to $n$ real eigenvalues.

• A symmetric matrix is **positive definite** if and only if all of its eigenvalues are positive.

• The orthonormal eigenvectors form a **basis** of $\mathbb{R}^n$ (similar to the standard coordinate axes).
Examples

• The 2x2 identity matrix has all of its eigenvalues equal to 1 (it is positive definite) with orthonormal eigenvectors $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$

• The matrix $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ has eigenvalues 0 and 2 with orthonormal eigenvectors $\begin{bmatrix} -1 \\ \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$ and $\begin{bmatrix} 1 \\ \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$

• The matrix $\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$ has eigenvalues 1 and 3 with orthonormal eigenvectors $\begin{bmatrix} -1 \\ \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$ and $\begin{bmatrix} 1 \\ \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$
Suppose $A \in \mathbb{R}^{n \times n}$ is symmetric

Any $x \in \mathbb{R}^n$ can be written as $x = \sum_{i=1}^{n} c_i v_i$ where $v_1, \ldots, v_n$ are the eigenvectors of $A$

\[
\begin{align*}
Ax &= \sum_{i=1}^{n} \lambda_i c_i v_i \\
A^2 x &= \sum_{i=1}^{n} \lambda_i^2 c_i v_i \\
&\vdots \\
A^t x &= \sum_{i=1}^{n} \lambda_i^t c_i v_i
\end{align*}
\]
Eigenvalues

• Suppose $A \in \mathbb{R}^{n \times n}$ is symmetric

• Any $x \in \mathbb{R}^n$ can be written as $x = \sum_{i=1}^{n} c_i v_i$ where $v_1, \ldots, v_n$ are the eigenvectors of $A$

  • $c_i = v_i^T x$, this is the projection of $x$ along the line given by $v_i$ (assuming that $v_i$ is a unit vector)
Eigenvalues of Symmetric Matrices

• Let $Q \in \mathbb{R}^{n \times n}$ be the matrix whose $i^{th}$ column is $v_i$ and $D \in \mathbb{R}^{n \times n}$ be the diagonal matrix such that $D_{ii} = \lambda_i$

• $Ax = QDQ^T x$

• Can throw away some eigenvectors to approximate this quantity

• For example, let $Q_k$ be the matrix formed by keeping only the top $k$ eigenvectors and $D_k$ be the diagonal matrix whose diagonal consists of the top $k$ eigenvalues
The Frobenius norm is a matrix norm given by

$$\|A\|_F = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} |A_{ij}|^2}$$

- $Q_kD_kQ_k^T$ is the best rank $k$ approximation of the symmetric matrix $A$ with respect to the Frobenius norm

$$Q_kD_kQ_k^T = \arg\min_{B \in \mathbb{R}^{n \times n} \text{ s.t. } \text{rank}(B)=k} \|A - B\|_F$$
Principal Component Analysis

• Principle component analysis

  • Can be used to reduce the dimensionality of the data while still maintaining a good approximation of the sample mean and variance

  • Can also be used for selecting good features that are combinations of the input features

  • Unsupervised – just finds a good representation of the data in terms of combinations of the input features
Principal Component Analysis

• Input a collection of data points sampled from some distribution $x_1, \ldots, x_p \in \mathbb{R}^n$

• Construct the matrix $W \in \mathbb{R}^{n \times p}$ whose $i^{th}$ column is

$$x_i - \frac{\sum_j x_j}{p}$$

• The matrix $WW^T$ is the sample covariance matrix

  • $WW^T$ is symmetric and positive semidefinite
Principal Component Analysis

• PCA finds a set of orthogonal vectors that best explain the variance of the sample covariance matrix

  • From our previous discussion, these are exactly the eigenvectors of $WW^T$

  • We can discard the eigenvectors corresponding to small magnitude eigenvalues to yield an approximation

  • Simple algorithm to describe, MATLAB and other programming languages have built in support for eigenvector computation
PCA in Practice

• Forming the matrix $WW^T$ can require a lot of memory (especially if $n \gg p$)

• Need a faster way to compute this without forming the matrix explicitly

• Typical approach: use the singular value decomposition
Singular Value Decomposition (SVD)

• Every matrix $B \in \mathbb{R}^{n \times p}$ admits a decomposition of the form

$$B = U \Sigma V^T$$

• where $U \in \mathbb{R}^{n \times n}$ is an orthogonal matrix, $\Sigma \in \mathbb{R}^{n \times p}$ is non-negative diagonal matrix, and $V \in \mathbb{R}^{p \times p}$ is an orthogonal matrix

• A matrix $C \in \mathbb{R}^{m \times m}$ is orthogonal if $C^T = C^{-1}$. Equivalently, the rows and columns of $C$ are orthonormal vectors
Singular Value Decomposition (SVD)

• Every matrix $B \in \mathbb{R}^{n \times p}$ admits a decomposition of the form

$$B = U \Sigma V^T$$

Diagonal elements of $\Sigma$ called singular values

• where $U \in \mathbb{R}^{n \times n}$ is an orthogonal matrix, $\Sigma \in \mathbb{R}^{n \times p}$ is non-negative diagonal matrix, and $V \in \mathbb{R}^{p \times p}$ is an orthogonal matrix

• A matrix $C \in \mathbb{R}^{m \times m}$ is orthogonal if $C^T = C^{-1}$. Equivalently, the rows and columns of $C$ are orthonormal vectors
SVD and PCA

• Returning to PCA
  
  • Let $W = U\Sigma V^T$ be the SVD of $W$
  
  • $W W^T = U\Sigma V^T V \Sigma^T U^T = U \Sigma \Sigma^T U^T$
  
  • If we can compute the SVD of $W$, then we don't need to form the matrix $W W^T$
SVD and PCA

• For any matrix \( A \), \( AA^T \) is symmetric and positive semidefinite

  • Let \( A = U \Sigma V^T \) be the SVD of \( A \)

  • \( AA^T = U \Sigma V^T V \Sigma^T U^T = U \Sigma \Sigma^T U^T \)

  • \( U \) must be a matrix of eigenvectors of \( AA^T \)

  • The eigenvalues of \( AA^T \) are all non-negative because \( \Sigma \Sigma^T = \Sigma^2 \) which are the square of the singular values of \( A \)
An Example: “Eigenfaces”

- Let’s suppose that our data is a collection of images of the faces of individuals
An Example: “Eigenfaces”

• Let’s suppose that our data is a collection of images of the faces of individuals

• The goal is, given the "training data", to correctly match new images to the training data

• Let’s suppose that each image is an \( s \times s \) array of pixels: \( x_i \in \mathbb{R}^n, n = s^2 \)

• As before, construct the matrix \( W \in \mathbb{R}^{n \times p} \) whose \( i^{th} \) column is \( x_i - \sum_j \frac{x_j}{p} \)
An Example: “Eigenfaces”

• Forming the matrix $WW^T$ requires a lot of memory
  
  • $s = 256$ means $WW^T$ is $65536 \times 65536$
  
  • Need a faster way to compute this without forming the matrix explicitly
  
  • Could use the singular value decomposition
An Example: “Eigenfaces”

• A different approach when \( p \ll n \)
  • Compute the eigenvectors of \( A^T A \) (this is an \( p \times p \) matrix)
  • Let \( v \) be an eigenvector of \( A^T A \) with eigenvalue \( \lambda \)
  • \( AA^T Av = \lambda Av \)
  • This means that \( Av \) is an eigenvector of \( AA^T \) with eigenvalue \( \lambda \) (or 0)
  • Save the top \( k \) eigenvectors - called eigenfaces in this example
An Example: “Eigenfaces”

• The data in the matrix is “training data”
  • Given a new image, we’d like to determine which, if any, member of the data set that it is most similar to

• Step 1: Compute the projection of the recentered, new image onto each of the $k$ eigenvectors
  • This gives us a vector of weights $c_1, \ldots, c_k$
An Example: “Eigenfaces”

- The data in the matrix is “training data”
  - Given a new image, we’d like to determine which, if any, member of the data set that it is most similar to
- Step 2: Determine if the input image is close to one of the faces in the data set
  - If the distance between the input and it's approximation is too large, then the input is likely not a face
An Example: “Eigenfaces”

- The data in the matrix is “training data”
  - Given a new image, we’d like to determine which, if any, member of the data set that it is most similar to

- Step 3: Find the person in the training data that is closest to the new input
  - Replace each group of training images by its average
  - Compute the distance to the $i^{th}$ average $\|c - a^i\|$ where $a^i$ are the coefficients of the average face for person $i$