Introduction to Machine Learning

Principal Component Analysis

Varun Chandola

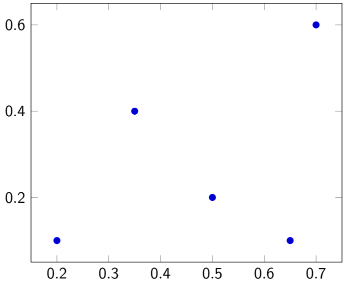
Computer Science & Engineering
State University of New York at Buffalo
Buffalo, NY, USA
chandola@buffalo.edu

University at Buffalo
Department of Computer Science and Engineering
School of Engeering and Applied Sciences

Outline

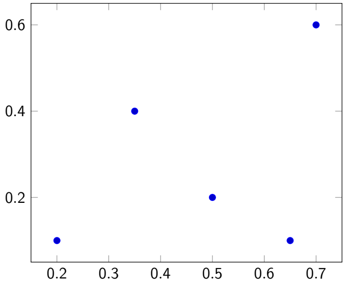
Introduction to PCA

- Consider the following data points



Introduction to PCA

- Consider the following data points



- *Embed* these points in 1 dimension
- What is the best way?
  - **Along the direction of the maximum variance**
  - Why?

Why Maximal Variance?

- Least loss of information
- Best capture the "spread"

Why Maximal Variance?

- Least loss of information
- Best capture the "spread"
- What is the direction of maximal variance?
- Given any direction ($\hat{\mathbf{u}}$), the projection of $\mathbf{x}$ on $\hat{\mathbf{u}}$ is given by:

$$\mathbf{x}_i^\top \hat{\mathbf{u}}$$

- Direction of maximal variance can be obtained by maximizing

$$\begin{aligned}\frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i^\top \hat{\mathbf{u}})^2 &= \frac{1}{N}\sum_{i=1}^{N}\hat{\mathbf{u}}^\top \mathbf{x}_i\mathbf{x}_i^\top \hat{\mathbf{u}} \\ &= \hat{\mathbf{u}}^\top \left(\frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i\mathbf{x}_i^\top\right)\hat{\mathbf{u}}\end{aligned}$$

Finding Direction of Maximal Variance

- Find:

$$\max_{\hat{\mathbf{u}}:\hat{\mathbf{u}}^\top \hat{\mathbf{u}}=1} \hat{\mathbf{u}}^\top \mathbf{S}\hat{\mathbf{u}}$$

  where:

$$\mathbf{S} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i \mathbf{x}_i^\top$$

- **S** is the sample (empirical) covariance matrix of the mean-centered data

Defining Principal Components

- First PC: Eigen-vector of the (sample) covariance matrix with largest eigen-value
- Second PC?

Defining Principal Components

- First PC: Eigen-vector of the (sample) covariance matrix with largest eigen-value
- Second PC?
- Eigen-vector with next largest value
- Variance of each PC is given by λ_i
- Variance captured by first L PC $(1 \leq L \leq D)$

$$\frac{\sum_{i=1}^{L} \lambda_i}{\sum_{i=1}^{D} \lambda_i} \times 100$$

- What are eigen vectors and values?

$$\mathbf{Av} = \lambda \mathbf{v}$$

  $\mathbf{v}$ is eigen vector and λ is eigen-value for the **square matrix A**
- Geometric interpretation?

Dimensionality Reduction Using PCA

- Consider first L eigen values and eigen vectors
- Let $\mathbf{W}$ denote the $D \times L$ matrix with first L eigen vectors in the columns (sorted by λ's)
- PC score matrix

$$\mathbf{Z} = \mathbf{X}\mathbf{W}$$

- Each input vector $(D \times 1)$ is replaced by a shorter $L \times 1$ vector

PCA Algorithm

1. *Center* $\mathbf{X}$
$$\mathbf{X} = \mathbf{X} - \hat{\mu}$$
2. Compute sample covariance matrix:
$$\mathbf{S} = \frac{1}{N-1}\mathbf{X}^\top\mathbf{X}$$
3. Find eigen vectors and eigen values for $\mathbf{S}$
4. $\mathbf{W}$ consists of first L eigen vectors as columns
   - Ordered by decreasing eigen-values
   - $\mathbf{W}$ is $D \times L$
5. Let $\mathbf{Z} = \mathbf{X}\mathbf{W}$
6. Each row in $\mathbf{Z}$ (or $\mathbf{z}_i^\top$) is the lower dimensional embedding of $\mathbf{x}_i$

Recovering Original Data

- Using $\mathbf{W}$ and $\mathbf{z}_i$

$$\hat{\mathbf{x}}_i = \mathbf{W}\mathbf{z}_i$$

- **Average Reconstruction Error**

$$J(\mathbf{W}, \mathbf{Z}) = \frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|^2$$

Theorem (Classical PCA Theorem)

Among all possible orthonormal sets of L basis vectors, PCA gives the solution which has the minimum reconstruction error.

- Optimal “embedding” in L dimensional space is given by $z_i = \mathbf{W}^\top \mathbf{x}_i$

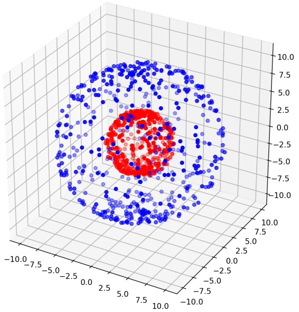
Using PCA for Face Recognition

EigenFaces [2]

- **Input:** A set of images (of faces)
- **Task:** Identify if a new image is a face or not.

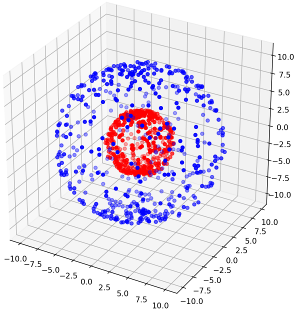
Issues with PCA

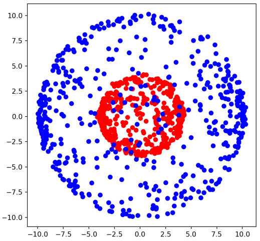
- A linear transformation of the data
- Might not work everytime



Issues with PCA

- A linear transformation of the data
- Might not work everytime





Kernel PCA - Applying the kernel trick [1] I

- Assume a non-linear transformation of input:

$$\mathbf{x_i} \Rightarrow \Phi(\mathbf{x_i}),\ \ \Phi(\mathbf{x_i}) \in \mathbb{R}^M$$

- We will assume that the new data is mean centered:

$$\frac{1}{N}\sum_{i=1}^{N}\Phi(\mathbf{x_i}) = 0$$

- The covariance matrix of the projected data, **C**:

$$\mathbf{C} = \frac{1}{N}\sum_{i=1}^{N}\Phi(\mathbf{x_i})\Phi(\mathbf{x_i})^\top$$

- The eigen vectors and eigen values of **C** are given by:

$$\mathbf{C}\mathbf{v}_k = \lambda_k \mathbf{v}_k$$

where $k = 1, 2, \ldots, M$

Kernel PCA - Applying the kernel trick [1] II

- Substituting the expression for **C**:

$$\frac{1}{N}\sum_{i=1}^{N}\Phi(\mathbf{x_i})\Phi(\mathbf{x_i})^\top \mathbf{v}_k = \lambda_k \mathbf{v}_k$$

- Rearranging and assuming $a_{ki} = \frac{\Phi(\mathbf{x_i})^\top \mathbf{v}_k}{N\lambda_k}$

$$\mathbf{v}_k = \sum_{i=1}^{N} a_{ki}\Phi(\mathbf{x_i})$$

- Substituting this back in the above equation

$$\frac{1}{N}\sum_{i=1}^{N}\Phi(\mathbf{x_i})\Phi(\mathbf{x_i})^\top \sum_{j=1}^{N} a_{kj}\Phi(\mathbf{x_j}) = \lambda_k \sum_{i=1}^{N} a_{ki}\Phi(\mathbf{x_i})$$

Note the subscript j in the second summation on the left hand side.

Kernel PCA - Applying the kernel trick [1] III

- Now multiplying both sides with $\Phi(\mathbf{x_l})^\top$:

$$\frac{1}{N}\Phi(\mathbf{x_l})^\top \sum_{i=1}^{N} \Phi(\mathbf{x_i})\Phi(\mathbf{x_i})^\top \sum_{j=1}^{N} a_{kj}\Phi(\mathbf{x_j}) = \lambda_k \Phi(\mathbf{x_l})^\top \sum_{i=1}^{N} a_{ki}\Phi(\mathbf{x_i})$$

which is the same as:

$$\sum_{i=1}^{N} \underbrace{\Phi(\mathbf{x_l})^\top\Phi(\mathbf{x_i})} \sum_{j=1}^{N} a_{kj} \underbrace{\Phi(\mathbf{x_i})^\top\Phi(\mathbf{x_j})} = N\lambda_k \sum_{i=1}^{N} \underbrace{a_{ki}\Phi(\mathbf{x_l})^\top\Phi(\mathbf{x_i})}$$

- Let $k()$ be a function, such that: $k(\mathbf{x_i}, \mathbf{x_j}) = \Phi(\mathbf{x_i})^\top\Phi(\mathbf{x_j})$
- The above expression can be written as:

$$\sum_{i=1}^{N} k(\mathbf{x_l}, \mathbf{x_i}) \sum_{j=1}^{N} a_{kj} k(\mathbf{x_i}, \mathbf{x_j}) = N\lambda_k \sum_{i=1}^{N} k(\mathbf{x_l}, \mathbf{x_i})$$

Kernel PCA - Applying the kernel trick [1] IV

- Consider the $N \times 1$ vector, $\mathbf{a}_k$ amd $N \times N$ matrix, $\mathbf{K}$, such that

$$\mathbf{a}_k = \begin{bmatrix} a_{k1} \\ a_{k2} \\ \vdots \\ a_{kN} \end{bmatrix}$$

  and,

$$\mathbf{K}[i][j] = k(\mathbf{x_i}, \mathbf{x_j})$$

- The above expression can be written, using matrix notation, as:

$$\mathbf{K}^2\mathbf{a}_k = \lambda_k N\mathbf{K}\mathbf{a}_k$$

- To solve for $\mathbf{a}_k$, we can solve the following:

$$\mathbf{K}\mathbf{a}_k = \lambda_k N\mathbf{a}_k$$

Projecting data using Kernel PCA

- For a new data instance, $\mathbf{x}^*$, the k^{th} entry of the corresponding $\mathbf{z}^*$ will be:

$$
\begin{aligned}
z_k^* &= \Phi(\mathbf{x}^*)^\top \mathbf{v}_k \\
&= \Phi(\mathbf{x}^*)^\top \sum_{i=1}^{N} a_{ki}\Phi(\mathbf{x_i}) \\
&= \sum_{i=1}^{N} a_{ki}\Phi(\mathbf{x}^*)^\top \Phi(\mathbf{x_i}) \\
&= \sum_{i=1}^{N} a_{ki} k(\mathbf{x}^*, \mathbf{x}_i)
\end{aligned}
$$

Centering the projected data

- How do we ensure that the projected new features have a zero mean, without doing the actual projection?
- Use the **Gram matrix**:

$$\tilde{\mathbf{K}} = \mathbf{K} - \mathbf{1}_N\mathbf{K} - \mathbf{K}\mathbf{1}_N + \mathbf{1}_N\mathbf{K}\mathbf{1}_N$$

where $\mathbf{1}_N$ is a $N \times N$ matrix with all entries equal to $\frac{1}{N}$

Kernel PCA Algorithm

1. Given the data set $\mathbf{X}$ and a kernel function $k()$, construct the kernel matrix $\mathbf{K}$
2. Compute the Gram matrix $\tilde{\mathbf{K}}$
3. Find eigen-vectors of $\tilde{\mathbf{K}}$
4. Use top M eigen-vectors to project a new data instance, $\mathbf{x}^*$ to the corresponding $\mathbf{z}^*$

Kernel PCA - Final Thoughts

- Very sensitive to the kernel chocie and kernel parameters
- Slow ($O(N^3)$)
- Recovering original data is not straightforward as linear PCA

References

B. Schölkopf, A. J. Smola, and K.-R. Müller.
Kernel Principal Component Analysis, page 327–352.
MIT Press, Cambridge, MA, USA, 1999.

M. Turk and A. Pentland.
Face recognition using eigenfaces.
In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR '91., IEEE Computer Society Conference on*, pages 586–591, Jun 1991.