Principal Component Analysis (PCA)
CSC411/2515 Tutorial

Harris Chan
Based on previous tutorial slides by Wenjie Luo, Ladislav Rampasek

University of Toronto

hchan@cs.toronto.edu

October 19th, 2017
Overview

1 Motivation
   - Dimensionality Reduction
   - Two Perspectives on Good Transformations

2 PCA
   - Maximum Variance
   - Minimum Reconstruction Error

3 Applications of PCA
   - Demo

4 Summary
We have some data $X \in \mathbb{R}^{N \times D}$, where $D$ can be very large.

We want a new representation of the data $Z \in \mathbb{R}^{N \times K}$ where $K < < D$.

- For computational reasons
- To better understand / visualize the data
- For compression
- etc.

We will restrict ourselves to \textbf{linear transformation.}
Example

- In this dataset, there are only 3 degrees of freedom: (1) horizontal translations; (2) vertical translations; (3) Rotations.

- But each image is $100 \times 100 = 10000$ pixels, so $X$ will be 10000 elements wide!
The goal is to find good directions $u$ that preserves ”important” aspects of the data.

In linear setting: $z = x^T u$

This will turn out to be the top-$K$ eigenvalues of the data covariance.

2 ways to view this:
1. Find directions of maximum variation
2. Find projections that minimizes the reconstruction error
Consider the $n$-th datapoint $x_n$ that has 2 dimensions, $x_1$ and $x_2$: 
We can pick a direction $\mathbf{u}_1$ to project $\mathbf{x}_n$ onto, creating a projected point $\hat{\mathbf{x}}_n$: 

![Diagram showing projection of $\mathbf{x}_n$ onto $\mathbf{u}_1$]
Two Derivations of PCA

By Pythagorean theorem:

\[ R^2 = D^2 + \epsilon^2 \]

Original Dist Variance Recons. Err

Since \( R^2 \) is fixed:

Problem Equivalence

Maximize \( D^2 \) (variance) ⇔ Minimize \( \epsilon^2 \) (reconstruction error)
Two Derivations of PCA

Figure 12.2 from Bishop’s Textbook:
Our goal is to maximize the variance of the projected data:

\[
\text{maximize } \frac{1}{2N} \sum_{n=1}^{N} (u_1^T x_n - u_1^T \bar{x}_n) = u_1^T S u_1 \tag{1}
\]

Where the sample mean and covariance is given by:

\[
\bar{x} = \frac{1}{N} \sum_{n=1}^{N} x_n \tag{2}
\]

\[
S = \frac{1}{N} \sum_{n=1}^{N} (x_n - \bar{x})(x_n - \bar{x})^T \tag{3}
\]
If we want to find a stationary point of a function of multiple variables \( f(x) \) subject to one or more constraints \( g(x) = 0 \):

1. Introduce Lagrangian function:

\[
L(x, \lambda) \equiv f(x) + \lambda g(x)
\]  

(5)

2. Find its stationary point w.r.t. both \( x \) and \( \lambda \)

If you are not familiar with it, check out Appendix E in Bishop’s book.
Finding $\mathbf{u}_1$

- We want to maximize $\mathbf{u}_1^T S \mathbf{u}_1$ subject to $\|\mathbf{u}_1\| = 1$ (since we are finding direction).
- Use Lagrange multiplier $\alpha_1$ to express this as:

$$\mathbf{u}_1^T S \mathbf{u}_1 + \alpha_1(1 - \mathbf{u}_1^T \mathbf{u}_1) \quad (6)$$

- Take derivative and set to 0:

$$S \mathbf{u}_1 - \alpha_1 \mathbf{u}_1 = 0 \quad (7)$$

$$S \mathbf{u}_1 = \alpha_1 \mathbf{u}_1 \quad (8)$$

- So $\mathbf{u}_1$ is an eigenvector of $S$ with eigenvalue $\alpha_1$.
- In fact, it must be the eigenvector with the maximum eigenvalue, since this maximizes the objective.
Finding \textbf{u}_2

- We want to maximize \( \mathbf{u}_2^T \mathbf{S} \mathbf{u}_2 \) subject to \( ||\mathbf{u}_2|| = 1 \) and \( \mathbf{u}_2^T \mathbf{u}_1 = 0 \) (orthogonal to \( \mathbf{u}_1 \))
- Use Lagrange form:

\[
\mathbf{u}_s^T \mathbf{S} \mathbf{u}_s + \alpha_s (1 - \mathbf{u}_s^T \mathbf{u}_2) - \beta \mathbf{u}_2^T \mathbf{u}_1
\]  \hspace{1cm} (9)

- Take derivative and set to 0 to find \( \beta \):

\[
\frac{\partial}{\partial \mathbf{u}_2} = \mathbf{S} \mathbf{u}_2 - \alpha_2 \mathbf{u}_2 - \beta \mathbf{u}_1 = 0 \hspace{1cm} (10)
\]
\[
\Rightarrow \mathbf{u}_1^T \mathbf{S} \mathbf{u}_2 - \alpha_2 \mathbf{u}_1^T \mathbf{u}_2 - \beta \mathbf{u}_1^T \mathbf{u}_1 = 0 \hspace{1cm} (11)
\]
\[
\Rightarrow \alpha_1 \mathbf{u}_1^T \mathbf{u}_2 - \alpha_2 \mathbf{u}_1^T \mathbf{u}_2 - \beta \mathbf{u}_1^T \mathbf{u}_1 = 0 \hspace{1cm} (12)
\]
\[
\Rightarrow \alpha_1 \cdot 0 - \alpha_2 \cdot 0 - \beta \cdot 1 = 0 \hspace{1cm} (13)
\]
\[
\Rightarrow \beta = 0 \hspace{1cm} (14)
\]
Finding $\mathbf{u}_2$

- We want to maximize $\mathbf{u}_2^T S \mathbf{u}_2$ subject to $\|\mathbf{u}_2\| = 1$ and $\mathbf{u}_2^T \mathbf{u}_1 = 0$ (orthogonal to $\mathbf{u}_1$)
- Use Lagrange form:

$$\mathbf{u}_s^T S \mathbf{u}_s + \alpha_s (1 - \mathbf{u}_s^T \mathbf{u}_2) - \beta \mathbf{u}_2^T \mathbf{u}_1$$  

(15)

- Take derivative and set to 0 to find $\alpha_2$:

$$\frac{\partial}{\partial \mathbf{u}_2} = S \mathbf{u}_2 - \alpha_2 \mathbf{u}_2 = 0$$  

(16)

$$\implies S \mathbf{u}_2 = \alpha_2 \mathbf{u}_2$$  

(17)

- So $\alpha_2$ must be the second largest eigenvalue of $S$. 
PCA In General

- We can compute the entire PCA solution by just computing the eigenvectors with the top-K eigenvalues.
- These can be found using the singular value decomposition (SVD) of $S$. 
Choosing the number of $K$

- How do we choose the number of components?
- Idea: Look at the spectrum of covariance, pick $K$ to capture most of the variation

More principled: Bayesian treatment (beyond this course)
We can also think of PCA as minimizing the *reconstruction error* of compressed data:

\[
\text{minimize} \quad \frac{1}{2N} \sum_{n=1}^{N} \|x_n - \tilde{x}_n\|^2
\]

(18)

We will omit some details for now, but the key is that we define some K-dimensional basis such that:

\[
\tilde{x} = Wx + \text{const}
\]

(19)

The solution will turn out to be the same as the maximum variance formulation.
We’ll apply PCA using scikit-learn in Python on various datasets for visualization / compression:

- Synthetic 2D data: Show the principal components learned and what the transformed data looks like
- MNIST digits: Compression and Reconstruction
- Olivetti faces dataset: Compression and Reconstruction
- Iris dataset: Visualization
For example: Olivetti Faces dataset. Apply PCA on the face images to find the principle components, and project the data down to $k$-dimensions.
PCA Application: Compression & Reconstruction

Reconstruction when using various values of $k$:
PCA Application: Visualization

- PCA can be used to find the 'best' viewing angle to project onto a 2-D plane (or 3D) to better understand the data.
- Example on the Iris dataset:
PCA is a linear projection of D-dimensional \( \{ \mathbf{x}_n \} \) to \( K \leq D \) vector space given by \( \{ \mathbf{u}_k \} \) basis vectors such that it:

- Maximizes variance in the projected data points
- Minimizes projection error (square loss)
- \( \{ \mathbf{u}_k \} \) are orthonormal
- \( \{ \mathbf{u}_k \} \) turns out to be the first \( K \) eigenvectors of the data covariance matrix with \( K \) largest eigenvalues
- Can be computed in \( O(KD^2) \)
PCA is good for:
- Dimensionality reduction
- Visualization
- Compression (with loss)
- Denoising (by removing small variances in the data)
- Can be used for data whitening = decorrelation, so that features have unit covariance

Caution! In classification task, if the class labels’ signal in the data has small variance, PCA may remove it completely
Thanks!