CSC 411 Lecture 12: Principal Component Analysis

Mengye Ren and Matthew MacKay

University of Toronto

Overview

- Today we'll cover the first unsupervised learning algorithm for this course: principal component analysis (PCA)
- Dimensionality reduction: map the data to a lower dimensional space
  - Save computation/memory
  - Reduce overfitting
  - Visualize in 2 dimensions
- PCA is a linear model, with a closed-form solution. It's useful for understanding lots of other algorithms.
  - Autoencoders
  - Matrix factorizations (next lecture)
- Today's lecture is very linear-algebra-heavy.
  - Especially orthogonal matrices and eigendecompositions.
  - Don't worry if you don't get it immediately — next few lectures won't build on it
  - Not on midterm (which only covers up through L10)

Projection onto a subspace

- Set-up: given a dataset $\mathcal{D} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\} \subset \mathbb{R}^D$
- Set $\boldsymbol{\mu}$ to the mean of the data, $\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}^{(i)}$
- Goal: find a K-dimensional subspace $\mathcal{S} \subset \mathbb{R}^D$ such that $\mathbf{x}^{(n)} - \boldsymbol{\mu}$ is "well-represented" by its projection onto $\mathcal{S}$
- Recall: The projection of a point $\mathbf{x}$ onto $\mathcal{S}$ is the point in $\mathcal{S}$ closest to $\mathbf{x}$.

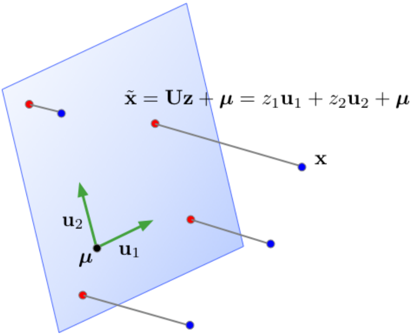
Projection onto a subspace

- Let $\{\mathbf{u}_k\}_{k=1}^{K}$ be an orthonormal basis of the subspace $\mathcal{S}$
- Approximate each data point $\mathbf{x}$ as:

$$\begin{aligned}\tilde{\mathbf{x}} &= \boldsymbol{\mu} + \mathrm{Proj}_{\mathcal{S}}(\mathbf{x} - \boldsymbol{\mu}) \\ &= \boldsymbol{\mu} + \sum_{k=1}^{K} z_k \mathbf{u}_k\end{aligned}$$

- From linear algebra: $z_k = \mathbf{u}_k^T(\mathbf{x} - \boldsymbol{\mu})$
- Let $\mathbf{U}$ be a matrix with columns $\{\mathbf{u}_k\}_{k=1}^{K}$ then $\mathbf{z} = \mathbf{U}^T(\mathbf{x} - \boldsymbol{\mu})$
- Also: $\tilde{\mathbf{x}} = \boldsymbol{\mu} + \mathbf{U}\mathbf{z}$

Projection onto a Subspace

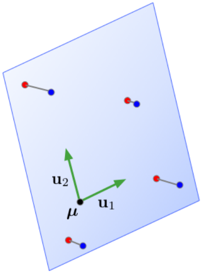


$$\mathbf{z} = \mathbf{U}^\top(\mathbf{x} - \boldsymbol{\mu})$$

- In machine learning, $\tilde{\mathbf{x}}$ is also called the reconstruction of $\mathbf{x}$.
- $\mathbf{z}$ is its representation, or code.

Projection onto a Subspace

- If we have a K-dimensional subspace in a D-dimensional input space, then $\mathbf{x} \in \mathbb{R}^D$ and $\mathbf{z} \in \mathbb{R}^K$.
- If the data points $\mathbf{x}$ all lie close to their reconstructions, then we can approximate distances, etc. in terms of these same operations on the code vectors $\mathbf{z}$.
- If $K \ll D$, then it's much cheaper to work with $\mathbf{z}$ than $\mathbf{x}$.
- A mapping to a space that's easier to manipulate or visualize is called a representation, and learning such a mapping is representation learning.
- Mapping data to a low-dimensional space is called dimensionality reduction.



Learning a Subspace

- How to choose a good subspace $\mathcal{S}$?
  - Need to choose $D \times K$ matrix $\mathbf{U}$ with orthonormal columns.
- Two criteria:
  - Minimize the reconstruction error

$$\min \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{x}^{(i)} - \tilde{\mathbf{x}}^{(i)}\|^2$$

  - Maximize the variance of the code vectors

$$\begin{aligned}
\max \sum_j \mathrm{Var}(z_j) &= \frac{1}{N} \sum_j \sum_i (z_j^{(i)} - \bar{z}_j)^2 \\
&= \frac{1}{N} \sum_i \|\mathbf{z}^{(i)} - \bar{\mathbf{z}}\|^2 \\
&= \frac{1}{N} \sum_i \|\mathbf{z}^{(i)}\|^2 \qquad \text{Exercise: show } \bar{\mathbf{z}} = 0
\end{aligned}$$

  - Note: here, $\bar{\mathbf{z}}$ denotes the mean, not a derivative.

Learning a Subspace

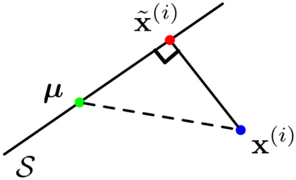
- These two criteria are equivalent! I.e., we'll show

$$\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}^{(i)} - \tilde{\mathbf{x}}^{(i)}\|^2 = \text{const} - \frac{1}{N}\sum_{i}\|\mathbf{z}^{(i)}\|^2$$

- Observation: by unitarity,

$$\|\tilde{\mathbf{x}}^{(i)} - \boldsymbol{\mu}\| = \|\mathbf{U}\mathbf{z}^{(i)}\| = \|\mathbf{z}^{(i)}\|$$

- By the Pythagorean Theorem,



$$\underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\tilde{\mathbf{x}}^{(i)} - \boldsymbol{\mu}\|^2}_{\text{projected variance}} + \underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}^{(i)} - \tilde{\mathbf{x}}^{(i)}\|^2}_{\text{reconstruction error}}$$

$$= \underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}^{(i)} - \boldsymbol{\mu}\|^2}_{\text{constant}}$$

Principal Component Analysis

Choosing a subspace to maximize the projected variance, or minimize the reconstruction error, is called principal component analysis (PCA).

Recall:

- Spectral Decomposition: a symmetric matrix $\mathbf{A}$ has a full set of eigenvectors, which can be chosen to be orthogonal. This gives a decomposition

$$\mathbf{A} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\top,$$

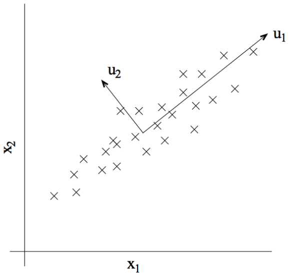
  where $\mathbf{Q}$ is orthogonal and $\boldsymbol{\Lambda}$ is diagonal. The columns of $\mathbf{Q}$ are eigenvectors, and the diagonal entries λ_j of $\boldsymbol{\Lambda}$ are the corresponding eigenvalues.
- I.e., symmetric matrices are diagonal in some basis.
- A symmetric matrix $\mathbf{A}$ is positive semidefinite iff each $\lambda_j \geq 0$.

Principal Component Analysis

- Consider the empirical covariance matrix:

$$\boldsymbol{\Sigma} = \frac{1}{N} \sum_{i=1}^{N} (\mathbf{x}^{(i)} - \boldsymbol{\mu})(\mathbf{x}^{(i)} - \boldsymbol{\mu})^\top$$

- Recall: Covariance matrices are symmetric and positive semidefinite.
- The optimal PCA subspace is spanned by the top K eigenvectors of $\boldsymbol{\Sigma}$.
  - More precisely, choose the first K of any orthonormal eigenbasis for $\boldsymbol{\Sigma}$.
  - The general case is tricky, but we'll show this for $K = 1$.
- These eigenvectors are called principal components, analogous to the principal axes of an ellipse.



Deriving PCA

- For $K = 1$, we are fitting a unit vector $\mathbf{u}$, and the code is a scalar $z = \mathbf{u}^\top(\mathbf{x} - \boldsymbol{\mu})$.

$$\begin{aligned}
\frac{1}{N}\sum_i [z^{(i)}]^2 &= \frac{1}{N}\sum_i (\mathbf{u}^\top(\mathbf{x}^{(i)} - \boldsymbol{\mu}))^2 \\
&= \frac{1}{N}\sum_{i=1}^{N} \mathbf{u}^\top(\mathbf{x}^{(i)} - \boldsymbol{\mu})(\mathbf{x}^{(i)} - \boldsymbol{\mu})^\top \mathbf{u} \\
&= \mathbf{u}^\top \left[\frac{1}{N}\sum_{i=1}^{N} (\mathbf{x}^{(i)} - \boldsymbol{\mu})(\mathbf{x}^{(i)} - \boldsymbol{\mu})^\top\right] \mathbf{u} \\
&= \mathbf{u}^\top \boldsymbol{\Sigma} \mathbf{u} \\
&= \mathbf{u}^\top \mathbf{Q} \boldsymbol{\Lambda} \mathbf{Q}^\top \mathbf{u} && \text{Spectral Decomposition} \\
&= \mathbf{a}^\top \boldsymbol{\Lambda} \mathbf{a} && \text{for } \mathbf{a} = \mathbf{Q}^\top \mathbf{u} \\
&= \sum_{j=1}^{D} \lambda_j a_j^2
\end{aligned}$$

Deriving PCA

- Maximize $\mathbf{a}^\top \boldsymbol{\Lambda} \mathbf{a} = \sum_{j=1}^{D} \lambda_j a_j^2$ for $\mathbf{a} = \mathbf{Q}^\top \mathbf{u}$.
    - This is a change-of-basis to the eigenbasis of $\boldsymbol{\Sigma}$.
- Assume the λ_i are in sorted order. For simplicity, assume they are all distinct.
- Observation: since $\mathbf{u}$ is a unit vector, then by unitarity, $\mathbf{a}$ is also a unit vector. I.e., $\sum_j a_j^2 = 1$.
- By inspection, set $a_1 = \pm 1$ and $a_j = 0$ for $j \neq 1$.
- Hence, $\mathbf{u} = \mathbf{Q}\mathbf{a} = \mathbf{q}_1$ (the top eigenvector).

- A similar argument shows that the kth principal component is the kth eigenvector of $\boldsymbol{\Sigma}$. If you're interested, look up the Courant-Fischer Theorem.

Decorrelation

- Interesting fact: the dimensions of $\mathbf{z}$ are decorrelated. For now, let Cov denote the empirical covariance.

$$
\begin{aligned}
\mathrm{Cov}(\mathbf{z}) &= \mathrm{Cov}(\mathbf{U}^\top(\mathbf{x} - \boldsymbol{\mu})) \\
&= \mathbf{U}^\top \mathrm{Cov}(\mathbf{x}) \mathbf{U} \\
&= \mathbf{U}^\top \boldsymbol{\Sigma} \mathbf{U} \\
&= \mathbf{U}^\top \mathbf{Q} \boldsymbol{\Lambda} \mathbf{Q}^\top \mathbf{U} \\
&= \begin{pmatrix} \mathbf{I} & \mathbf{0} \end{pmatrix} \boldsymbol{\Lambda} \begin{pmatrix} \mathbf{I} \\ \mathbf{0} \end{pmatrix} && \text{by orthogonality} \\
&= \text{top left } K \times K \text{ block of } \boldsymbol{\Lambda}
\end{aligned}
$$

- If the covariance matrix is diagonal, this means the features are uncorrelated.
- This is why PCA was originally invented (in 1901!).

Recap

Recap:

- Dimensionality reduction aims to find a low-dimensional representation of the data.
- PCA projects the data onto a subspace which maximizes the projected variance, or equivalently, minimizes the reconstruction error.
- The optimal subspace is given by the top eigenvectors of the empirical covariance matrix.
- PCA gives a set of decorrelated features.

Applying PCA to faces

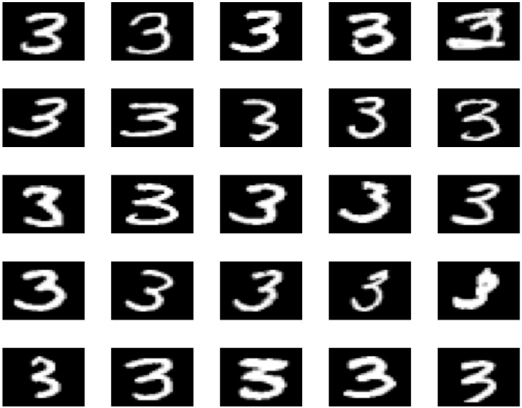
- Consider running PCA on 2429 19x19 grayscale images (CBCL data)
- Can get good reconstructions with only 3 components

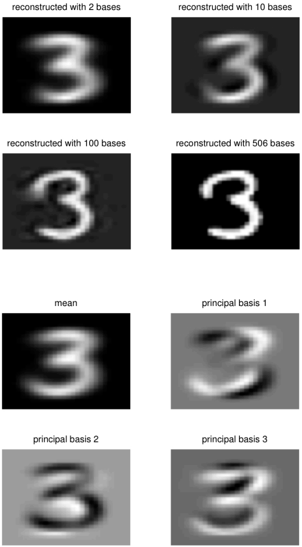
- PCA for pre-processing: can apply classifier to latent representation
  - For face recognition PCA with 3 components obtains 79% accuracy on face/non-face discrimination on test data vs. 76.8% for a Gaussian mixture model (GMM) with 84 states. (We'll cover GMMs later in the course.)
- Can also be good for visualization

Applying PCA to faces: Learned basis

Principal components of face images ("eigenfaces")

Applying PCA to digits





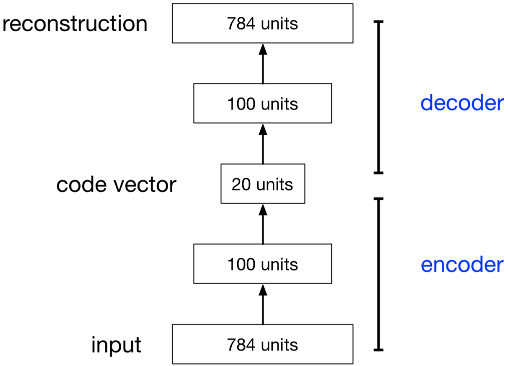
Next

Next: two more interpretations of PCA, which have interesting generalizations.

1. Autoencoders
2. Matrix factorization (later lecture)

Autoencoders

- An autoencoder is a feed-forward neural net whose job it is to take an input $\mathbf{x}$ and predict $\mathbf{x}$.
- To make this non-trivial, we need to add a bottleneck layer whose dimension is much smaller than the input.



Linear Autoencoders

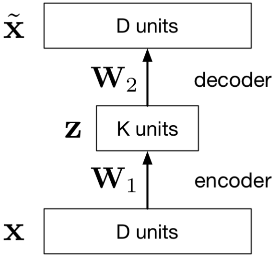
Why autoencoders?

- Map high-dimensional data to two dimensions for visualization
- Learn abstract features in an unsupervised way so you can apply them to a supervised task
  - Unlabled data can be much more plentiful than labeled data

Linear Autoencoders

- The simplest kind of autoencoder has one hidden layer, linear activations, and squared error loss.

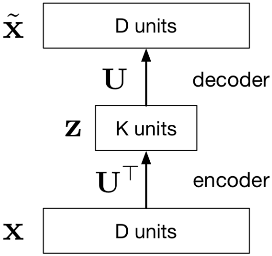
$$\mathcal{L}(\mathbf{x}, \tilde{\mathbf{x}}) = \|\mathbf{x} - \tilde{\mathbf{x}}\|^2$$



- This network computes $\tilde{\mathbf{x}} = \mathbf{W}_2\mathbf{W}_1\mathbf{x}$, which is a linear function.
- If $K \geq D$, we can choose $\mathbf{W}_2$ and $\mathbf{W}_1$ such that $\mathbf{W}_2\mathbf{W}_1$ is the identity matrix. This isn't very interesting.
- But suppose $K < D$:
  - $\mathbf{W}_1$ maps $\mathbf{x}$ to a K-dimensional space, so it's doing dimensionality reduction.

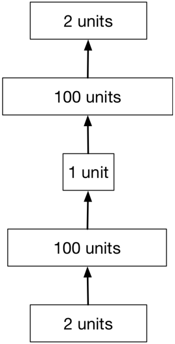
Linear Autoencoders

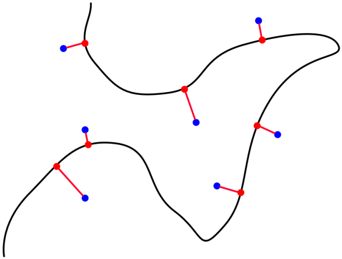
- Observe that the output of the autoencoder must lie in a K-dimensional subspace spanned by the columns of $\mathbf{W}_2$.
- We saw that the best possible K-dimensional subspace in terms of reconstruction error is the PCA subspace.
- The autoencoder can achieve this by setting $\mathbf{W}_1 = \mathbf{U}^\top$ and $\mathbf{W}_2 = \mathbf{U}$.
- Therefore, the optimal weights for a linear autoencoder are just the principal components!



Nonlinear Autoencoders

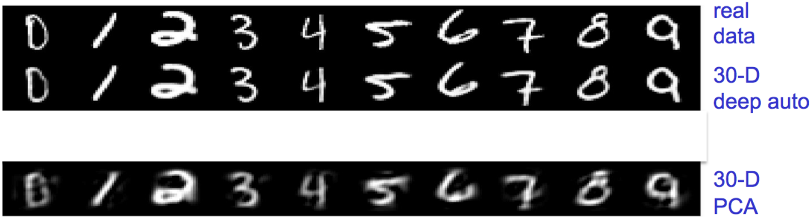
- Deep nonlinear autoencoders learn to project the data, not onto a subspace, but onto a nonlinear manifold
- This manifold is the image of the decoder.
- This is a kind of nonlinear dimensionality reduction.





Nonlinear Autoencoders

- Nonlinear autoencoders can learn more powerful codes for a given dimensionality, compared with linear autoencoders (PCA)



Nonlinear Autoencoders

Here's a 2-dimensional autoencoder representation of newsgroup articles. They're color-coded by topic, but the algorithm wasn't given the labels.