Principal Component Analysis

Rui Zhu

Overview

Unsupervised learning

Unsupervised learning: learning without a teacher.

- A set of N observations $(x_1, x_2, ..., x_N)$ of a random p-vector X having joint density Pr(X).
- The goal is to directly infer the properties of Pr(X) without the help of a supervisor or teacher providing correct answers or degree-of-error for each observation.
- It is difficult to ascertain the validity of inferences drawn from the output of most unsupervised learning algorithms.
- This uncomfortable situation has led to heavy proliferation of proposed methods, since effectiveness is a matter of opinion and cannot be verified directly.

Unsupervised learning

Popular tasks: dimension reduction, clustering

- Dimension reduction: identify low-dimensional manifolds within the X-space that represent high data density. This provides information about the associations among the variables and whether or not they can be considered as functions of a smaller set of "latent" variables.
- Clustering: find multiple convex regions of the X-space that contain modes of $\Pr(X)$. This can tell whether or not $\Pr(X)$ can be represented by a mixture of simpler densities representing distinct types or classes of observations.

Curse of dimensionality: effects on local methods

Consider the **nearest-neighbour** procedure for inputs uniformly distributed in a p-dimensional unit hypercube.

Suppose we send out a hypercubical neighbourhood about a target point to capture a fraction r of the observations.

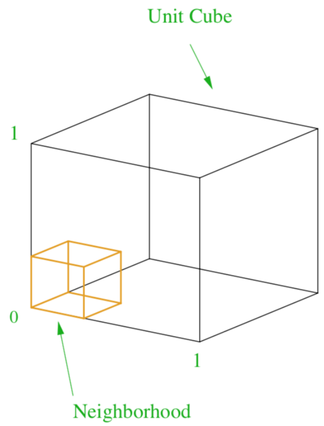
Since this corresponds to a fraction r of the unit volume, the expected edge length will be

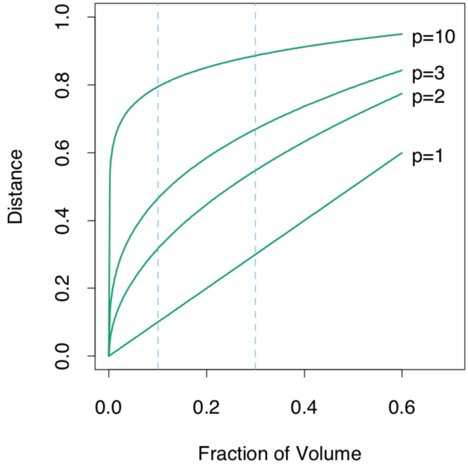
$$e_p(r) = r^{1/p}.$$

In ten dimensions $e_{10}(0.01) = 0.63$ and $e_{10}(0.1) = 0.80$, while the entire range for each input is only 1.0.

So to capture 1% or 10% of the data to form a local average, we must cover 63% or 80% of the range of each input variable. Such neighbourhoods are no longer "local".

Curse of dimensionality: effects on local methods





Curse of dimensionality: effects on local methods

Another problem is that all sample points are close to an edge of the sample.

Consider N data points uniformly distributed in a p-dimensional unit ball centred at the origin. Suppose we consider a nearest-neighbour estimate at the origin.
The median distance from the origin to the closest data point is given by the expression

$$d(p, N) = \left(1 - (1/2)^{1/N}\right)^{1/p}.$$

For $N = 500$, $p = 10$, $d(p, N) \approx 0.52$. Hence most data points are closer to the boundary of the sample space than to any other data point.

Prediction is much more difficult near the edges of the training sample.

High-dimensional data

High-dimensional data ($p \gg N$): High variance and overfitting are a major concern.

Classification methods discussed before:

- kNN: usually not very effective on high-dimensional data because of curse of dimensionality
- LDA: can't be applied to high-dimensional data directly because it's hard to get S_w^{-1}
- SVM: suggest to use linear kernel when p is very large
- RF, NN: computational cost is very high when p is very large

High-dimensional data

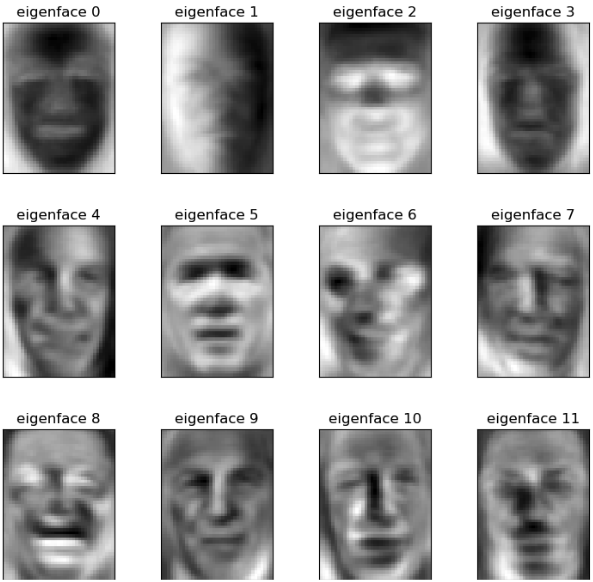
Common approaches when we'd like to use classification methods on high-dimensional data:

- Dimension reduction first, e.g. PCA+kNN
- Add regularisation terms, e.g. ridge penalty

Principal component analysis (PCA)

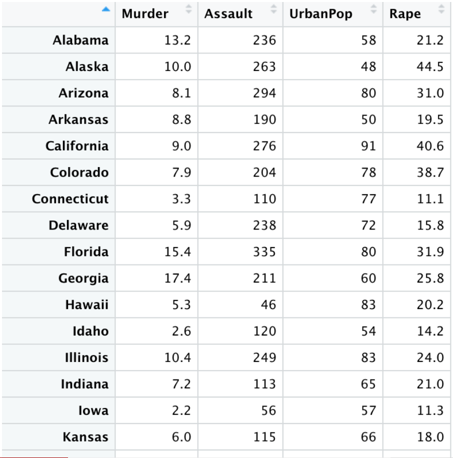
- Two examples of PCA
- Two formulations of PCA
- Neural networks and PCA

Example 1: eigenface



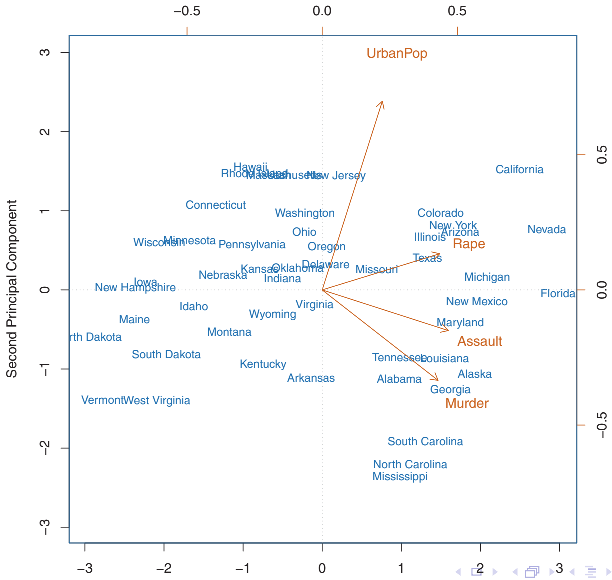
Example 2: the `USArrests` data

For each of the 50 states in the United States, the data set contains the number of arrests per 100,000 residents for each of three crimes: `Assault`, `Murder`, and `Rape`. We also record `UrbanPop` (the percent of the population in each state living in urban areas).



	Murder	Assault	UrbanPop	Rape
Alabama	13.2	236	58	21.2
Alaska	10.0	263	48	44.5
Arizona	8.1	294	80	31.0
Arkansas	8.8	190	50	19.5
California	9.0	276	91	40.6
Colorado	7.9	204	78	38.7
Connecticut	3.3	110	77	11.1
Delaware	5.9	238	72	15.8
Florida	15.4	335	80	31.9
Georgia	17.4	211	60	25.8
Hawaii	5.3	46	83	20.2
Idaho	2.6	120	54	14.2
Illinois	10.4	249	83	24.0
Indiana	7.2	113	65	21.0
Iowa	2.2	56	57	11.3
Kansas	6.0	115	66	18.0

Example 2: the USArrests data



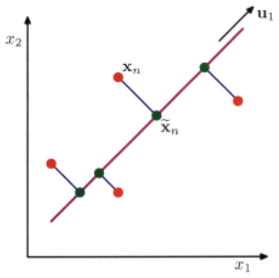
Two formulations of PCA

- The orthogonal projection of the data onto a lower dimensional linear space, such that the variance of the projected data is maximised.
- The linear projection that minimises the average projection cost, defined as the mean squared distance between the data points and their projections.

Two formulations of PCA: maximum variance

Consider a dataset of observations $\{\mathbf{x}_n\}$, where $n = 1, 2, 3, \ldots, N$, and $\mathbf{x}_n \in \mathbb{R}^p$. Our goal is to project the data onto an M dimensional space that maximises the variance of the projected data.

To begin with, consider $M = 1$ and define the direction of this space using $\mathbf{u}_1 \in \mathbb{R}^p$.



Two formulations of PCA: maximum variance

- $\mathbf{u}_1$ is a unit vector with $\mathbf{u}_1^T\mathbf{u}_1 = 1$.
- Each data point $\mathbf{x}_n$ is projected to a scalar value $\mathbf{u}_1^T\mathbf{x}_n$, a linear combination of the original features.

$$\max_{\mathbf{u}_1} \ \frac{1}{N}\sum_{n=1}^{N}\{\mathbf{u}_1^T\mathbf{x}_n - \mathbf{u}_1^T\bar{\mathbf{x}}\}^2 = \mathbf{u}_1^T\mathbf{S}\mathbf{u}_1$$

$$\text{s.t. } \mathbf{u}_1^T\mathbf{u}_1 = 1,$$

where

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^T$$

Two formulations of PCA: maximum variance

$$\mathbf{S}\mathbf{u}_1 = \lambda_1 \mathbf{u}_1$$

$$\mathbf{u}_1^T \mathbf{S}\mathbf{u}_1 = \lambda_1$$

- $\mathbf{u}_1$ is the first eigenvector of $\mathbf{S}$, which corresponds to the largest eigenvalue.
- The largest eigenvalue λ_1 is the maximum variance.
- Choosing new direction to maximise the projected variance amongst all possible directions orthogonal to those already considered.
- M eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_M$ of $\mathbf{S}$ corresponding to the M largest eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_M$.

Two formulations of PCA: maximum variance

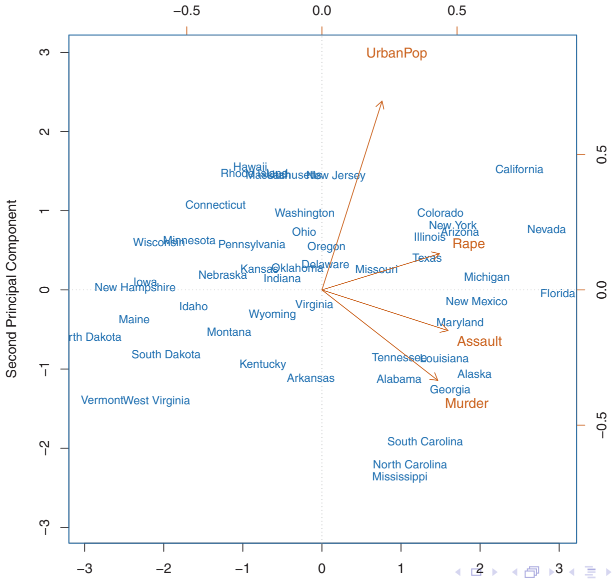
- Principal components (loadings): $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_M$
- Principal component scores: $\mathbf{u}_i^T \mathbf{x}_n$

Two formulations of PCA: maximum variance

	PC1	PC2
Murder	0.5358995	-0.4181809
Assault	0.5831836	-0.1879856
UrbanPop	0.2781909	0.8728062
Rape	0.5434321	0.1673186

Table: The principal component loading vectors, $\mathbf{u}_1$ and $\mathbf{u}_2$, for the `USArrests` data.

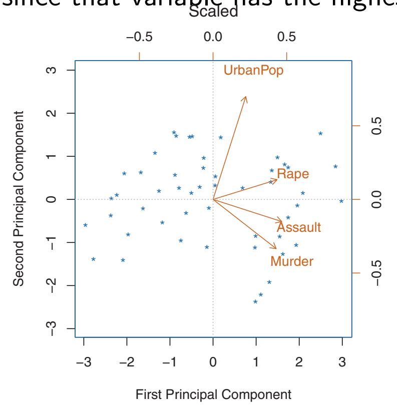
Two formulations of PCA: maximum variance

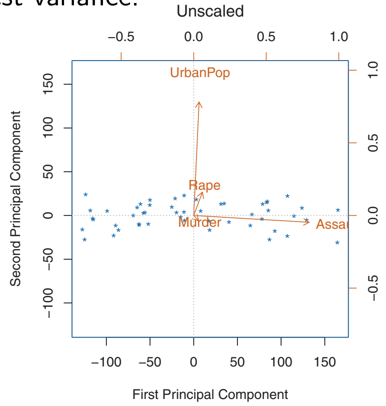


Two formulations of PCA: maximum variance

Preprocessing before using PCA:

- Variances of `Murder`, `Rape`, `Assault` and `UrbanPop`: 18.97, 87.73, 6945.16 and 209.5.
- If no scaling, the first PC will have a very large loading for `Assault`, since that variable has the highest variance.





Two formulations of PCA: maximum variance

Uniqueness of PCs: each principal component loading vector is unique, up to a sign flip.

- Flipping the sign has no effect as the direction does not change.

Two formulations of PCA: maximum variance

The proportion of variance explained

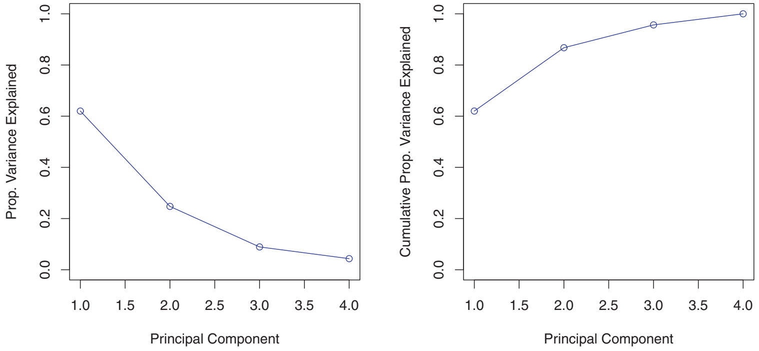
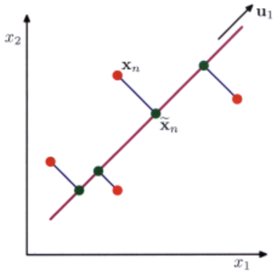


Figure: Left: a scree plot depicting the proportion of variance explained by each of the four principal components in the `USArrests` data. Right: the cumulative proportion of variance explained by the four principal components in the `USArrests` data.

Two formulations of PCA: minimum reconstruction error



Two formulations of PCA: minimum reconstruction error

$$\min \quad \frac{1}{N}\sum_{n=1}^{N} ||\mathbf{x}_n - \tilde{\mathbf{x}}_n||_F^2,$$

$$\tilde{\mathbf{x}}_n = \sum_{i=1}^{M} z_{ni}\mathbf{u}_i + \sum_{i=M+1}^{p} b_i\mathbf{u}_i$$

- $\{z_{ni}\}$ depend on the particular data point
- $\{b_i\}$ are constants that are the same for all data points

Two formulations of PCA: minimum reconstruction error

Taking derivatives with respect to z_{ni} and b_i and setting them to zeros, we obtain

$$z_{nj} = \mathbf{x}_n^T \mathbf{u}_j, \;\; j = 1, 2, \ldots, M$$

$$b_j = \bar{\mathbf{x}}^T \mathbf{u}_j, \;\; j = M+1, \ldots, p$$

$$\mathbf{x}_n - \tilde{\mathbf{x}}_n = \sum_{i=M+1}^{p} \{(\mathbf{x}_n - \bar{\mathbf{x}})^T \mathbf{u}_i\} \mathbf{u}_i$$

Thus we now aim to minimise

$$J = \frac{1}{N} \sum_{n=1}^{N} \sum_{i=M+1}^{p} (\mathbf{x}_n^T \mathbf{u}_i - \bar{\mathbf{x}}^T \mathbf{u}_i)^2 = \sum_{i=M+1}^{p} \mathbf{u}_i^T \mathbf{S} \mathbf{u}_i$$

Two formulations of PCA: minimum reconstruction error

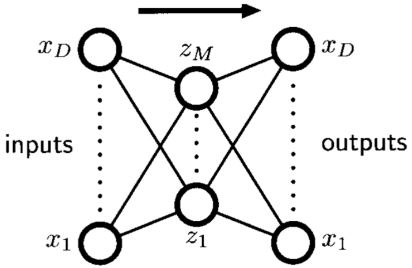
$$J = \frac{1}{N}\sum_{n=1}^{N}\sum_{i=M+1}^{p}(\mathbf{x}_n^T\mathbf{u}_i - \bar{\mathbf{x}}^T\mathbf{u}_i)^2 = \sum_{i=M+1}^{p}\mathbf{u}_i^T\mathbf{S}\mathbf{u}_i$$

- $p = 2, M = 1$: we have to choose $\mathbf{u}_2$ to minimise J, which is the eigenvector corresponding to the smaller of the two eigenvalues.
- Thus we should choose the principal subspace to be aligned with the eigenvector having the larger eigenvalue.

To minimise the average squared projection distance, we should choose the principal component subspace to pass through the mean of the data points and to be aligned with the directions of maximum variance.

Neural networks and PCA

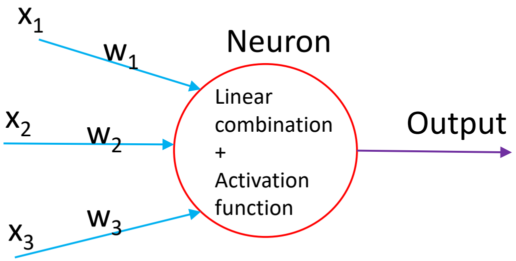
A Neural network with a single hidden layer and the number of units $M < p$ has a close relationship with PCA.



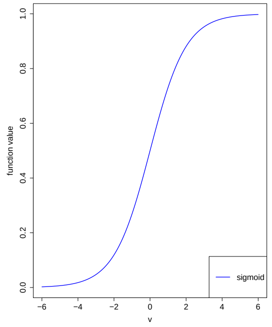
In this figure, $D = p$.

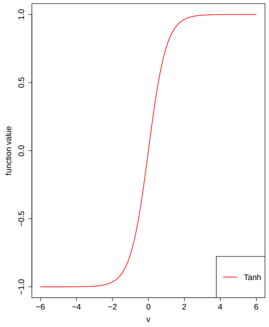
Neuron

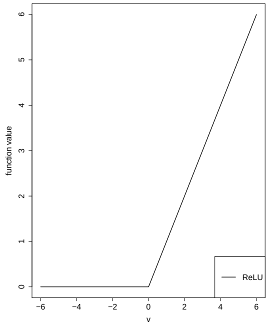
A closer look at the neuron



Activation functions







Neural networks and PCA

Autoassociative neural networks, or Autoencoder

- Has the same number of outputs as inputs.
- Map each input vector onto itself: autoassociative mapping.
- $M < p$: a perfect reconstruction of the input vecors is not in general possible.
- Minimise the reconstruction error

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} ||\mathbf{y}(\mathbf{x}_n, \mathbf{w}) - \mathbf{x}_n||_2^2$$

Neural networks and PCA

Relationship to PCA:

If the hidden units have linear activation functions, then it can be shown that $E(\mathbf{w})$ has a unique global minimum, and that at this minimum the network performs a projection onto the M-dimensional subspace which is spanned by the first M principal components of the data.

Neural networks and PCA

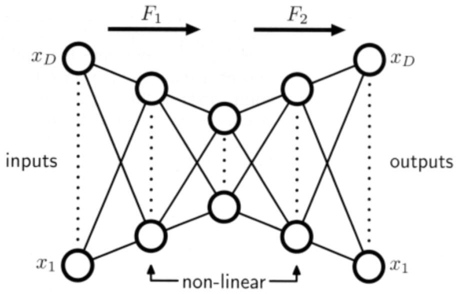
There is no advantage in using two-layer neural networks to perform dimensionality reduction.

Standard techniques for PCA:

- Guarantee to give the correct solution in finite time
- Generate an ordered set of eigenvalues with corresponding orthonormal eigenvectors

Neural networks and nonlinear PCA

Things are different if we add additional hidden layers.

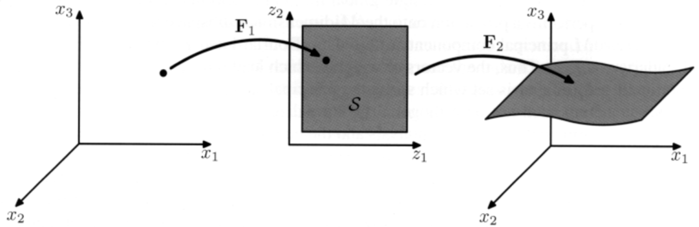


In this figure, $D = p$.

Neural networks and nonlinear PCA

- F_1: projects the original p-dimensional data onto an M-dimensional subspace S defined by the activations of the units in the second hidden layer. [Encoder]
- F_2: projects from the M-dimensional subspace back into the original p-dimensional input space. [Decoder]
- The mappings are general and is not restricted to being linear, because the nonlinear hidden layers.

Neural networks and nonlinear PCA



In this figure, $p = 3$, $M = 2$.

Neural networks and nonlinear PCA

This network effectively performs a nonlinear PCA.

- Not being limited to linear transformations.
- Computationally intensive optimisation techniques must be used.
- The dimensionality of the subspace must be specified before training the network.