Machine Learning and Data Analytics
ME 5013- Fall 2019

Lectures 14

- Dimensionality Reduction

UTSA
The University of Texas at San Antonio™

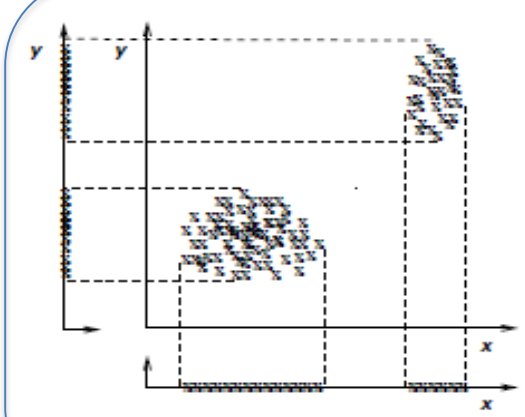
Adel Alaeddini, PhD
Associate Professor of Mechanical Engineering
Advanced Data Engineering Lab
adel.alaeddini@utsa.edu

Disclosure: the slides are adopted from various online resources

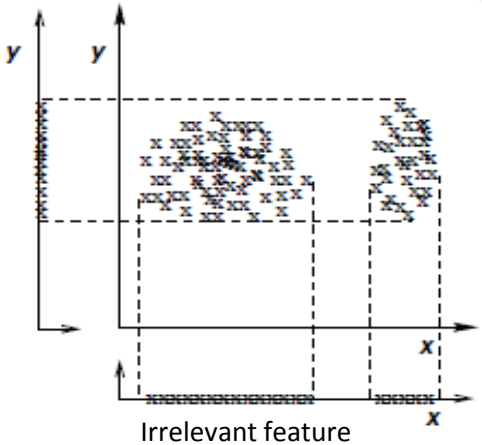
Dimensionality Reduction

Motivation:

- Computational performance
  - Data compression
- Interpretability
  - Data visualization
- Predictive Performance
  - More representative features



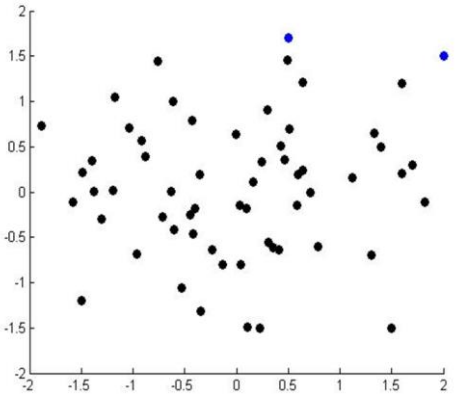
Redundant feature



Irrelevant feature

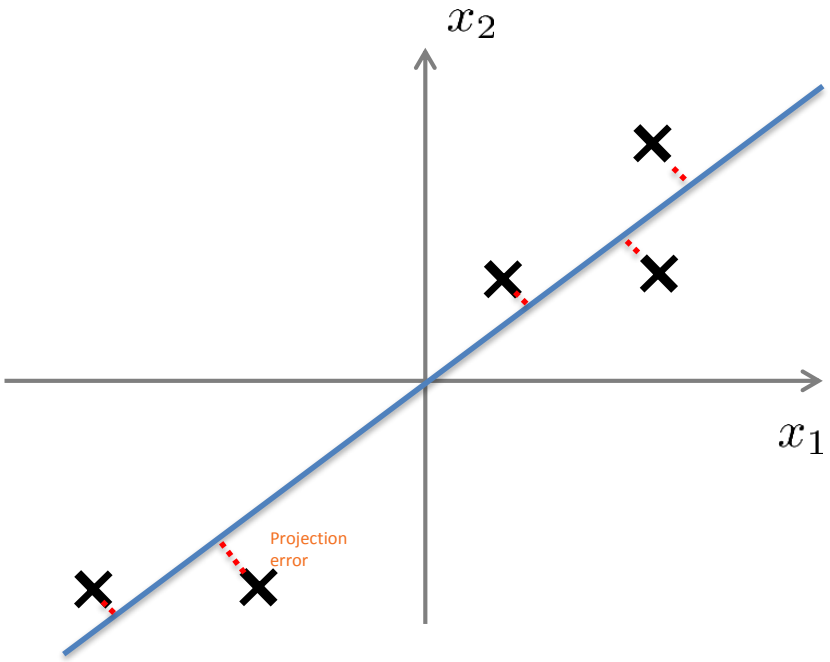
Country	GDP (trillions of US$)	Per capita GDP (thousands of intl. $)	Human Develop-ment Index	Life expectancy	Poverty Index (Gini as percentage)	Mean household income (thousands of US$)	...
Canada	1.577	39.17	0.908	80.7	32.6	67.293	...
China	5.878	7.54	0.687	73	46.9	10.22	...
India	1.632	3.41	0.547	64.7	36.8	0.735	...
Russia	1.48	19.84	0.755	65.5	39.9	0.72	...
Singapore	0.223	56.69	0.866	80	42.5	67.1	...
USA	14.527	46.86	0.91	78.3	40.8	84.3	...
...	...	...	...	...	...	...	

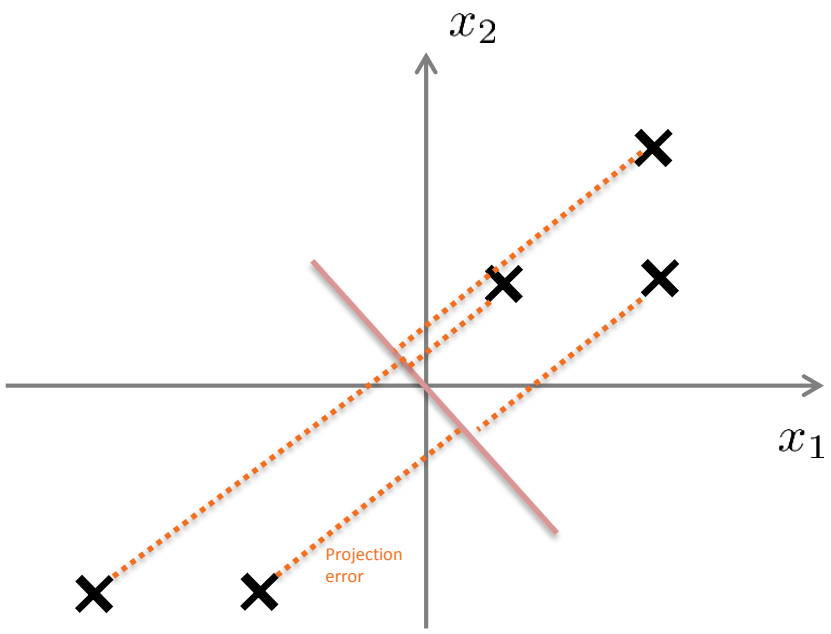
Country		
Canada	1.6	1.2
China	1.7	0.3
India	1.6	0.2
Russia	1.4	0.5
Singapore	0.5	1.7
USA	2	1.5
...	...	...



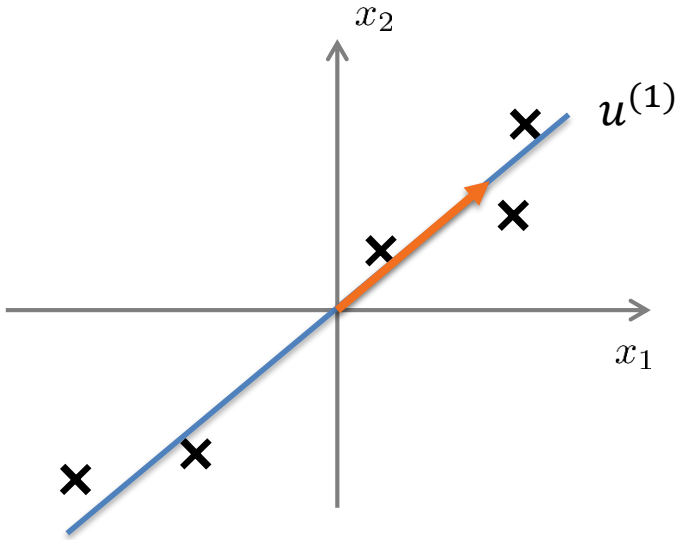
Principal Component Analysis problem formulation

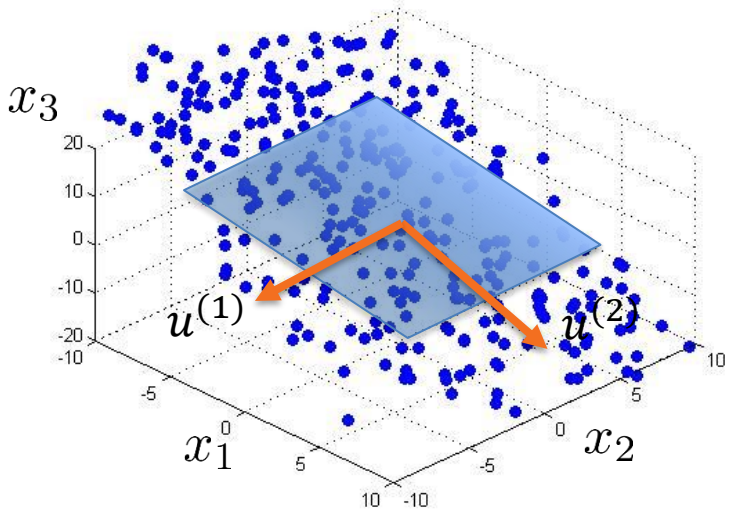
- Transform the predictors
- Project the points in a new space. i.e. line, plane, hyperplane
- The first principal component is that (normalized) linear combination of the variables with the largest variance.
- The second principal component has largest variance, subject to being uncorrelated with the first.
- Hence with many correlated original variables, we replace them with a small set of principal components that capture their joint variation.





Principal Component Analysis (PCA)

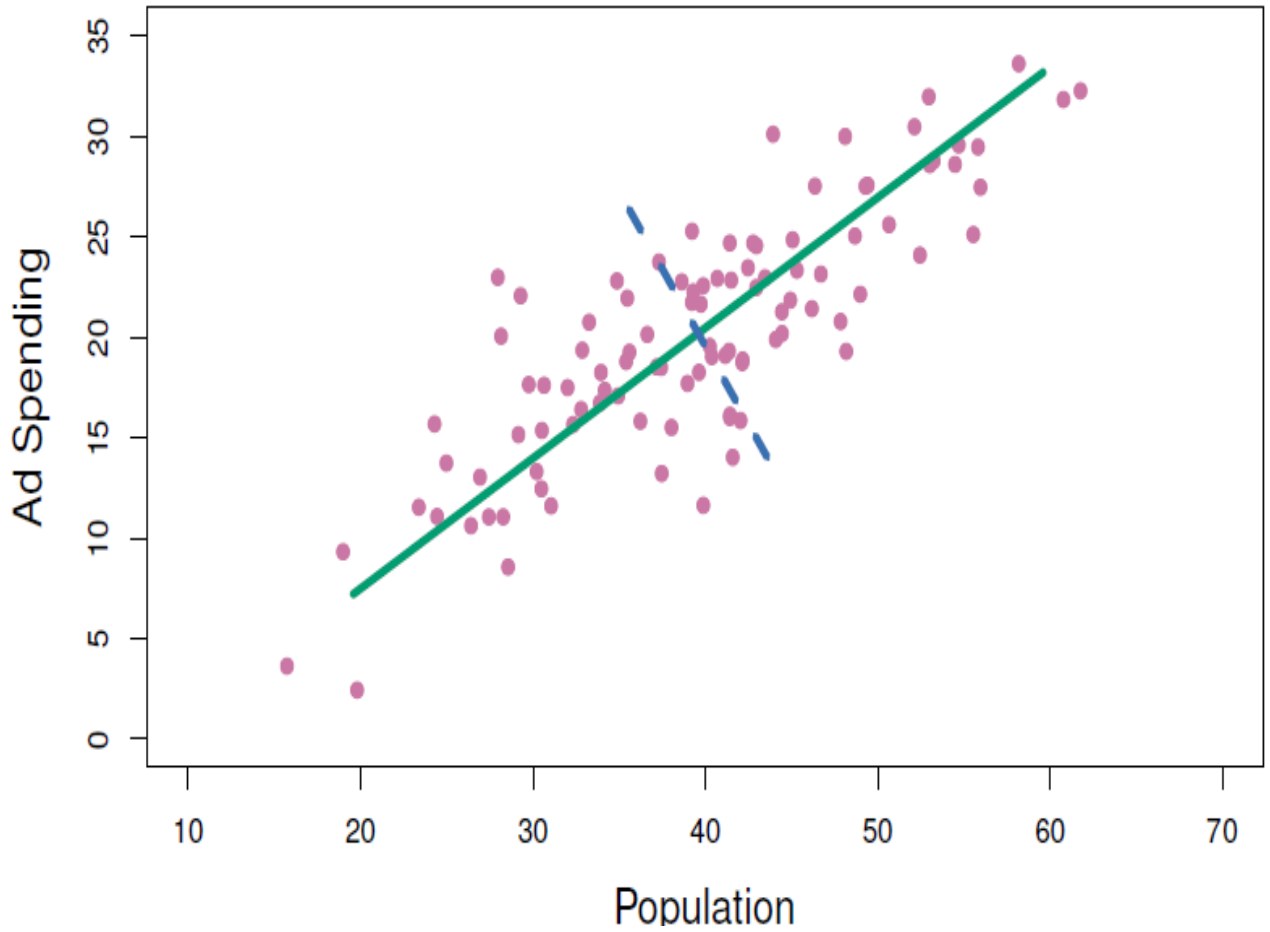




Reduce from 2-dimension to 1-dimension: Find a direction (a vector $u^{(1)} \in \mathbb{R}^n$) onto which to project the data so as to minimize the projection error.

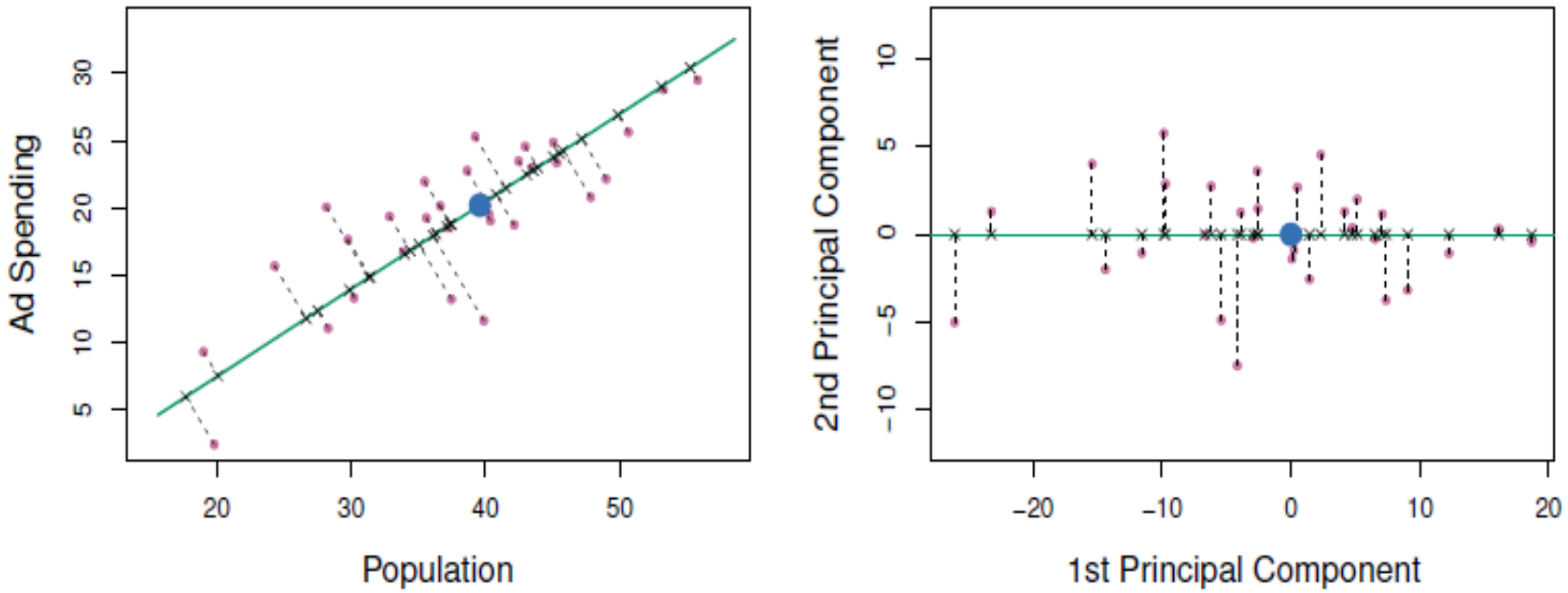
Reduce from n-dimension to k-dimension: Find k vectors $u^{(1)}, u^{(2)}, \ldots, u^{(k)}$ onto which to project the data, so as to minimize the projection error.

Example



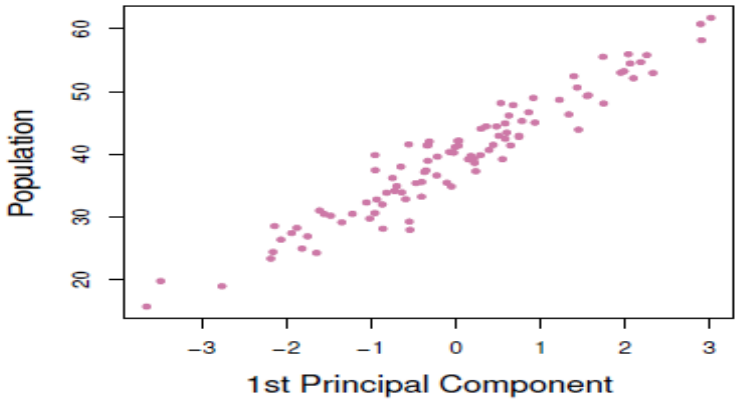
The population size (pop) and ad spending (ad) for 100 different cities are shown as purple circles. The green solid line indicates the first principal component, and the blue dashed line indicates the second principal component.

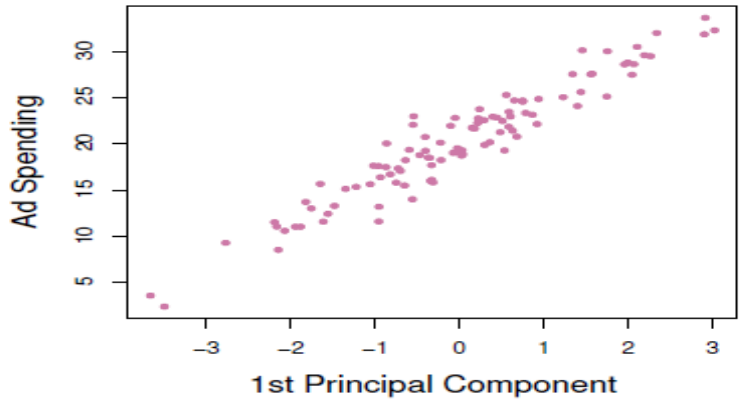
Example



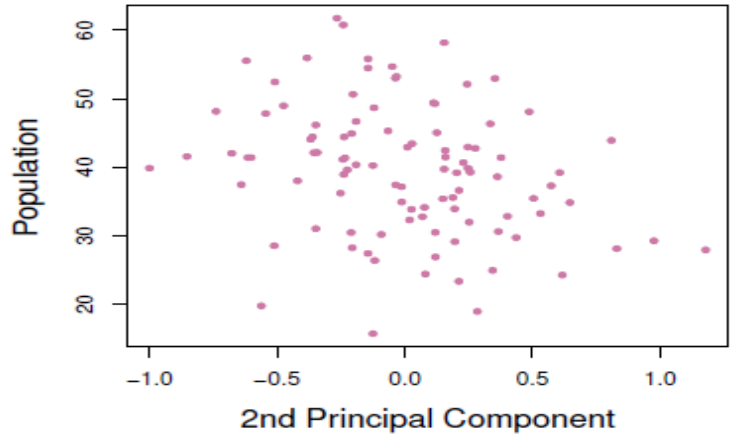
A subset of the advertising data. Left: The first principal component, chosen to minimize the sum of the squared perpendicular distances to each point, is shown in green. These distances are represented using the black dashed line segments. Right: The left-hand panel has been rotated so that the first principal component lies on the x-axis.

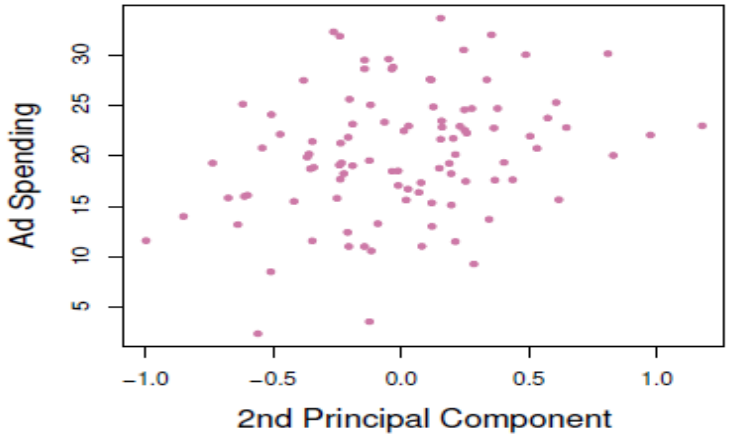
Example





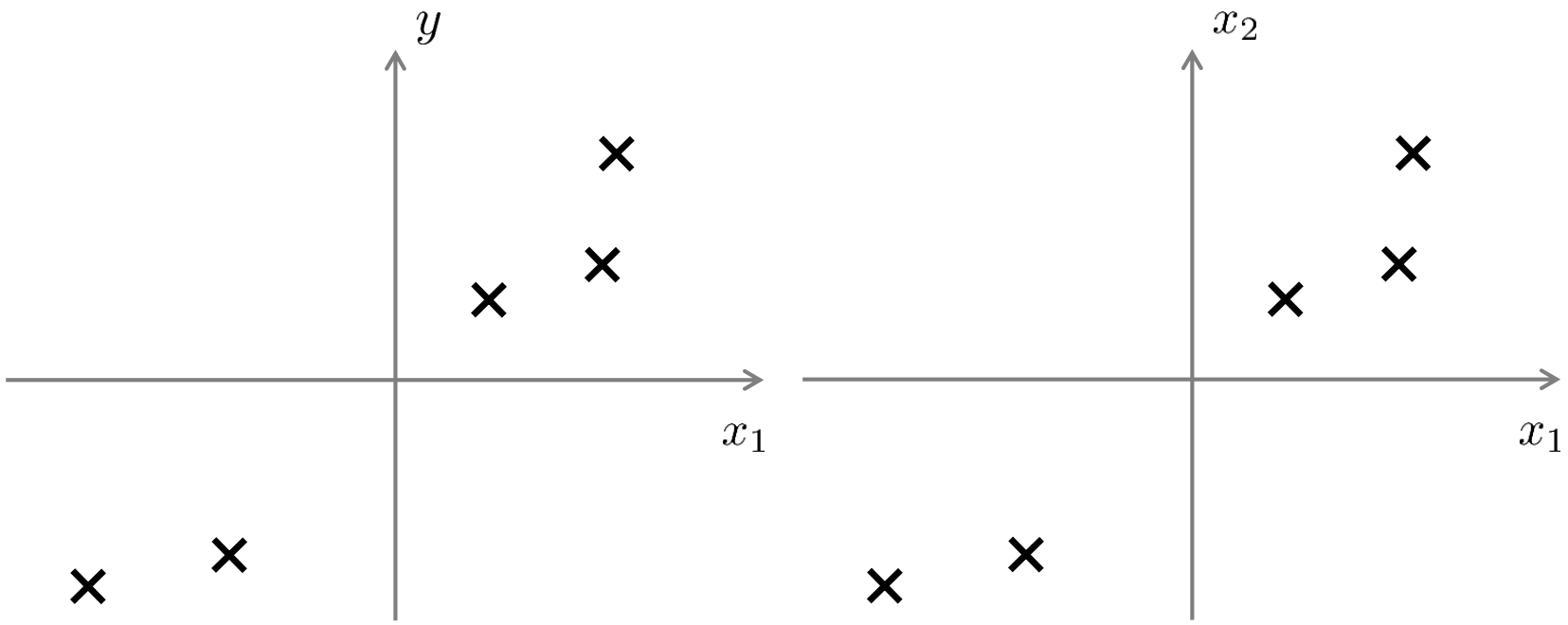
Plots of the first principal component scores versus pop and ad. The relationships are strong.





Plots of the second principal component scores versus pop and ad. The relationships are weak.

PCA vs linear regression



Principal Component Analysis algorithm

Data preprocessing

Training set: $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$

Preprocessing (feature scaling/mean normalization):

$$\mu_j = \frac{1}{m} \sum_{i=1}^{m} x_j^{(i)}$$

Replace each $x_j^{(i)}$ with $x_j - \mu_j$.

If different features on different scales (e.g., $x_1 =$ size of house, $x_2 =$ number of bedrooms), scale features to have comparable range of values.

Principal Component Analysis algorithm

$$x^{(i)} \in \mathbb{R}^p \rightarrow z^{(i)} \in \mathbb{R}^k \quad k \leq p$$

Reduce data from n-dimensions to k-dimensions

Compute "covariance matrix":

$$\text{covariance}\ \Sigma = \frac{1}{m}\sum_{i=1}^{n}(x^{(i)})(x^{(i)})^T$$

Compute "eigenvectors" of matrix Σ:

$$Au = \lambda u$$

Covariance Matrix — Eigenvector — Eigenvalue

 # Python Command

v, u LA.eig(A)

- *W: Eigenvalues*
- *u: Eigenvectors*

Example:

```
import numpy as np
from numpy import linalg as LA
A = np.array([[1,2,3],[3,2,1],[1,0,-1]])
w, v = LA.eig(A)
```

Principal Component Analysis algorithm

$$U = \begin{bmatrix} | & | & & | \\ U^{(1)} & U^{(2)} & \dots & U^{(p)} \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{p \times p}$$

U is also known as Loadings

Take firs k columns

$$U_{reduced} = \begin{bmatrix} | & | & & | \\ U^{(1)} & U^{(2)} & \dots & U^{(k)} \\ | & | & & | \end{bmatrix}_{p \times k}$$

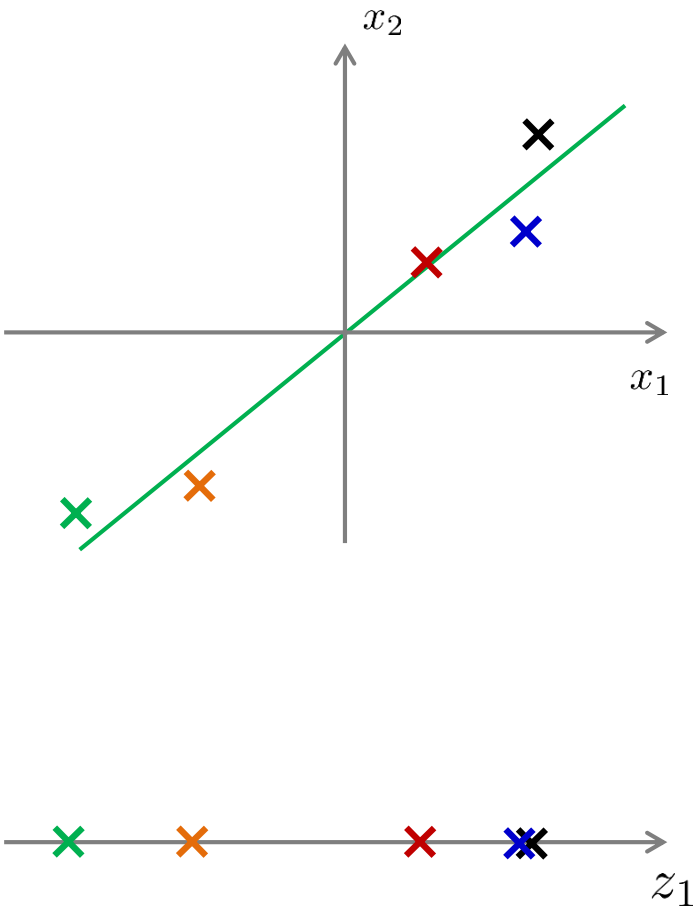
$$[Z]_{n \times k} = [X]_{n \times p} \begin{bmatrix} | & | & & | \\ U^{(1)} & U^{(2)} & \dots & U^{(k)} \\ | & | & & | \end{bmatrix}_{p \times k}$$

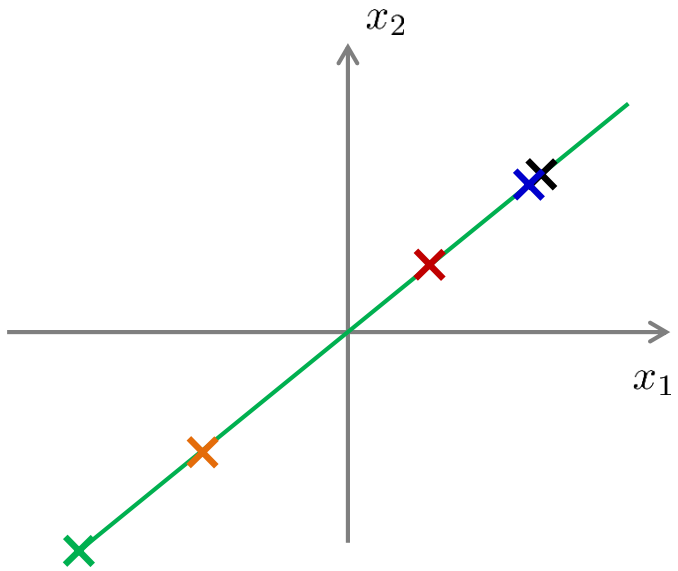
Z is also known as Scores

Reconstruction from compressed representation

$$[Z]_{n\times k} = [X]_{n\times p}[U_{Reduced}]_{p\times k}$$

$$[X^T]_{p\times n} = [Z]_{n\times k}\left[U_{Reduced}^T\right]_{k\times p} + Mean$$





Choosing k (number of principal components)

Average squared projection error: $\frac{1}{n}\sum_{i=1}^{n} \|x^{(i)} - x_{approx}^{(i)}\|^2$

Total variation in the data: $\frac{1}{n}\sum_{i=1}^{n} \|x^{(i)}\|^2$

Typically, choose k to be smallest value so that

$$\frac{\sum_{i=1}^{m} \|x^{(i)} - x_{approx}^{(i)}\|^2}{\sum_{i=1}^{n} \|x^{(i)}\|^2} \leq 0.01 \qquad (1\%)$$

"99% of variance is retained"

Keep increasing k till achieving 99%

Choosing k (number of principal components)

Pick smallest value of k for which

$$\frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{m} S_{ii}} \geq 0.99$$

Eigenvalues

(99% of variance retained)

Advice for applying PCA

Supervised learning speedup

$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})$

Extract inputs:

Unlabeled dataset: $x^{(1)}, x^{(2)}, \ldots, x^{(n)} \in \mathbb{R}^{10000}$

$\downarrow PCA$

$z^{(1)}, z^{(2)}, \ldots, z^{(n)} \in \mathbb{R}^{1000}$

New training set:

$(z^{(1)}, y^{(1)}), (z^{(2)}, y^{(2)}), \ldots, (z^{(n)}, y^{(n)})$

Note: Mapping $x^{(i)} \rightarrow z^{(i)}$ should be defined by running PCA only on the training set. This mapping can be applied as well to the examples $x_{cv}^{(i)}$ and $x_{test}^{(i)}$ in the cross validation and test sets.

Application of PCA

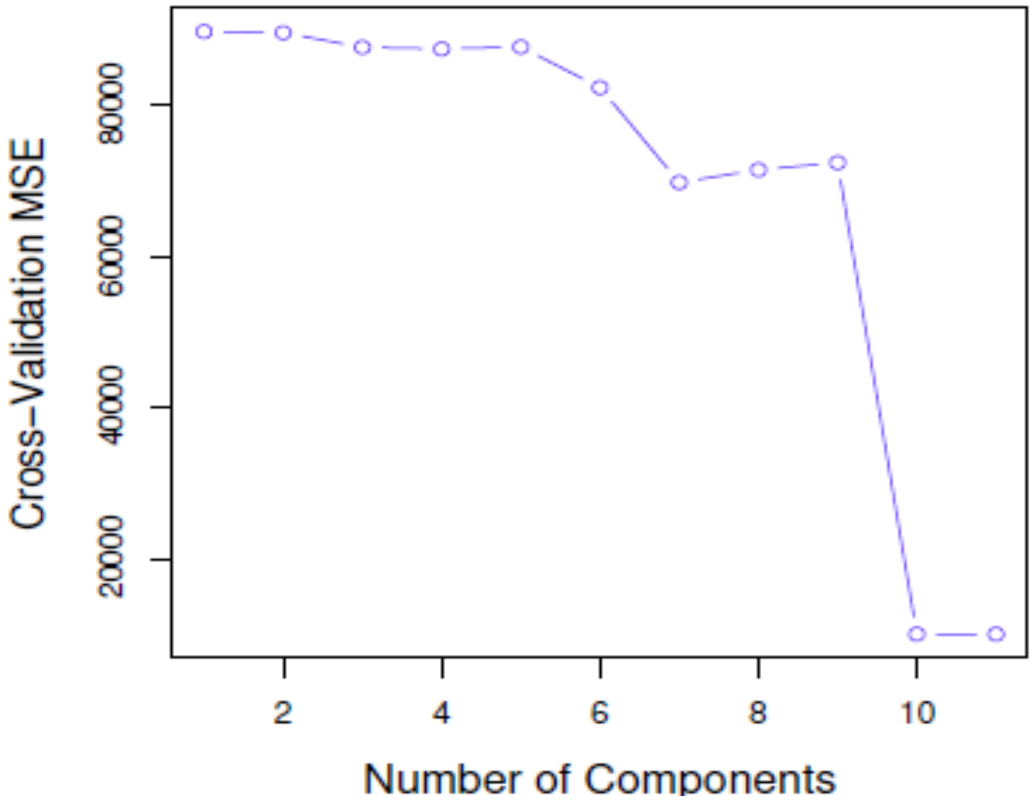
- Compression
  - Reduce memory/disk needed to store data
  - Speed up learning algorithm

    choose k by % of variance retained
- Visualization

Original

Reconstruction using 50 PCs

Example



The 10-fold cross validation MSE obtained using PCR, as a function of M.

Advice for applying PCA

Potential problem with using PCA to avoid overfitting

Use $z^{(i)}$ instead of $x^{(i)}$ to reduce the number of features to $k < n$.

Thus, fewer features, less likely to overfit!

This might work OK, but isn't a good way to address overfitting. Use regularization instead.

$$\min_{\theta} \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m} \sum_{j=1}^{n} \theta_j^2$$

Advice for applying PCA

PCA is sometimes used where it shouldn't be

Design of ML system:

- Get training set
- Run PCA to reduce $x^{(i)}$ in dimension to get $z^{(i)}$
- Train logistic regression on
- Test on test set: Map $x^{(i)}_{test}$ to $z^{(i)}_{test}$. Run on test set.

How about doing the whole thing without using PCA?

Before implementing PCA, first try running whatever you want to do with the original/raw data $x^{(i)}$. Only if that doesn't do what you want, then implement PCA and consider using $z^{(i)}$.