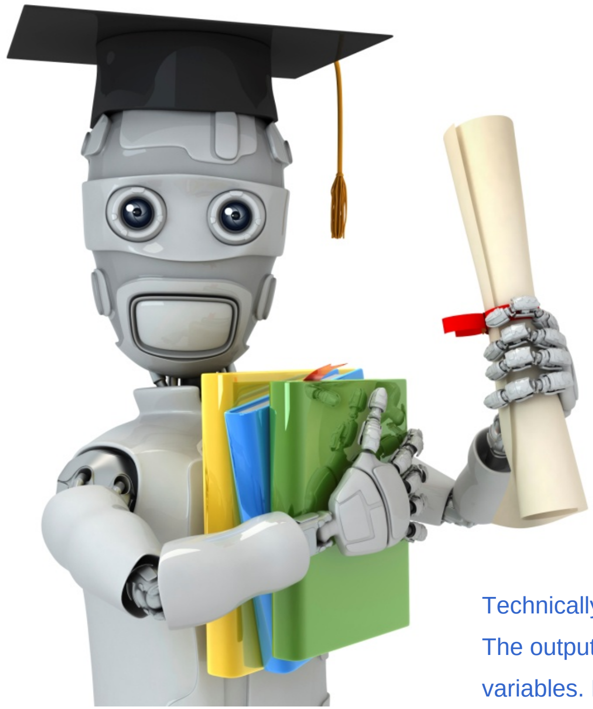


Dimensionality Reduction

Motivation I: Data Compression

Machine Learning

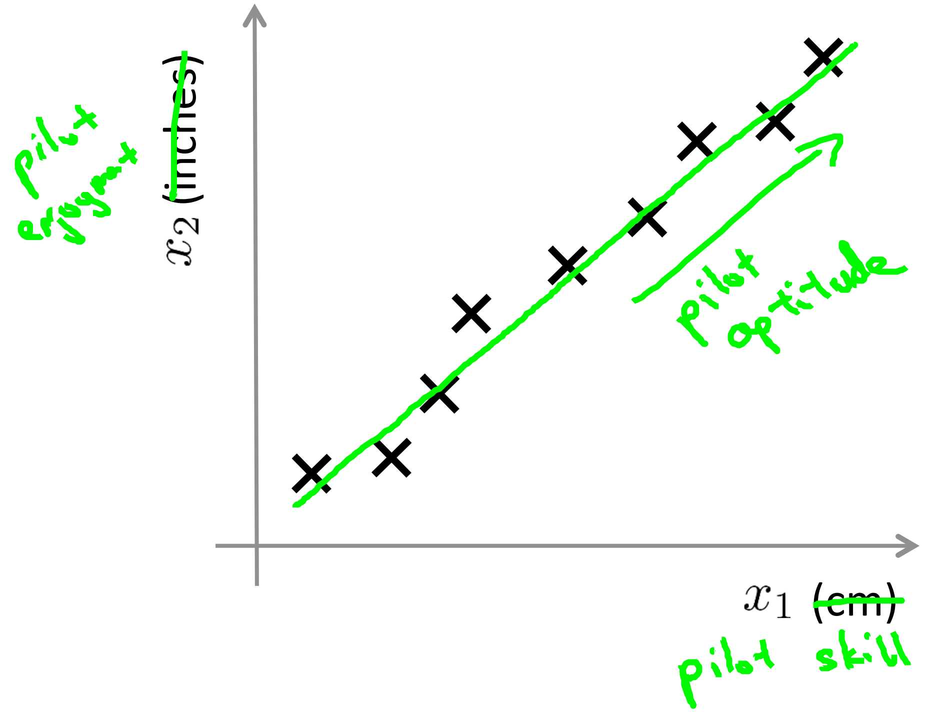
Technically, a principal component can be defined as a linear combination of optimally-weighted observed variables. The output of PCA are these principal components, the number of which is less than or equal to the number of original variables. Less, in case when we wish to discard or reduce the dimensions in our dataset.The PCs possess some useful properties which are listed below:

1- The PCs are essentially the linear combinations of the original variables, the weights vector in this combination is actually the eigenvector found which in turn satisfies the principle of least squares.

2- The PCs are orthogonal, as already discussed.

3- The variation present in the PCs decrease as we move from the 1st PC to the last one, hence the importance.

Data Compression

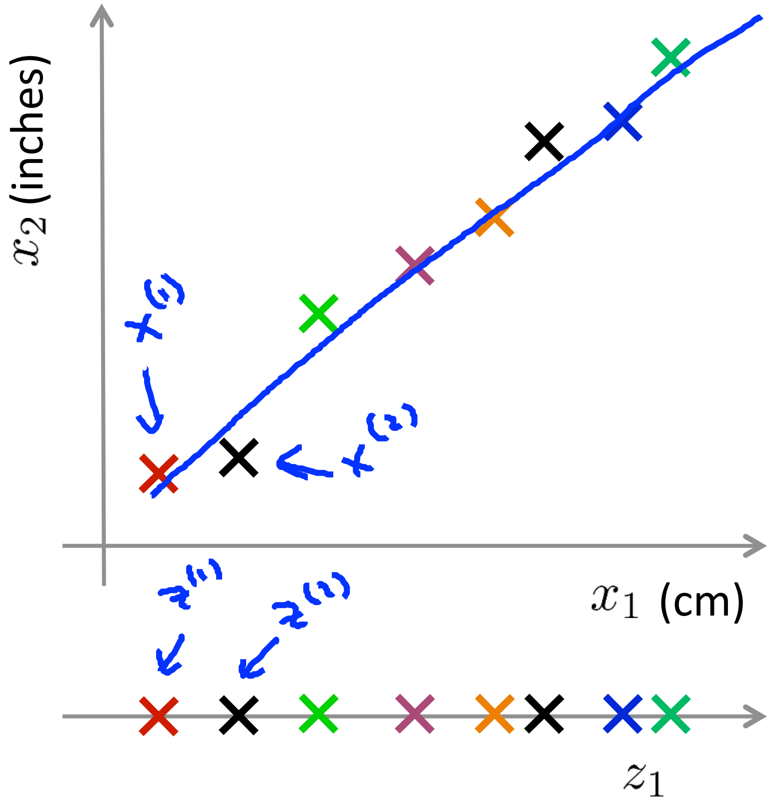


Reduce data from 2D to 1D

The main idea of principal component analysis (PCA) is to reduce the dimensionality of a data set consisting of many variables correlated with each other, either heavily or lightly, while retaining the variation present in the dataset, up to the maximum extent.

Intuitively, Principal Component Analysis can supply the user with a lower-dimensional picture, a projection or "shadow" of this object when viewed from its most informative viewpoint.

Data Compression



Reduce data from 2D to 1D

$x^{(1)} \in \mathbb{R}^2 \rightarrow z^{(1)} \in \mathbb{R}$

$x^{(2)} \in \mathbb{R}^2 \rightarrow z^{(2)} \in \mathbb{R}$

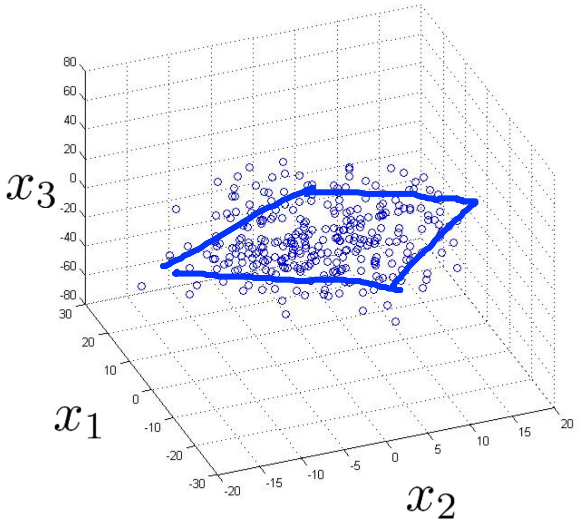
$\vdots$

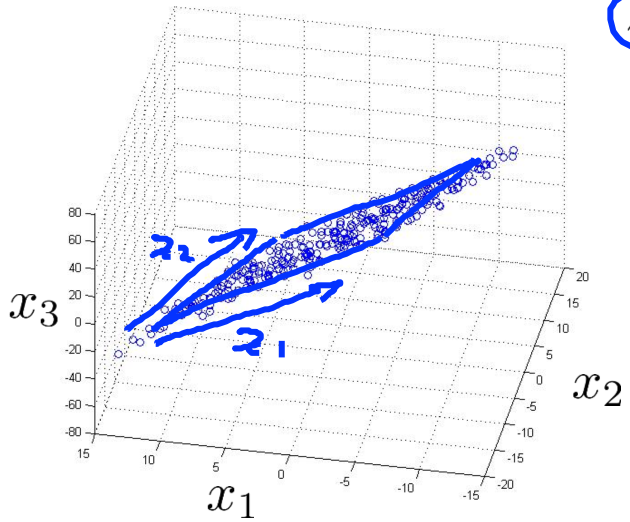
$x^{(m)} \in \mathbb{R}^2 \rightarrow z^{(m)} \in \mathbb{R}$

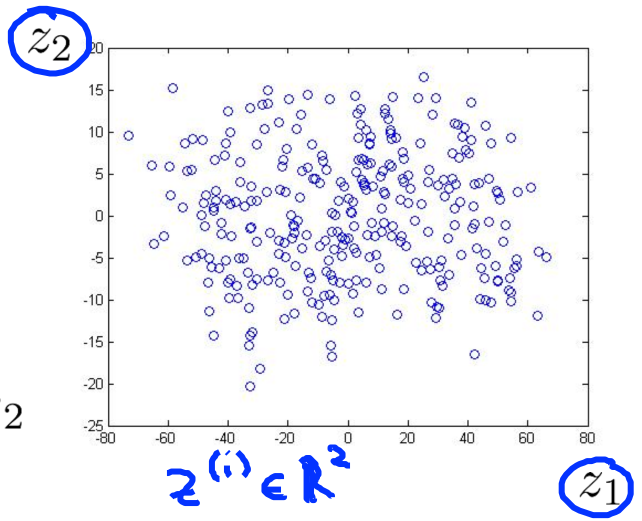
Data Compression

1000D $\rightarrow$ 100D

Reduce data from 3D to 2D





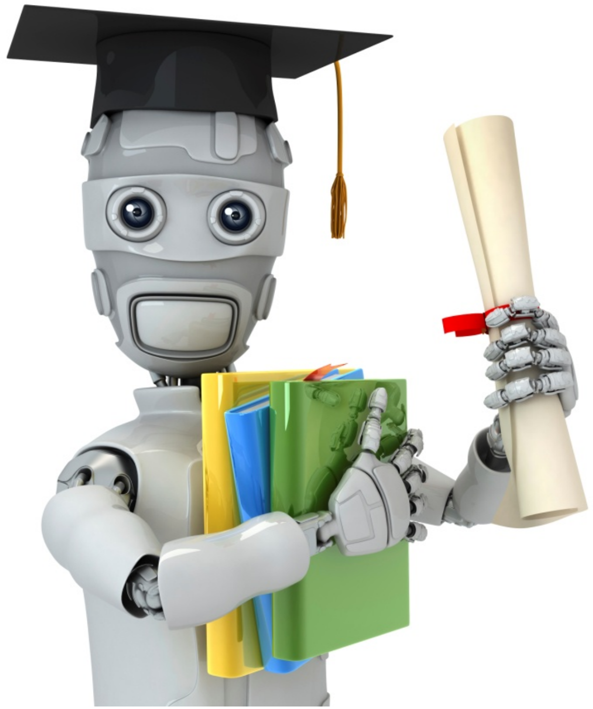


$x^{(i)} \in \mathbb{R}^3$

$z^{(i)} \in \mathbb{R}^2$

$z = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \leftarrow$ $\quad z^{(i)} = \begin{bmatrix} z_1^{(i)} \\ z_2^{(i)} \end{bmatrix} \leftarrow$

Andrew Ng



Machine Learning

Dimensionality Reduction

Motivation II: Data Visualization

The first principal component expresses the most amount of variance & most information. Each additional component expresses less variance and more noise, so representing the data with a smaller subset of principal components preserves the signal and discards the noise.

The correlation between each principal component should be zero as subsequent components capture the remaining variance. Correlation between any pair of eigenvalue/eigenvector is zero so that the axes are orthogonal, i.e., perpendicular to each other in the data space.

Data Visualization

$x \in \mathbb{R}^{50}$ $x^{(i)} \in \mathbb{R}^{50}$

Country	x_1 GDP (trillions of US$)	x_2 Per capita GDP (thousands of intl. $)	x_3 Human Develop-ment Index	x_4 Life expectancy	x_5 Poverty Index (Gini as percentage)	x_6 Mean household income (thousands of US$)	…
$\rightarrow$ 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	…
…	…	…	…	…	…	…	

[resources from en.wikipedia.org]

Andrew Ng

Data Visualization

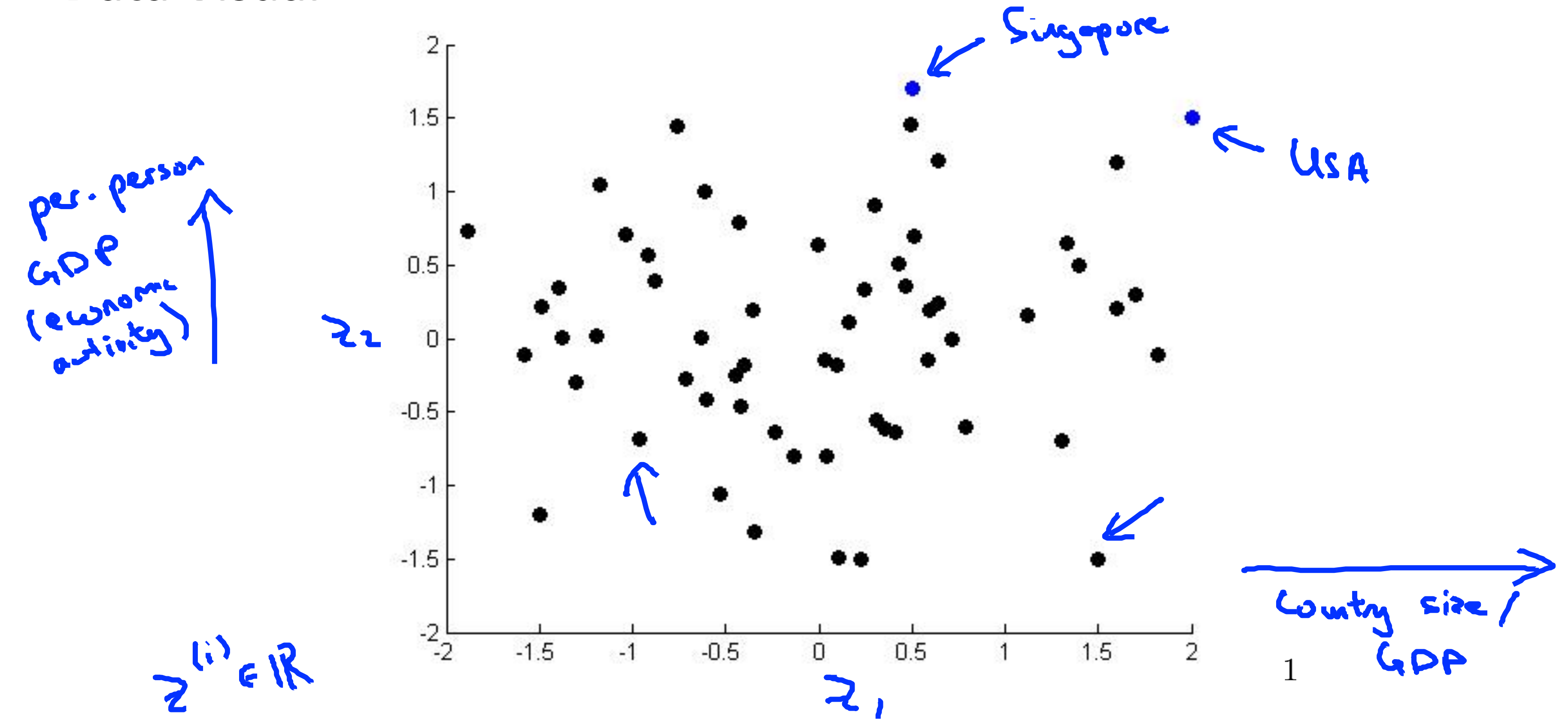
Sometimes Pcs (z1,z2) have no physical meaning

$z^{(i)} \in \mathbb{R}^2$

Country	z_1	z_2
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
…	…	…

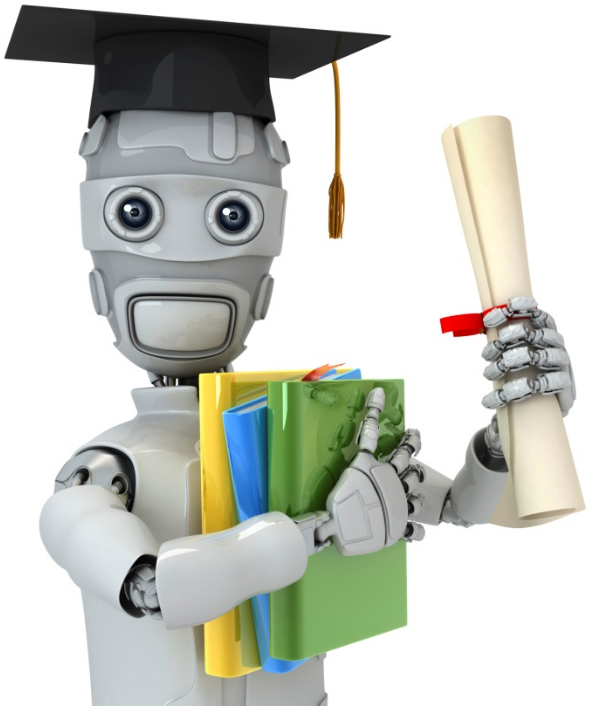
Reduce data from 50D to 2D

Data Visualization



1

Andrew Ng



Machine Learning

Dimensionality Reduction

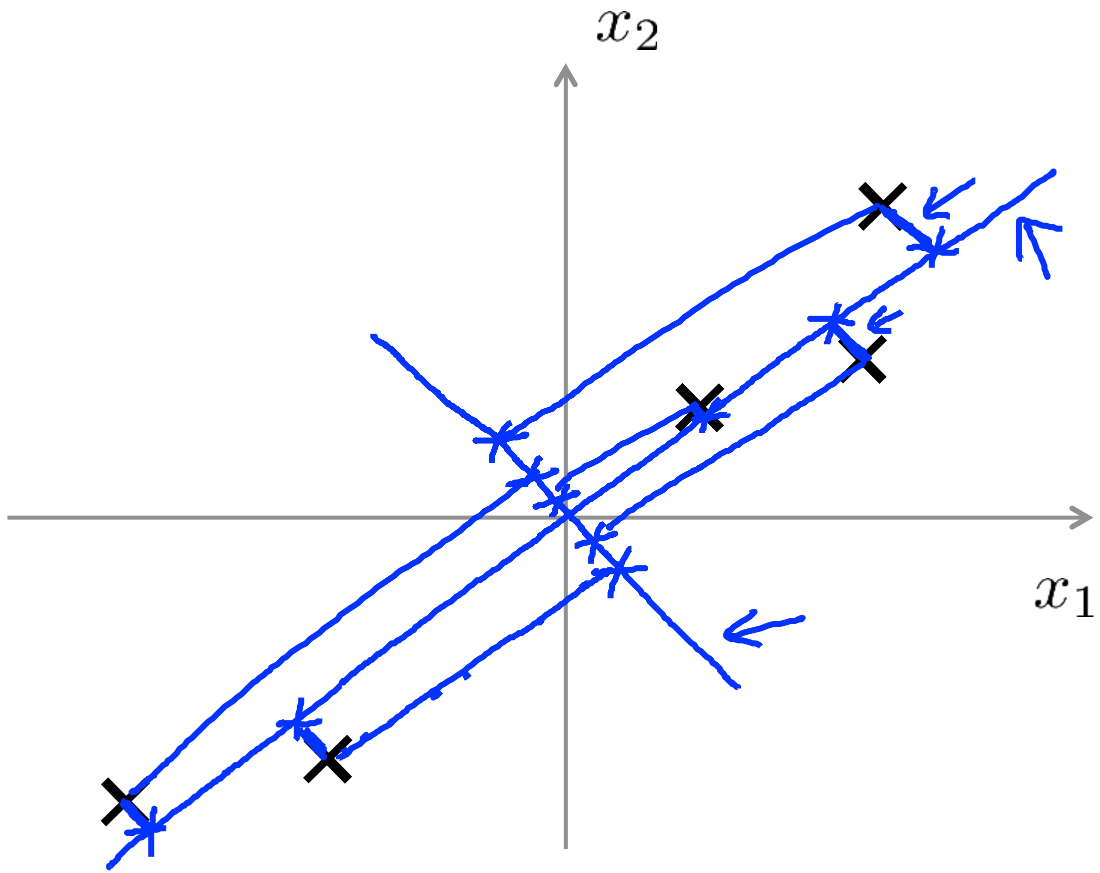
Principal Component Analysis problem formulation

Dimensionality: It is the number of random variables in a dataset or simply the number of features

Correlation: It shows how strongly two variable are related to each other. The value of the same ranges for -1 to +1

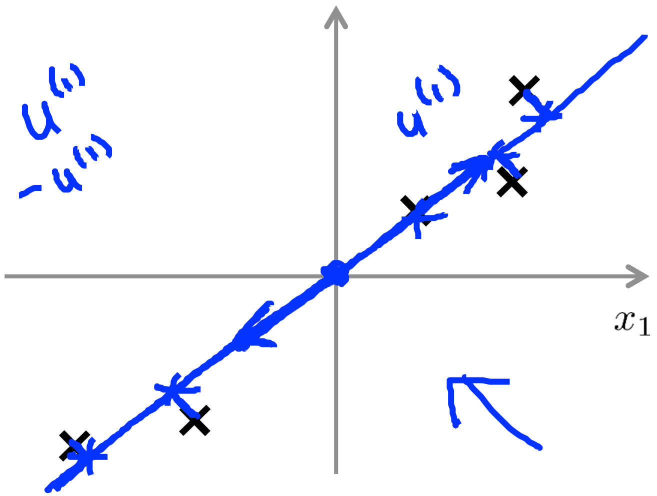
Orthogonal: Uncorrelated to each other, i.e., correlation between any pair of variables is 0.

Principal Component Analysis (PCA) problem formulation



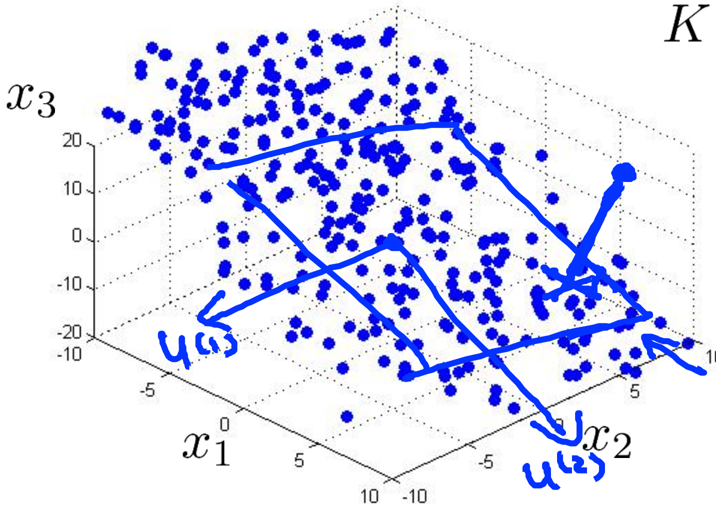
$x \in \mathbb{R}^2$

Principal Component Analysis (PCA) problem formulation



$3D \rightarrow 2D$

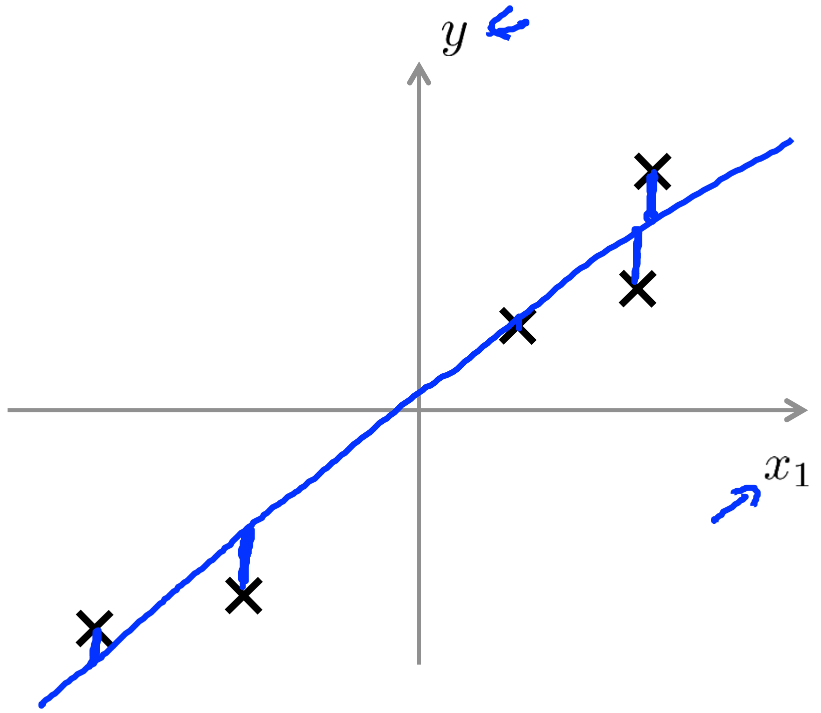
$K = 2$

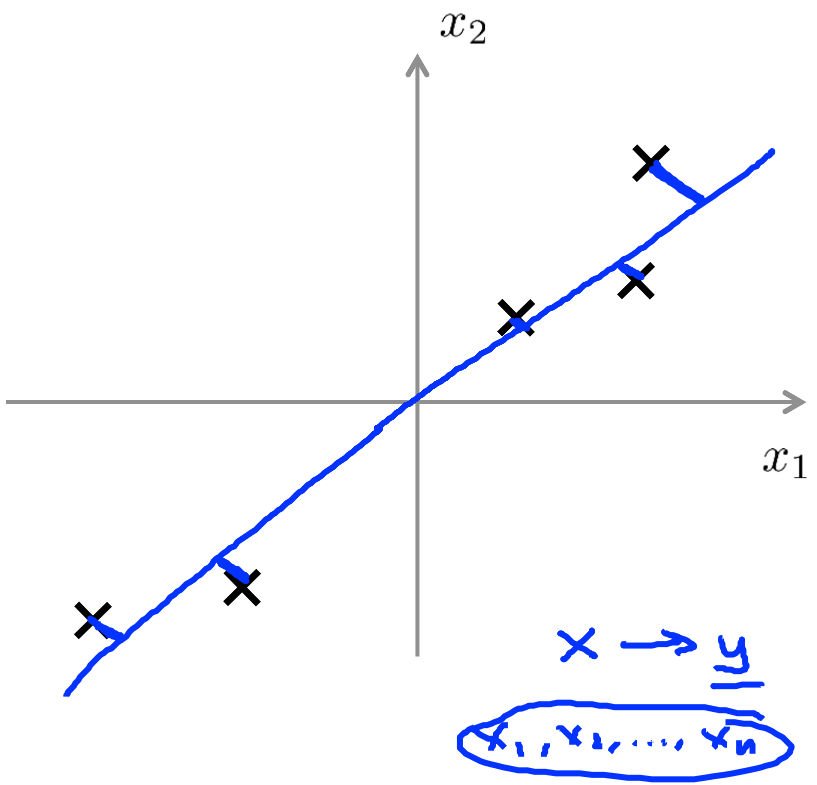


Reduce from 2-dimension to 1-dimension: Find a direction (a vector $u^{(1)} \in \mathbb{R}^n$) onto which to project the data so as to minimize the projection error.

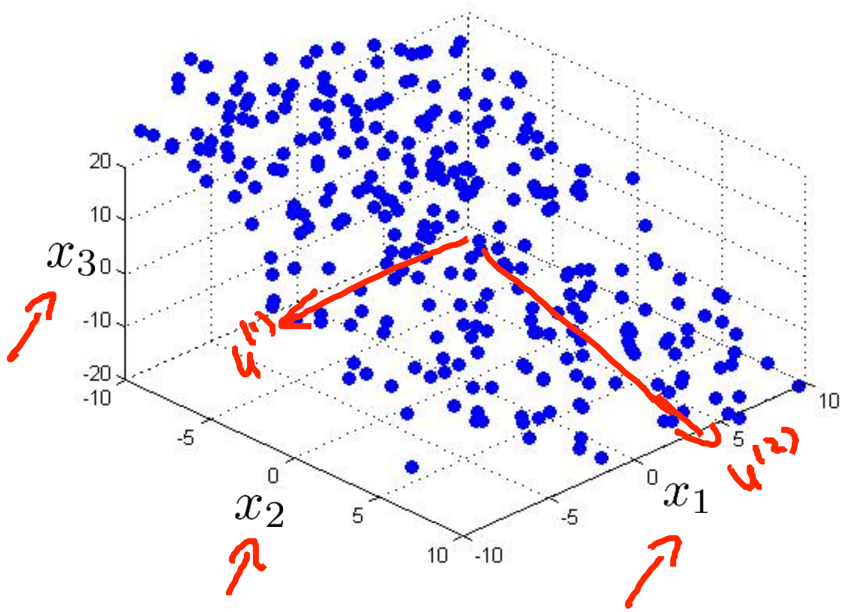
Reduce from n-dimension to k-dimension: Find k vectors $u^{(1)}, u^{(2)}, \ldots, u^{(k)}$ onto which to project the data, so as to minimize the projection error.

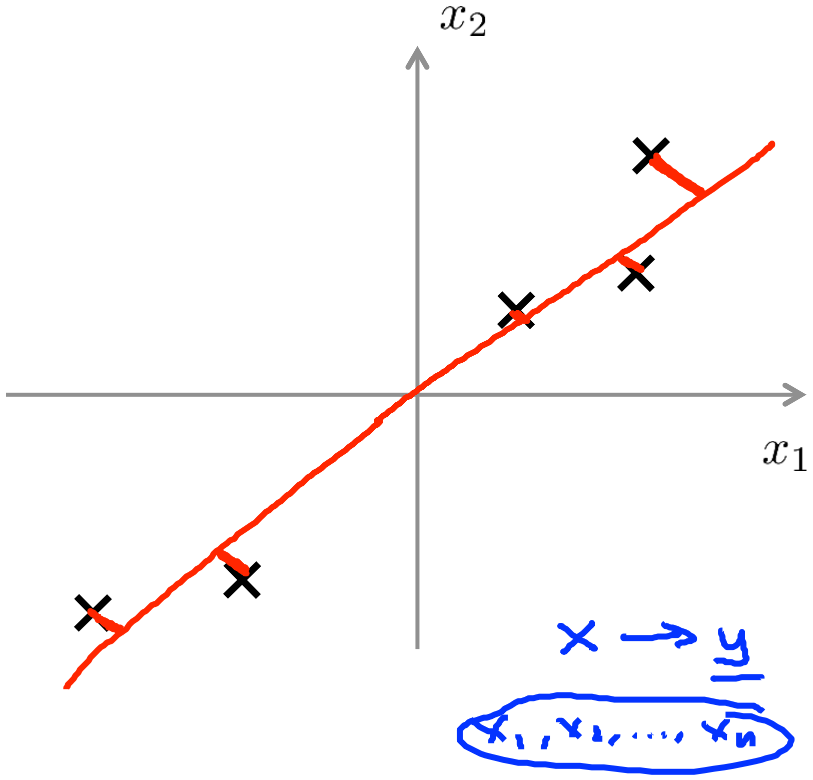
PCA is not linear regression



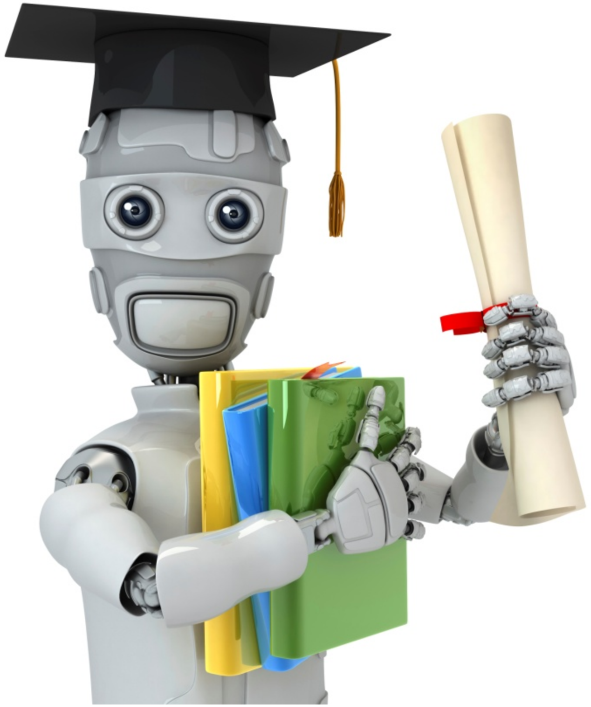


PCA is not linear regression





Andrew Ng



Machine Learning

Dimensionality Reduction

Principal Component Analysis algorithm

Data preprocessing

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

Preprocessing (feature scaling/mean normalization):

The dataset on which PCA technique is to be used must be scaled.

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

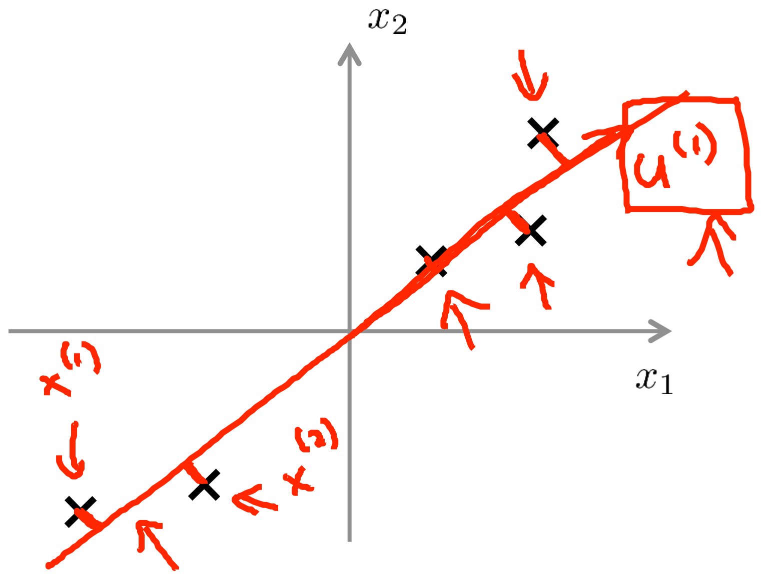
Replace each $x_j^{(i)}$ with $x_j - \mu_j$.

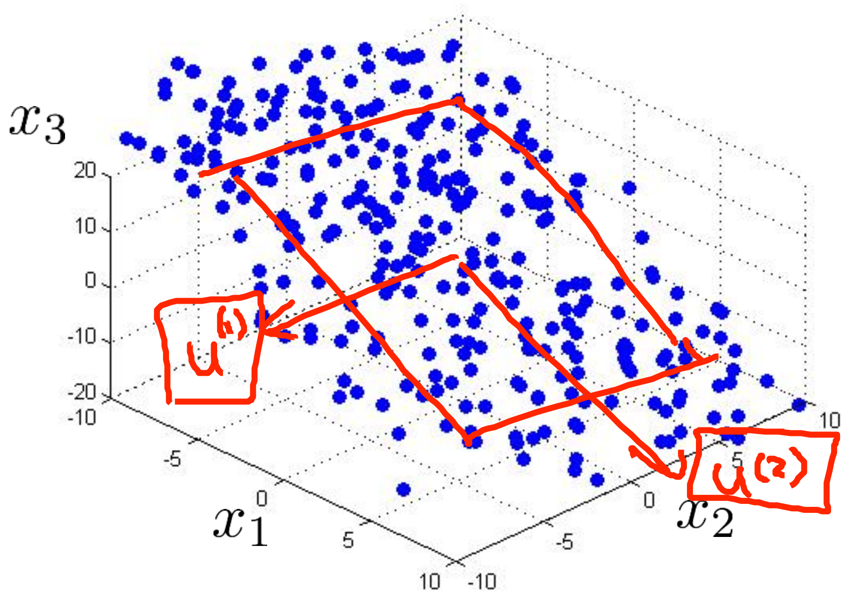
If different features on different scales (e.g., $x_1 =$ size of house, $x_2 =$ number of bedrooms), scale features to have comparable range of values.

$$x_j^{(i)} \leftarrow \frac{x_j^{(i)} - \mu_j}{s_j}$$

Andrew Ng

Principal Component Analysis (PCA) algorithm



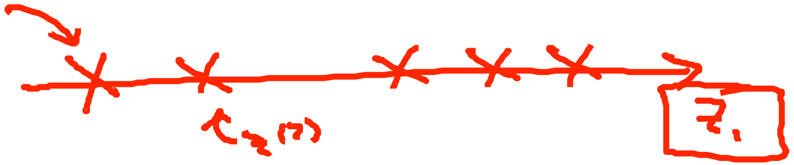


Reduce data from 2D to 1D

Reduce data from 3D to 2D

$z^{(1)}$ $x^{(i)} \in \mathbb{R}^2 \rightarrow z^{(i)} \in \underline{\mathbb{R}}$

$x^{(i)} \in \mathbb{R}^3 \rightarrow z^{(i)} \in \mathbb{R}^2$



$$z = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

Andrew Ng

The line which maximizes the variance of the data once it is projected into the data space is equivalent to finding the path which minimizes the least-squares distance of the projection.

Principal Component Analysis (PCA) algorithm

Reduce data from n-dimensions to $\underline{k\text{-dimensions}}$

Compute "covariance matrix":

Sigma not summation

$$\underline{\Sigma} = \frac{1}{m}\sum_{i=1}^{n} \boxed{\underbrace{(x^{(i)})}_{n\times 1}\underbrace{(x^{(i)})^T}_{1\times n}} \quad n\times n$$

Sigma

Compute "eigenvectors" of matrix Σ:

→ Singular value decomposition

eig(Sigma)

```
[U,S,V] = svd(Sigma);
```

n×n matrix

$$U = \begin{bmatrix} | & | & | & & | \\ u^{(1)} & u^{(2)} & u^{(3)} & \cdots & u^{(n)} \\ | & | & | & & | \end{bmatrix}$$

$\underbrace{u^{(1)}\ u^{(2)}\ u^{(3)}}_{k}$

$U \in \mathbb{R}^{n\times n}$

$u^{(1)}, \ldots, u^{(k)}$

Andrew Ng

Principal Component Analysis (PCA) algorithm

From `[U,S,V] = svd(Sigma)`, we get:

$$\rightarrow U = \underbrace{\begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \dots & u^{(n)} \\ | & | & & | \end{bmatrix}}_{k} \in \mathbb{R}^{n \times n}$$

Can be ex. of a training set or validation set or test set

$$x \in \mathbb{R}^n \longrightarrow z \in \mathbb{R}^k$$

$$z^{(i)} = \underbrace{\begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \dots & u^{(k)} \\ | & | & & | \end{bmatrix}}_{n \times k}^{T} x^{(i)} = \underbrace{\underbrace{\begin{bmatrix} — (u^{(1)})^T — \\ \vdots \\ — (u^{(k)})^T — \end{bmatrix}}_{k \times n} \underset{n \times 1}{x^{(i)}}}_{k \times 1}$$

U_{reduce}

$z \in \mathbb{R}^k$

Andrew Ng

Principal Component Analysis (PCA) algorithm summary

PCA will rotate the data in a possibly undesired way if feature normalization isn't applied

→ After mean normalization (ensure every feature has zero mean) and optionally feature scaling:

```
Sigma =
```
$$\underline{\frac{1}{m}\sum_{i=1}^{m}(x^{(i)})(x^{(i)})^T} \leftarrow$$

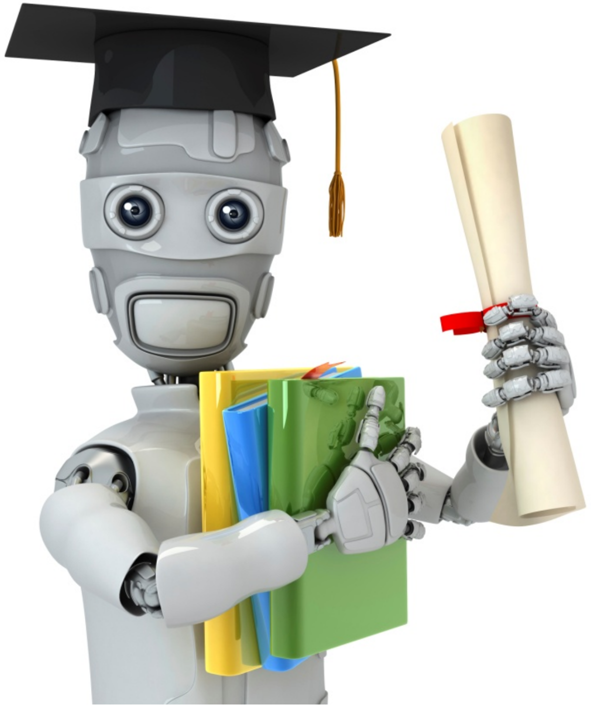
$$\underline{X} = \begin{bmatrix} — & x^{(1)T} & — \\ & \vdots & \\ — & x^{(m)T} & — \end{bmatrix}$$

→ $\boxed{\text{Sigma} = (1/m) * X' * X;}$

```
→ [U,S,V] = svd(Sigma);
→ Ureduce = U(:,1:k);
→ z = Ureduce'*x;
```

$x \in \mathbb{R}^n$ $\cancel{x_0 = 1}$

Andrew Ng

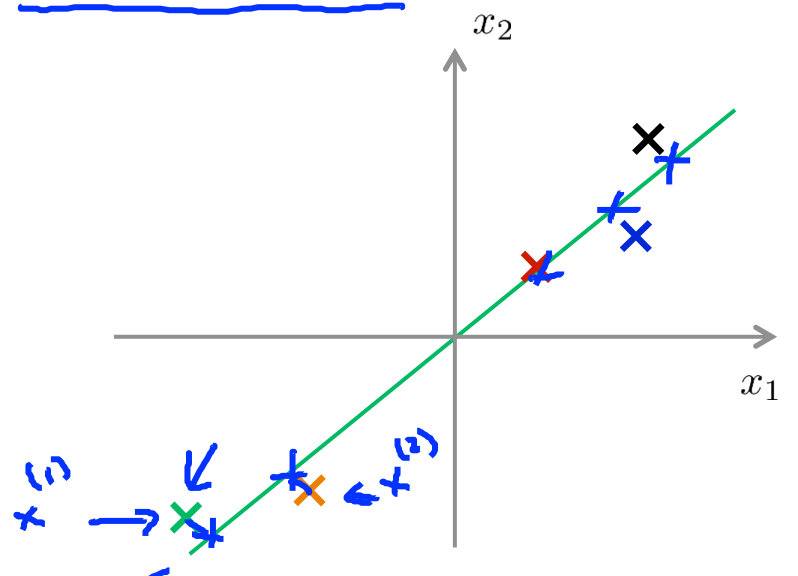


Machine Learning

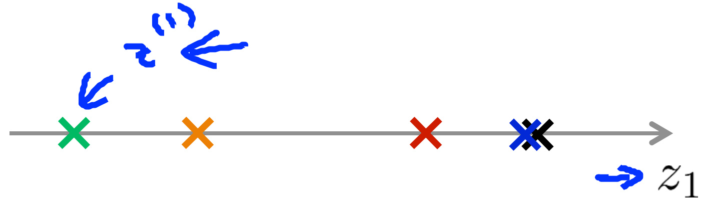
Dimensionality Reduction

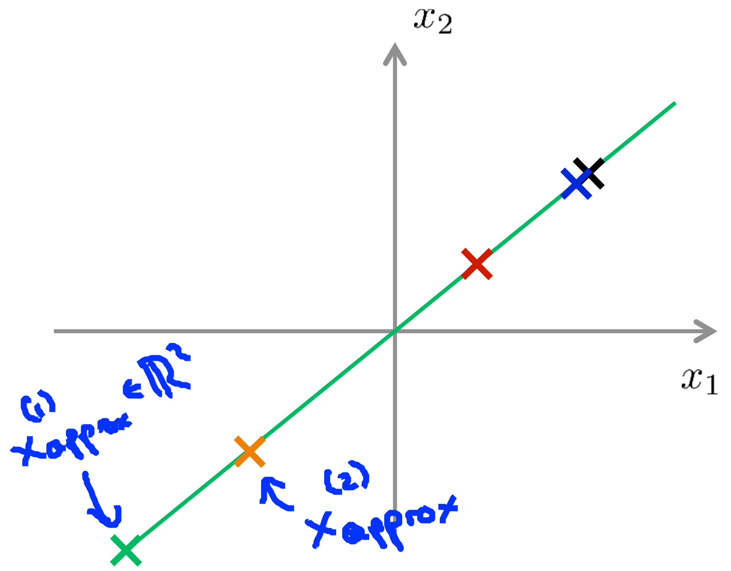
Reconstruction from compressed representation

Reconstruction from compressed representation



$$\rightarrow z = U_{reduce}^T x$$



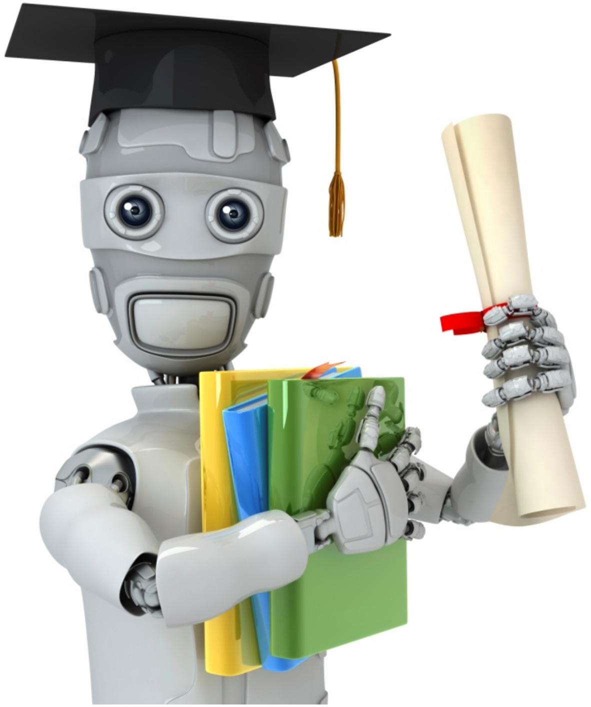


$$z \in \mathbb{R} \rightarrow x \in \mathbb{R}^2$$

$$\approx x$$

$$\underbrace{x_{approx}^{(1)}}_{\in \mathbb{R}^n} = \underbrace{\underbrace{U_{reduce}}_{n\times k} \cdot \underbrace{z^{(1)}}_{k\times 1}}_{n\times 1}$$

Andrew Ng



Machine Learning

Dimensionality Reduction

Choosing the number of principal components

Choosing k (number of principal components)

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

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

Typically, choose k to be smallest value so that

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

$\cancel{0.05}$ $\qquad$ $\cancel{5\%}$

0.10 $\qquad$ (10%)

$\swarrow$

$\rightarrow$ "~~99%~~ of variance is retained"

~~95%~~ 90%

Choosing k (number of principal components)

Algorithm:

Try PCA with ~~$k = 1$~~ ~~k=2~~ ~~k=3~~ k=4 ...

Compute $U_{reduce}, \underline{z^{(1)}}, \underline{z^{(2)}}, \ldots, \underline{z^{(m)}}, x^{(1)}_{approx}, \ldots, x^{(m)}_{approx}$

Check if

$$\boxed{\frac{\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)} - x^{(i)}_{approx}\|^2}{\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)}\|^2}} \le 0.01?$$

k=17

→ `[U,S,V] = svd(Sigma)`

$$\rightarrow S = \begin{bmatrix} s_{11} & & & & 0 \\ & s_{22} & & & \\ & & s_{33} & & \\ & & & \ddots & \\ 0 & & & & s_{nn} \end{bmatrix}$$

k=3

For given k

$$\rightarrow 1 - \frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}} \le 0.01$$

$$\rightarrow \boxed{\frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}} \ge \underline{0.99}}$$

Choosing k (number of principal components)

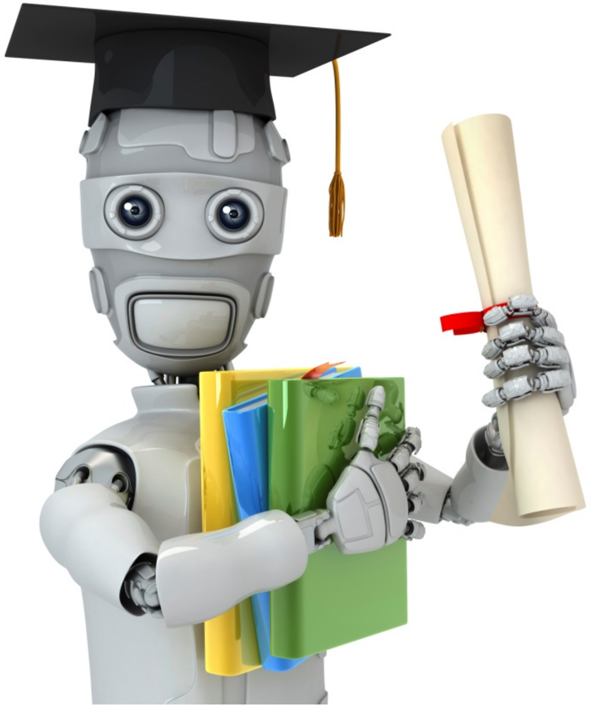
$\rightarrow$ `[U,S,V] = svd(Sigma)`

Pick smallest value of k for which

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

$k=100$

(99% of variance retained)



Machine Learning

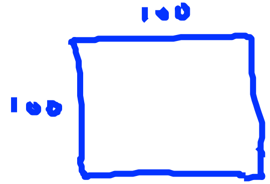
Dimensionality Reduction

Advice for applying PCA

Supervised learning speedup

$x^{(i)} \in \mathbb{R}^{10,000} \leftarrow$

$\rightarrow \boxed{(\underline{x}^{(1)}, y^{(1)})}, (\underline{x}^{(2)}, y^{(2)}), \ldots, (\underline{x}^{(m)}, y^{(m)})$



Extract inputs:

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

$$\downarrow PCA$$

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

$$\begin{tikzcd} x \arrow[d] \\ z \end{tikzcd}$$

U_{reduce}

New training set:

$\boxed{(z^{(1)}, y^{(1)})}, \underline{(z^{(2)}, y^{(2)})}, \ldots, \underline{(z^{(m)}, y^{(m)})}$

$h_\theta(z) = \frac{1}{1+e^{-\theta^T z}}$

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.

$\underline{x \rightarrow z}$

Andrew Ng

Application of PCA

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

Choose k by % of variance retain

- Visualization

k=2 or k=3

Bad use of PCA: To prevent overfitting

$\rightarrow$ Use $\underline{z^{(i)}}$ instead of $\underline{x^{(i)}}$ to reduce the number of features to $\underline{\overset{1000}{k}} < \underline{n}$. — 10000

Thus, fewer features, less likely to overfit.

Bad!

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

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

Andrew Ng

PCA is sometimes used where it shouldn't be

Design of ML system:

- Get training set $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$
- ~~Run PCA to reduce $x^{(i)}$ in dimension to get $z^{(i)}$~~
- Train logistic regression on $\{(\overset{x^{(i)}}{\cancel{z^{(1)}}}, y^{(1)}), \ldots, (\underset{x^{(m)}}{\cancel{z^{(m)}}}, y^{(m)})\}$
- Test on test set: Map $x_{test}^{(i)}$ to $z_{test}^{(i)}$. Run $h_\theta(z)$ on $\{(z_{test}^{(1)}, y_{test}^{(1)}), \ldots, (z_{test}^{(m)}, y_{test}^{(m)})\}$

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 $\boxed{x^{(i)}}$. Only if that doesn't do what you want, then implement PCA and consider using $\underline{z^{(i)}}$.

Andrew Ng