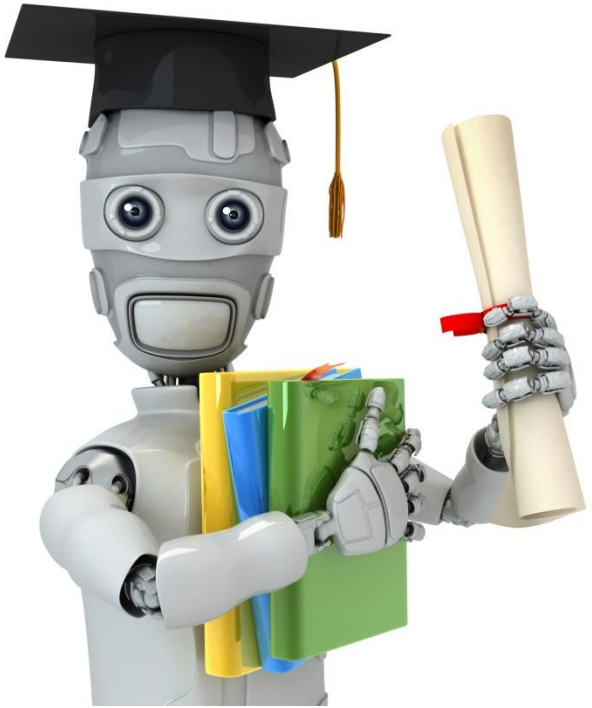
CSE 4621
Machine Learning

Lecture 13

Md. Hasanul Kabir, PhD.

Professor, CSE Department

Islamic University of Technology (IUT)

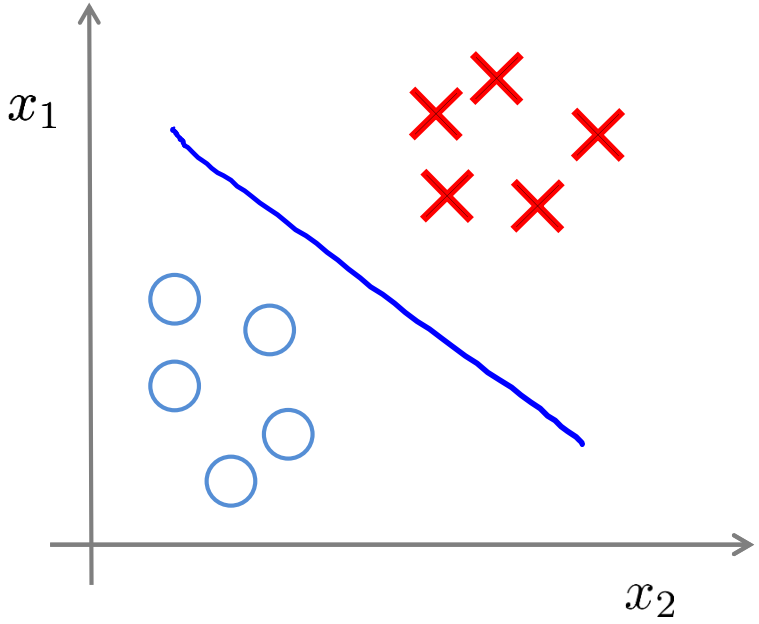


Machine Learning

Unsupervised Learning

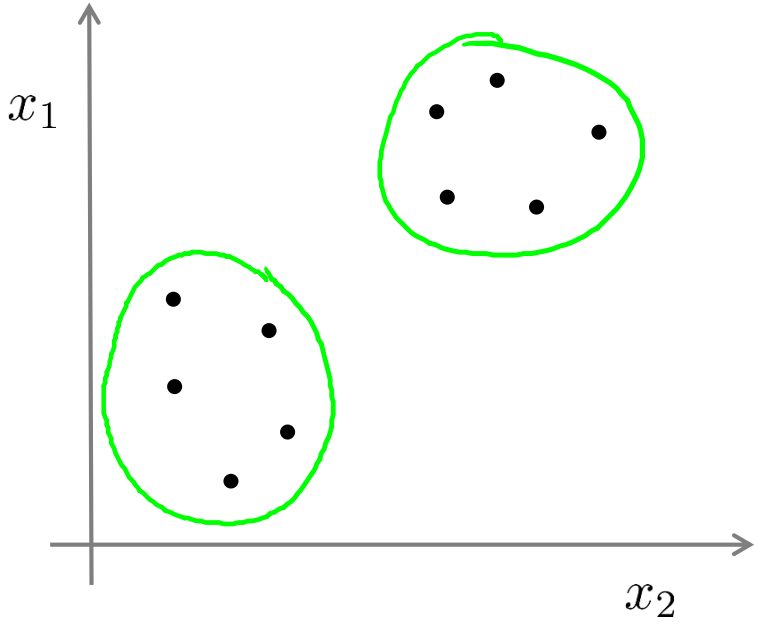
Introduction

Supervised learning



Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \ldots, (x^{(m)}, y^{(m)})\}$

Unsupervised learning



Clustering algorithm

Training set: $\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(m)}\}$

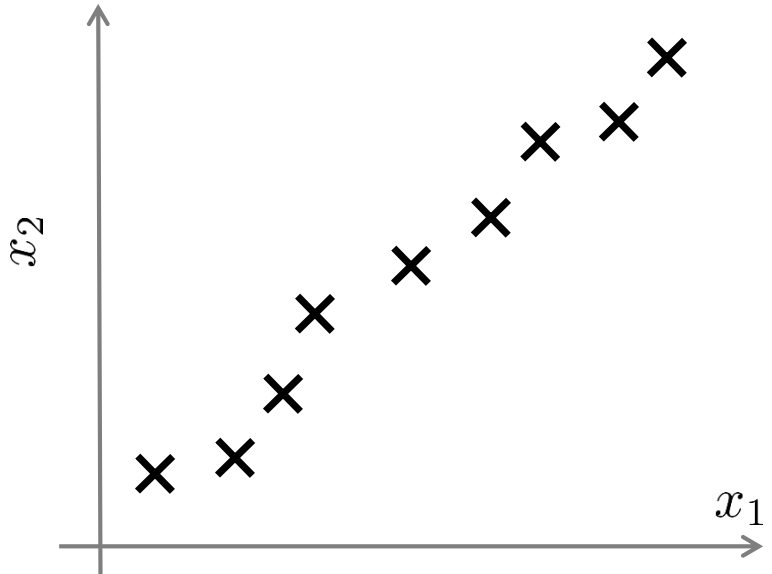
Unsupervised Learning

- Finds undetected patterns in a data set with no pre-existing labels and with a minimum of human supervision.
- Two of the main methods used in unsupervised learning are
  - principal component analysis (PCA), and
  - cluster analysis.

- Cluster Analysis
  - learning to group, or segment, datasets with shared attributes
- Principal Component Analysis
  - learning strategy is to learn a new feature space that captures the characteristics of the original space by maximizing some objective function or minimising some loss function.

Principal Component

- The principal components of a collection of points in a real p-space are a sequence of *p* direction vectors, where the *i*-th vector is the direction of a line that best fits the data while being orthogonal to the first (*i-1*) vectors.
  - Here, a best-fitting line is defined as one that minimizes the average squared distance from the points to the line.



Principal Component Analysis

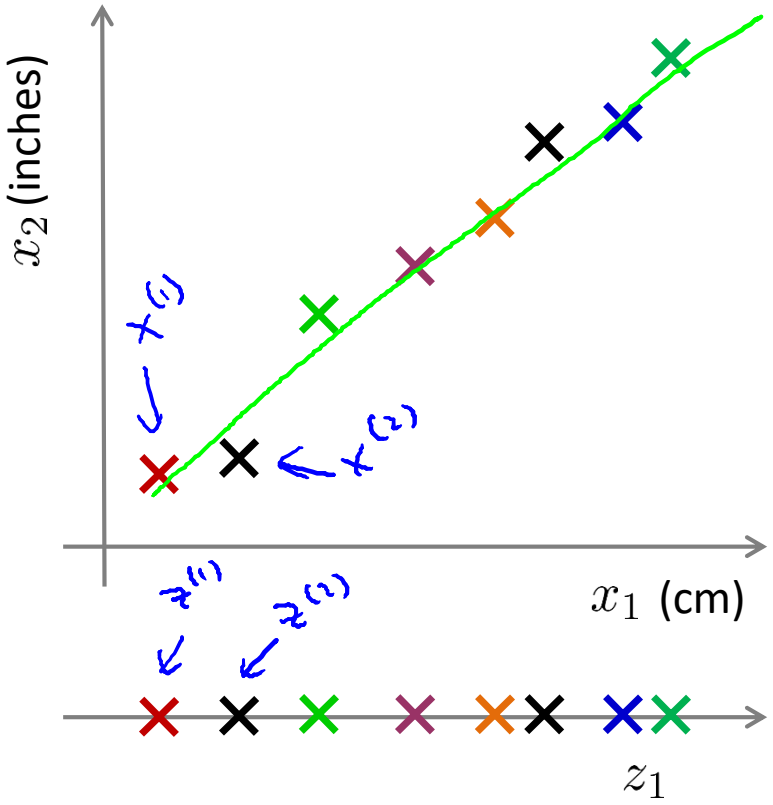
- **Principal component analysis** (**PCA**) is the process of computing the principal components and using them to perform a change of basis on the data,
  - Also known as Karhunen–Loève transform (KLT)
- Probably the most widely-used and well-known of the “standard” multivariate methods.
- Invented by Pearson (1901) and Hotelling (1933)
- Principal component analysis (PCA) is a way to reduce data dimensionality
- PCA projects the data in the least square sense– it captures big (principal) variability in the data and ignores small variability

Principal Component Analysis

- Principle
  - **Linear projection method** to reduce the number of variables
  - Transfer a set of correlated variables into a new set of uncorrelated variables
  - Map the data into a space of lower dimensionality
  - Form of unsupervised learning

- Properties
  - It can be viewed as a rotation of the existing axes to new positions in the space defined by original variables
  - New axes are orthogonal and represent the directions with maximum variability

Data Compression



Reduce data from 2D to 1D

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

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

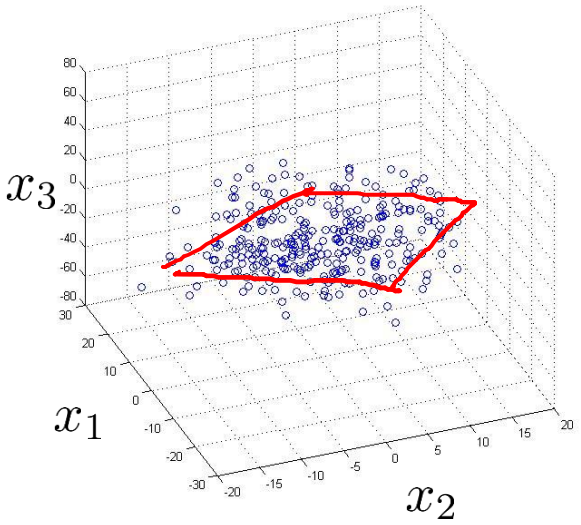
$$\vdots$$

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

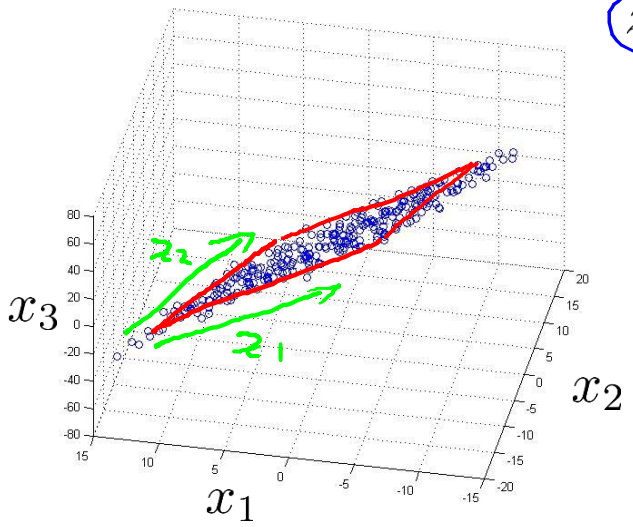
Data Compression

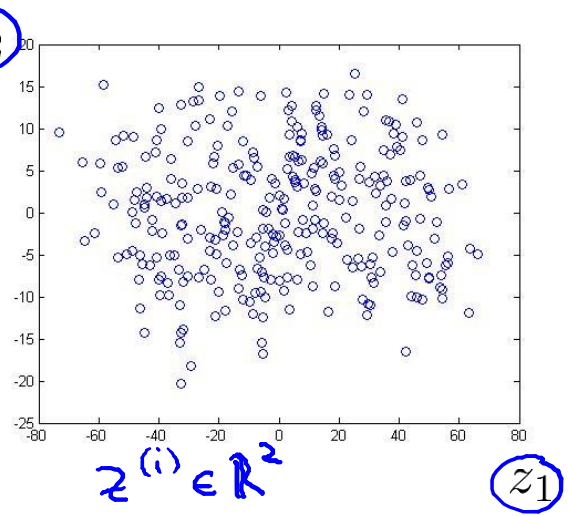
1000D → 100D

Reduce data from 3D to 2D



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



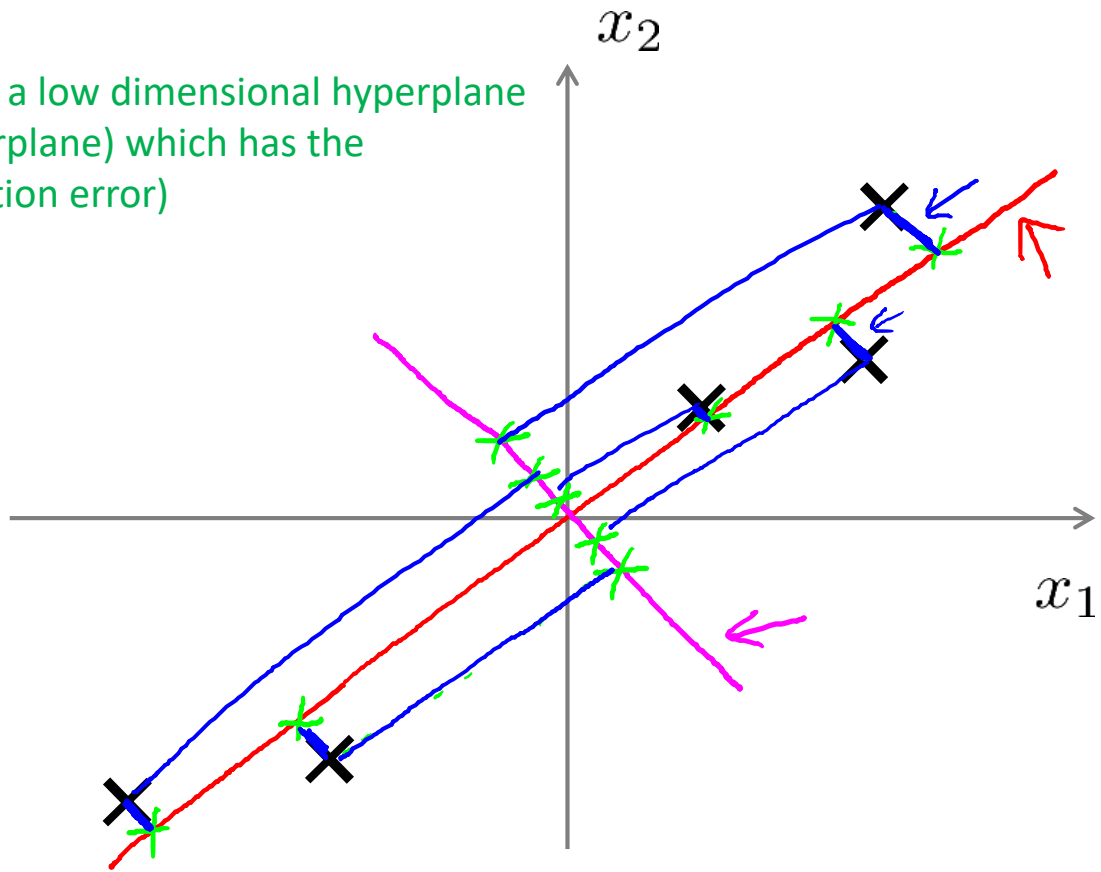


$z^{(i)} \in \mathbb{R}^2$

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

Principal Component Analysis (PCA) problem formulation

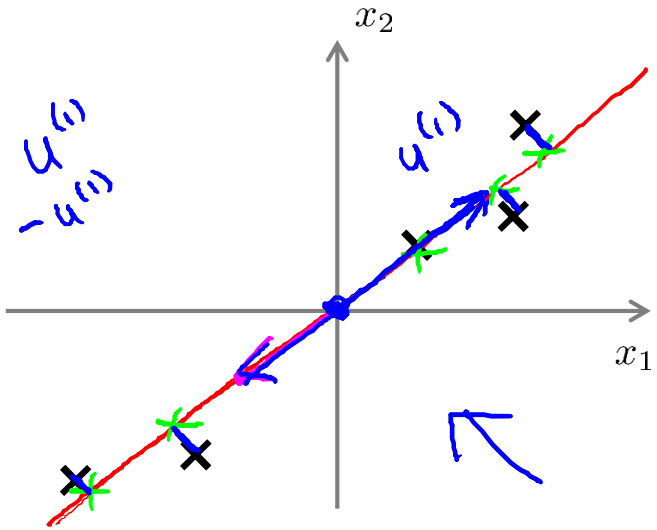
(PCA tries to find a low dimensional hyperplane (projection hyperplane) which has the minimum projection error)

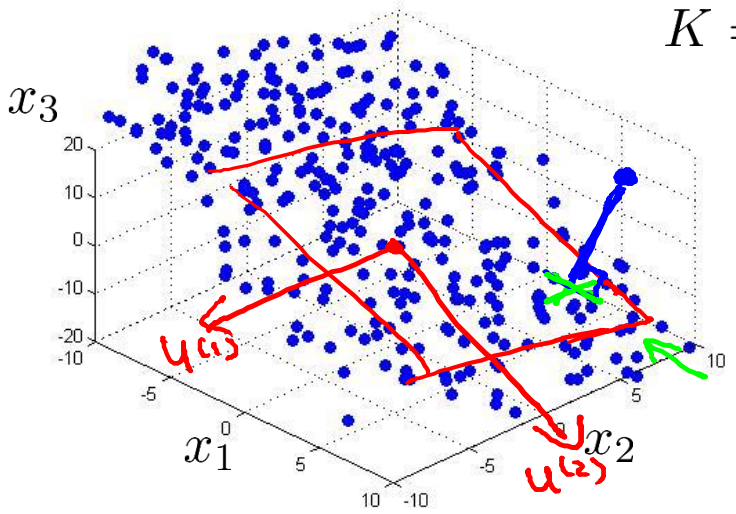


$x \in \mathbb{R}^2$

Principal Component Analysis (PCA) problem formulation

$3D \rightarrow 2D$
$K = 2$



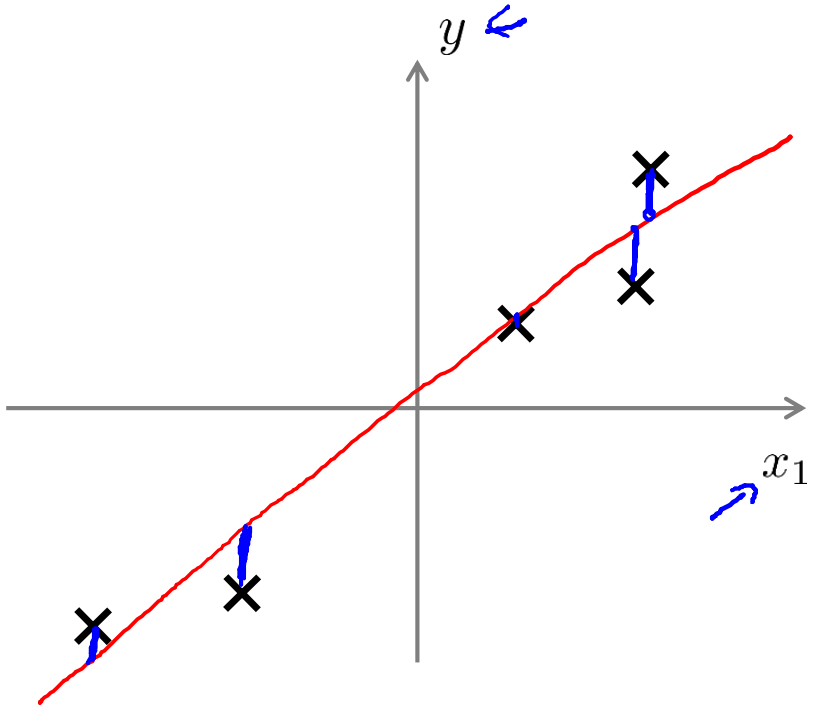


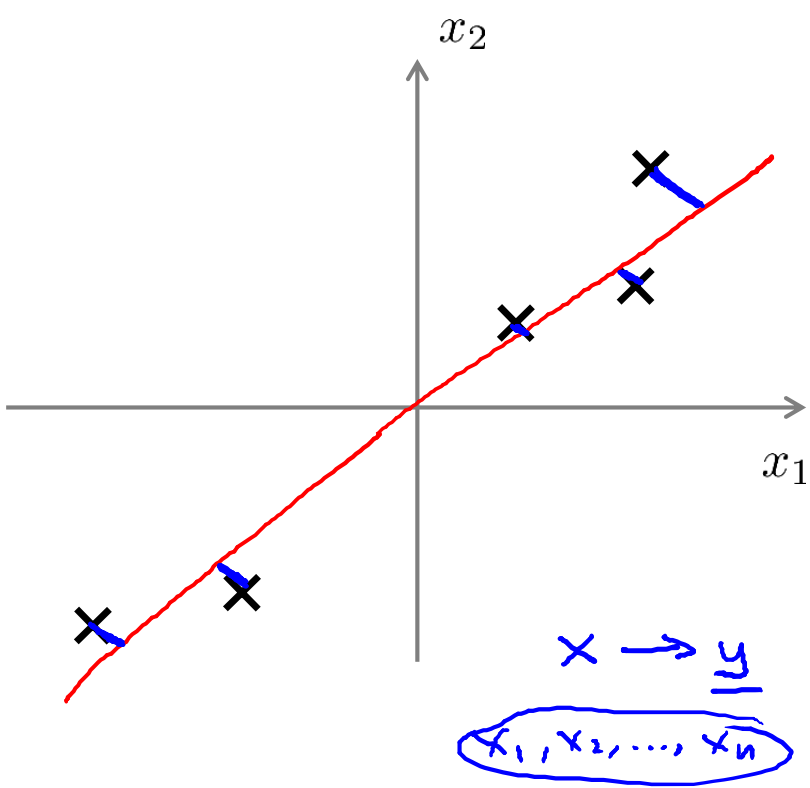
Reduce from 2-dimension to 1-dimension: Find a direction (a vector $u^{(1)} \in \mathbb{R}^n$) onto which to project the data so as to minimize the projection error.

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.

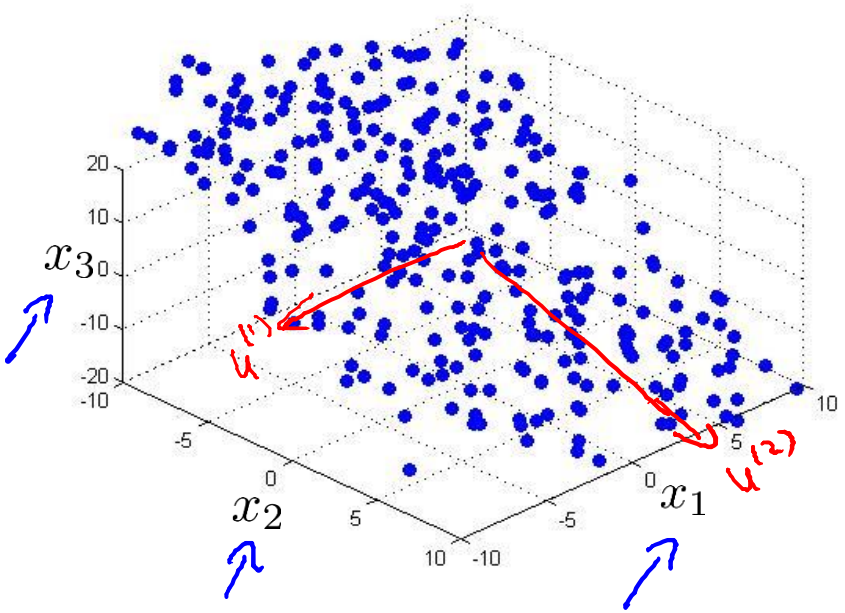
PCA is not linear regression

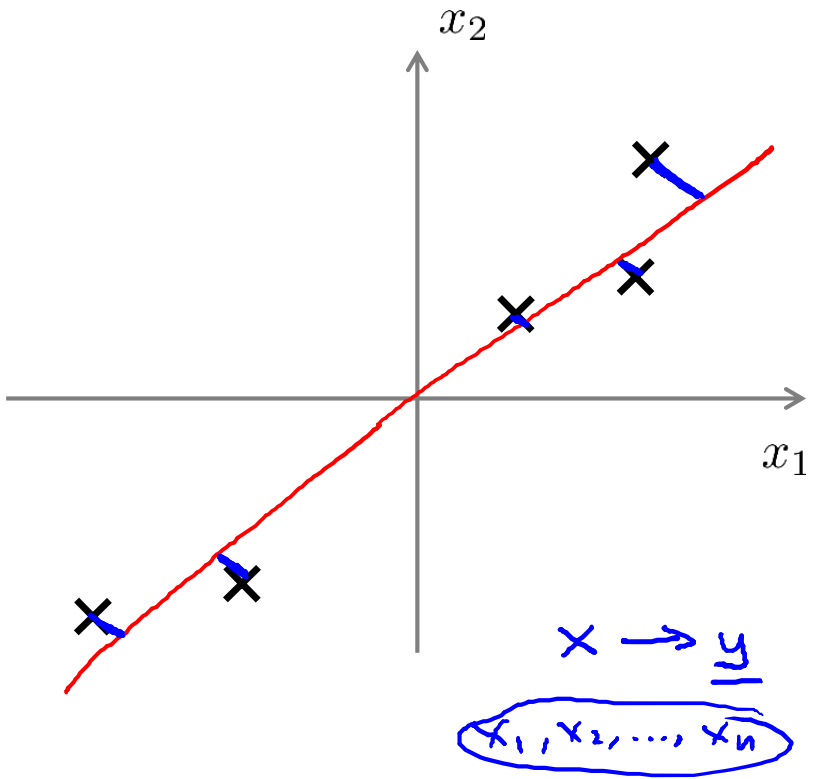
(PCA tries to minimize (squared) projection error not MSE)

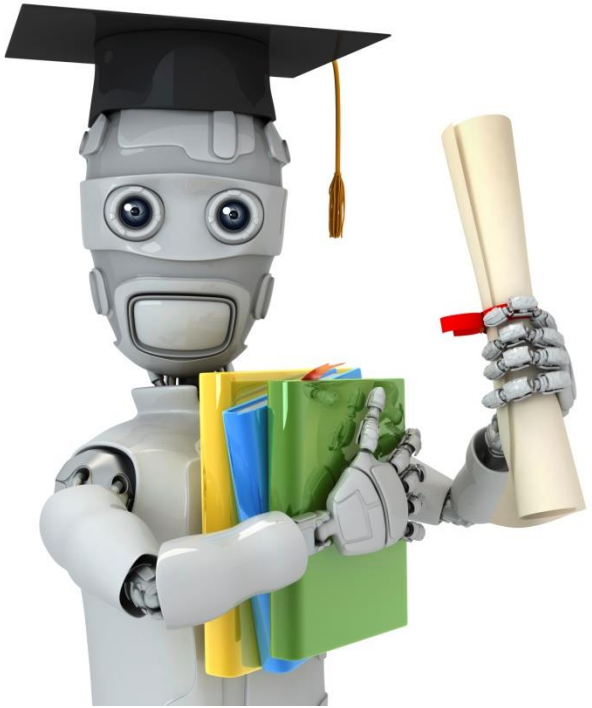




PCA is not linear regression







Machine Learning

Dimensionality Reduction

Principal Component Analysis algorithm

Steps

1. **Subtract the mean**
2. **Calculate the covariance matrix**
3. **Calculate the eigenvectors and eigenvalues**
4. **Choosing components and forming a feature vector**

1. Subtract Mean

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 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.

1. Subtract Mean

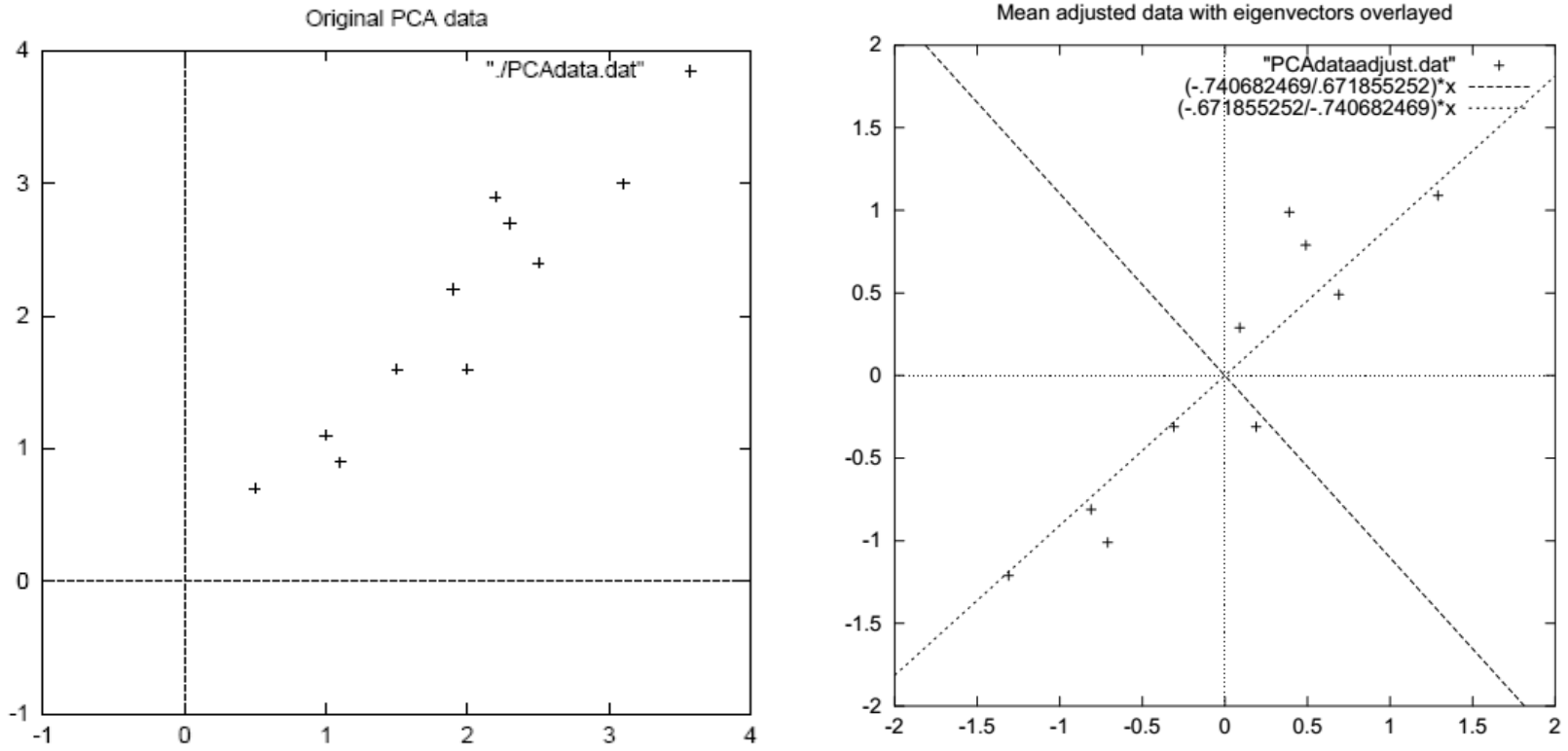


Figure 3.1: PCA example data, original data on the left, data with the means subtracted on the right, and a plot of the data

1. Subtract Mean

Data =

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9

DataAdjust =

x	y
.69	.49
-1.31	-1.21
.39	.99
.09	.29
1.29	1.09
.49	.79
.19	-.31
-.81	-.81
-.31	-.31
-.71	-1.01

Subtracting the mean makes variance and covariance calculation easier by simplifying their equations. The variance and co-variance values are not affected by the mean value.

2. Calculate the covariance matrix

- Compute “covariance matrix”:

$$\Sigma = \frac{1}{m}\sum_{i=1}^{n}(x^{(i)})(x^{(i)})^T \qquad cov = \begin{pmatrix} .616555556 & .615444444 \\ .615444444 & .716555556 \end{pmatrix}$$

  - Since the non-diagonal elements in this covariance matrix are positive, we should expect that both the x and y variable increase together.

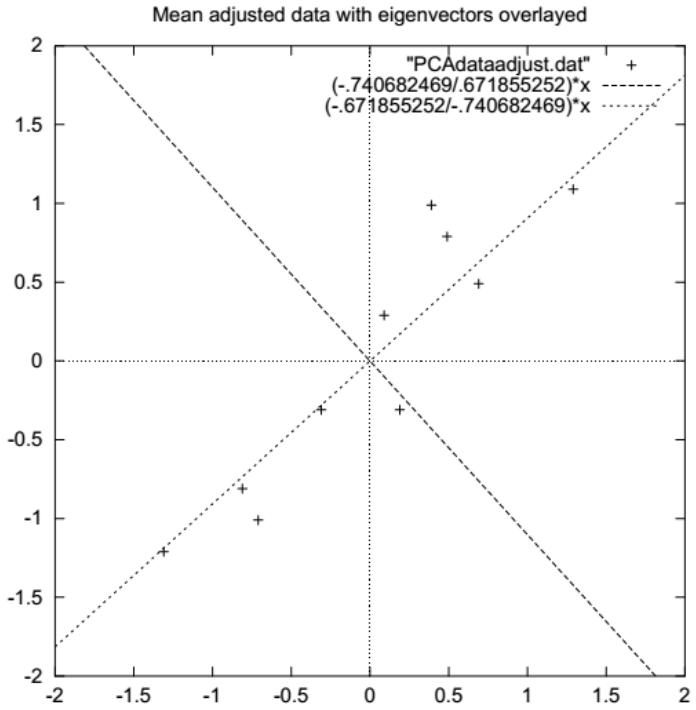
- Gives information about shape of data distribution.

3. Calculate the eigenvectors and eigenvalues

- Singular Value Decomposition to get the eigenvectors and eigenvalues:

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

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



4. Choosing components and forming a feature vector

- The eigenvector with the *highest* eigenvalue is the *principle component* of the data set.
- Once eigenvectors are found from the covariance matrix, the next step is to order them by eigenvalue, highest to lowest.
  - This gives you the components in order of significance.
- Form a new feature vector by projecting onto the selected eigenvectors.

PCA with all Eigenvectors

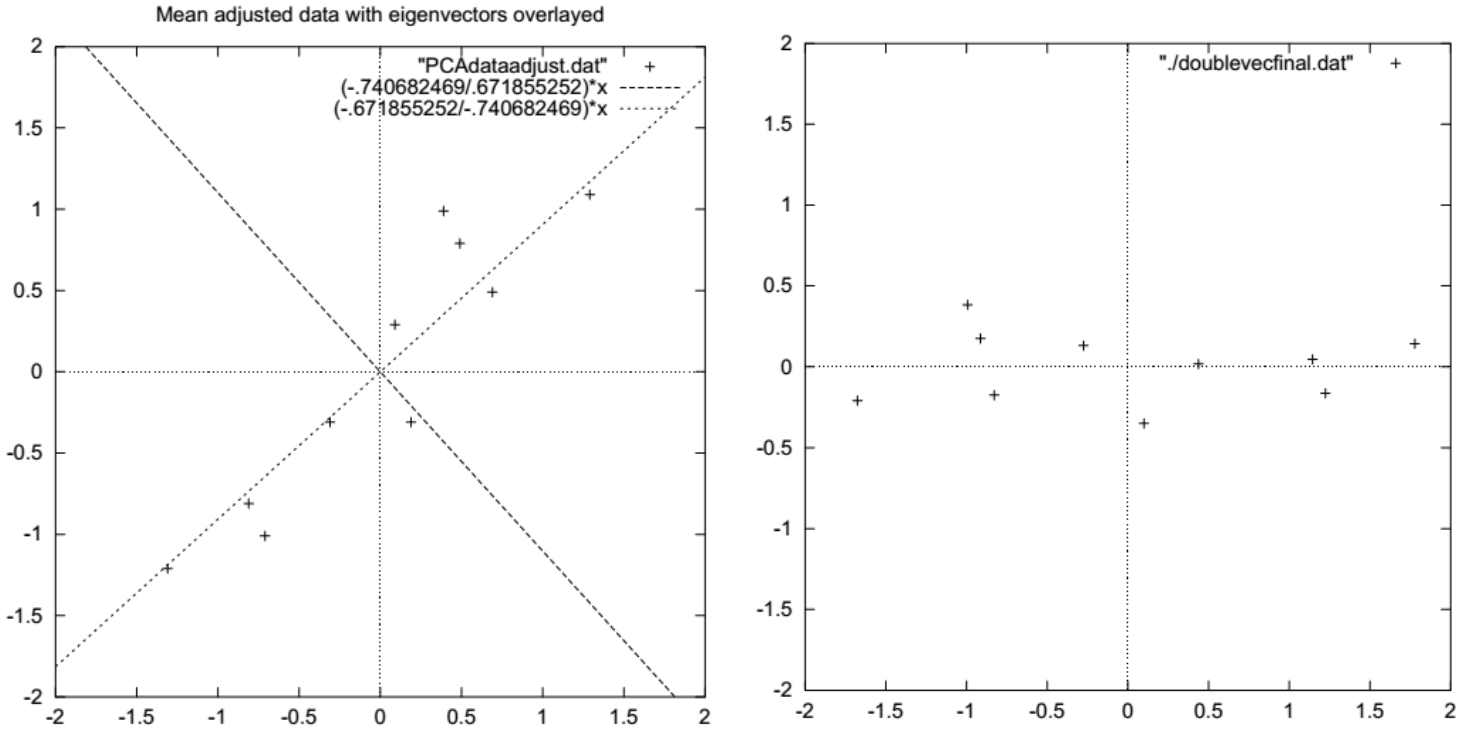
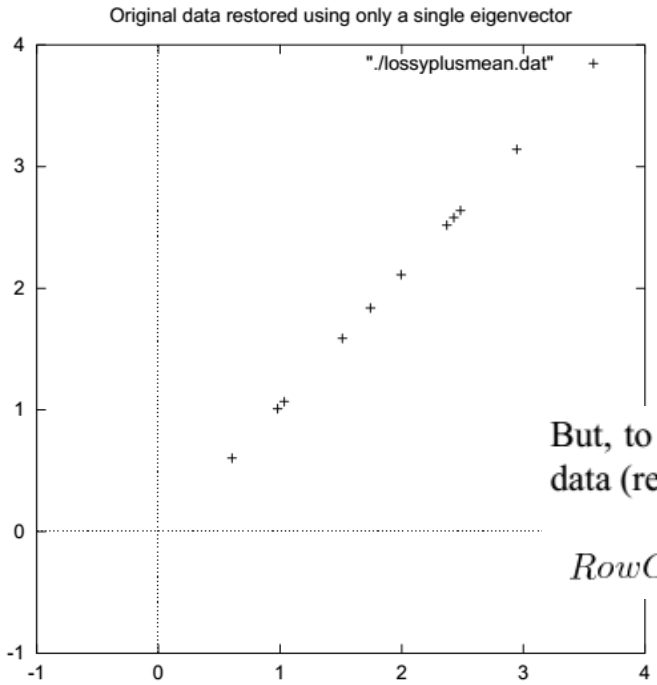


Figure 3.3: The table of data by applying the PCA analysis using both eigenvectors, and a plot of the new data points.

Data Reconstruction



Recall that the final transform is this:

$$FinalData = RowFeatureVector \times RowDataAdjust,$$

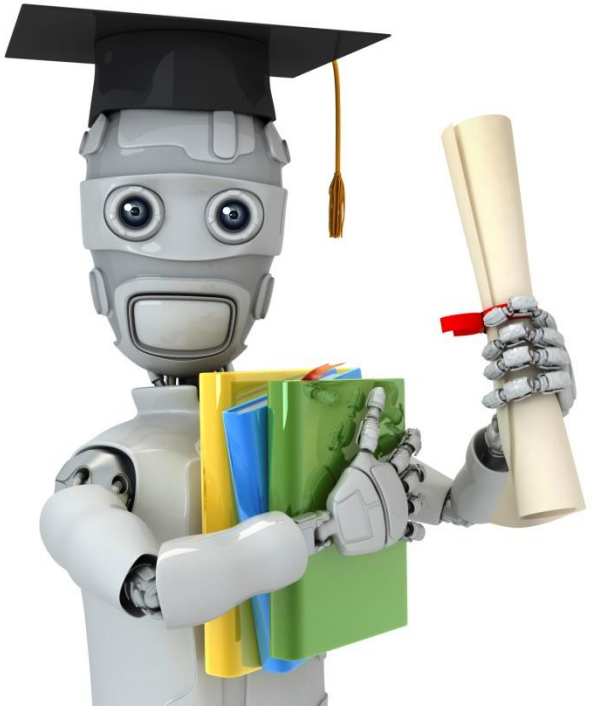
which can be turned around so that, to get the original data back,

$$RowDataAdjust = RowFeatureVector^{-1} \times FinalData$$

But, to get the actual original data back, we need to add on the mean of that original data (remember we subtracted it right at the start). So, for completeness,

$$RowOriginalData = (RowFeatureVector^{T} \times FinalData) + OriginalMean$$

Figure 3.5: The reconstruction from the data that was derived using only a single eige vector



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

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

0.05 0.10 5% (10%)

"99% of variance is retained"

95% 90%

Choosing k (number of principal components)

Algorithm:

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

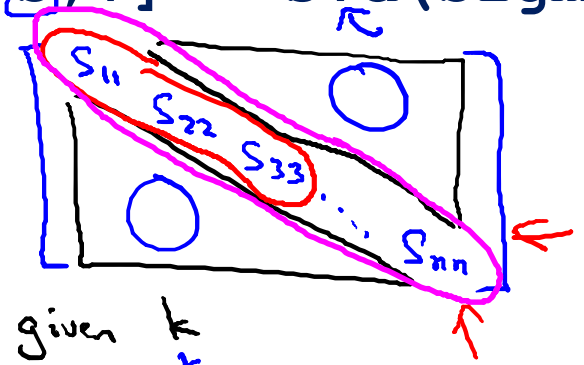
Compute $U_{reduce}, z^{(1)}, z^{(2)}, \ldots, z^{(m)}, x^{(1)}_{approx}, \ldots, x^{(m)}_{approx}$

Check if

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

$k = 17$

→ [U, S, V] = svd(Sigma)

→ S = 

$k=3$

For given k

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

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

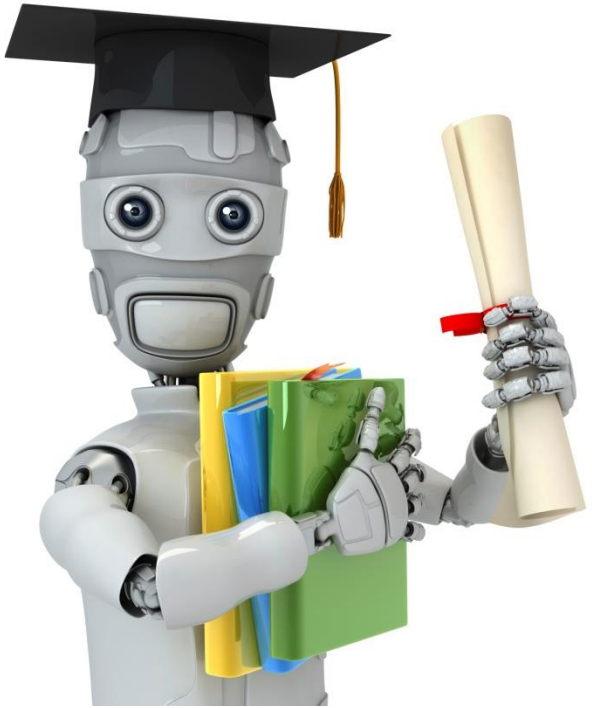
Choosing k (number of principal components)

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

Pick smallest value of k for which

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

(99% of variance retained)



Machine Learning

Dimensionality Reduction

Advice for applying PCA

Supervised learning speedup

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

Extract inputs:

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

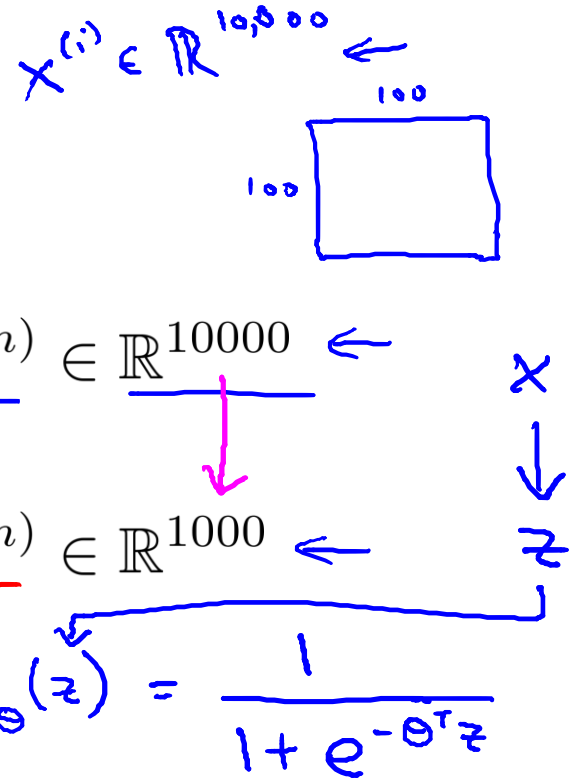
$\downarrow PCA$

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

New training set:

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

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 retain

- Visualization

$k = 2$ or $k = 3$

Bad use of PCA: To prevent overfitting

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

100 — 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.

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

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 $\{(z^{(1)}, y^{(1)}), \ldots, (z^{(m)}, y^{(m)})\}$
- Test on test set: Map $x^{(i)}_{test}$ to $z^{(i)}_{test}$. Run $h_\theta(z)$ on $\{(z^{(1)}_{test}, y^{(1)}_{test}), \ldots, (z^{(m)}_{test}, y^{(m)}_{test})\}$

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 $z^{(i)}$.