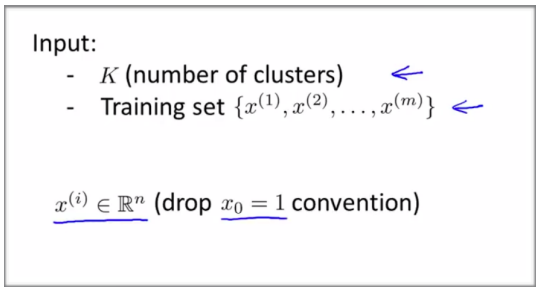
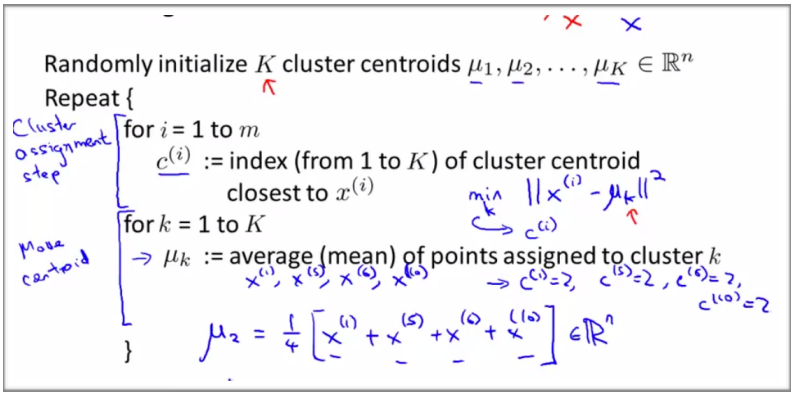
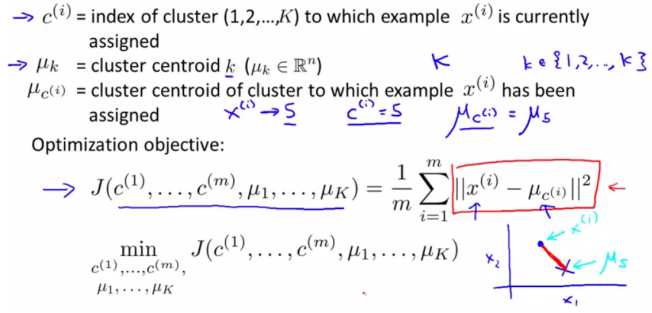
- **Unsupervised Learning Introduction**
- Supervised Learning
  - Find the decision boundary
- Unsupervised learning
  - no labels
  - give unlabeled training set to the algorithm and find some structure to the data
  - clustering
- **Clustering K-Means**
- Cluster centroids
  - the number of groups that are wanted
- Iterative process
  - compute the average of all the points and reassign the centroid centers
- K means



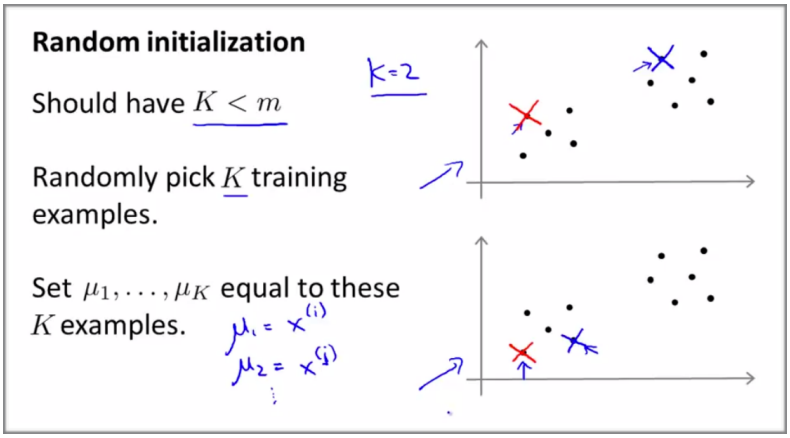
- K-Means algorithm



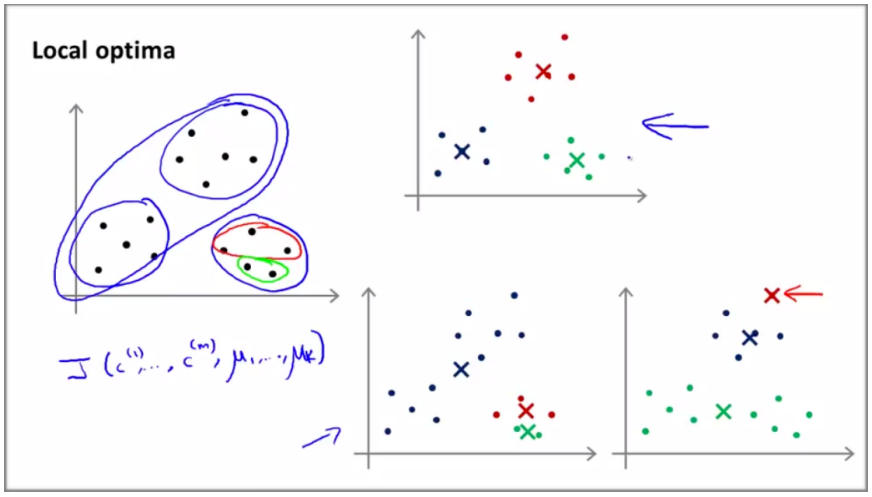
- **Optimization Objective or Cost Function**
- K-means optimization objective



- **Random initialization**
- Random initialization of cluster centers



- Local optima



  - To prevent this run k means multiple times
- Random initialization

For i = 1 to 100 { 50 - 1000

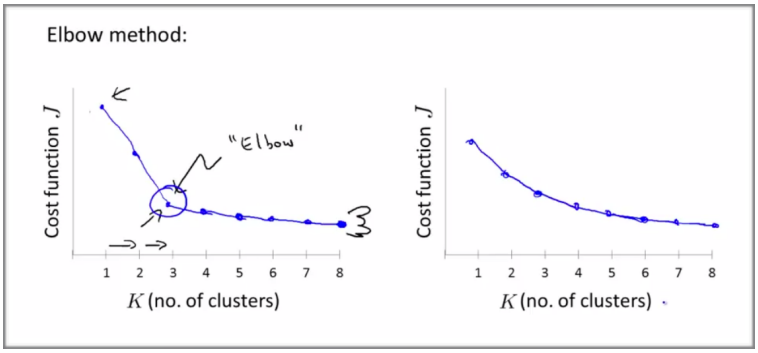
→ Randomly initialize K-means.
Run K-means. Get $c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K$.
Compute cost function (distortion)
→ $J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$
}

Pick clustering that gave lowest cost $J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$

K = 2 - 10

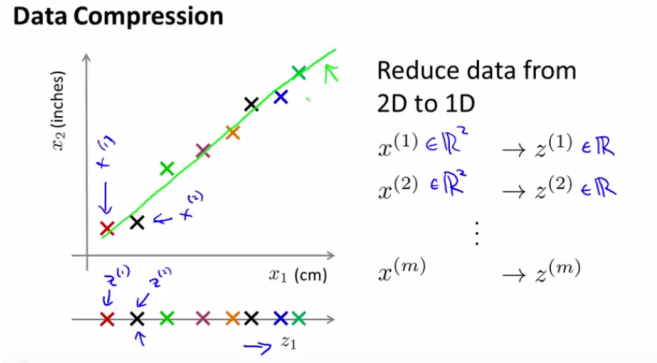
- If key is very large run multiple random initialization will make not that big of a difference

- **Choosing the number of clusters**
- Choosing the value of K
  - Elbow method

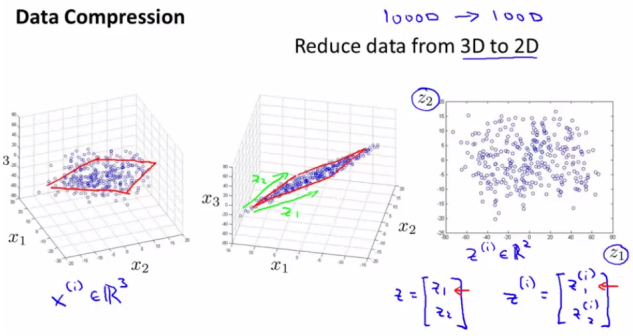


- Choosing the value of K
  - Sometimes, you're running K-means to get cluster to use for some later/downstream purpose. Evaluate K-means based on a metric for how well it performs for that later purpose.

- **Dimensionality Reduction**
- Motivation 1: Data compression

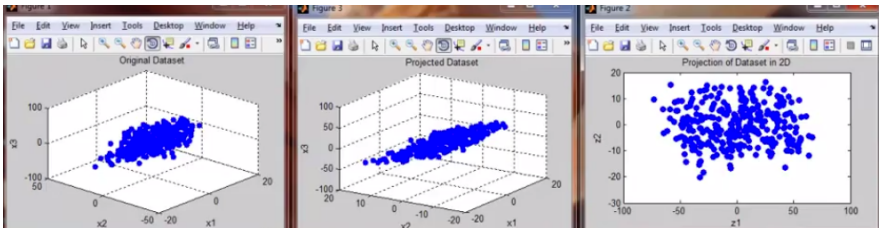


- Data compression Reduce data from 3d to 2d



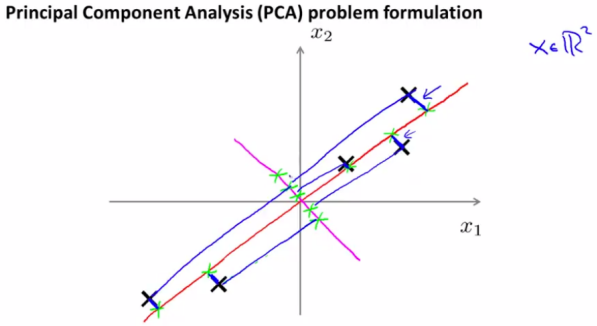
  - Left is a 3d point cloud
  - Middle is a projection of the data onto a 2d surface or plane

- Right now only need to numbers to represent the data on the plane

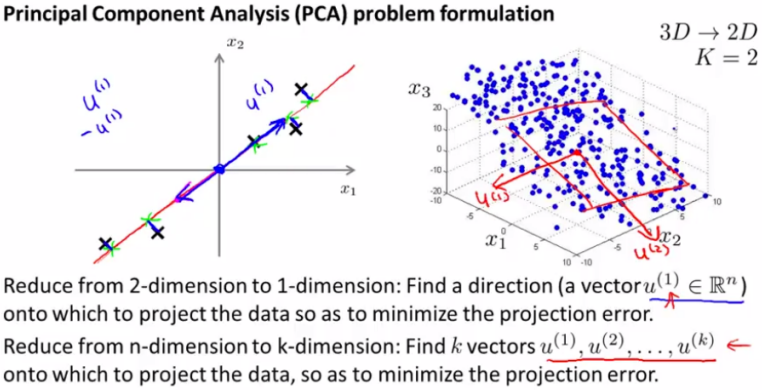


- Process to reduce the data from 3 dimensional to 2 dimensional

- **Motivation 2: Data Visualization**
- **Principal Component Analysis**
- Principal Component
- Analysis (PCA) problem formulation
  - tries to find a lower dimensional surface or a line in this case such that the sum of squares of the blue line segments is minimized. This is called the projection error.
    - should perform mean normalization and feature scaling before perform PCA

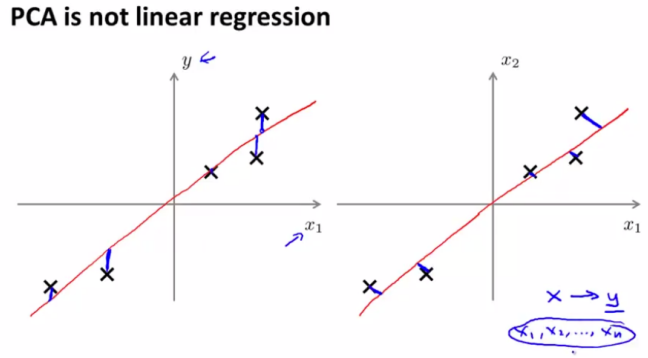


- Principal Component Analysis (PCA) problem formulation

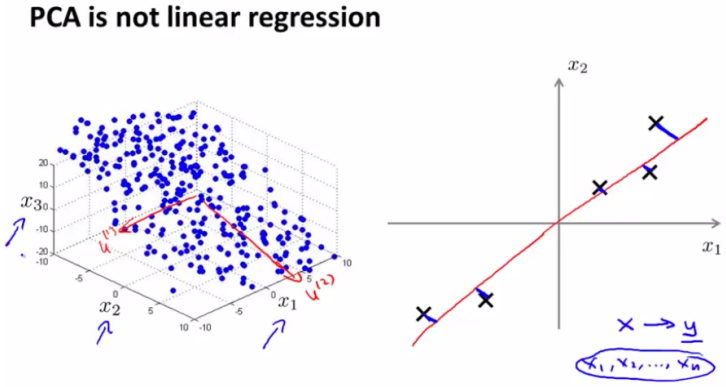


- PCA is not Linear regression
  - Linear regression we are minimizing the square magnitude of the blue lines
    - the blue lines are the value of the point and the value predicted by the hypothesis
  - Linear regression is taking the values of x and trying to predict y

- PCA tries to minimize the magnitude of the blue lines which are drawn at an angle which are the shortest orthogonal or shortest distance between the point x and the red line
- PCA list of features where all the features are treated equally

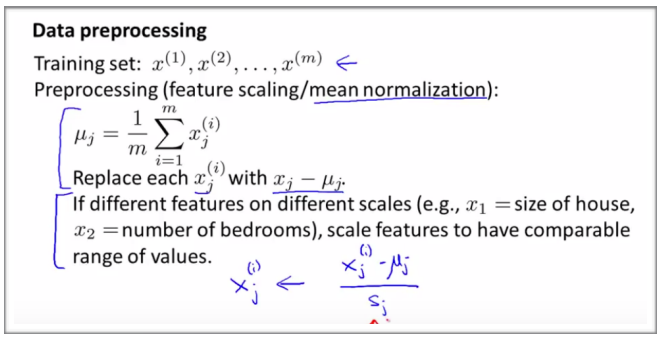


- 3 dimensional data and want to reduce to 2 dimensional



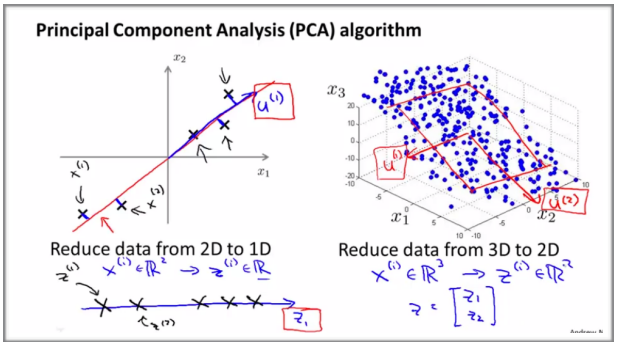
- PCA trying to find a lower dimensional surface to project the data

- **Principal Component Analysis Algorithm**
- Data preprocessing

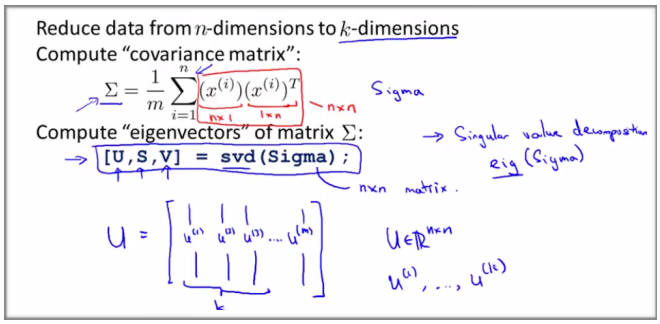


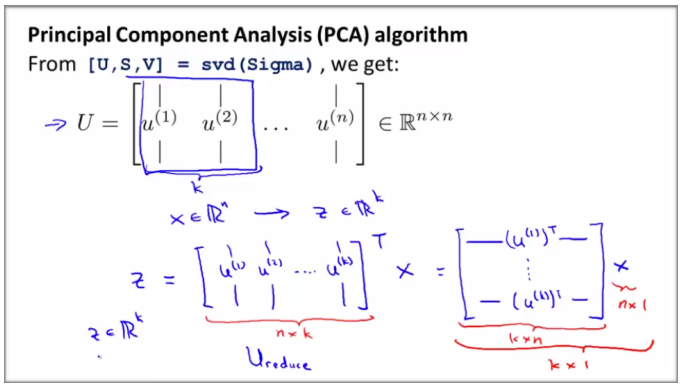
- sj can be the max - min value or more common the std

- Principal Component Analysis (PCA) algorithm

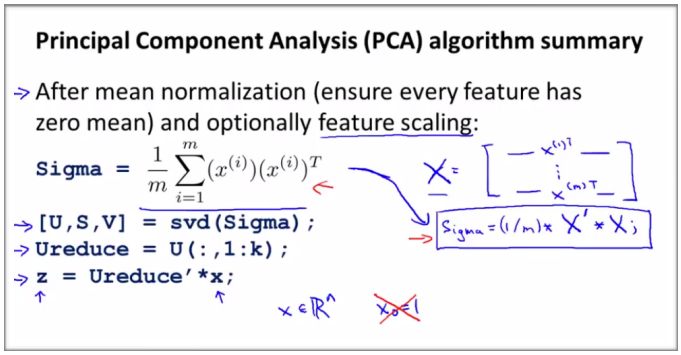


- Principal Component Analysis (PCA) algorithm
  - reduce data from n-dimensions to k-dimensions
  - Compute “covariance matrix”:
  - SVD singular value decomposition

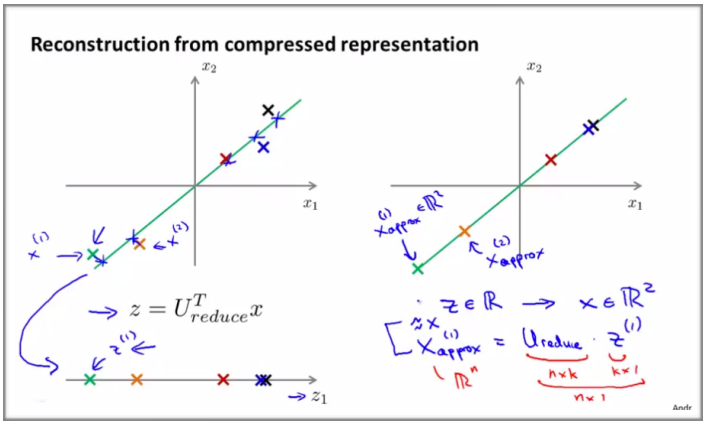




  - Principal Component Analysis (PCA) algorithm summary



- **Applying PCA**
- **Reconstruction from Compressed Representation**
- Reconstruction from compressed representation



- **Choosing the Number of Principal Components**
- Choosing k (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

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

"99% of variance is retained"

- Choosing K (number of principal components)

Choosing k (number of principal components)

Algorithm:

Try PCA with $k = 1$

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

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

$k = 17$

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

$S = \begin{bmatrix} S_{11} & & & \\ & S_{22} & & \\ & & S_{33} & \\ & & & \ddots \\ & & & & S_{nn} \end{bmatrix}$

For given k

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

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

- allows to select the value of K without having to run PCA over and over
- Choosing k (number of principal components)

Choosing k (number of principal components)

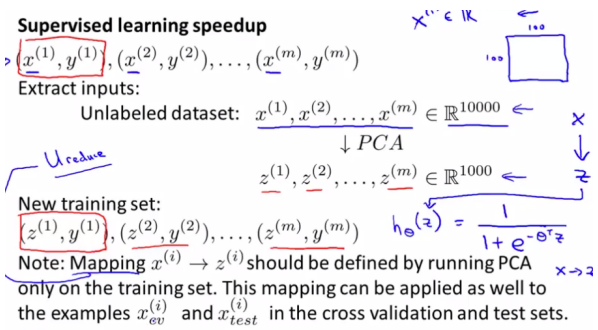
[U,S,V] = svd(Sigma)

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)

- **Advice for applying PCA**
- Supervised Learning speedup



- Application of PCA
  - Compression
    - Reduce memory/disk needed to store data
    - Speed up learning algorithm
    - Choose k by percentage of variance retained
  - Visualization
    - k = 2 or k = 3
- Bad use of PCA: To prevent overfitting

Bad use of PCA: To prevent overfitting

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

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 reduces the dimension of the data without knowing what the value of y is, but this will throw away some of the valuable information

- PCA is sometimes used where it shouldn't be

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 $z^{(i)}$.