Machine Learning Part IV

Unsupervised Learning
K-means, PCA

K-means Algorithm

Randomly initialize K cluster centroids $\underline{\mu_1}, \underline{\mu_2}, \ldots, \underline{\mu_K} \in \mathbb{R}^n$

Repeat {

Cluster assignment step

for i = 1 to m

$\underline{c^{(i)}}$:= index (from 1 to K) of cluster centroid closest to $x^{(i)}$

$\min_k \|x^{(i)} - \mu_k\|^2 \rightarrow c^{(i)}$

Move centroid

for k = 1 to K

$\rightarrow \mu_k$:= average (mean) of points assigned to cluster k

$x^{(1)}, x^{(5)}, x^{(6)}, x^{(10)} \rightarrow c^{(1)} = 2, \; c^{(5)} = 2, \; c^{(6)} = 2, \; c^{(10)} = 2$

$$\mu_2 = \frac{1}{4}\left[x^{(1)} + x^{(5)} + x^{(6)} + x^{(10)}\right] \in \mathbb{R}^n$$

}

Optimization Objective

$c^{(i)}$ = index of cluster (1,2,…,K) to which example $x^{(i)}$ is currently assigned

μ_k = cluster centroid $\underline{k}$ ($\mu_k \in \mathbb{R}^n$) K $k \in \{1,2,\ldots,K\}$

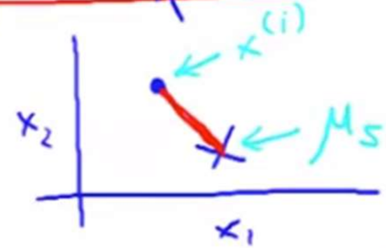
$\mu_{c^{(i)}}$ = cluster centroid of cluster to which example $x^{(i)}$ has been assigned $x^{(i)} \rightarrow \underline{5}$ $\underline{c^{(i)} = 5}$ $\mu_{\underline{c^{(i)}}} = \mu_5$

Optimization objective:

$$\underline{J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)} = \frac{1}{m} \sum_{i=1}^{m} \boxed{||x^{(i)} - \mu_{c^{(i)}}||^2}$$

$$\min_{\substack{c^{(1)}, \ldots, c^{(m)}, \\ \mu_1, \ldots, \mu_K}} J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$$

Distortion



Optimization Objective

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$

Cluster assignment step

Minimize $J(\ldots)$ wrt $\boxed{c^{(1)}, c^{(2)}, \ldots, c^{(m)}}$ ←

(holding $\mu_1, \ldots, \mu_K$ fixed)

Repeat {

for i = 1 to m

$c^{(i)}$:= index (from 1 to K) of cluster centroid closest to $x^{(i)}$

move centroid

for k = 1 to K

μ_k := average (mean) of points assigned to cluster k

}

minimize $J(\ldots)$ wrt $\boxed{\mu_1, \ldots, \mu_K}$

Random Initialization

Random initialization

$K=2$

Should have $K < m$

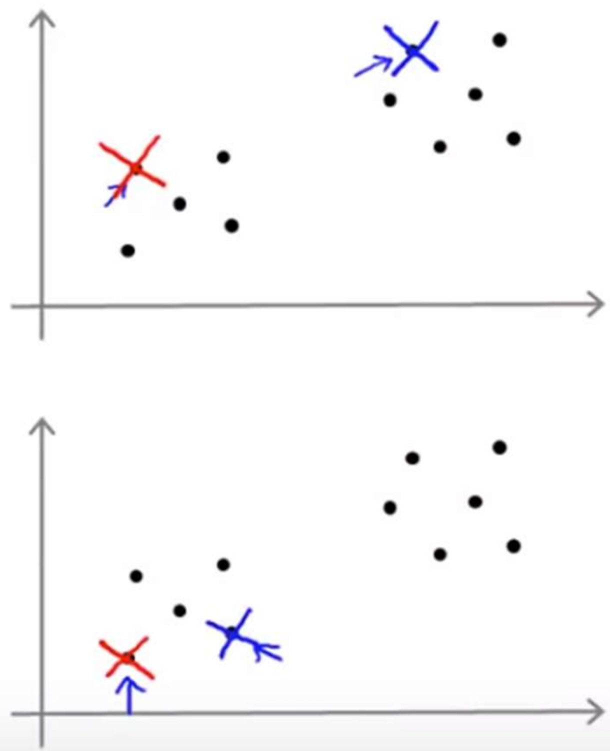
Randomly pick K training examples.

Set $\mu_1, \ldots, \mu_K$ equal to these K examples.

$\mu_1 = x^{(i)}$

$\mu_2 = x^{(j)}$

$\vdots$



Multiple Random Initialization

For i = 1 to 100 {   50 - 1000

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

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

K=2-10

Choosing number of clusters

Suppose you run k-means using k = 3 and k = 5. You find that the cost function J is much higher for k = 5 than for k = 3. What can you conclude?

- [ ] This is mathematically impossible. There must be a bug in the code.
- [ ] The correct number of clusters is k = 3.
- [x] In the run with k = 5, k-means got stuck in a bad local minimum. You should try re-running k-means with multiple random initializations.

Correct

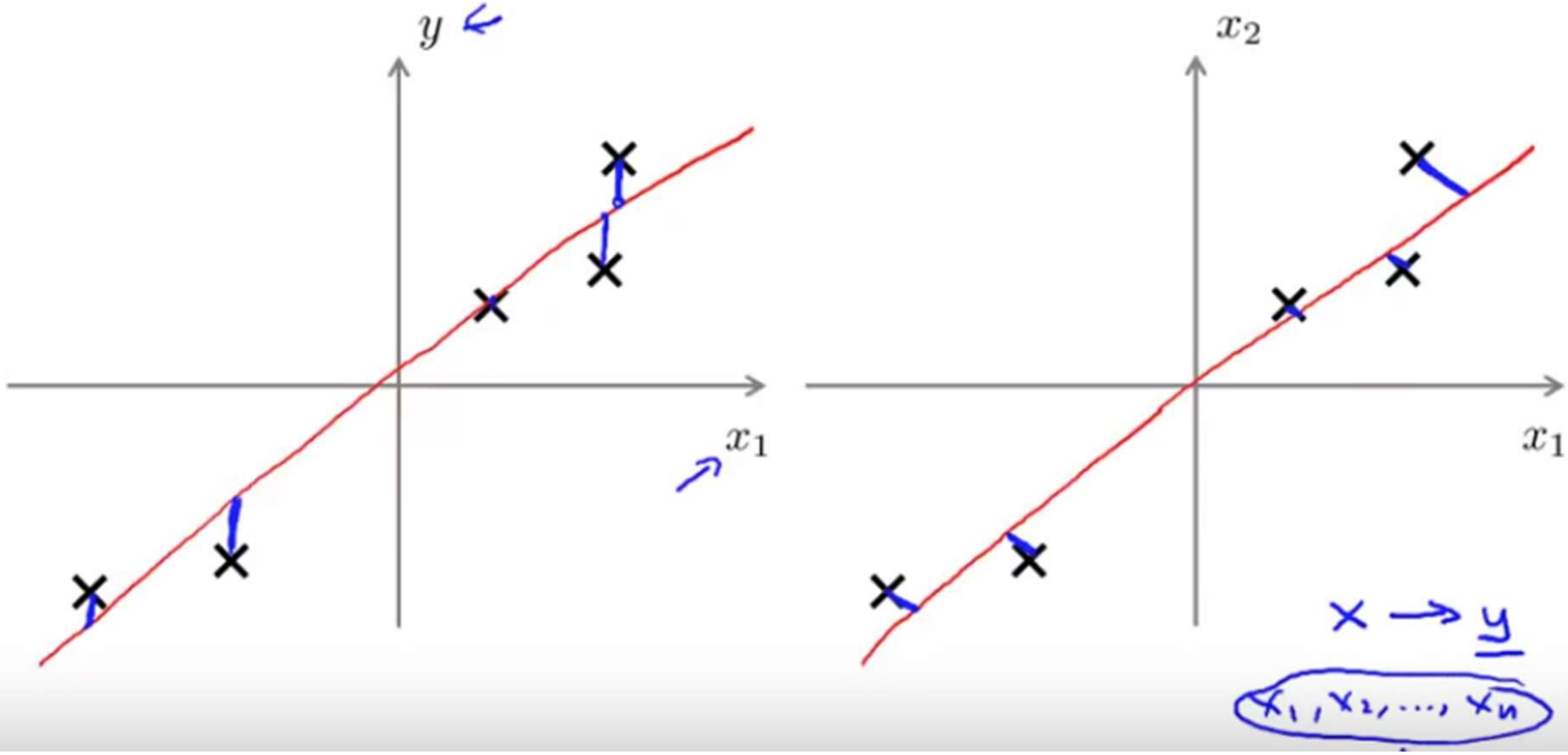
- [ ] In the run with k = 3, k-means got lucky. You should try re-running k-means with k = 3 and different random initializations until it performs no better than with k = 5.

Application

- Customer segmentation
- Topic group
- Image compression (as in ex7)

Principal Component Analysis

PCA is not linear regression



Principal Component Analysis (PCA) algorithm

Reduce data from n-dimensions to k-dimensions

Compute "covariance matrix":

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

Compute "eigenvectors" of matrix Σ:

$$\boxed{\texttt{[U,S,V] = svd(Sigma);}}$$

$n\times n$ matrix.

→ Singular value decomposition

eig(Sigma)

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

k

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

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

$Z = X * U[:, 1:K]$

X is m by n,
U is n by k,
Z is m by k

Principal Component Analysis (PCA) algorithm

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

$$U = \Bigg[\underbrace{\boxed{\begin{matrix} | & | \\ u^{(1)} & u^{(2)} \\ | & | \end{matrix}}}_{k} \; \begin{matrix} & | \\ \ldots & u^{(n)} \\ & | \end{matrix} \Bigg] \in \mathbb{R}^{n \times n}$$

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

$$z^{(i)} = \underbrace{\begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \ldots & u^{(k)} \\ | & | & & | \end{bmatrix}}_{\substack{n \times k \\ U_{reduce}}}{}^{T} x^{(i)} = \underbrace{\underbrace{\begin{bmatrix} \text{—} (u^{(1)})^T \text{—} \\ \vdots \\ \text{—} (u^{(k)})^T \text{—} \end{bmatrix}}_{k \times n} \underbrace{x^{(i)}}_{n \times 1}}_{k \times 1}$$

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

Principal Component Analysis (PCA) algorithm summary

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

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

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

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

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

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

In PCA, we obtain $z \in \mathbb{R}^k$ from $x \in \mathbb{R}^n$ as follows:

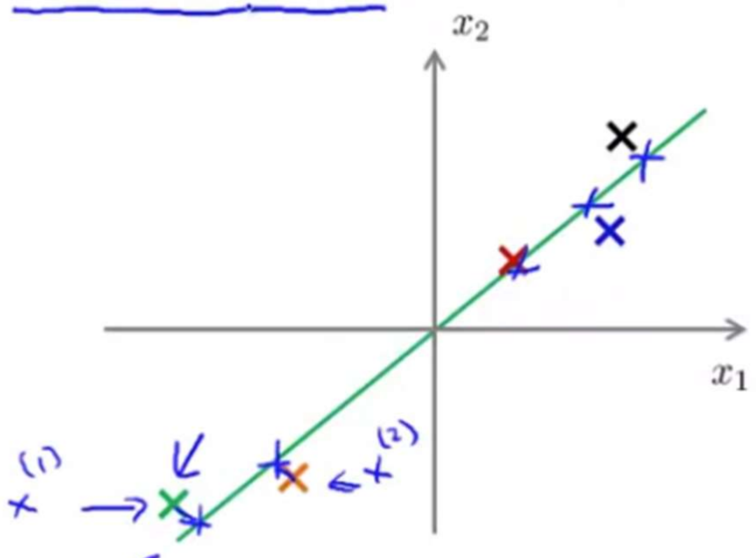
$$z = \begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \dots & u^{(k)} \\ | & | & & | \end{bmatrix}^T x = \begin{bmatrix} \text{---} & (u^{(1)})^T & \text{---} \\ \text{---} & (u^{(2)})^T & \text{---} \\ & \vdots & \\ \text{---} & (u^{(k)})^T & \text{---} \end{bmatrix} x$$

Which of the following is a correct expression for z_j?

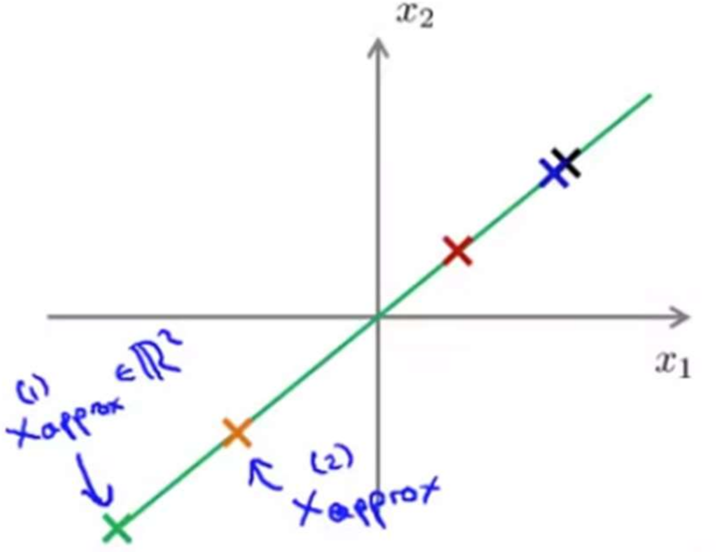
- ○ $z_j = (u^{(k)})^T x$
- ○ $z_j = (u^{(j)})^T x_j$
- ○ $z_j = (u^{(j)})^T x_k$
- ◉ $z_j = (u^{(j)})^T x$

Correct

Reconstruction from compressed representation



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



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

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

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

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

"99% of variance is retained"

Choosing k (number of principal components)

Algorithm:

Try PCA with $\cancel{k=1}$ $\cancel{k=2}$ $\cancel{k=3}$ $k=4$ ...

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

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)
```

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

$k=3$

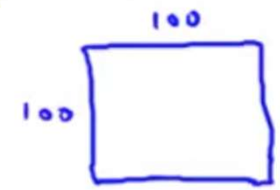
For given k

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

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

Supervised learning speedup

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



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

Extract inputs:

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

$\downarrow PCA$

U_{reduce}

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

$x \downarrow z$

New training set:

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

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

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.

$x \rightarrow z$