PCA

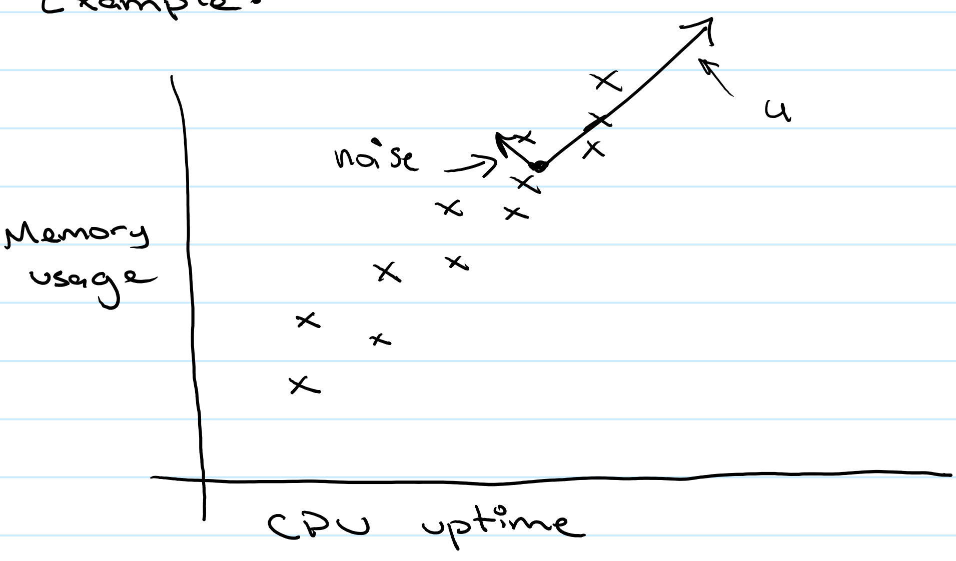
Sunday, December 13, 2015 11:05 AM

Let's say youre given a training set

$\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$, where $x^{(i)} \in \mathbb{R}^n$.

Now say that some of the features are linearly dependent and the data lies on a k-dimensional subspace. The PCA problem is to find the k-dimensional subspace, such that $k < n$ (often $k \ll n$)

Example:



The above data can be reduced into the subspace represented by u, which might capture the feature "usage intensity"

Before applying PCA, we need to do

some pre-processing

Pre-processing:

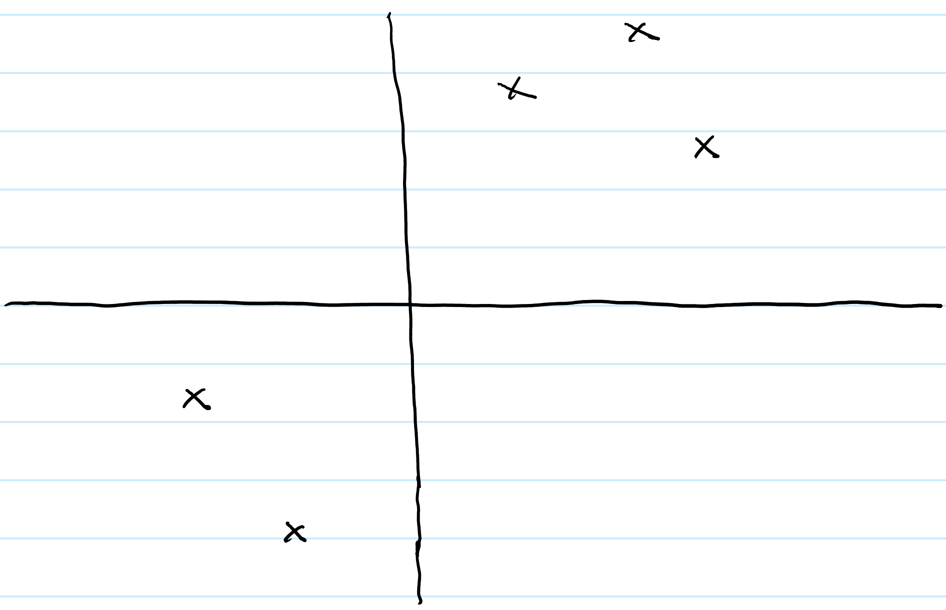
1. Compute $\mu = \frac{1}{m} \sum_{i=1}^{m} x^{(i)}$
2. $x^{(i)} \leftarrow x^{(i)} - \mu$

} zero out mean

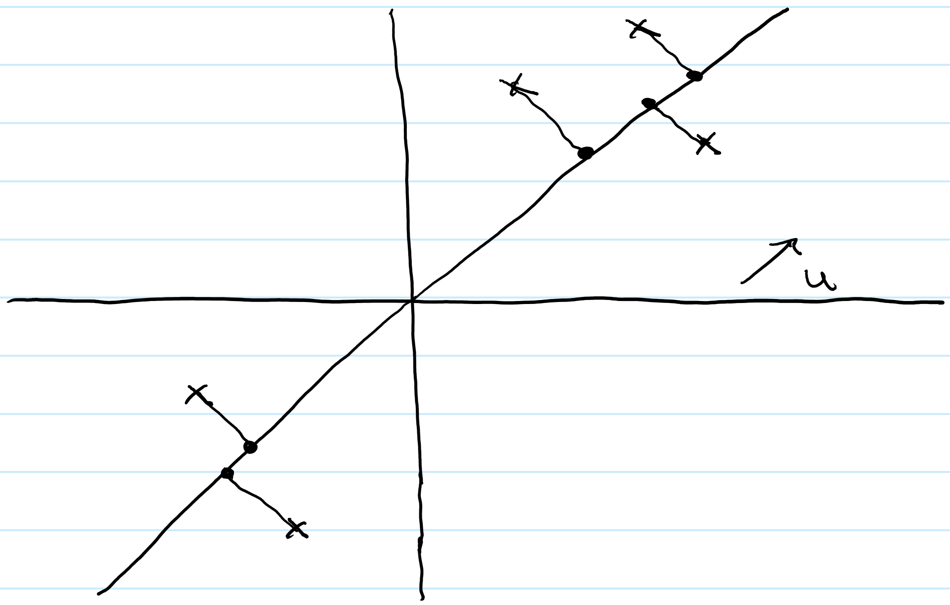
3. Compute $\sigma_j^2 = \frac{1}{m} \sum_{i=1}^{m} (x_j^{(i)})^2$
4. $x_j^{(i)} \leftarrow \frac{x_j^{(i)}}{\sigma_j}$

} normalize to unit variance

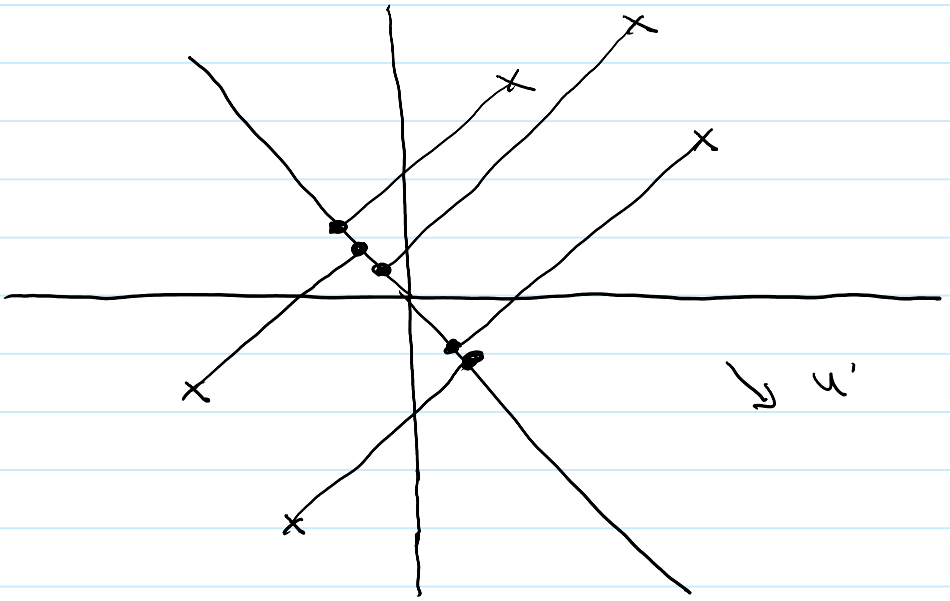
Now consider this example



In order to reduce this, we would want to find a subspace like this:



And not a subspace like this:



If you notice the difference between the two, you can see for the optimal one that the variance of the projections is high. For the sub-optimal line, the variance of the projections is quite low.

To formalize this notion, we can define

the PCA problem as:

$$\max_u \frac{1}{m}\sum_{i=1}^{m}\left(x^{(i)T}u\right)^2$$, where $x^{(i)T}u$ is the length of the projection of $x^{(i)}$ onto u.

s.t. $\|u\| = 1$

$$\frac{1}{m}\sum_{i=1}^{m}\left(x^{(i)T}u\right)^2 = \frac{1}{m}\sum_{i=1}^{m}\left(u^T x^{(i)}\right)\left(x^{(i)T}u\right)$$

$$= u^T\left[\frac{1}{m}\sum_{i=1}^{m}x^{(i)}\left(x^{(i)T}\right)\right]u = u^T\Sigma u$$

where Σ is the covariance matrix.

So the PCA problem can be rewritten as

$$\max_u u^T\Sigma u$$, where $\Sigma = \frac{1}{m}\sum_{i=1}^{m}x^{(i)}\left(x^{(i)T}\right)$

s.t. $u^T u = 1$

To solve this optimization problem, we construct the Lagrangian

$$L(u,\lambda) = u^T\Sigma u - \lambda(u^T u - 1)$$

$$\nabla_u L(u,\lambda) = 2\Sigma u - 2\lambda u$$

$$2\Sigma u - 2\lambda u = 0 \Rightarrow \boxed{\Sigma u = \lambda u}$$

$\therefore$ The solutions to this problem are the principal eigenvectors of Σ, the empirical covariance matrix.

To form a k-dimensional subspace, choose the top k principal eigenvectors, that is the eigenvectors with the k largest eigenvalues.

A Faster Algorithm

computing Σ can be quite expensive, especially for large values of m and n.

Instead of computing Σ and then finding the eigenvectors, another approach is to use the singular-value decomposition, also known as the SVD.

SVD

Any $m \times n$ matrix M can be factored as

$$M = U \Sigma V^T$$

where U is a $m \times m$ orthogonal matrix
Σ is a $m \times n$ diagonal matrix
V^T is a $n \times n$ orthogonal matrix

Σ contains non-negative real numbers on the diagonal.

The SVD is related to the eigendecomposition as well:

$$\begin{aligned} M^T M &= (U \Sigma V^T)^T (U \Sigma V^T) \\ &= (V \Sigma^T U^T)(U \Sigma V^T) \\ &= V \Sigma U^T U \Sigma V^T \\ &= V \Sigma I \Sigma V^T \\ &= V (\Sigma \Sigma) V^T \end{aligned}$$

Which is the eigendecomposition since

$$AX = X\Lambda \quad , \quad A \text{ is } n \times n \text{ matrix}$$
and X's columns are eigenvectors

$$\Rightarrow A = X \Lambda X^{-1}$$

Since V is orthogonal, $V^{-1} = V^T$ and since

$$\Sigma \Sigma = \begin{bmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_n \end{bmatrix} \begin{bmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_n \end{bmatrix} = \begin{bmatrix} \sigma_1^2 & & 0 \\ & \ddots & \\ 0 & & \sigma_n^2 \end{bmatrix}$$

This means V's columns are the eigenvectors of $M^T M$.

Now coming back to PCA, the covariance matrix Σ can be computed as:

$$\frac{1}{m} X^T X = \frac{1}{m} \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots & x^{(m)} \\ | & | & & | \end{bmatrix} \begin{bmatrix} \text{---} & x^{(1)T} & \text{---} \\ \text{---} & x^{(2)T} & \text{---} \\ & \vdots & \\ \text{---} & x^{(m)T} & \text{---} \end{bmatrix}$$

$$= \frac{1}{\;} \begin{bmatrix} \text{---} & x_1^{(i)} & \text{---} \end{bmatrix} \begin{bmatrix} | & | & & | \end{bmatrix}$$

$$\begin{bmatrix} — & x^{(m)} & — \end{bmatrix}$$

$$= \frac{1}{m} \begin{bmatrix} — & x_1^{(i)} & — \\ — & x_2^{(i)} & — \\ & \vdots & \\ — & x_n^{(i)} & — \end{bmatrix} \begin{bmatrix} | & | & & | \\ x_1^{(i)} & x_2^{(2)} & \cdots & x_n^{(i)} \\ | & | & & | \end{bmatrix}$$

$$= \frac{1}{m} \begin{bmatrix} \sum_{i=1}^{m} \left(x_1^{(i)}\right)^2 & \sum_{i=1}^{m} x_1^{(i)} x_2^{(i)} & \cdots & \sum_{i=1}^{m} x_1^{(i)} x_n^{(i)} \\ \sum_{i=1}^{m} x_2^{(i)} x_1^{(i)} & \cdots\cdots & & \sum_{i=1}^{m} x_2^{(i)} x_n^{(i)} \\ \vdots & & & \\ \sum_{i=1}^{m} x_n^{(i)} x_1^{(i)} & \cdots\cdots & & \sum_{i=1}^{m} x_n^{(i)} x_n^{(i)} \end{bmatrix}$$

$$= \frac{1}{m} \sum_{i=1}^{m} \begin{bmatrix} x_1^{(i)} x_1^{(i)} & x_1^{(i)} x_2^{(i)} & \cdots & x_1^{(i)} x_n^{(i)} \\ x_2^{(i)} x_1^{(i)} & x_2^{(i)} x_2^{(i)} & \cdots & x_2^{(i)} x_n^{(i)} \\ \vdots & \vdots & \ddots & \\ x_n^{(i)} x_1^{(i)} & x_n^{(i)} x_2^{(i)} & \cdots & x_n^{(i)} x_n^{(i)} \end{bmatrix}$$

$$= \frac{1}{m} \sum_{i=1}^{m} \begin{bmatrix} x_1^{(i)} \\ x_2^{(i)} \\ \vdots \\ x_n^{(i)} \end{bmatrix} \begin{bmatrix} x_1^{(i)} & x_2^{(i)} & \cdots & x_n^{(i)} \end{bmatrix}$$

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

So in order to apply PCA, we don't need to compute Σ. Instead, we can take the singular value decomposition of X:

$$X = U \Sigma V^T$$

and use the top k singular values in Σ with the first k rows in V^T (or equivalently the first k columns in V) to form the basis of the subspace, since the columns of V are eigenvectors of $\Sigma = \frac{1}{m} X^T X$.

Final Algorithm

1. Pre-process data:

   a) Compute $\mu = \frac{1}{m} \sum_{i=1}^{m} x^{(i)}$

   b) Set $x^{(i)} \leftarrow x^{(i)} - \mu$

   c) Compute $\sigma_j^2 = \frac{1}{m} \sum_{i=1}^{m} (x_j^{(i)})^2$

   d) Set $x_j^{(i)} \leftarrow \dfrac{x_j^{(i)}}{\sigma_j}$

2. Compute $X = U \Sigma V^T$ using SVD

3. Pick k principal eigenvectors, now compute:

$$\hat{X} = X V_{n, 1:k}$$

where $\hat{X}$ is the new representation of the data