Unsupervised Learning: K-Means and Principal Component Analysis (PCA)

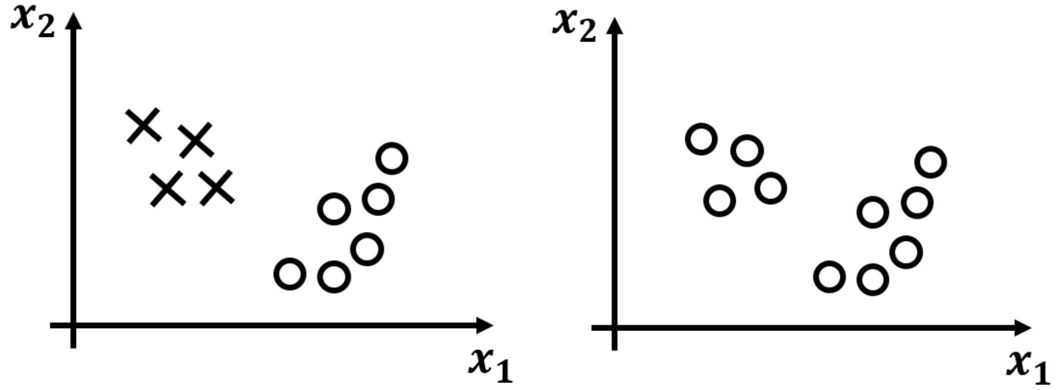
Jae Yun JUN KIM[*]

March 2, 2020

Different from supervised-learning problems, for unsupervised learning case, the input data are not labeled a priori, and it is the job of unsupervised learning to find the labels (solutions) for the data. Hence, the problem of the unsupervised learning consists of finding structures of the input data and assign labels (solutions) to these data. In this lecture, we are going to see two basic unsupervised algorithms: K-means and Principal Component Analysis (PCA).

1 K-means algorithm

1.1 Motivation



(a) Supervised learning (b) Unsupervised learning

Figure 1: Comparison between supervised and unsupervised learning

One can show the convergence of the K-means algorithm, by defining the *distortion function* as

$$E(k,\mu) = \sum_{i=1}^{I} \|x^{(i)} - \mu_k^{(i)}\|^2 \tag{1}$$

Then you can show that the K-means is the *coordinate ascent* on E. That is, you need to hold k and optimize for μ and hold μ and optimize for k. Then, one can see that the value decreases monotonically over E until reaching the optimal point. On the other hand, there is a way

[*]ECE Paris Graduate School of Engineering, 37 quai de Grenelle 75015 Paris, France; jae-yun.jun-kim@ece.fr

Algorithm 1: K-means

```
1 Input: $\{x^{(1)}, x^{(2)}, \cdots, x^{(I)}\} \in \mathbb{R}^{N\times I}$ and $K$;
2 Initialize cluster centroids: $\{\mu_1, \cdots, \mu_K\} \in \mathbb{R}^{N\times K}$ ;
3 while until convergence do
4     $y^{(i)} \leftarrow \arg\min_k \|x^{(i)} - \mu_k\|_2$;
5     $\mu_k \leftarrow \frac{\sum_{i=1}^{I} \mathbb{I}\{y^{(i)} = k\}x^{(i)}}{\sum_{i=1}^{I} \mathbb{I}\{y^{(i)} = k\}}$;
6 return $\{\mu_1, \cdots, \mu_K\}$;
```

to choose automatically the number of clusters, but it is better that we choose it manually. Further, $E(k, \mu)$ is not convex in general. Hence, there could be multiple local optima. To deal with this problem, we need to choose multiple initial conditions to find the best one.

2 Principal Component Analysis

2.1 Motivation

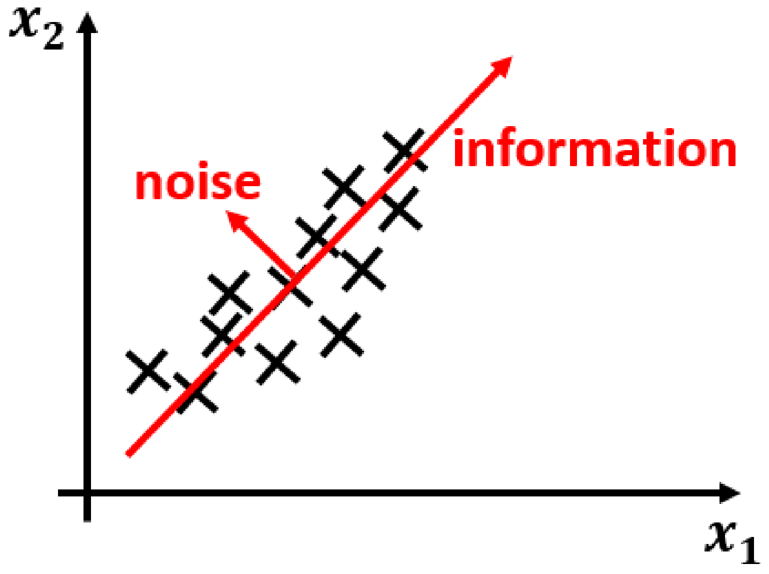


Figure 2: Principal component analysis

2.2 Intuition

Given $\{x^{(1)}, \cdots, x^{(I)}\}$ where $x^{(i)} \in \mathbb{R}^N$, reduce it to P-dimensional data, where $P < N$.
The first four lines of the PCA algorithm correspond to the data pre-processing procedures: The first two lines are for zeroing out the mean and the latter two lines are for normalizing the variance. These steps are critical when the scales of the variables are different.

Algorithm 2: Principal component analysis

```
1 Set $\mu \leftarrow \frac{1}{I}\sum_{i=1}^{I} x^{(i)}$;
2 Compute $\tilde{x}^{(i)}$ by $x^{(i)} - \mu$;
3 Compute $\Sigma \leftarrow \frac{1}{I}\sum_{i=1}^{I} \tilde{x}^{(i)}\tilde{x}^{(i)^T}$;
4 Find the eigenvectors corresponding to the $P$-largest eigenvalues of $\Sigma$: $\{u_1, \cdots, u_P\}$;
5 Project the pre-processed data ($\widetilde{X}$) onto $U$: $\widetilde{Y}$;
6 Post-process $\widetilde{Y}$ to get $Y$;
7 return $Y$;
```

How to find the main axes along which the data vary? That is, how to find the principal axes of the data variation?

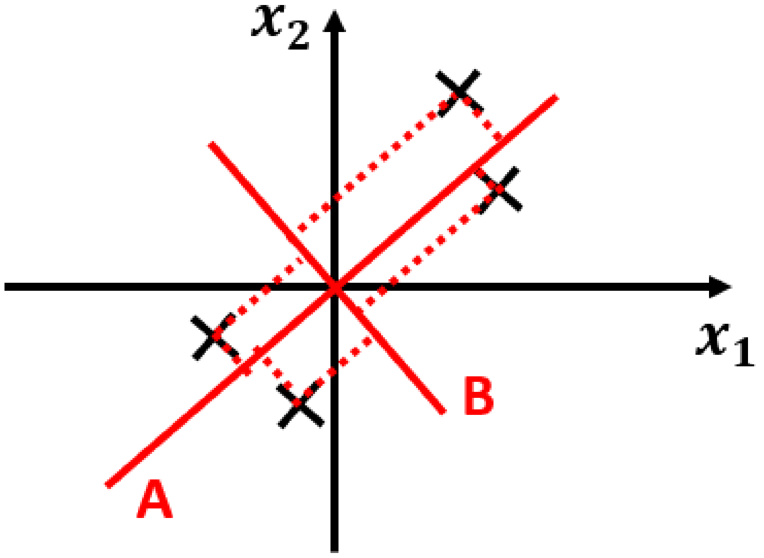


Figure 3: Searching for principal components

Suppose that I consider the hyperplane A and project each training example onto this hyperplane. Let us also consider the hyperplane B and project the training examples onto this hyperplane. Then I observe that the variance of the projected points onto the hyperplane is smaller than that of the hyperplane A. One way to formalize the notion of finding the main axes of the data variation is to find the vector (direction) u so that when the training examples are projected on that direction, the projected points vary as much as possible.
If $\|u\|$=1, then the vector $\tilde{x}^{(i)}$ projected on u has length $\left(\tilde{x}^{(i)}\right)^T u$.
Then, our problem can be mathematically expressed as

$$\begin{aligned} \max_u \quad & \frac{1}{I}\sum_{i=1}^{I}\left(\tilde{x}^{(i)^T} u\right)^2 \\ \text{s.t.} \quad & \|u\| = 1. \end{aligned} \tag{2}$$

On the other hand, one can develop the above objective function as

$$\frac{1}{I}\sum_{i=1}^{I}\left(\tilde{x}^{(i)^T}u\right)^2 = \frac{1}{I}\sum_{i=1}^{I} u^T\tilde{x}^{(i)}\tilde{x}^{(i)^T}u = u^T\left[\frac{1}{I}\sum_{i=1}^{I}\tilde{x}^{(i)}\tilde{x}^{(i)^T}\right]u = u^T\Sigma u. \tag{3}$$

In summary, for a given training data set, one can find the principal axes of the variation of data by constructing first the covariance matrix (Σ) and then by finding the principal eigenvectors of Σ, which gives the best hyperplane onto which the data is projected. More generally, to find the P-dimensional hyperplane, choose $\{u_1, \cdots, u_P\}$ to be the top P eigenvectors of Σ (that is, the eigenvectors corresponding to the P highest eigenvalues). Now that we have the input data centered to the origin ($\{\tilde{x}^{(1)}, \cdots, \tilde{x}^{(I)}\}$) and the P principal eigenvectors ($\{u_1, \cdots, u_P\}$), we can find the original input data projected onto the hyperplane spanned by the P principal eigenvectors as:

$$y^{(i)} = (u_1^T\tilde{x}^{(i)}, \cdots, u_P^T\tilde{x}^{(i)}), \tag{4}$$

where $y^{(i)} \in \mathbb{R}^{P\times 1}$.

2.2.1 Matrix deficiency

Some matrices are deficient, and, therefore, they do not have full set of eigenvectors (i.e., an $n \times n$ matrix that does not have n different eigenvectors). But, this is not the case with the covariance matrix Σ because it is a symmetric matrix, and, therefore, it can not be deficient. Hence, it will always have a full set of eigenvectors.

2.2.2 Free rotation of the eigenvectors

Sometimes the eigenvectors can rotate freely over their spanned subspace. Hence, it is not meaningful to look at each individual eigenvector and do the associated analysis. PCA is only good for analyzing the subspace spanned by the principal eigenvectors because tiny changes in the data can cause the eigenvectors rotate freely, but the subspace spanned by the P top eigenvectors will remain the same.

2.3 Another point of view of PCA

Choose a direction onto which the training data is projected so to minimize the summed squared difference between the projected points and the original points.

2.4 Applications of PCA

1. Visualization of 50 dimensions $\rightarrow$ 3D
2. Compression
3. Learning: train a learning algorithm using the reduced dimensional data
4. Solving overfitting problems by reducing the number of features
5. Anomaly detection: although it is not the best algorithm for this purpose, but it works fine.
6. Matching/distance calculations

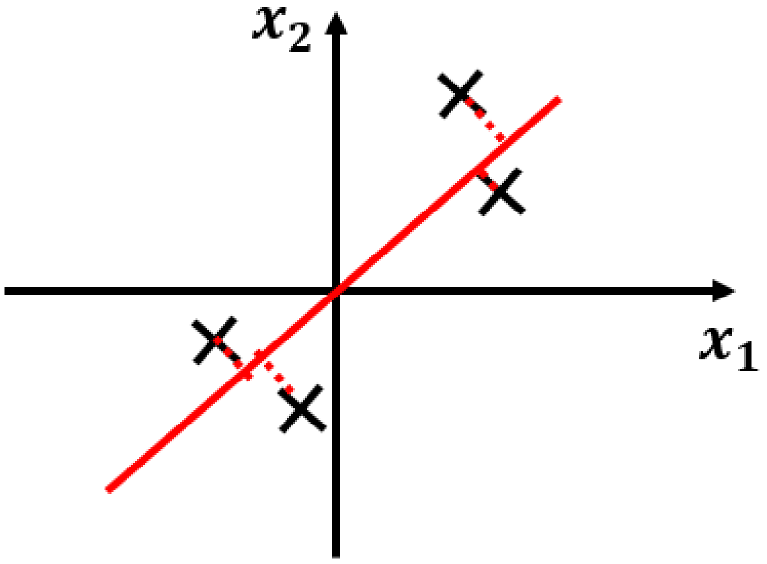


Figure 4: Another interpretation of the PCA

2.5 Recapitulation of the eigenvalue and the eigenvector of a matrix

If $Au = \lambda u$, then u is eigenvector of A and λ is eigenvalue of A. One can convert the following constrained optimization problem into an unconstrained optimization problem as follows. First, the constrained optimization problem can be stated as

$$\begin{aligned} \max_{u} \quad & u^T \Sigma u \\ \text{s.t.} \quad & u^T u = 1. \end{aligned} \tag{5}$$

The, this constrained optimization problem can be transformed into an unconstrained optimization problem as

$$\max_{u} \quad L(u, \lambda), \tag{6}$$

where $L(u, \lambda) = u^T \Sigma u - \lambda(u^T u - 1)$.
One can solve this problem by setting $\nabla L = \Sigma u - \lambda u \equiv 0$ and finding the u that satisfies this condition. Hence, u is an eigenvector of Σ and λ is eigenvalue of Σ. And, it turns out that u is the principal eigenvector.