Week 2

Issues with PCA

1. Time complexity issue
2. Non-linear data

1. Time complexity issue:

$$C = \frac{1}{n}XX^T$$

- $O(d^3)$

If $d >> n$, algorithm become computationally lengthy.

How to fix??

fact 1:

The PCs can be written as linear combination of data points.

$$Cw_k = \lambda_k w_k$$

$$\Rightarrow w_k = \sum_{i=1}^{n} \left(\frac{x_i^T w_k}{n\lambda_k} \right) x_i$$

$$w_k = X\alpha_k$$

- If we can find α_k, we are done.
- elements in α_k also contains the w_k.

Doing some algebra, we get

$$K\alpha_k = n\lambda_k\alpha_k$$

where $K = X^T X$

So,the α_ks, calculated using solving the above eigenequation, are same as we are looking for????

Are these eigenvectors the same as eigenvectots of the covariance matrix??

These α_k must satisfy the following

$$\alpha_k^T K \alpha_k = 1$$

Fact 2

The nonzero eigenvalues of XX^T and X^TX are same.

Using this fact and a bit more algebra, we can say that these α_k are

$$\alpha_k = \frac{\beta_k}{\sqrt{n\lambda_k}}$$

Where β_k is the unit eigenvector corresponding to k^{th} largest eigenvalue $n\lambda_k$.

Algorithm:

- **Step 1:** Compute $K = X^T X$
- **Step 2:** Compute eigen-decomposition of K
    - Let $\{\beta_1, \beta_2, \ldots, \beta_l\}$ are the **unit** eigenvectors corresponding to eigenvalues $\{n\lambda_1, n\lambda_2, \ldots, n\lambda_l\}$.
- **Step 3:** set $\alpha_k = \dfrac{\beta_k}{\sqrt{n\lambda_k}} = \dfrac{\text{unit eigenvector}}{\sqrt{\text{eigenvalue of } K}}$
- **step 4** $w_k = X\alpha_k$

Non-linear data:

Data may not live in low dimensional linear subspace.

Consider the following relationship between the features:

$$f_1^2 - 2af_1 + a^2 + f_2^2 - 2bf_2 + b^2 - 1 = 0$$

Can we transform the dataset into higher dimension space such that the data live in linear subspace of that higher dimension space???

For the above data, consider the following transformation:

$$[f_1, f_2] \rightarrow [f_1^2, f_2^2, f_1 f_2, f_1, f_2, 1]$$

Then for $u = [1, 1, 0, -2a, -2b, a^2 + b^2 - 1]$

$$\phi(x_i)^T u = 0$$

- u is orthogonal to $\phi(x_i)$ for all i.

That is $\phi(x_i)$ live in a linear subspace.

$\phi(x_1), \phi(x_2), \ldots, \phi(x_n)$ live in a linear subspace of a higher dimension space for an appropriate ϕ.

- Apply PCA on $\phi(x_i)$s.

If x_is are centered, will the $\phi(x_i)s$ be centered??

If we apply the PCA on the transformed dataset, the K matrix will be given as

$$K = [\phi(x_i)^T \phi(x_j)]_{n \times n}$$

We need to transform each and every data point and compute the dot product for each pair.

- There is better way to do it!!

Kernel function

A function $k : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ is called a valid kernel if there exist a transformation mapping $\phi : \mathbb{R}^d \rightarrow \mathbb{R}^D$ such that

$$k(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

How to check whether a function is a valid kernel?

1. Find out an appropriate mapping ϕ

   Or

2. Check the following two conditions:

- Symmetric: check $k(x_i, x_j) = k(x_j, x_i)$ For all i, j
- for any dataset $\{x_1, x_2, \ldots, x_n\}$, the matrix $K = [k(x_i, x_j)]$ is a positive semi definite.
  - Check if all the eigenvalues are non-negative or not.

Now, K matrix can easily be computed as

$$K_{ij} = k(x_i, x_j)$$

Now, we have everything ready. Just apply the algorithm discussed in issue 1.

Algorithm:

step 1 Compute K using the kernel function.

step 2 Center the matrix:

$$K_C = K - I_nK - KI_n + I_nkI_n$$

where I_n is the (n, n) matrix with all entries equal to $1/n$.

Step 3: Compute eigen-decomposition of K

- Let $\{\beta_1, \beta_2, \ldots, \beta_l\}$ are the **unit** eigenvectors corresponding to eigenvalues $\{n\lambda_1, n\lambda_2, \ldots, n\lambda_l\}$.

Step 4: set $\alpha_k = \dfrac{\beta_k}{\sqrt{n\lambda_k}} = \dfrac{\text{unit eigenvector}}{\sqrt{\text{eigenvalue of } K}}$

step 5 $w_k = \phi(X)\alpha_k$

w_k can not be computed without knowing ϕ explicitly.

But we still can compute the projetion:

$$\phi(x_i)^T w_k = \sum_{j=i}^{n} \alpha_{kj} k(x_i, x_j)$$