S21CSE471: Statistical Methods in AI **Lec Date: 15/03**

Principal Component Analysis

Prepared by Team 29: Arjun, Arpit, Arvind

1 Introduction

Principal Component Analysis (PCA) is an unsupervised learning algorithm that is used for the dimensionality reduction in machine learning. Large amount of data is good for machine learning but when the number of dimensions are too high then it increases computation time and it also involves some redundant and irrelevant features. So with dimensionality reduction important features are taken into account.

PCA does dimensionality reduction by finding correlation and pattern among various attributes and makes data small without loss of any information.It is very important for problems which have complex data. It involves calculations of covariance and eigen values

2 Understanding PCA

2.1 Principal Components

Principal Components-It is the new set of variables that is obtained from initial values and they are more significant and independent of each other and with this concept we get more useful information rather than before which was more scattered. For example- If we have n-dimensional data then PCA 1 gives more information than PCA 2. PCA 2 gives more info than PCA 3 and so on.

2.2 Use of covariance

Covariance provides a measure of the strength of the correlation between two or more sets of random variates. The covariance for two random variates X and Y, each with sample size N, is defined by the expectation value.

$$Cov_{x,y} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{N-1}$$

We will use covariance matrix to find principal component.

2.3 Dimensionality reduction using PCA

PCA algorithm is divided into 5 steps.

- Standardization of the data
- Computing the covariance matrix
- Calculating the eigenvectors and eigenvalues
- Computing the Principal Components

- Reducing the dimensions of the data set

2.3.1 Data Standardization

Data Standardization is to scale the data in such a way that all data lies in same range and this helps in achieving better models otherwise model becomes biased towards factors which have high value bit contribute more or less same. Standardization is carried out by subtracting each value in the data from the mean and dividing it by the overall deviation in the data set.

$$Z = \frac{variablevalue - mean}{standarddeviation}$$

2.3.2 Find covariance matrix

A covariance matrix is a p × p matrix, where p represents the dimensions of the data set. Each entry in the matrix represents the covariance of the corresponding variables.

Consider a case where we have a 2-Dimensional data set with variables a and b, the covariance matrix is a 2×2 matrix as shown below:

$$\begin{bmatrix} Cov(a,a) & Cov(a,b) \\ Cov(b,a) & Cov(b,b) \end{bmatrix}$$

In the above matrix: Cov(a, a) represents the covariance of a variable with itself, which is nothing but the variance of the variable ‘a’ Cov(a, b) represents the covariance of the variable ‘a’ with respect to the variable ‘b’. And since covariance is commutative, Cov(a, b) = Cov(b, a)

Covariance matrix shows dependency of 2 variables on each other. A positive value shows 2 variables are directly proportional to each other and negative value shows inversely proportional.

2.3.3 Find eigen values

Consider a 2-Dimensional data set, for which 2 eigenvectors (and their respective eigenvalues) are computed. We use covariance matrix to understand where variance is most in data. Since more variance in the data denotes more information about the data, eigenvectors are used to identify and compute Principal Components.

2.3.4 Find Principal Component

After calculating eigen values and eigen vectors we sort the in descending order and then the first highest eigen value and eigen vector forms the first principal component and then the component with less values can be removed and thus helps us in removing features which are of less importance and reduce dimensions.

The final step in computing the Principal Components is to form a matrix known as the feature matrix that contains all the significant data variables that possess maximum information about the data.

2.3.5 Reduce dimensions

We will re-arrange the original data with the final principal components(which we calculated above) and they represent the maximum and the most significant information of the data set. To replace data with new axis we simply multiply the transpose of the original data set by the transpose of the obtained feature vector(in above step).

3 Derivation

3.1 Proving that the top n principal components means extracting top n eigen values of Covariance matrix

The most common definition of PCA is that, for a given set of data vectors $x_i \in [1...n]$, the p principal axes are those orthonormal axes onto which the variance retained under the projection is maximal

Consider a dataset belonging to $\|R\|^d$ consisting of n points, that is $X_{d\times n}$, $\bar{X}$ is a $d \times 1$ dimensional vector representing the mean of n points in each dimension defined as

$$\bar{X} = \frac{1}{n} \times X \bullet [\text{ — } \quad 1 \quad \text{ — }]^T_{1\times n}$$

Let's first project X onto 1 component with unit vector u with dimension $d \times 1$. Thus

$$\begin{aligned} X_u &= u^T X \\ \bar{X_u} &= u^T \bar{X} \end{aligned}$$

From definition, the goal is to maximize the variance of the projected points, thus

$$\begin{aligned} &max\ Var(u^T X) \\ &ST:\ \|u\| = 1 \end{aligned} \tag{1}$$

The constraint signifies u being a unit vector, else the value of u would tend to infinity. Now, in standard form, $Var(u^T X)$ can be written as

$$Var(u^T X) = \frac{1}{n} \times \sum_{i=1}^{n} (u^T x_i - u^T \bar{X})^2$$

Where x_i is the i^{th} row. By removing the summation from the above equation.

$$\begin{aligned} Var(u^T X) &= \frac{1}{n} \times (u^T X - u^T \bar{X})(u^T X - u^T \bar{X})^T \\ &= \frac{1}{n} \times u^T (X - \bar{X})(X - \bar{X})^T u \\ &= u^T S u \end{aligned}$$

Where $S_{d\times d} = \frac{1}{n} \times (X - \bar{X})(X - \bar{X})^T$ is the Covariance matrix

Formulating (1) in Lagrangian form

$$J(u, \lambda) = u^T S u - \lambda * (u^T u - 1)$$

At optimum $\frac{\partial J}{\partial u} = 0$, thus

$$\begin{aligned} &\frac{\partial J}{\partial u} = 2Su - 2\lambda * u = 0 \\ &Su = \lambda * u \end{aligned} \tag{2}$$

(2) satisfies the definition of eigenvector where u is the eigenvector and λ is the corresponding eigen value. Since S is a d dimensional square matrix, in all we will have d eigenvalue-vector pairs

Multiplying (2) by u^T

$$\begin{aligned} &u^T Su = \lambda * u^T u \\ &u^T Su = \lambda \end{aligned} \tag{3}$$

Which says that the variance would be maximum if we set u to eigenvector u_i having largest eigenvalue λ_i. This would be our first principal component. So, in order to extract m principal components, $m < d$, we would take top m eigenvalues and select the corresponding eigenvectors

Now, another property that these m principal components should satisfy is that they should be orthonormal. This condition is also satisfied by (2) since we know S is a symmetric matrix. This can be proved as follows

Let λ_1, λ_2 be eigen values of eigen vectors u_1, u_2 respectively

$$Su_1 = \lambda_1 * u_1 \tag{4}$$

$$Su_2 = \lambda_2 * u_2 \tag{5}$$

Multiplying (4) by u_2^T and (5) by u_1^T

$$u_2^T Su_1 = \lambda_1 * u_2^T u_1 \tag{6}$$

$$u_1^T Su_2 = \lambda_2 * u_1^T u_2 \tag{7}$$

Taking transpose of (7)

$$u_2^T Su_1 = \lambda_2 * u_2^T u_1 \tag{8}$$

Subtracting (6) and (8), we get

$$(\lambda_2 - \lambda_1) * u_2^T u_1 = 0 \tag{9}$$

Symmetric matrices always have distinct eigenvalues, thus $u_2^T u_1 = 0$

3.2 Extracting Principal components using SVD

Singular Value Decomposition (SVD) is a commonly used technique to decompose a matrix into several smaller matrices known as component matrices. Doing this allows us to determine/observe many useful and interesting properties of the original matrix. A matrix A can be decomposed into following

$$\begin{aligned} A &= U\sigma V^T \\ Where\ A &= m \times n\ matrix \\ U &= m \times m\ matrix \\ \sigma &= m \times m\ matrix \\ V &= n \times m\ matrix \end{aligned} \tag{10}$$

A basic approach to actually calculating PCA on a computer would be to perform the eigenvalue decomposition of S directly. It turns out that doing so would introduce some potentially serious numerical issues that could be avoided by using SVD. Since S is a symmetric matrix, it can be decomposed as

$$S = ULU^T$$

Where U is a set of eigenvectors of S and L is a diagonal matrix consisting of eigenvalues

Since the data is standardized, we can consider $\bar{X}$ to be 0 If we perform SVD on X, we will get

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

Now

$$\begin{aligned} S &= \frac{1}{n} XX^T \\ &= \frac{1}{n} U\sigma V^T V \sigma^T U^T \\ &= \frac{1}{n} U(\sigma\sigma^T)U^T \end{aligned}$$

meaning that right singular vectors U are principal directions and that singular values are related to the eigenvalues of covariance matrix via $\lambda_i = \sigma_i^2/n$. Principal components are given by

$$U^T X = \sigma V^T \tag{11}$$

So to extract top k principal components, after finding the SVD of X, we can just select top k rows of V and top k rows and columns of σ

3.3 Applications of SVD

1. Image Compression
   An image can be classified as a matrix A which can then be decomposed via the singular value decomposition as $A = U \sum V^T$
2. Rank of matrix
3. Quantification of sensitivity of a linear system to numerical errors
4. Optimal lower-rank approximation to the matrix
5. Distance (measured by matrix norm) to the nearest rank $i1$ matrix

4 Applications of PCA

1. Quantiative Finance
   Fund managers with a large portfolios can extract X number of Principal Components which best represent the variance in the stocks best. This would reduce the complexity of problem while still explaining the movement of all the stocks of the portfolio (instead of creating a corelational matrix of size 'no. of stocks' * 'no. of stocks'

2. Neuroscience
   PCA is used to find the identity of a neuron from the shape of its action potential, detect coordinated activities of large neuronal ensembles, determining collective variables (order parameters) during phase transitions in the brain, etc

3. Detection and Visualization of Computer Network Attacks

4. Image Compression

References

[1] How are Principal Component Analysis and Singular Value Decomposition Linked
Link: https://intoli.com/blog/pca-and-svd/

[2] Relationship between PCA and SVD
Link: https://stats.stackexchange.com/questions/134282/relationship-between-svd-and-pca-how-to-use-svd-to-perform-pca

[3] A tutorial on Principal component analysis, Link: https://arxiv.org/pdf/1404.1100.pdf

[4] Pattern recognition and machine learning: Section 12.1: Principal component analysis

[5] Pattern recognition and machine learning: Section 12.1: Principal component analysis

[6] https://www.cs.cmu.edu/ mgormley/courses/606-607-f18/slides606/lecture11-pca.pdf

[7] Pattern recognition and machine learning: Section 12.1: Principal component analysis

[8] Principal Component Analysis, MIT 18.650 Statistics for Applications Link: https://youtu.be/WW3ZJHPwvyg