Principal Components Analysis

Cheng Li, Bingyu Wang

November 3, 2014

1   What's PCA

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. The number of principal components is less than or equal to the number of original variables.

In another word, PCA tries to identify the subspace in which the data approximately lies. Here are two examples given by Andrew Ng's Notes:

1) Given a dataset $\{x^{(i)}; i = 1, ..., m \}$ of attributes of m different types of automobiles, such as their maximum speed, turn radius, and so on. Let $x^{(i)} \in \mathbb{R}^n$ for each i. But unknown to us, two different features-some x_i and x_j-respectively give a car's maximum speed measured in miles per hour, and maximum speed measured in kilometers per hour. These two features are therefore almost linearly dependent, up to only small differences introduced by rounding off to the nearest mph and kph. Thus the data really lies approxmimately on an $n - 1$ dimensinal subspace. How can we automatically detect, and perhaps remove, this redundancy?

2) For a less contrived example, consider a dataset resulting from a survey of pilots for radio-controlled helicopters, where x_1^i is a measure of the piloting skill of pilot i, and x_2^i captures how much he/she enjoys flying. Because RC helicopters are very difficult to fly, only the most committed students, ones that truly enjoy flying, become a good pilots. So, the two features x_1 and x_2 are strongly correlated. Indeed, we might posit that the data actually likes along some diagonal axis (the u_1 direction) capturing the intrinsic piloting "karma" of a person, with small amount of noise lying off this axis. (See Figure1) How can we automatically compute this u_1 direction?

2   Prior PCA algorithm

Before running PCA, typically we first pre-process the data to normalize its mean and variance, as following:

1. Let $u = \frac{1}{m} \sum_{i=1}^{m} x^{(i)}$

2. Replace each $x^{(i)}$ with $x^{(i)} - u$

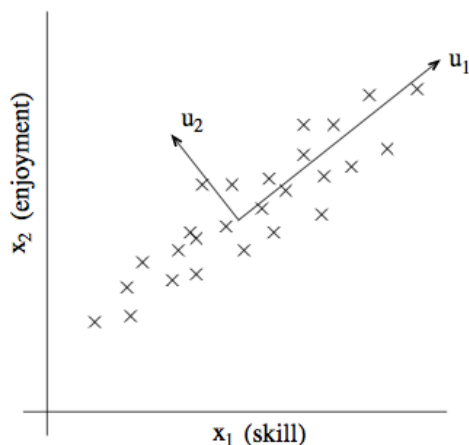3. Let $\sigma_j^2 = \frac{1}{m} \sum_{i=1}^{m} (x_j^{(i)})^2$

Figure 1: Pilot's skill and enjoyment relationship.

4. Replace each $x_j^{(i)}$ with $\frac{x_j^{(i)}}{\sigma_j}$

Step $1-2$ zero out of the mean of the data, and may be omitted for data known to have zero mean(for instance, time series corresponding to speech or other acoustic signals.)

Step $3-4$ rescale each coordinate to have unit variance, which ensures that different attributes are all treated on the same "scale". For instance, if x_1 was car's maximum speed in mph(taking values from $\{10-200\}$) and x_2 are the number of seats(taking values from $\{2-8\}$), then this renormalization rescales the different features to make them more comparable. Step $3-4$ may be omitted if we had apriori knowledge that the different features are all on the same scale(for instance, the MNIST digits dataset).

3 PCA Theory

After the prior Steps $1-4$, which has been described above, all we have to do is just only an eigenvector calculation. Then we just choose the top k eigenvectors as the new subspace. Why does it work and what is the theory behind the PCA? In fact, there are many theories can explain the PCA. But we just choose one of them, called Maximum Variance theory.

3.1 Maximum Variance Theory

Having carried out the normalization, how do we compute the "major axis of variation" u-that is, the direction on which the data approximately lies? One way to pose this problem is as finding the unit vector u so that when the data is projected onto the direction corresponding to u, the variance of the projected onto the direction corresponding to u, the variance of the projected data is maximized. Intuitively, the data starts off with some amount of variance/information in it. We would like to choose a direction u so that if we were to approximate the

data as lying in the direction/subspace corresponding to u, as much as possible of this variance is still retained.

3.2 Analysis

Consider the following dataset(See Figure 2), on which we have already carried out the normalization steps: Now, suppose we pick u to correspond the direction
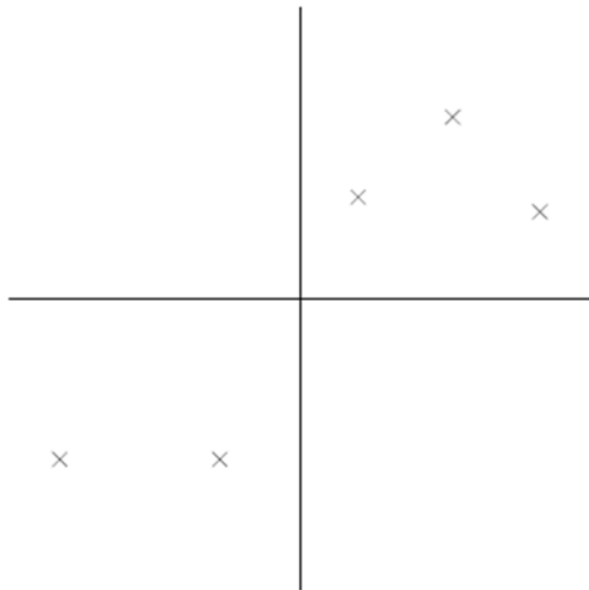

Figure 2: Sample data in 2-Dimention

shown in Figure 3. The circles denotes the projections of the original data onto this line.

We see that in Figure 3 the projected data still has a fairly large variance, and the points tend to be far from zero. In contrast, suppose had instead picked the following direction(Figure 4):

In the Figure 4, the projections have a significantly smaller variance, and are much closer to the origin.

We would like to automatically select the direction u corresponding to the Figure 3 with maximum variance. First we should know how to calculate the distance of the point's projection onto u from the origin. Note that given a unit vector u and a point x, the length of the projection of x onto u is given by $x^T u$. So the problem can be transformed to mathematic problem as following: choose u so that:

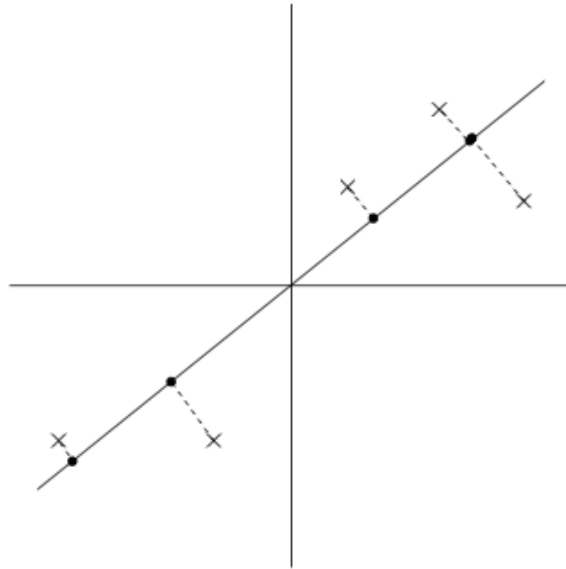$$\max_{u:\|u\|=1} \frac{1}{m} \sum_{i=1}^{m} (x^{(i)T} u)^2$$

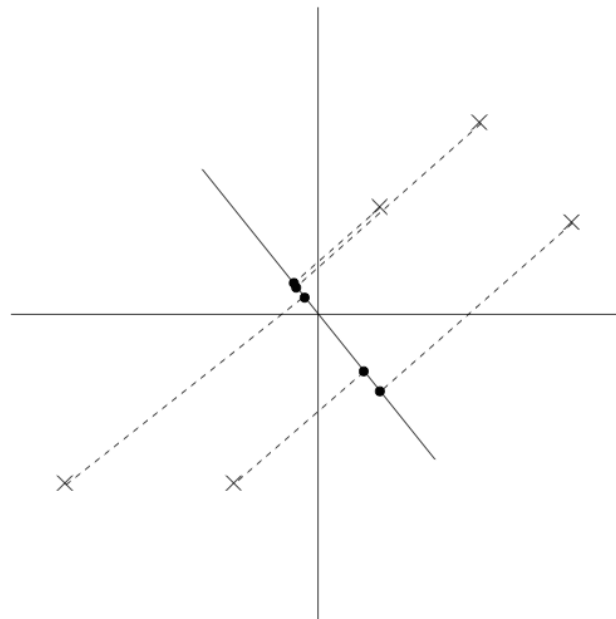Figure 3: Sample data with Maximum Variance

Figure 4: Sample data with Minimum Variance

Next, we will derivate the fomular as following:

4

$$\implies \max_{u: \|u\|=1} \frac{1}{m} \sum_{i=1}^{m} (x^{(i)T} u)^2$$

$$\implies \max_{u: \|u\|=1} \frac{1}{m} \sum_{i=1}^{m} (x^{(i)T} u)^T (x^{(i)T} u)$$

$$\implies \max_{u: \|u\|=1} \frac{1}{m} \sum_{i=1}^{m} (u^T x^{(i)})(x^{(i)T} u)$$

$$\implies \max_{u: \|u\|=1} \frac{1}{m} \sum_{i=1}^{m} u^T x^{(i)} x^{(i)T} u$$

$$\implies \max_{u: \|u\|=1} u^T (\frac{1}{m} \sum_{i=1}^{m} x^{(i)} x^{(i)T}) u$$

$$\implies \max_{u: \|u\|=1} u^T \Sigma u, \quad where \quad \Sigma = \frac{1}{m} \sum_{i=1}^{m} x^{(i)} x^{(i)T}$$

Now we can use Lagrange multiplier to continue:

$$\implies \max u^T \Sigma u \qquad subject \quad u^T u = 1 (Because \|u\| = 1)$$
$$\implies \mathcal{L}(u, \lambda) = u^T \Sigma u + \lambda (u^T u - 1)$$
$$\implies \frac{\partial \mathcal{L}(u, \lambda)}{\partial u} = \frac{\partial (u^T \Sigma u + \lambda u^T u)}{\partial u} = \frac{\partial (u^T \Sigma u)}{\partial u} + \frac{\partial (\lambda u^T u)}{\partial u}$$

Since $u \in \mathbb{R}^n$, which is a column vector. According to Matrix calculus[1], which is shown as following:

$$\frac{\partial x^T A x}{\partial x} = 2Ax$$
$$\frac{\partial x^T x}{\partial x} = 2x$$

where A is not a function of x, A is symmetric, and x is a row vector. Then we get:

$$\frac{\partial (u^T \Sigma u)}{\partial u} + \frac{\partial (\lambda u^T u)}{\partial u}$$
$$= 2\Sigma u + 2\lambda u \overset{Set}{=} 0$$

Since λ is a scalar, we can replace λ by $-\lambda$. Finally, we get the $\Sigma u = \lambda u$, where u is the engivector of Σ and λ is the engivalue of Σ.

To summarize, we have found that if we wish to find a 1-dimensional subspace with to approximate the data, we should choose u to be the principal eigenvector

[1] http://en.wikipedia.org/wiki/Matrix_calculus

of Σ. More generally, if we wish to project our data to k-dimensional subspace $(k < n)$, we should choose $u_1, u_2, ..., u_k$ to be the top k vectors of Σ. The u_i's now form a new, orthogonal basis for the data.[2]

Then, to represent $x^{(i)}$ in this basis, we need only compute the corresponding vector

$$\hat{x}^{(i)} = \begin{bmatrix} u_1^T x^{(i)} \\ u_2^T x^{(i)} \\ ... \\ u_k^T x^{(i)} \end{bmatrix} \in \mathbb{R}^k.$$

Thus, whereas $x^{(i)} \in \mathbb{R}^n$, the vector $\hat{x}^{(i)}$ now gives a lower, k-dimensional, approximation/representation for $x^{(i)}$. That is to say, $x^{(i)}$ is the original datapoint and $\hat{x}^{(i)}$ is the new datapoint after PCA. PCA is therefore also referred to as a **dimensionality reduction** algorithm. The vectors $u_1, ..., u_k$ are called the first k **principal components** of the data.

4 References

- PCA Lecture Notes by Andrew Ng(Stanford Univ.).

---

[2]Because Σ is symmetric, the u_i's will (or always can be chosen to be) orthogonal to each other.