**Principal Component Analysis**

**by Ricky Ho,** [**http://horicky.blogspot.com/2009/11/principal-component-analysis.html**](http://horicky.blogspot.com/2009/11/principal-component-analysis.html) **(with minor corrections)**

One common problem of machine learning (ML) is the "**curse of high dimensionality**". When there are too many attributes in the input data, many of the ML algorithms will be very inefficient or some of them will even be non-performing (e.g. in nearest neighbor computation, data points in a high-dimensional space are pretty much equally distant to each other).

It is quite possible that the attributes we selected are inter-dependent on each other. If so, we may be able to extract a smaller subset of independent attributes that may still be very useful to describe the data characteristics. In other words, we may be able to reduce the number of dimensions significantly without losing much fidelity of the data.

"**Dimension Reduction**" is a technique to determine how we can reduce the number of dimensions while minimizing the loss of fidelity of data characteristics. It is typically applied during the data cleansing stage before feeding into the machine learning algorithm.

"**Feature Selection**" is a simple technique to select a subset of features that is more significant or relevant. A very simple "filtering" approach can be used by looking at each attribute independently, then rank the attributes’ significance using some measurement (e.g. information gain) and throwing away those that have only minimum significance. A more sophisticated "wrapper" approach is to evaluate different subsets of features. There are two common models in the "wrapper" approach, "forward selection" and "backward elimination".

In **forward selection**, we start with zero attributes and then start to pick the attributes with the highest statistical significance, (ie: prediction improves a lot as measured by cross validation tests). After picking the first attribute, we next select the best second attribute and find the one conferring the most significant improvement in the cross-validation check. We keep growing the set of attributes until we don't find significant improvements. One issue of the "forward selection" approach is that it may miss "grouped features". For example, attribute1 and attribute2 may be insignificant when they are considered alone, but combining them will give very big improvement.

**Backward elimination** can be used to handle this problem. It basically goes the opposite direction, starting with the full set of attributes and dropping those attributes that have the least statistical significance (ie: for which prediction degrades very little in cross-validated tests). The downside of "backward elimination" is that it is much more expensive to run.

A more powerful approach called "**Feature Extraction**" is more commonly used to extract a different set of attributes by linearly combining the existing set of attributes. Principal Component Analysis "PCA" is a very popular technique in this arena. PCA can analyze the interdependency between pairs of attributes and identify those significant combinations.

**The intuition behind PCA**
The intuition is that PCA rearranges the existing m attributes to form another set of m attributes that are a linear combination of the original ones. The new set of attributes has the characteristic that

* Each attribute is independent of each other (i.e., are (orthogonal)
* The attributes are ranked according to the variation in the data that they explain

Note that attributes explaining only small amounts of variation don't provide much information to describe the data samples and so can be ignored with minimal lost of fidelity. So we can safely remove those to reduce the dimensionality of the data.

The question is: How do we rearrange the m attributes to exhibit the above 2 characteristics? Let’s take a deeper look into it.

**Underlying theory of PCA**
Assume there are N data points in the input data set and each data point is described by M attributes. We use the statistical definition for the "mean", "variance" of each attribute and "co-variance" for every pair of attributes. Co-variance is an indicator of dependencies of two attributes with zero implies independence.


In an ideal situation, we want COV-x to be a diagonal matrix, which means COV(i, j) to be zero. In other words, all pairs of attribute-i and attribute-j are independent to each other. We also want the diagonal to be ranked in descending order.

So the problem can be reduced to finding a different combination of the m attributes to form a new set of m attributes (Y = P. X) such that COV-y is a ranked diagonal matrix.

How do we determine P?


**Some Matrix theory**
Here is a review of Matrix theory that will be used



Let’s find the transformation matrix P


So the PCA process can be summarized in following ...

1. Input: X, a matrix of (m \* n), a set of N sample data points, each with M attributes.
2. Compute Cov-X, a matrix of (m \* m), the Covariance matrix of X
3. Compute the m Eigenvectors and m Eigenvalues of Cov-X
4. Order the Eigenvectors according to the Eigenvalues
5. Now find the transformation matrix P, which is a matrix of (m \* m). Note that each row vector of P corresponding to an eigenvector, which is effectively the axis of the new co-ordinate system.
6. Truncate P to just take the top k rows. Now P' is a (k \* m) matrix.
7. Apply P' . X to all input data to result in a matrix of (k \* n). This is effectively reducing each data vector from m-dimension to k-dimension.

**References**
[A very good paper](http://www.snl.salk.edu/~shlens/pub/notes/pca.pdf)
[Some Matrix math review and step by step PCA calculation](http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf)