Unsupervised Methods:
An Overview for Actuaries

By Brian Holland

Actuaries might have some familiarity with unsupervised methods. In this article I'd like to focus on these methods, their differences from supervised methods, and some examples from actuarial practice that we might not generally know.

First of all: what are supervised methods? Who is supervising? You are supervising. Supervised methods involve you, the modeler, saying which observation depends on the other observations. Examples we all know are regression and classification. In regression, you say which number is *y* and which is x. That decision is the supervision. In classification you do the same thing: you know target categories, and are fitting items into categories based on characteristics. Linear regression, generalized linear models, and generalized additive models all fit into this category.

So what could unsupervised methods be? How could you do anything without saying what you're trying to model? That was my first question at least. Unsupervised methods have no labels with known meaning. Their goal is to find structure in the data. Think of the task as describing the space.

An example is clustering. Whatever is an *x* or a *y* or a *z*, you might first want to know if there are clusters. Are there clumps of items here or there? That question might be good to ask. There are some canned routines to compute clusters already in the Python language library scikit-learn for machine learning. Scikit-learn is the subject of Jeff Heaton's article in this newsletter. The scikit-learn documentation at http://scikit-learn.org/stable/modules/clustering.html includes a helpful comparison of types of clusters, types of algorithms to detect clusters, and the results of those algorithms.

Applications of clustering are right at hand:

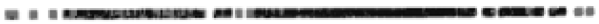
- Actuarial models: how detailed should they be? Model points could be clustered.
  - Freedman and Reynolds (2008): "Cluster analysis: a spatial approach to actuarial modeling."
  - How much granularity do you need for premiums, assumptions, and models?
- Recommender systems, collaborative filtering
  - Customers who liked certain things might have found products that you also might like. Those customers form a cluster.
  - Clustering of types of objects based on similar characteristics.

Underwriting categories – we're clustering by appropriate premium level.

  - If categories are already set: I'd say this is a classification problem: supervised.
  - If categories are being developed, I'd say this is a clustering problem: unsupervised.

Facial recognition: which faces are similar?

In clustering the number of dimensions or attributes matters, whether you know what the attributes represent or not. For example, with only one attribute here, we see a couple of groups.



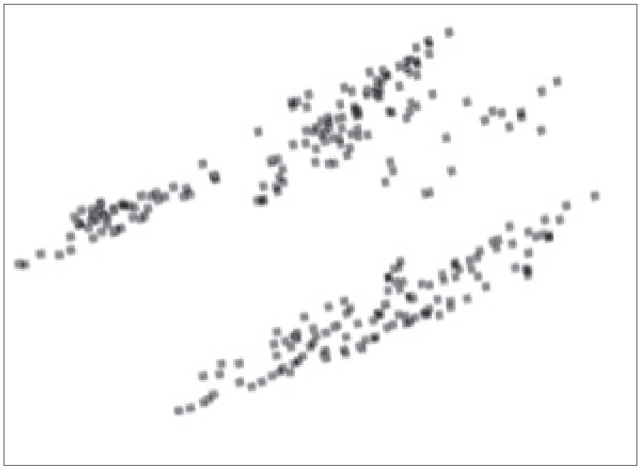
With two attributes the picture is much richer, and some groups in the 1-D example would get split into more, just from looking at it. The following image represents tilting the 2-D view forward in 3-D, to give a sense that there is a bit more going on here. In the bottom island in 2-D there is maybe a ridge of points which are closer together.

Brian D. Holland, FSA, MAAA, is director and actuary, Experience Studies at AIG. He can be reached at brian.holland@aig.com.

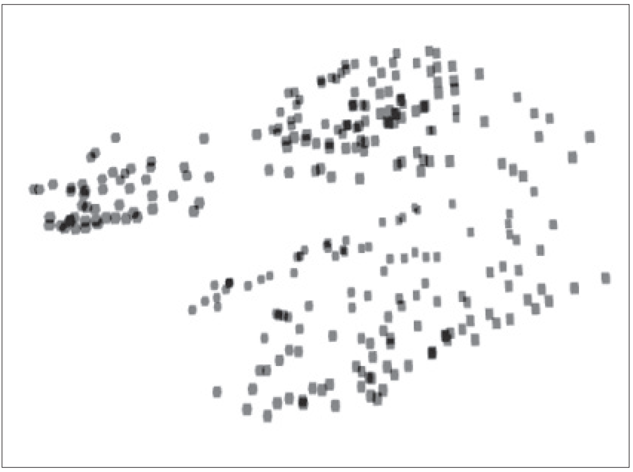
CONTINUED ON **PAGE 16**

UNSUPERVISED METHODS … | FROM PAGE 16

Points in 2-D

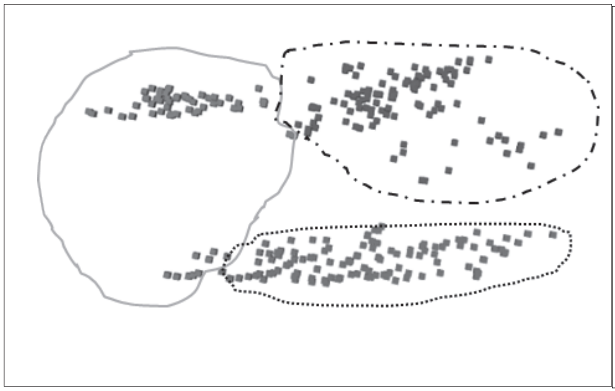


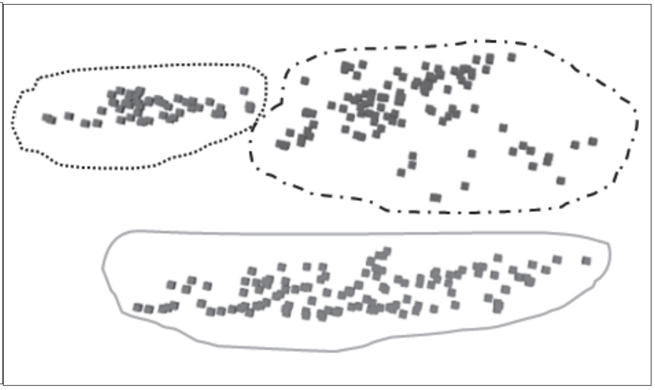
Points in 3-D, Tilted Away



So how would you group these points into clusters? I recommend using well-studied and documented routines if in doubt. That way, the procedure is at least written up and known.

K-means: given number of groups (here: 3), pick k starting points to represent groups, assign each point to the nearest representative position (centroid), recompute the average after assignment, and do again. This is an iterative process. Here, the border between the three regions separates nearby points.	DBSCAN: emphasizes proximity of nearby points. The results here are generally the same classes, but there are some differences from K-Means. The lower cluster is all in one group.





The number of clusters we pick clearly depends on the number of dimensions we're considering. But what does "dimensions" mean anyway? An intuitive answer is that it means how many numbers are needed to describe the situation. For example, a line is 1-D even in 3-D space. As long as you can rotate and move your axes the right way, you can represent the whole line with one dimension. A financial example profit = income – expense. You might have three columns for the three numbers. If you know two, you know the third, so are there really three dimensions? There are not, not if you turn it the right way. The upshot is that you might not have to deal with as many numbers, columns, etc. as it appears at first, as long as each revised axis describes the right combination of original features.

DIMENSION REDUCTION

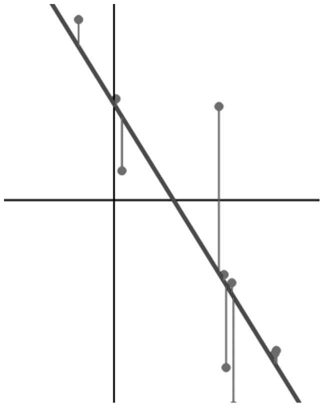
Why would we want to reduce the number of dimensions instead of using all the available information? I have two solid reasons for you:

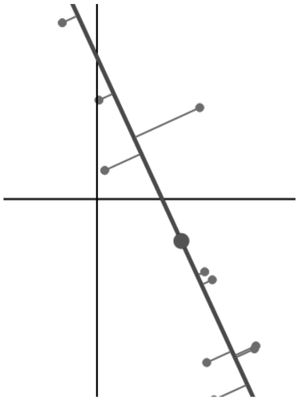
1. Visualization: to see the main features. We can order the dimensions by variability along the new axes to see the main features.

2. Clustering: it is cheaper to calculate dimensions with fewer coordinates.

A technique called singular value decomposition (SVD) can be used to find and to order new dimensions. It determines the main dimensions or axes, which is those that pick up the most variance of the data, orders them, and quantifies the spread of the data around those dimensions. It differs from regression in that all coordinates (x and y for example) are treated the same. It minimizes squared distance to a line—a 1-dimensional subspace—not from a value y to a predicted value up or down the y axis. Then dropping that dimension from the new coordinates, there is one dimension left, and the procedure can be repeated until none are left. The sum of squared coordinates along a new axis indicates the variance along that axis, and orders the axes naturally.

Linear regression: minimize sum of squared distances from point up or down to the regression line.	SVD: minimize distance from point to the line. Here, the line goes through the average point, shown as a larger point.





CONTINUED ON **PAGE 18**

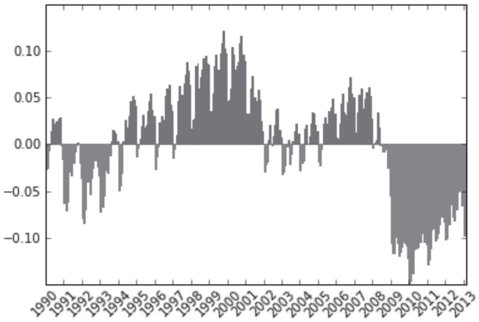
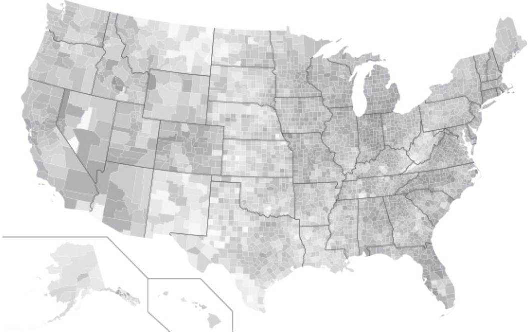
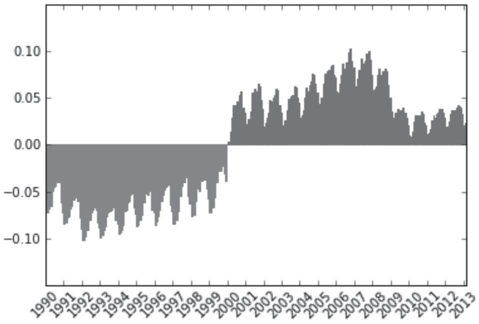
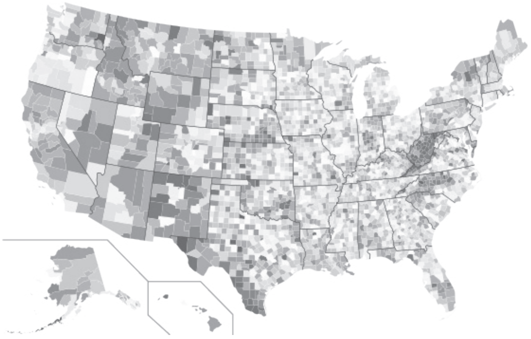
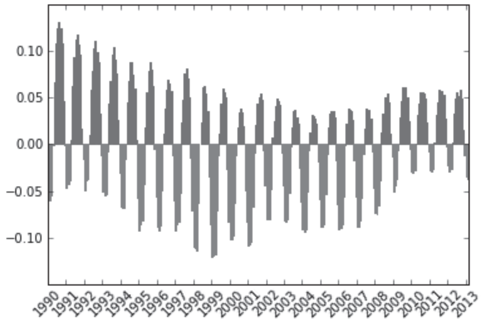
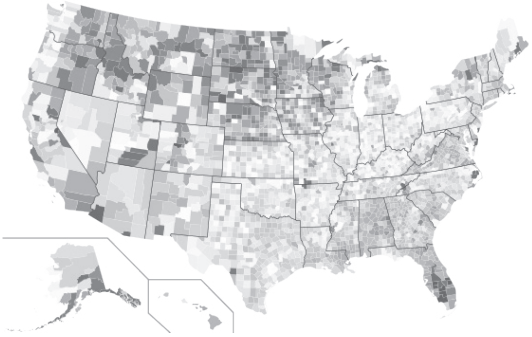
UNSUPERVISED METHODS … | FROM PAGE 17

When the decomposition is finished, the original data matrix *X* is represented as a product of three matrices:

$$X = U\ S\ V'$$

U and V are both orthonormal matrices. Their columns represent the new axes we could say. S is a diagonal matrix, with values decreasing down the diagonal. Multiplying these matrices amounts to taking the first column of U to scale each row of the result; the first column of V (transposed) to scale across the columns, times the first diagonal element of S for an overall scale level. Doing the same with the second columns of U and V and the second diagonal element of S.

Can SVD help us find the main features in a larger dataset? To illustrate, I've tried out SVD on unemployment by county by month. I used unemployment because we all have some domain knowledge just from following the news. An animation of this detailed unemployment can be seen at *http://bdholland.blogspot.com/2013/05/visualizing-unemployment-by-county.html.* Some patterns are visible from watching the animation. SVD of the matrix of unemployment rates does show recognizable patterns. If we have X = U S V', a matrix with rows of months and columns of counties, then columns of U can be represented by time series; and columns of V can be represented as maps. We would make X by adding up layers: taking pairs of month and county vectors, blowing them out to make a matrix, and scaling them. The first three pairs are shown below. The first pair of columns of U, V is the familiar macroeconomic story: across most of the United States, there is worsening unemployment to 1992, improvement to the dot-com bust, then improvement to the mortgage crisis at which time there was a big spike in unemployment. The second layer is a regional correction. The third layer is my favorite: a mostly seasonal layer by date, with a map that clearly matches seasons. Note that the maps originally had red and green values: green for positive and red for negative. Values near white are near zero in any case.

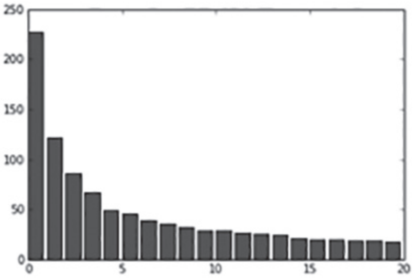
<table>
<tr><th>Vector</th><th>Month singular vector</th><th>County singular vector</th></tr>
<tr><td>1</td><td>
</td><td></td></tr>
<tr><td>2</td><td>
</td><td></td></tr>
<tr><td>3</td><td>
</td><td></td></tr>
</table>

CONTINUED ON **PAGE 20**

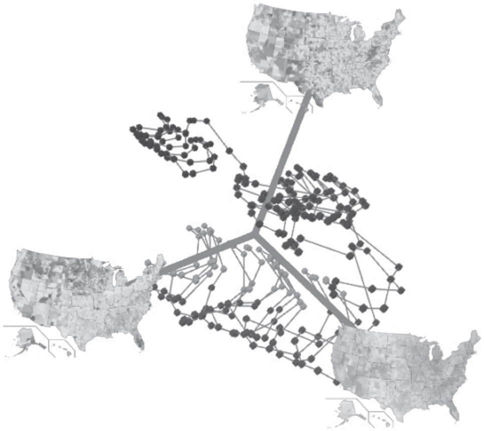
UNSUPERVISED METHODS … | FROM PAGE 19

The scaling factors, the diagonal of S, are called singular values. They drop off by design, as later pairs are smaller or less important for describing the landscape. The singular values show the relative magnitude of the different layers, here for the first 20 of the nearly 280 layers. It is clear that much of the variability in unemployment by time and county is captured in the first three pairs of singular vectors shown above:

Singular values 1-20 of centered unemployment by county

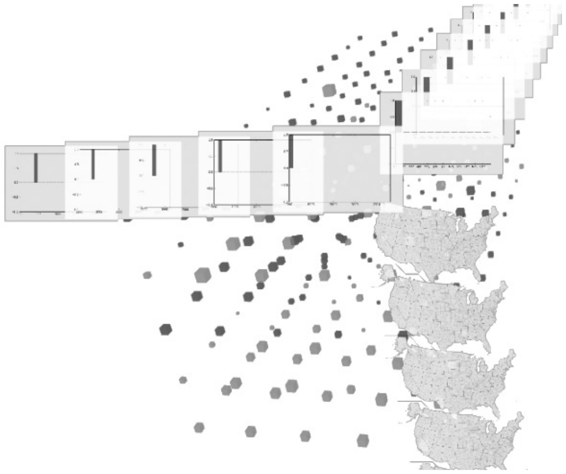


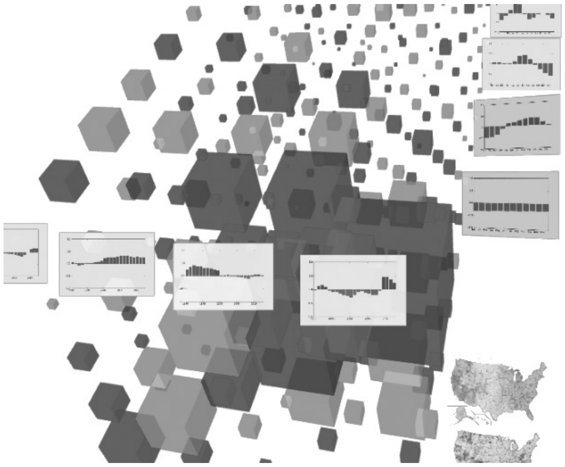
We can look at clustering of months as well by examining calendar months by the first three singular vectors. The clustering for different dimensions shown earlier was actually the calendar months. Below the months are connected as a time series, showing the three axes' meanings: the main singular vectors by county, represented as maps. It is now clear that the seasonal axis is the reason that the values bounce in and out through the seasons. The blue cluster (upper-left-most) is the period after the mortgage crisis. The macro axis, going down to the right, picks up the better and worse periods generally. From this exercise it is also clear what a weakness of this method is. We have to name the axis: if we're lucky, they make some sense, but they can be hard to explain. This issue can come up with matrix decompositions.



An actuarial application is the Lee-Carter mortality improvement model. Lee and Carter published their model in 1992. There have been changes since and that was some years ago, but that is the point: SVD has been used some time ago in actuarial applications. The model was not initially stated in terms of SVD, but Lee and Carter noted that the solution could be found with SVD. Note the language in the original paper: "… there are no given regressors." That is not supervised regression, but an unsupervised model. The authors effectively decomposed a matrix of mortality rates by age and calendar year, taking the first singular vectors for each of age and calendar year. For the projection of mortality rates, autoregressive integrated moving average (ARIMA) was used on the calendar year singular vector.

Higher-order singular value decomposition (HOSVD) is a logical next step beyond SVD. HOSVD means "higher-order" SVD. It goes by several names and has arisen in several contexts. It amounts to a way to decompose a tensor—effectively an array for actuaries—with more indices than the two indices of an array. The example above shows seasonality in each of the first three month-singular vectors. I decomposed a tensor of unemployment rates which was just a rearrangement of the same numbers into an array by calendar year, calendar month, and county. The unemployment rates by month (to the upper right), year (to the left) and county are shown on the left below for the top portion of the tensor. The decomposed tensor on the right shows the same county maps, but the calendar months show different patterns, and the calendar years are only the annual effects. The original tensor or array is replicated by scaling each combination of month, year and county singular vector triplet by the volume of the corresponding cube, and adding all such layers.





There are several closely related topics that are worth mentioning briefly:

SVD

$X = U S V'$ possible for any matrix.

X: the (centered) data

U, V: their columns are called left and right singular vectors, respectively.

U, V are orthonormal, which also means we can see them as a rotation.

S: diagonal matrix, with values decreasing on the diagonal.

PCA: principal components analysis

V: columns are the *principal components*

U S: contains *principal component scores*

Covariance matrix of mean-centered matrix X is $X' X / n = V S^2 V' /n$ since $U^{-1} = U'$

To my mind, the main point to remember about unsupervised learning methods is that they are used to find structure in data, without any domain knowledge of the source data or explicit modeling. They can be used to show the main features, which might be clusters of data, or high-level features. Clustering methods give mathematical support and convenience to functions that actuaries regularly perform.