Practical Methods for the Analysis of "Big Data"

Module 4: Clustering, Decision Trees, and Ensemble Methods

Philip A. Schrodt

The Pennsylvania State University
schrodt@psu.edu

Workshop at the Odum Institute
University of North Carolina, Chapel Hill
20-21 May 2013

Topics: Module 4

Clustering

- K-Means

Hierarchical Clustering: Dendograms

Comparisons

- Generating Dendograms from LDA Topics

Classification Trees

Ensemble Methods

- Bayesian Model Averaging
- Random Forests^{TM}
- Boosting

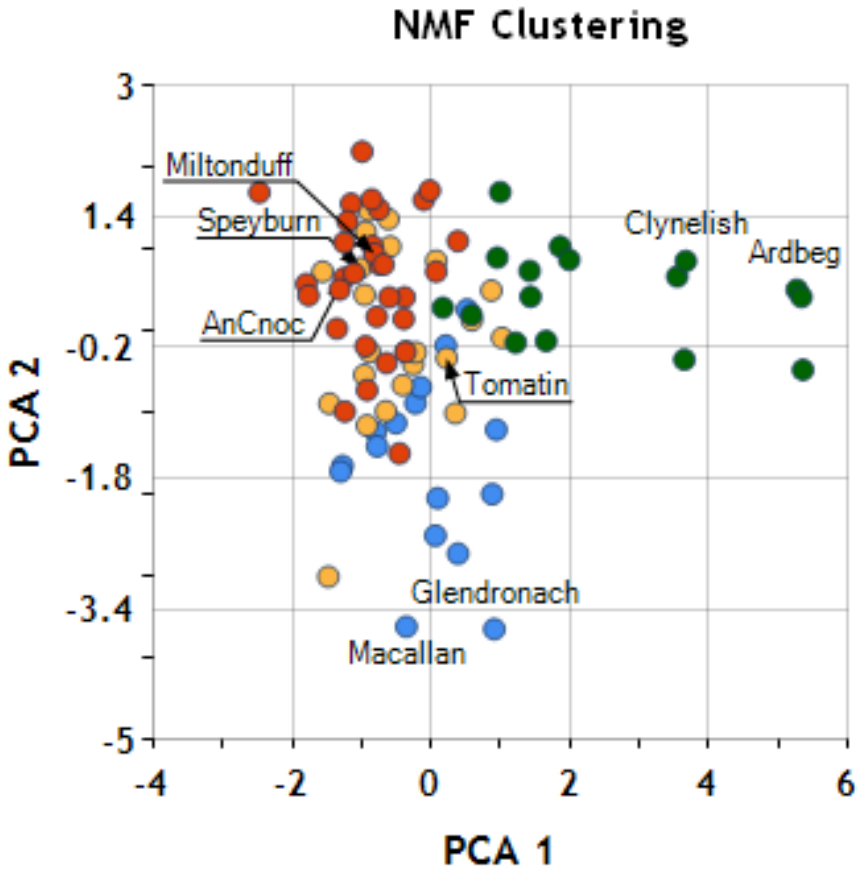
Topics: Module 4

- Clustering
  - K-Means
- Hierarchical Clustering: Dendograms
- Comparisons
  - Generating Dendograms from LDA Topics
- Classification Trees
- Ensemble Methods
  - Bayesian Model Averaging
  - Random Forests™
  - Boosting

General comments

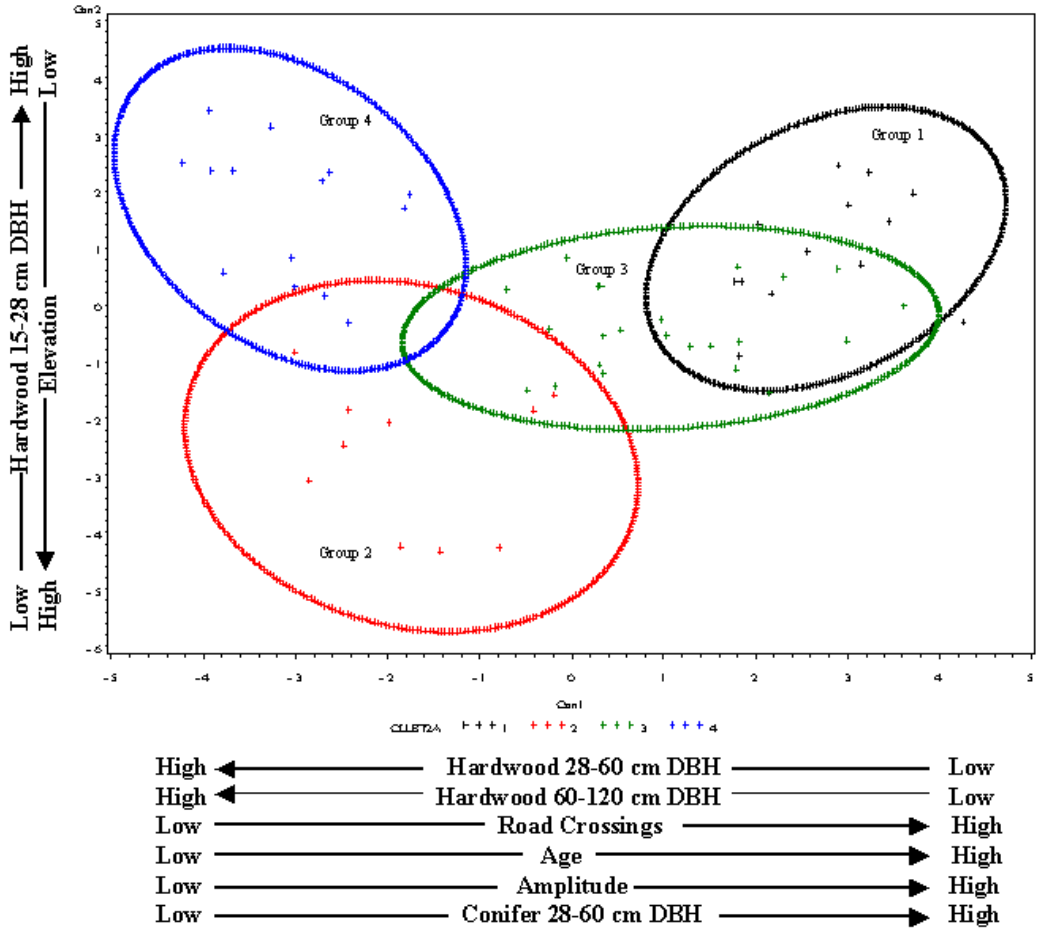
- Requires a metric—and there are many—for the distance between the cases
- In contrast to linear approaches—but similar to SVM—this assumes heterogeneous subpopulations
- Clustering is typically depicted in two dimensions but usually is computed in an arbitrarily large space
-

Cluster Example 1



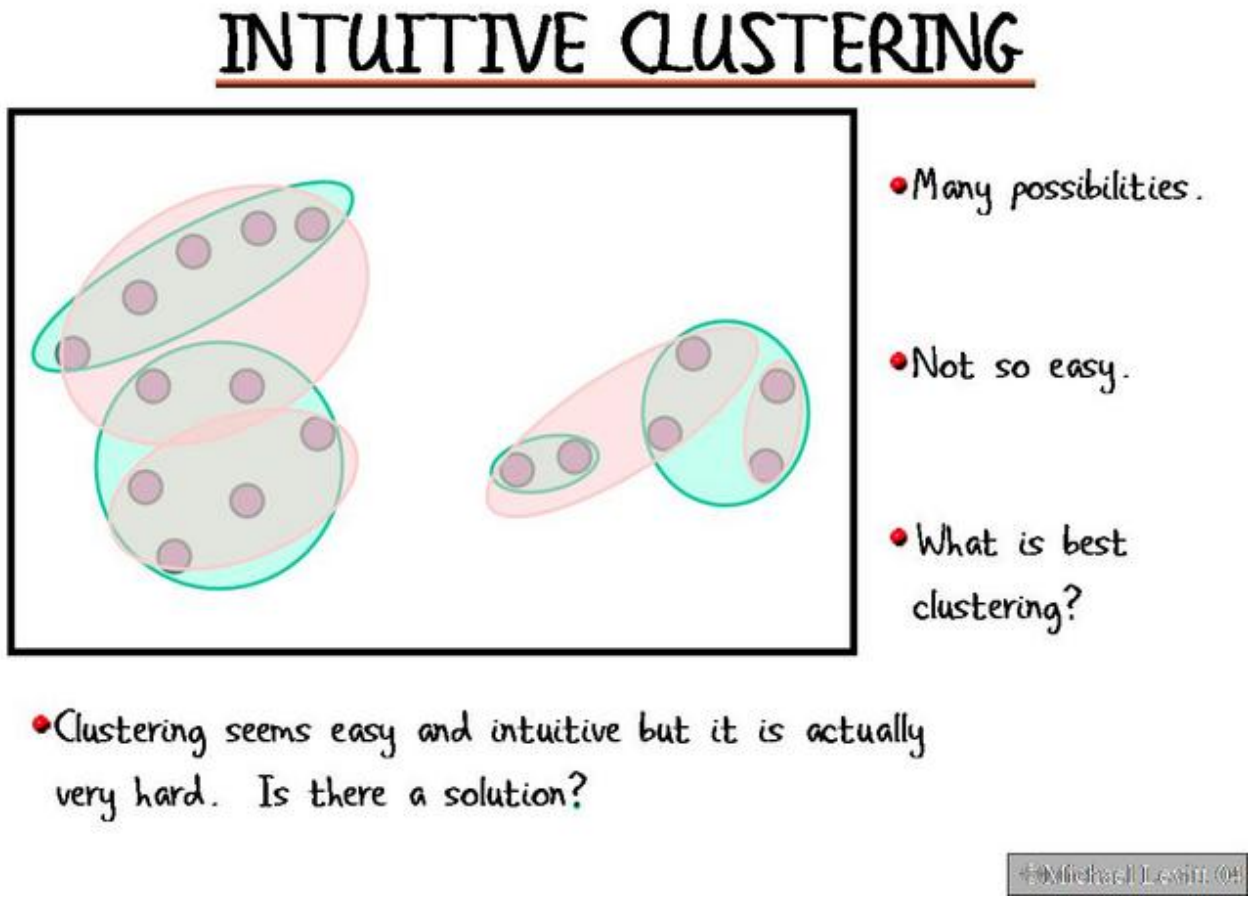
Exercise: search Google images for “cluster analysis” for a zillion examples

Cluster Example 2



[this had something to do with herpetology, perhaps explaining the importance of “road crossings”]

Intuitive Clustering



Diagrams from Michael Levitt, Structural Biology, Stanford

Source: http://csb.stanford.edu/class/public/lectures/lec4/Lecture6/Data_Visualization/images/Intuitive_Clustering.jpg

Overview of distance metrics

Distance Measurements Between Data Points

This parameter specifies how the distance between data points in the clustering input is measured. The options are:

- Euclidean: Use the standard Euclidean (as-the-crow-flies) distance.
- Euclidean Squared: Use the Euclidean squared distance in cases where you would use regular Euclidean distance in Jarvis-Patrick or K-Means clustering.
- Manhattan: Use the Manhattan (city-block) distance.
- Pearson Correlation: Use the Pearson Correlation coefficient to cluster together genes or samples with similar behavior; genes or samples with opposite behavior are assigned to different clusters.
- Pearson Squared: Use the squared Pearson Correlation coefficient to cluster together genes with similar or opposite behaviors (i.e. genes that are highly correlated and those that are highly anti-correlated are clustered together).
- Chebychev: Use Chebychev distance to cluster together genes that do not show dramatic expression differences in any samples; genes with a large expression difference in at least one sample are assigned to different clusters.
- Spearman: Use Spearman Correlation to cluster together genes whose expression profiles have similar shapes or show similar general trends (e.g. increasing expression with time), but whose expression levels may be very different.

Distance Measurements Between Clusters

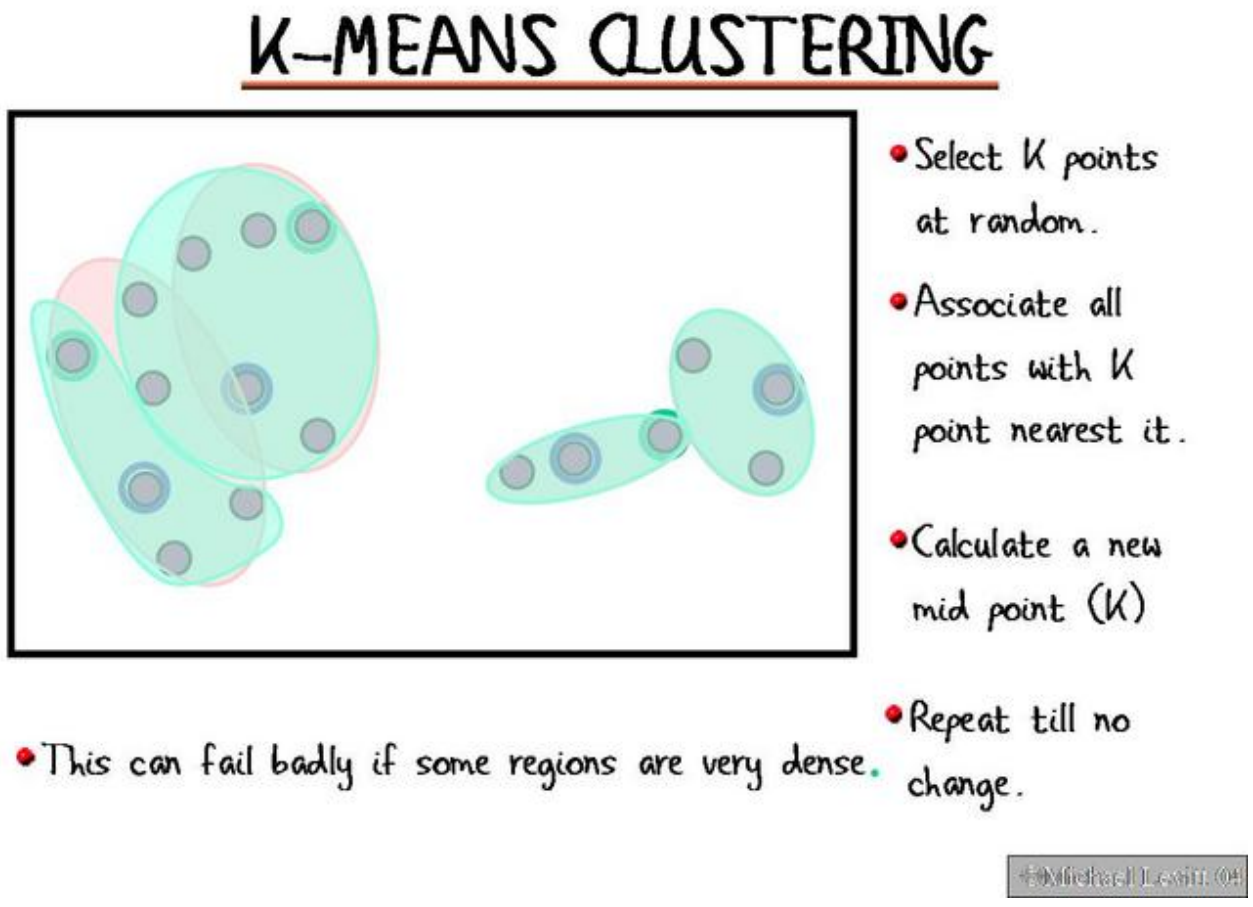
This parameter specifies how the distance between clusters is measured. The options are:

- ***Average Linkage***: The distance between two clusters is the average of the distances between all the points in those clusters.
- ***Single Linkage***: The distance between two clusters is the distance between the nearest neighbors in those clusters.
- ***Complete Linkage***: The distance between two clusters is the distance between the furthest points in those clusters.

Source:

http://www.improvedoutcomes.com/docs/WebSiteDocs/Clustering/Clustering_Parameters/Distance_Metrics_Overview.htm

K-Means



Source: http://csb.stanford.edu/class/public/lectures/lec4/Lecture6/Data_Visualization/images/K-Means_Clustering.jpg

K-Means algorithm

K-means Clustering

- Randomly assign each of $x_1 \ldots, x_N$ to K user specified clusters
- Compute the average value of the points, or centroid, of each cluster
- For each i=1, ..,N compute the distance between x_i and each of the cluster centroids
- Assign x_i to the cluster with the closest centroid and recalculate the centroids of the affected clusters
- Iterate until no more reassignments are made

Source: http://biology.unm.edu/biology/maggieww/Public_Html/K-means.gif

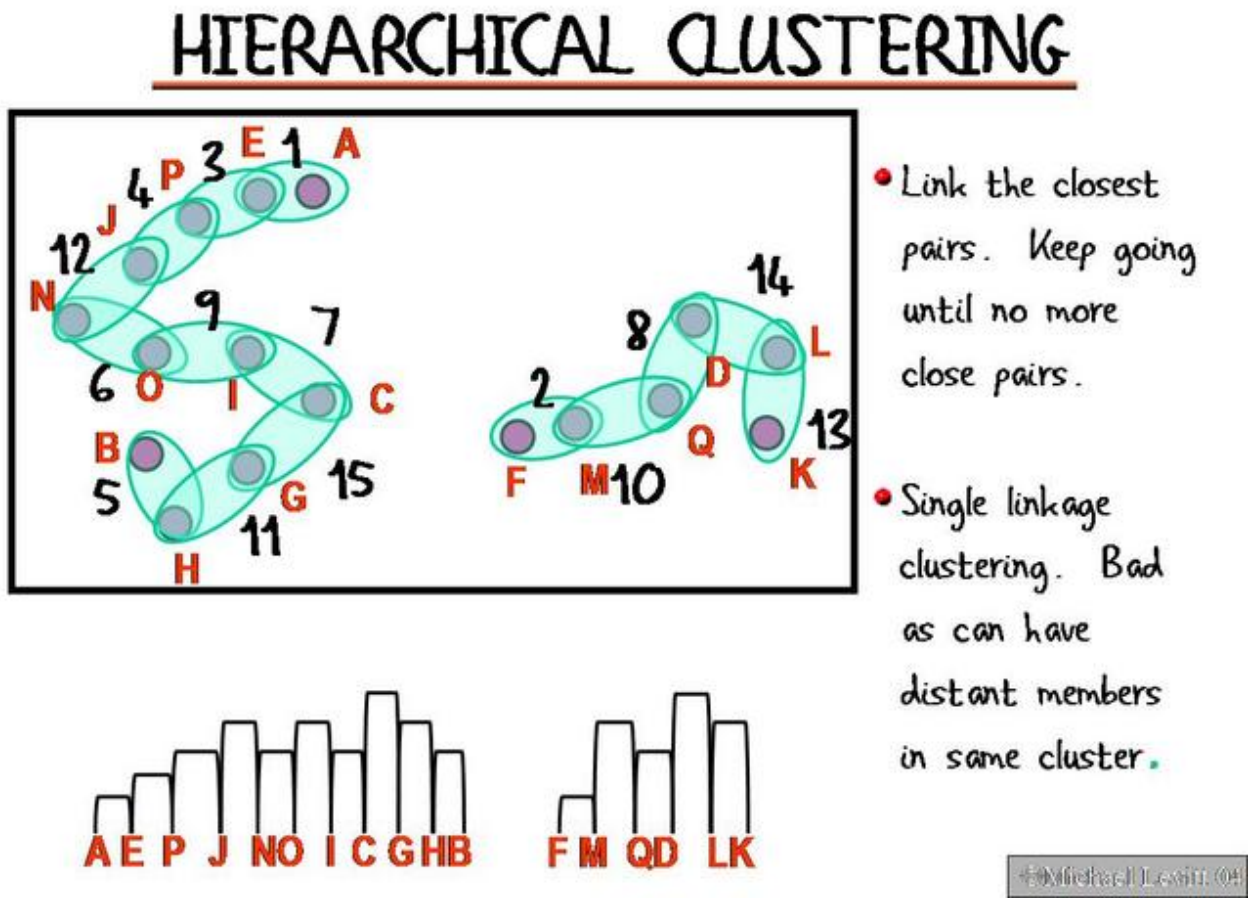
K-Means: Issues

- Results vary depending on the number of clusters
- Results vary depending on the random starting points: one approach is to do a number of these and see which clusters consistently emerge

Let’s go exploring!

Google Image Search: “k means clustering”

Hierarchical Clustering



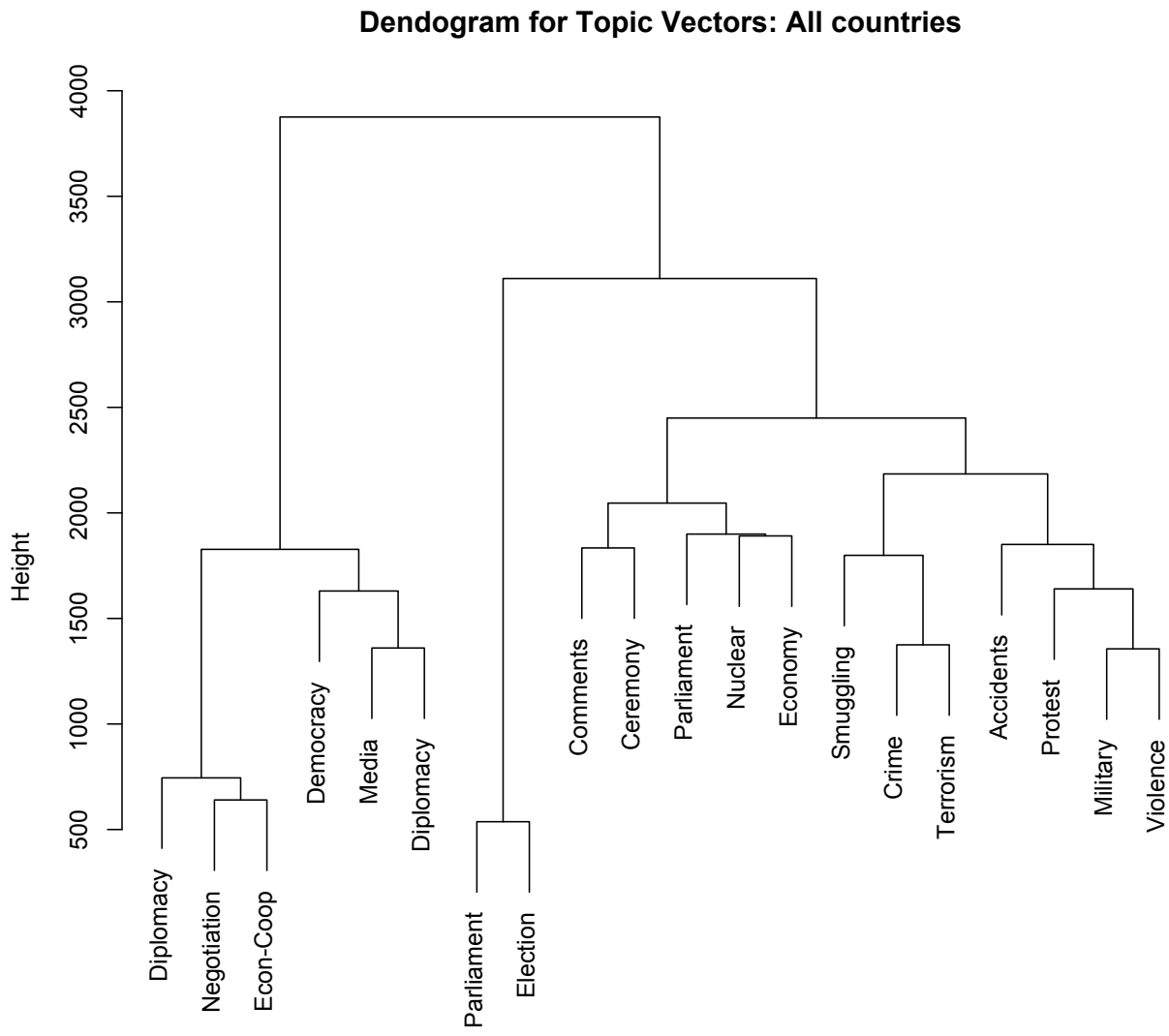
Source: http://csb.stanford.edu/class/public/lectures/lec4/Lecture6/Data_Visualization/images/Hierarchical_Clustering.jpg

Comparison Strategy

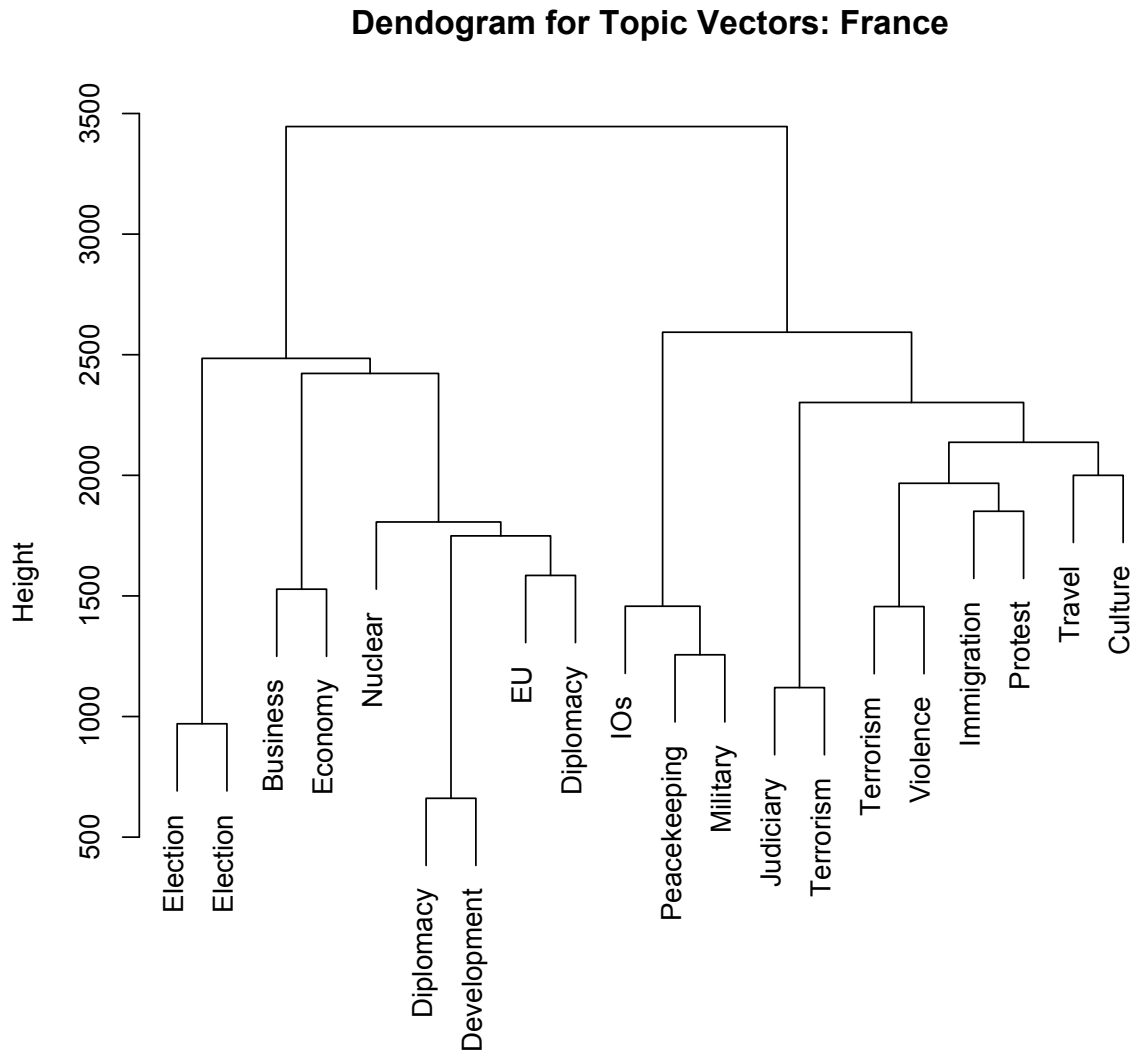
- Words that are similar should co-occur in topics more frequently
- For a pair of ‘top-words’, let their similarity-weight be equal to:
  - No. of times that the pair appears *within* all ‘top-word’ vectors
- Distance between two vectors:
  - A constant minus the sum of the similarity-weights for

    word-pairs that occur *across* the two ‘top-word’ vectors

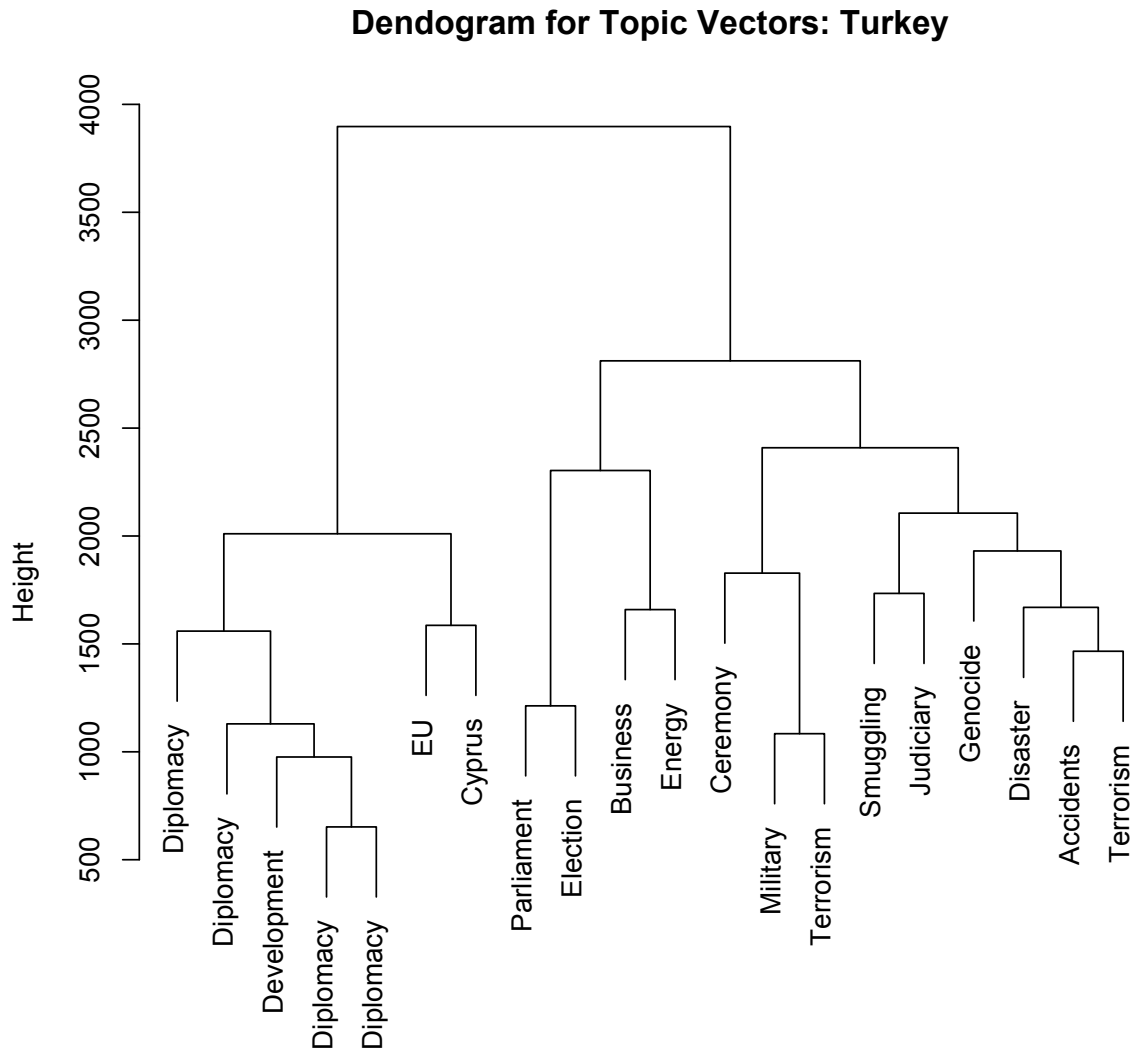
Comparing Topics: Combined Sample



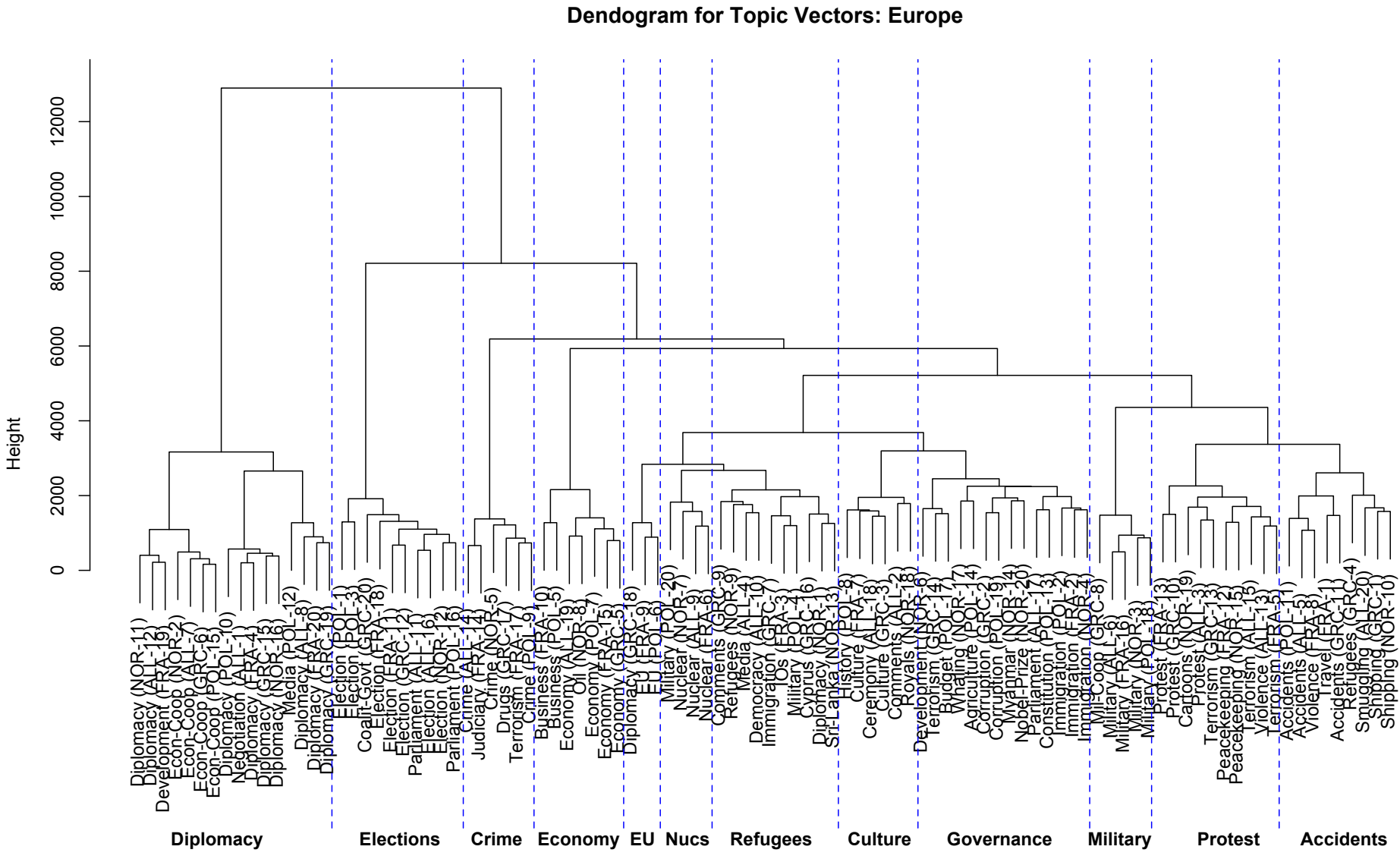
Comparing Topics: France



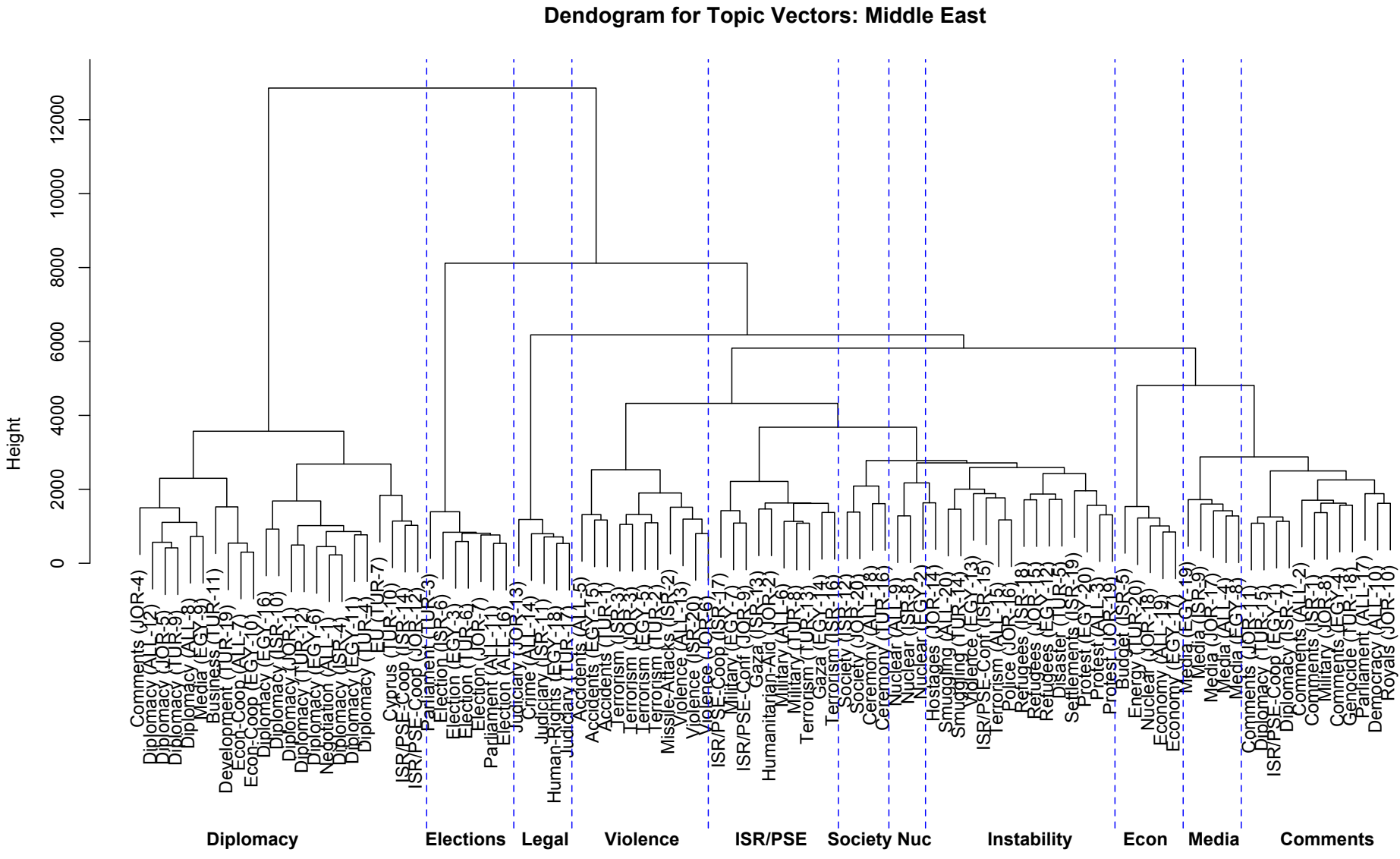
Comparing Topics:Turkey



Comparing Topics across Countries: Europe



Comparing Topics across Countries: Middle East



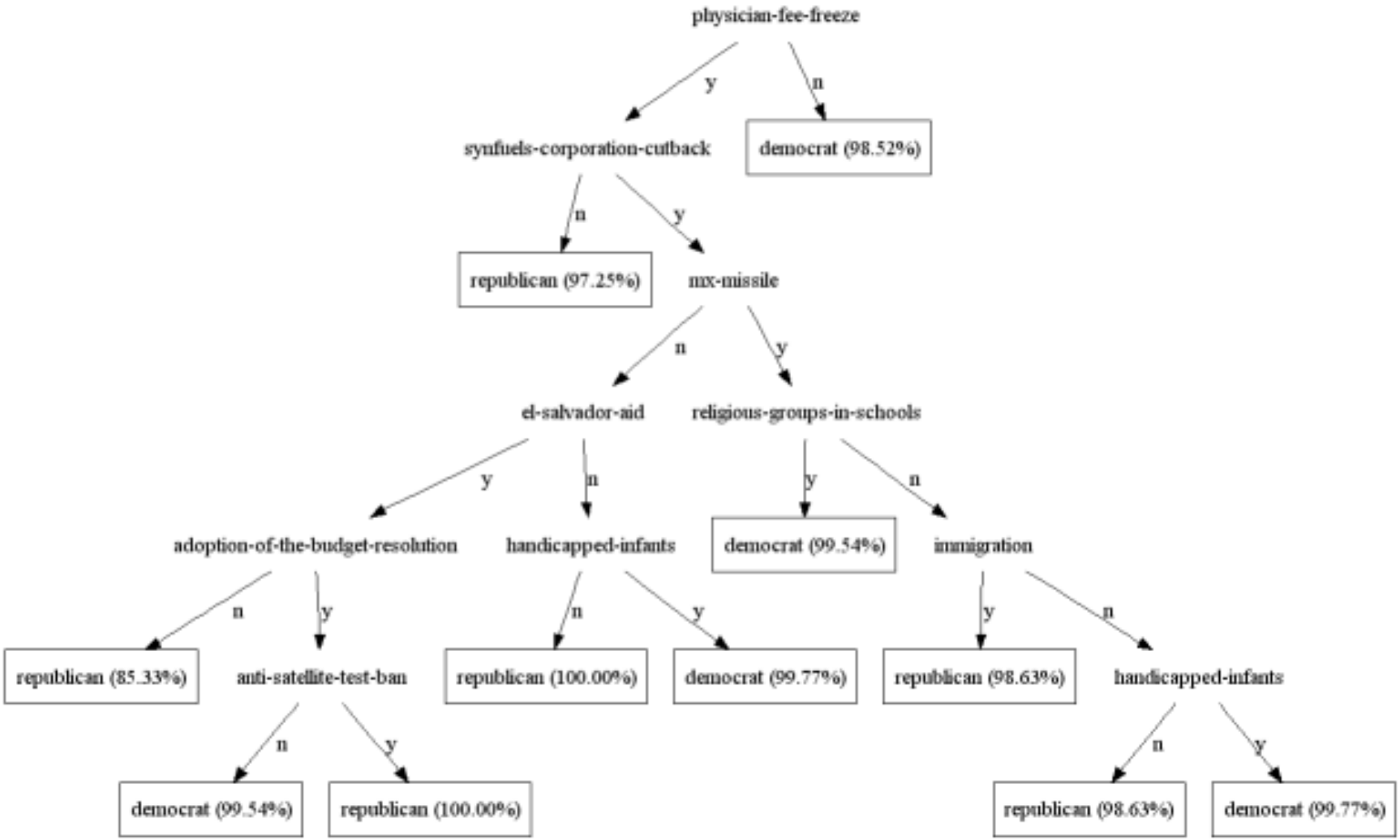
Let’s go exploring!

Google Image Search: “dendograms”

Topics: Module 4

- Clustering
  - K-Means
- Hierarchical Clustering: Dendograms
- Comparisons
  - Generating Dendograms from LDA Topics
- Classification Trees
- Ensemble Methods
  - Bayesian Model Averaging
  - Random Forests™
  - Boosting

Classification Tree Example



Source: http://orange.biolab.si/doc/ofb/c_otherclass.htm

Classification Tree Example

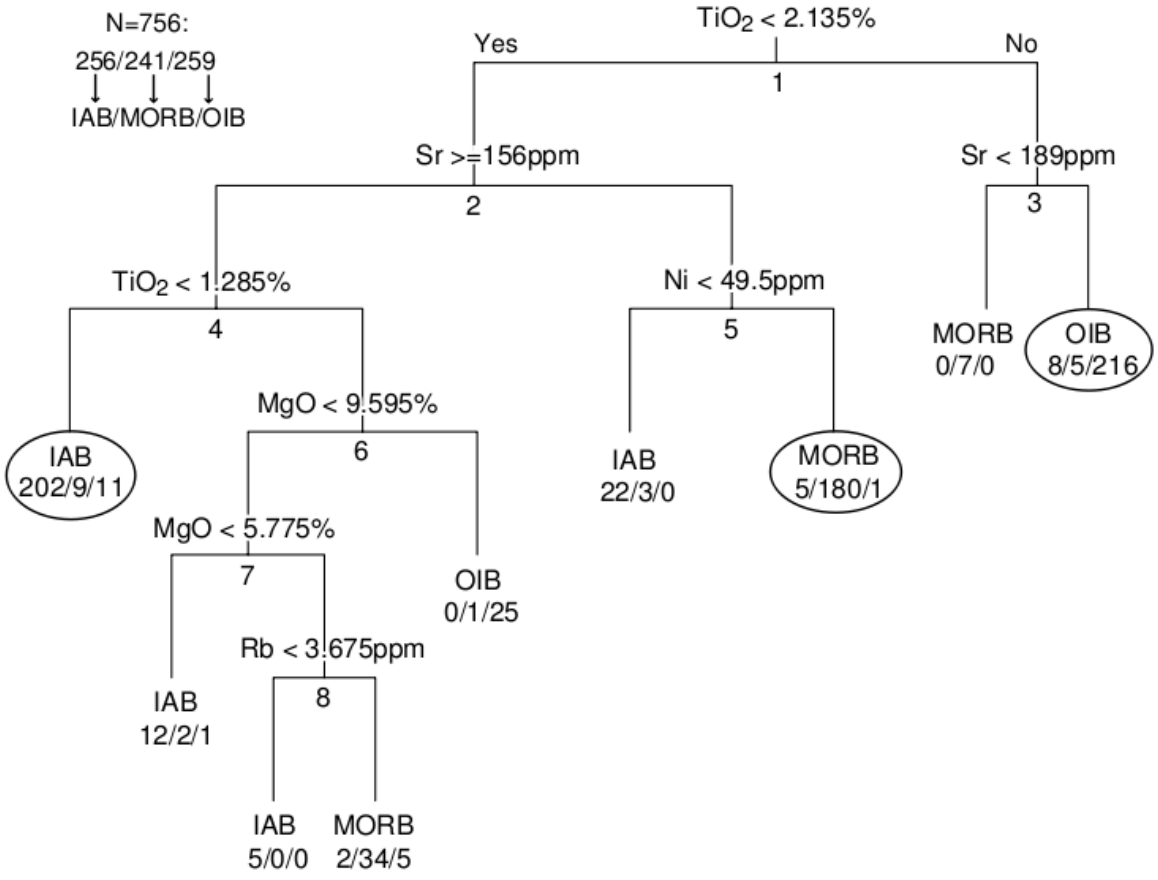
output of running the tree.py script

```
physician-fee-freeze=n: democrat (98.52%)
physician-fee-freeze=y
|    synfuels-corporation-cutback=n: republican (97.25%)
|    synfuels-corporation-cutback=y
|    |    mx-missile=n
|    |    |    el-salvador-aid=y
|    |    |    |    adoption-of-the-budget-resolution=n: republican (85.33%)
|    |    |    |    adoption-of-the-budget-resolution=y
|    |    |    |    |    anti-satellite-test-ban=n: democrat (99.54%)
|    |    |    |    |    anti-satellite-test-ban=y: republican (100.00%)
|    |    |    el-salvador-aid=n
|    |    |    |    handicapped-infants=n: republican (100.00%)
|    |    |    |    handicapped-infants=y: democrat (99.77%)
|    |    mx-missile=y
|    |    |    religious-groups-in-schools=y: democrat (99.54%)
|    |    |    religious-groups-in-schools=n
|    |    |    |    immigration=y: republican (98.63%)
|    |    |    |    immigration=n
|    |    |    |    |    handicapped-infants=n: republican (98.63%)
|    |    |    |    |    handicapped-infants=y: democrat (99.77%)
```

Let's go exploring!

Google Image Search: "classification tree"

Classification Tree with Continuous Breakpoints



[this has something to do with classifying basalts]

Source: http://www.ucl.ac.uk/∼ucfbpve/papers/VermeeschGCA2006/W3441-rev37x.png

ID3 Algorithm

- Calculate the entropy of every attribute using the data set S
- Split the set S into subsets using the attribute for which entropy is minimum (or, equivalently, information gain is maximum)
- Make a decision tree node containing that attribute
- Recurse on subsets using remaining attributes

Source: http://en.wikipedia.org/wiki/ID3_algorithm

Entropy: definition

Definition [edit]

Named after Boltzmann's H-theorem, Shannon denoted the entropy H of a discrete random variable X with possible values $\{x_1, ..., x_n\}$ and probability mass function $P(X)$ as,

$$H(X) = E[I(X)] = E[-\ln(P(X))].$$

Here E is the expected value operator, and I is the information content of X.[8][9] $I(X)$ is itself a random variable.

When taken from a finite sample, the entropy can explicitly be written as

$$H(X) = \sum_i P(x_i)\, I(x_i) = -\sum_i P(x_i) \log_b P(x_i) = -\sum_i \frac{n_i}{N} \log_b \frac{n_i}{N} = \log_b N - \frac{1}{N} \sum_i n_i \log_b n_i,$$

where b is the base of the logarithm used. Common values of b are 2, Euler's number e, and 10, and the unit of entropy is bit for $b = 2$, nat for $b = e$, and dit (or digit) for $b = 10$.[10]

In the case of $p(x_i) = 0$ for some i, the value of the corresponding summand $0 \log_b(0)$ is taken to be 0, which is consistent with the well-known limit:

$$\lim_{p \to 0+} p \log(p) = 0.$$

Source: http://en.wikipedia.org/wiki/Entropy_%28information_theory%29

C4.5 Algorithm

C4.5 builds decision trees from a set of training data in the same way as ID3, using the concept of information entropy. The training data is a set $S = s_1, s_2, \ldots$ of already classified samples. Each sample s_i consists of a p-dimensional vector $(x_{1,i}, x_{2,i}, \ldots, x_{p,i})$, where the x_j represent attributes or features of the sample, as well as the class in which s_i falls.

At each node of the tree, C4.5 chooses the attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. The splitting criterion is the normalized information gain (difference in entropy). The attribute with the highest normalized information gain is chosen to make the decision. The C4.5 algorithm then recurses on the smaller sublists.

Source: http://en.wikipedia.org/wiki/C4.5_algorithm

C4.5 vs. ID3

C4.5 made a number of improvements to ID3. Some of these are:

- Handling both continuous and discrete attributes: In order to handle continuous attributes, C4.5 creates a threshold and then splits the list into those whose attribute value is above the threshold and those that are less than or equal to it.
- Handling training data with missing attribute values—C4.5 allows attribute values to be marked as ? for missing. Missing attribute values are simply not used in gain and entropy calculations.
- Handling attributes with differing costs.
- Pruning trees after creation—C4.5 goes back through the tree once it's been created and attempts to remove branches that do not help by replacing them with leaf nodes

Source: http://en.wikipedia.org/wiki/C4.5_algorithm

Neural networks

Developed by Geoffrey Hinton, who through the magic of the internet, is here to explain...
https://www.coursera.org/course/neuralnets

Topics: Module 4

- Clustering
  - K-Means
- Hierarchical Clustering: Dendograms
- Comparisons
  - Generating Dendograms from LDA Topics
- Classification Trees
- Ensemble Methods
  - Bayesian Model Averaging
  - Random Forests^{TM}
  - Boosting

Bayesian Model Averaging

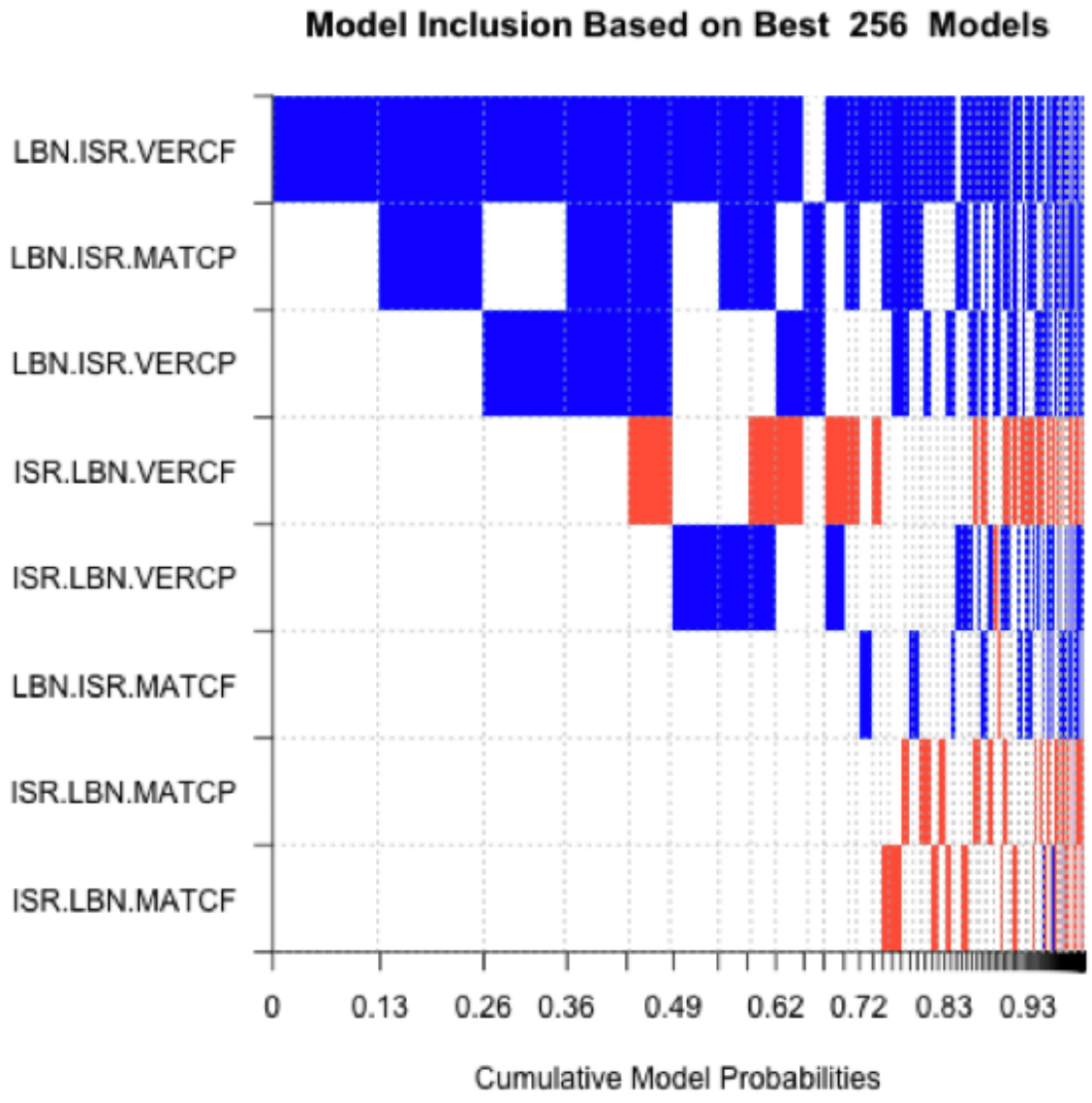
- Systematically integrates the information provided by all combinations of variables
- Result is the overall posterior probability that a variable is important
  - Without having to generate hundreds of papers and thousands of nonrandomly discarded models
- Machine learning suggests that systematic assessment of models gives about 10% better accuracy with much less information, and completely eliminates the need for vaguely defined indicators
- Predictions can be made using an ensemble of all of the models
  - In meteorology and finance, these models are generally more robust in out-of-sample evaluations
- Framework is Bayesian rather than frequentist, which eliminates a long list of philosophical and interpretive problems with the frequentist approach

The problem of ”controls”

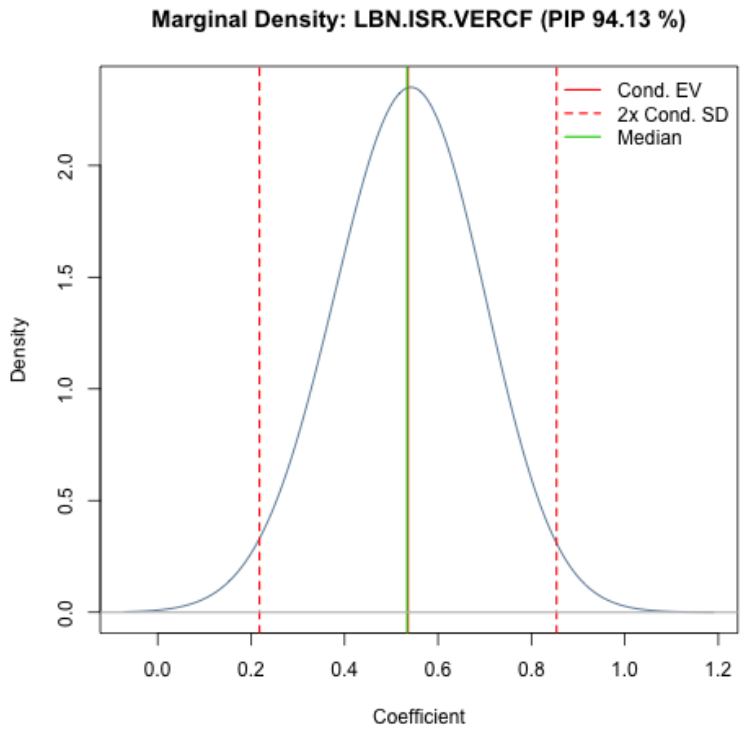
- For starters, they aren't “controls”, they are just another variable
  - Often in a really bad neighborhood
  - Nature bats last in $(X'X)^{-1}X'y$
  - For something closer to a control, use case matching or Bayesian priors
- Numerous studies over the past 50 years—all ignored—have suggested that simple models are better
- In many forecasting models, there is no obvious theoretical reason for using any particular measure, so instead we have to assess multiple measures of the same latent concept: “power”, “legitimacy”, “authoritarianism”
  - This is a feature, not a bug
  - Regression approaches have terrible pathologies in these situations
  - Currently, we laboriously work through all of these options across scores of journal and conference papers presented over the course of years*

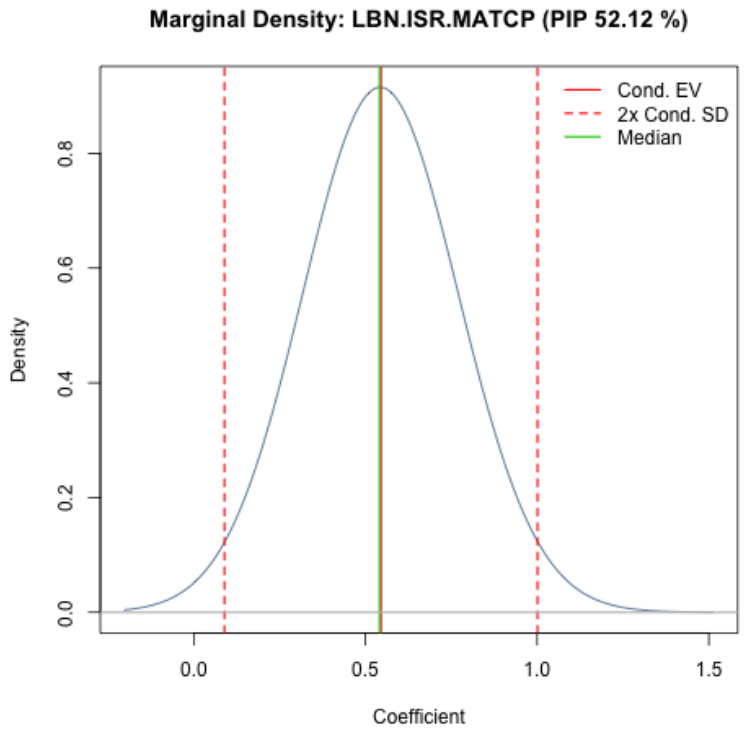
* So if BMA really catches on, a number of journals—and tenure cases—are doomed. On the former, how sad. On the latter, be afraid, be very afraid.

BMA: variable inclusion probabilities



BMA: Posterior probabilities





Random Forests^{TM} : Breiman's Algorithm

Each tree is constructed using the following algorithm:

1. Let the number of training cases be N, and the number of variables in the classifier be M.
2. We are told the number m of input variables to be used to determine the decision at a node of the tree; m should be much less than M.
3. Choose a training set for this tree by choosing n times with replacement from all N available training cases (i.e., take a bootstrap sample). Use the rest of the cases to estimate the error of the tree, by predicting their classes.
4. For each node of the tree, randomly choose m variables on which to base the decision at that node. Calculate the best split based on these m variables in the training set.
5. Each tree is fully grown and not pruned (as may be done in constructing a normal tree classifier).

For prediction a new sample is pushed down the tree. It is assigned the label of the training sample in the terminal node it ends up in. This procedure is iterated over all trees in the ensemble, and the mode vote of all trees is reported as the random forest prediction.

Source: http://en.wikipedia.org/wiki/Random_forest

This sucker is trade-marked!

Random Forests(tm) is a trademark of Leo Breiman and Adele Cutler and is licensed exclusively to Salford Systems for the commercial release of the software.
Our trademarks also include RF(tm), RandomForests(tm), RandomForest(tm) and Random Forest(tm).

For details:
http://www.stat.berkeley.edu/∼breiman/RandomForests/cc_home.htm

Features of Random Forests

Breiman et al claim the following:

- It is unexcelled in accuracy among current algorithms.
- It runs efficiently on large data bases.
- It can handle thousands of input variables without variable deletion.
- It gives estimates of what variables are important in the classification.
- It generates an internal unbiased estimate of the generalization error as the forest building progresses.
- It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data are missing.
- It has methods for balancing error in class population unbalanced data sets.
- Generated forests can be saved for future use on other data.
- Prototypes are computed that give information about the relation between the variables and the classification.
- It computes proximities between pairs of cases that can be used in clustering, locating outliers, or (by scaling) give interesting views of the data.
- The capabilities of the above can be extended to unlabeled data, leading to unsupervised clustering, data views and outlier detection.
- It offers an experimental method for detecting variable interactions.

Random forests^{TM} may also cure acne, remove cat hair from upholstery and show promise for bringing peace to the Middle East, though Breiman et al do not explicitly make these claims.

Source: http://www.stat.berkeley.edu/∼breiman/RandomForests/cc_home.htm#features

AdaBoost

AdaBoost, short for Adaptive Boosting, is a machine learning algorithm, formulated by Yoav Freund and Robert Schapire. It is a meta-algorithm, and can be used in conjunction with many other learning algorithms to improve their performance. AdaBoost is adaptive in the sense that subsequent classifiers built are tweaked in favor of those instances misclassified by previous classifiers. AdaBoost is sensitive to noisy data and outliers. In some problems, however, it can be less susceptible to the overfitting problem than most learning algorithms.

The classifiers it uses can be weak (i.e., display a substantial error rate), but as long as their performance is slightly better than random (i.e. their error rate is smaller than 0.5 for binary classification), they will improve the final model. Even classifiers with an error rate higher than would be expected from a random classifier will be useful, since they will have negative coefficients in the final linear combination of classifiers and hence behave like their inverses.

AdaBoost generates and calls a new weak classifier in each of a series of rounds t = 1,. . . ,T. For each call, a distribution of weights D_t is updated that indicates the importance of examples in the data set for the classification. On each round, the weights of each incorrectly classified example are increased, and the weights of each correctly classified example are decreased, so the new classifier focuses on the examples which have so far eluded correct classification.

Source: http://en.wikipedia.org/wiki/AdaBoost

Go to: adaboost_matas.pdf

http://www.robots.ox.ac.uk/∼az/lectures/cv/adaboost_matas.pdf