START OF DAY 8
Midterm

• Go over questions
• General issues only
• Specific issues: visit with me
• Regrading may make your grade go up OR down
Classification Evaluation
Issues with Accuracy

• Measuring accuracy
  – Is 99% accuracy good? Is 20% accuracy bad?
  – Can be excellent, good, mediocre, poor, terrible

• Why?
  – Depends on problem complexity
  – Depends on base accuracy (i.e., majority learner)
  – Depends on cost of error (e.g., ICU, etc.)

• Problem: assumes equal cost for all errors
## Confusion Matrix

<table>
<thead>
<tr>
<th>True Output (Target)</th>
<th>Predicted Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>True Positive (TP): Hits</td>
</tr>
<tr>
<td>0</td>
<td>False Positive (FP): False Alarm</td>
</tr>
<tr>
<td>1</td>
<td>False Negative (FN): Misses</td>
</tr>
<tr>
<td>0</td>
<td>True Negative (TN): Correct Rejections</td>
</tr>
</tbody>
</table>

### Accuracy

$$\text{Accuracy} = \frac{(\text{TP}+\text{TN})}{(\text{TP}+\text{TN}+\text{FP}+\text{FN})}$$

Single number: loses information
Precision

The percentage of predicted true positives that are target true positives (of those I predict true, how many are actually true)

Precision = \frac{TP}{TP+FP}
Recall

<table>
<thead>
<tr>
<th>True Output (Target)</th>
<th>Predicted Output</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>True Positive (TP) Hits</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>False Negative (FN) Misses</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>False Positive (FP) False Alarm</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>True Negative (TN) Correct Rejections</td>
</tr>
</tbody>
</table>

Recall = TP/(TP+FN)

The percentage of target true positives that were predicted as true positives (of those that are true, how many do I predict are true
P/R Trade-off (I)

• ICU monitoring:
  – Is precision the goal? Not so much, rather not miss any
  – Recall is the goal: Don’t want to miss any, and would rather err towards accepting some false positives (check patient when unnecessary) and minimize false negatives (not check on a needy patient)

• Google search:
  – Is recall the goal? Not really, because we never get to the millionth page, rather get a few very good results early
  – Precision is the goal: Don’t want to see irrelevant documents (false positives), can tolerate missing some (false negatives), there are plenty of sites anyways and we don’t need to get all

• Trade-off:
  – Easy to maximize precision – only classify the one or few most confident candidates as true
  – Easy to maximize recall – classify everything as true
  – Neither is particularly useful!
P/R Trade-off (II)

Complete P/R curve
Breakeven Point defined by P=R

Alternatively, F-measure:

\[
F = \frac{2 \times (\text{PRECISION} \times \text{RECALL})}{\text{PRECISION} + \text{RECALL}}
\]
Other Measures

• Sensitivity (Recall):
  – TP / (TP + FN)

• Specificity:
  – TN / (TN + FP)

• Positive Predictive Value (Precision):
  – TP / (TP + FP)

• Negative Predictive Value:
  – TN / (TN + FN)
ROC Curves and AUC (I)

• Receiver Operating Characteristic Curve
  – Developed in WWII to statistically model false positive and false negative detections of radar operators
• Standard measure in medicine and biology
• Graphs true positive rate (sensitivity) vs. false positive rate (1-specificity)
• Goal: Maximize TPR and minimize FPR
  – Max TPR: classify everything positive
  – Min FPR: classify everything negative
  – Neither is acceptable, of course!
ROC Curves and AUC (II)

Each point on the ROC curve represents a different tradeoff (cost ratio) between TPR and FPR.

AUC is area under the curve: represents performance averaged over all possible cost ratios.

Single summary number

Perfect model has AUC = 1.0
Random model has AUC = 0.5
ROC Properties

• AUC properties
  – 1.0 - Perfect prediction
  – .9 - Excellent
  – .7 - Mediocre
  – .5 - Random

• ROC Curve properties
  – If two ROC curves do not intersect then one method dominates the other
  – If they do intersect then one method is better for some cost ratios, and is worse for others
    • Blue alg better for precision, yellow alg for recall, red neither

• Can choose method and balance based on goals
Lift (I)

• In some situations, we are not interested in the accuracy over the entire data set
  – Accurate predictions for 5%, 10%, or 20% of data
  – Don’t care about the rest

• Prototypical application: direct marketing
  – Baseline: random targeting of population
  – Can we do better?
    • Want to know how much better a targeted offer is on a fraction of the population
Lift (II)

\[
\text{Lift} = \frac{\left[ \frac{\text{TP}}{\text{TP} + \text{TN}} \right]}{\left[ \frac{\text{TP} + \text{FP}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \right]}
\]

How much better a model is over random predictions
Lift (III)

\[
\text{Lift}(t) = \frac{\text{CR}(t)}{t}
\]

E.g., \( \text{Lift (25\%)} = \frac{\text{CR}(25)}{25} = \frac{62}{25} = 2.5 \)

If we select 25\% of prospects using our model, they are 2.5 times more likely to respond than if we selected them randomly.

Can vary \( t \) to make decisions (e.g., cost/benefit analysis)
Summary

• Several measures
  – Single value vs. range of thresholds
• Restricted to binary classification
  – Could always cast problem as a set of two class problems but that can be inconvenient
• Accuracy handles multi-class outputs
• Key point:
  – The measure you optimize makes a difference
  – The measure you report makes a difference
  – Measure what you want to optimize/report (i.e., use measure appropriate to task/domain)
Clustering
What is Clustering?

- Unsupervised learning
- Seeks to organize data into “reasonable” groups
- Often based on some similarity (or distance) measure defined over data elements
- Quantitative characterization may include
  - Centroid / Medoid
  - Radius
  - Diameter
  - Compactness
  - Etc.
Clustering Taxonomy

• Partitional methods:
  – Algorithm produces a single partition or clustering of the data elements

• Hierarchical methods:
  – Algorithm produces a series of nested partitions, each of which represents a possible clustering of the data elements

• Symbolic Methods:
  – Algorithm produces a hierarchy of concepts
Distance-based Clustering
• What is the maximum likelihood hypothesis for the mean of a single Normal distribution given observed instances from it?

• From section 6.4, it is the hypothesis that minimizes the sum of squared errors:

\[
\mu_{ML} = \arg \min_{\mu} \sum_{i=1}^{n} (x_i - \mu)^2
\]
Estimating 1 Mean (II)

• And, in this case, the sum of squared errors is minimized by the sample mean:

$$\mu_{ML} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

What if there are k means to estimate?
Intuition

• Consider $k$ hidden variables
• Each instance is thus extended from $x_i$ to $< x_i, z_{i1}, z_{i2}, ..., z_{ik}>$:
  – $x_i$ is the observed instance
  – $z_{ij}$ are the hidden variables
  – $z_{ij} = 1$ if $x_i$ was generated by the $j$th Gaussian and 0 otherwise
The (EM) $k$-Means Algorithm

- **Initialization:**
  - Set $h = <\mu_1, \mu_2, \ldots, \mu_k>$ where the $\mu_i$’s are arbitrary values

- **Step 1:**
  - Calculate the expected value $E[z_{ij}]$ for each hidden variable $z_{ij}$, assuming the current hypothesis $h$

- **Step 2:**
  - Calculate a new maximum likelihood hypothesis $h' = <\mu'_1, \mu'_2, \ldots, \mu'_k>$, assuming the value taken on by each $z_{ij}$ is its expected value $E[z_{ij}]$ as calculated in Step 1
  - Replace $h$ by $h'$
  - If stopping condition is not met, go to Step 1
The $k$-Means Algorithm

• Instead of considering actual probabilities, we simplify as follows:
  
  – $E[z_{ij}]$ is 1 if $x_i$ is closest to $\mu_j$ and 0 otherwise (i.e., we “threshold” the expected value)
  
  – $\mu_j'$ is the mean of the $x_i$’s for which $E[z_{ij}] = 1$ (i.e., we compute the unweighted center of gravity of those points closest to $\mu_j$ )
**K-means Overview**

- Algorithm builds a single $k$-subset partition
- Works with numeric data only
- Starts with $k$ random centroids
- Uses iterative re-assignment of data items to clusters based on some distance to centroids until all assignments remain unchanged
K-means Algorithm

1) Pick a number, $k$, of cluster centers (at random, do not have to be data items)
2) Assign every item to its nearest cluster center (e.g., using Euclidean distance)
3) Move each cluster center to the mean of its assigned items
4) Repeat steps 2 and 3 until convergence (e.g., change in cluster assignments less than a threshold)
Pick 3 initial cluster centers (randomly)
Assign each point to the closest cluster center.

$k$-Means Example (II)
Move each cluster center to the mean of each cluster.
Reassign points closest to a different new cluster center

Q: Which points are reassigned?
$k$-Means Example (V)

A: three points with animation
$k$-Means Example (VI)

re-compute cluster means
**k-Means Example (VII)**

- **Move cluster centers to cluster means**
- **Re-assign**
- **No change**
$k$-Means Demo

https://www.youtube.com/watch?v=zHbxbb2ye3E&feature=youtu.be
**K-means Discussion**

- Result can vary significantly depending on initial choice of seeds
- Can get trapped in local minimum
  - Example:
    - Restart with different random seeds
- Does not handle outliers well
- Does not scale very well
**K-means Summary**

**Advantages**
- Simple, understandable
- Items automatically assigned to clusters

**Disadvantages**
- Must pick number of clusters beforehand
- All items forced into a cluster
- Sensitive to outliers

Time complexity is $O(mkn)$ where $m$ is # of iterations
$K$-medoids Overview

• Aka Partitioning Around Medoids (PAM)
• Similar to $K$-means:
  – Algorithm builds a single $k$-subset partition
  – Works with numeric data only
  – Uses iterative re-assignment of medoids as long as overall clustering quality improves
• Difference with $K$-means:
  – Starts with $k$ random medoids and sticks to medoids (i.e., data points)
**K-medoids Quality Measures**

- Clustering quality:
  - Sum of all distances from a non-medoid object to the medoid for the cluster it is in (an item is assigned to the cluster represented by the medoid to which it is closest)

- Quality impact:
  - $C_{jih} =$ cost change for item $j$ associated with swapping medoid $i$ for non-medoid $h$
  - Total impact to clustering quality by medoid change ($h$ replaces $i$):

$$TC_{ih} = \sum_{j=1}^{n} C_{jih}$$
**K-medoids Algorithm**

1) Pick a number, $k$, of random data items as medoids

2) Calculate

\[
\arg\min_{(n,m)} \ TC_{mn}
\]

3) If $TC_{mn} < 0$, replace $m$ by $n$ and go back to 2

4) Assign every item to its nearest medoid

The pair $(n,m)$ of medoid/non-medoid with the smallest impact on clustering quality
\textbf{K-medoids Example (I)}

Assume $k=2$
Select X5 and X9 as medoids

\begin{tabular}{c|cc}
\textbf{X1} & 2 & 6 \\
\textbf{X2} & 3 & 4 \\
\textbf{X3} & 3 & 8 \\
\textbf{X4} & 4 & 7 \\
\textbf{X5} & 6 & 2 \\
\textbf{X6} & 6 & 4 \\
\textbf{X7} & 7 & 3 \\
\textbf{X8} & 7 & 4 \\
\textbf{X9} & 8 & 5 \\
\textbf{X10} & 7 & 6 \\
\end{tabular}

\textbf{Distances to X5 to X9}

\begin{tabular}{c|cc}
\textbf{X1} & 6 & 7 \\
\textbf{X2} & 5 & 6 \\
\textbf{X3} & 9 & 8 \\
\textbf{X4} & 7 & 6 \\
\textbf{X6} & 2 & 3 \\
\textbf{X7} & 2 & 3 \\
\textbf{X8} & 3 & 2 \\
\textbf{X10} & 5 & 2 \\
\end{tabular}

Current clustering: \{X1,X2,X5,X6,X7\},\{X3,X4,X8,X9,X10\}
**K-medoids Example (II)**

Must try to replace X5 by X1, X2, X3, X4, X6, X7, X8, X10
Must try to replace X9 by X1, X2, X3, X4, X6, X7, X8, X10

<table>
<thead>
<tr>
<th>Repl. X5 by X1</th>
<th>Bef</th>
<th>Aft</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>6</td>
<td>0</td>
<td>-6</td>
</tr>
<tr>
<td>X2</td>
<td>5</td>
<td>3</td>
<td>-2</td>
</tr>
<tr>
<td>X3</td>
<td>8</td>
<td>3</td>
<td>-5</td>
</tr>
<tr>
<td>X4</td>
<td>6</td>
<td>3</td>
<td>-3</td>
</tr>
<tr>
<td>X5</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>X6</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>X7</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>X8</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>X9</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>X10</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Repl. X5 by X2</th>
<th>Bef</th>
<th>Aft</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>6</td>
<td>3</td>
<td>-3</td>
</tr>
<tr>
<td>X2</td>
<td>5</td>
<td>0</td>
<td>-5</td>
</tr>
<tr>
<td>X3</td>
<td>8</td>
<td>4</td>
<td>-4</td>
</tr>
<tr>
<td>X4</td>
<td>6</td>
<td>4</td>
<td>-2</td>
</tr>
<tr>
<td>X5</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>X6</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>X7</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>X8</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>X9</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>X10</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

| Replace X5 by X4: | -9 |
| Replace X5 by X6: | -5 |
| Replace X5 by X7: | 0  |
| Replace X5 by X8: | -1 |
| Replace X5 by X10: | 1  |

<table>
<thead>
<tr>
<th>Repl. X5 by X3</th>
<th>Bef</th>
<th>Aft</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>6</td>
<td>3</td>
<td>-3</td>
</tr>
<tr>
<td>X2</td>
<td>5</td>
<td>3</td>
<td>-2</td>
</tr>
<tr>
<td>X3</td>
<td>8</td>
<td>0</td>
<td>-8</td>
</tr>
<tr>
<td>X4</td>
<td>6</td>
<td>2</td>
<td>-4</td>
</tr>
<tr>
<td>X5</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>X6</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>X7</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>X8</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>X9</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>X10</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

⇒ Replace X5 by X3
**K-medoids Example (III)**

X3 and X9 are new medoids

<table>
<thead>
<tr>
<th>Distances</th>
<th>to X3</th>
<th>to X9</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>X2</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>X4</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>X5</td>
<td>9</td>
<td>5</td>
</tr>
<tr>
<td>X6</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>X7</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>X8</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>X10</td>
<td>6</td>
<td>2</td>
</tr>
</tbody>
</table>

Current clustering: {X1,X2,X3,X4},{X5,X6,X7,X8,X9,X10}

No change in medoids yields better quality ⇒ DONE!

(I think)
K-medoids Discussion

• As in K-means, user must select the value of $k$, but the resulting clustering is independent of the initial choice of medoids
• Handles outliers well
• Does not scale well
  – CLARA and CLARANS improve on the time complexity of K-medoids by using sampling and neighborhoods
K-medoids Summary

Advantages
• Simple, understandable
• Items automatically assigned to clusters
• Handles outliers

Disadvantages
• Must pick number of clusters beforehand
• High time complexity
Hierarchical Clustering

• Two Approaches
  – Agglomerative
    • Each instance is initially its own cluster. Most similar instance/clusters are then progressively combined until all instances are in one cluster. Each level of the hierarchy is a different set/grouping of clusters.
  – Divisive:
    • Start with all instances as one cluster and progressively divide until all instances are their own cluster. You can then decide what level of granularity you want to output.
Hierarchical Agglomerative Clustering

1. Assign each data item to its own cluster
2. Compute pairwise distances between clusters
3. Merge the two closest clusters
4. If more than one cluster is left, go to step 2
Cluster Distances

• Complete-link
  – Maximum pairwise distance between the items of two different clusters

• Single-link
  – Minimum pairwise distance between the items of two different clusters

• Average-link
  – Average pairwise distance between the items of two different clusters
HAC Example

<table>
<thead>
<tr>
<th></th>
<th>P₁</th>
<th>P₂</th>
<th>P₃</th>
<th>P₄</th>
<th>P₅</th>
<th>P₆</th>
<th>P₇</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₁</td>
<td>0</td>
<td>9</td>
<td>100</td>
<td>102</td>
<td>25</td>
<td>25</td>
<td>75</td>
</tr>
<tr>
<td>P₂</td>
<td>0</td>
<td>105</td>
<td>107</td>
<td>26</td>
<td>30</td>
<td>90</td>
<td></td>
</tr>
<tr>
<td>P₃</td>
<td>0</td>
<td>4</td>
<td>101</td>
<td>99</td>
<td>35</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P₄</td>
<td>0</td>
<td>105</td>
<td>103</td>
<td>39</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P₅</td>
<td>0</td>
<td>4</td>
<td>56</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P₆</td>
<td>0</td>
<td>54</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P₇</td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Assume single-link
HAC Demo

http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/AppletH.html
HAC Discussion

• No need to specify number of clusters

• Still need to know when to stop:
  • Too early ⇒ clustering too fine
  • Too late ⇒ clustering too coarse

• Trade one parameter \((K)\) for another (distance threshold)?

* May also be done after the dendrogram is built
Picking “the” Threshold

• Guessing (sub-optimal)
• Looking for “jumps” in the distance function (subjective)
• Human examination (expensive, unreasonable)
• Semi-supervised learning
  – Must-link, cannot-link constraints
  – Stated explicitly or implicitly (e.g., through labels)
Simple Solution

- Select random sample $S$ of items
- Label items in $S$
- Cluster $S$
- Find the threshold value $T$ that maximizes some clustering quality measure on $S$
- Cluster complete dataset up to $T$

$|S|=50$ was shown to give reasonable results
HAC Summary

• Best implementation is $O(n^2\log n)$
• All clusterings in one pass
• Impact of cluster distance
  – Single link – can lead to long chained clusters where some points are quite far from each other
  – Complete link – finds more compact clusters
  – Average link – used less because have to re-compute the average each time
Symbolic Clustering
Symbolic approach to category formation.

Uses global quality metrics to determine number of clusters, depth of hierarchy, and category membership of new instances.

Categories are probabilistic. Instead of category membership being defined as a set of attribute values that must be matched by an object, COBWEB represents the probability with which each attribute value is present.

Incremental algorithm. Any time a new instance is presented, COBWEB considers the overall quality of either placing it in an existing category or modifying the hierarchy to accommodate it.
Consider the following quantity:

\[
\sum_k \sum_i \sum_j P(A_i = v_{ij})P(C_k \mid A_i = v_{ij})P(A_i = v_{ij} \mid C_k)
\]

Intuitively, this quantity balances intra-class similarity and inter-class differences:

- \(P(A_i = v_{ij} \mid C_k)\) is the probability that an object has value \(v_{ij}\) for attribute \(A_i\) given that the object belongs to category \(C_k\). The greater this probability, the more likely two objects in a category share the same attribute values.
- \(P(C_k \mid A_i = v_{ij})\) is the probability with which an object belongs to category \(C_k\) given that it has value \(v_{ij}\) for attribute \(A_i\). The greater this probability, the less likely objects not in the category will have those attribute values.
- \(P(A_i = v_{ij})\) serves as a weight. It ensures that frequently occurring attribute values exert a stronger influence.
Using Bayes’ rule, we have
\[ P(A_i = v_{ij})P(C_k \mid A_i = v_{ij}) = P(C_k)P(A_i = v_{ij} \mid C_k) \]
so that, by substitution, the above quantity becomes
\[ \sum_k P(C_k) \sum_i \sum_j P(A_i = v_{ij} \mid C_k)^2 \]
And finally, Category Utility is defined as the increase in the expected number of attribute values that can be correctly guessed given a partition (into \( n \) categories) over the expected number of correct guesses without that knowledge.

\[ CU = \frac{\sum_k P(C_k)[\sum_i \sum_j P(A_i = v_{ij} \mid C_k)^2 - \sum_i \sum_j P(A_i = v_{ij})^2]}{n} \]
Tree Representation

- Each node stores:
  1. Its probability of occurrence, \( P(C_k) \) (\( = \text{num. instances at node} / \text{total num. instances} \))
  2. All possible values of every attribute observed in the instances, and for each such value, its predictability \( P(A_i = v_{ij} | C_k) \)

- Leaf nodes correspond to observed instances.
- All links are “is-a” links (i.e., no test on attribute values).
- Tree is initialized with a single node whose probabilities are those of the first instance.
- For each subsequent instance \( I \), Cobweb(\( \text{Root}, I \)) is invoked.
Algorithm Cobweb(Node, Instance)

If Node is a leaf
    Create 2 children, $L_1$ and $L_2$ of Node, with $P(C_{L_1}) = P(C_{L_2}) = 0.5$
    Set the probabilities of $L_1$ to those of Node
    Initialize the probabilities of $L_2$ to those of Instance
    Add Instance to Node, updating Node’s probabilities
Else
    Add Instance to Node, updating Node’s probabilities
    For each child $C$ of Node
        Compute CU of taxonomy obtained by placing Instance in $C$
        Let $S_1$ be the score of the best categorization $C_1$
        Let $S_2$ be the score of the next best categorization $C_2$
        Let $S_3$ be the score of placing Instance in a new category
        Let $S_4$ be the score of merging $C_1$ and $C_2$ into one category
        Let $S_5$ be the score of splitting $C_1$
        If $S_1$ is the best score
            Cobweb($C_1$, Instance)
        Else if $S_3$ is the best score
            Create child node $L$ for the new category
            Initialize the probabilities of $L$ to those of Instance
            Update the category probabilities of $L$’s siblings
        Else is $S_4$ is the best score
            Let $C_m$ be the result of merging $C_1$ and $C_2$
            Cobweb($C_m$, Instance)
        Else if $S_5$ is the best score
            Split $C_1$
            Cobweb(Node, Instance)
COBWEB Demo

http://www-ai.cs.uni-dortmund.de/kdnet/auto?self=$81d91eaae317b2bebb
Discussion

- Nice probabilistic model with no parameters set a priori.
- Only handles nominal features (CLASSIT extends to numerical).
- Sensitive to order of presentation of instances.
- Retains each instance, which may cause problems with noisy data.
Homework: Clustering

END OF DAY 8