Unsupervised Learning and Data Mining

Clustering

Supervised Learning

- Decision trees
- Artificial neural nets
- K-nearest neighbor
- Support vectors
- Linear regression
- Logistic regression
- ...

Supervised Learning

- $F(x)$: true function (usually not known)
- $D$: training sample drawn from $F(x)$

| 87.7 | 59.5 | 159.8 | 129.7 | 86.0 | 100.0 | 60.1 | 100.0 | 100.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 78.4 | 160.1 | 130.4 | 108.4 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 60.1 | 160.1 | 159.8 | 60.1 | 0.0 | 0.1 | 0.0 | 0.1 | 0.0 | 0.1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 18.5 | 159.8 | 130.4 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 84.1 | 130.4 | 85.9 | 9.8 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 90.1 | 130.4 | 96.2 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 40.5 | 159.8 | 35.9 | 32.0 | 0.1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 74.8 | 130.4 | 18.2 | 38.2 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 75.1 | 140.1 | 32.5 | 86.6 | 0.0 | 0.1 | 0.0 | 0.0 | 0.1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
### Supervised Learning

- **F(x):** true function (usually not known)
- **D:** training sample drawn from F(x)

| 77,M,195,0.125,95,39,0.1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 | 0 |
| 78,M,160,1,130,100,37,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 | 1 |
| 69,F,180,0,115,85,40,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 | 0 |
| 18,M,165,0,110,80,44,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 | 0 |
| 54,F,135,0,115,95,39,35,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 | 1 |

- **G(x):** model learned from training sample D

| 69,M,180,1,130,105,38,20,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 | 0 |

- **Goal:** \( E<(F(x)-G(x))^2> \) is small (near zero) for future samples drawn from F(x)

### Clustering

- **≠** Supervised Learning

### Unsupervised Learning

- **Clustering**

- **=** Unsupervised Learning

### Supervised Learning

- **Well Defined Goal:**

  Learn G(x) that is a good approximation to F(x) from training sample D

- **Know How to Measure Error:**

  Accuracy, RMSE, ROC, Cross Entropy, ...

### Clustering ≠ Supervised Learning
Supervised vs. Unsupervised Learning

**Supervised**
- \( y = F(x) \): true function
- \( D \): labeled training set
- \( D: \{x,y\} \)
- \( y = G(x) \): model trained to predict labels \( D \)
- Goal:
  \[ E<(F(x)-G(x))^2> \approx 0 \]
- Well defined criteria: Accuracy, RMSE, ...

**Unsupervised**
- Generator: true model
- \( D \): unlabeled data sample
- \( D: \{x\} \)
- Learn
  \[ ???????? \]
- Goal:
  \[ ???????? \]
- Well defined criteria:
  \[ ???????? \]

What to Learn/Discover?

- Statistical Summaries
- Generators
- Density Estimation
- Patterns/Rules
- Associations
- Clusters/Groups
- Exceptions/Outliers
- Changes in Patterns Over Time or Location

Goals and Performance Criteria?

- Statistical Summaries
- Generators
- Density Estimation
- Patterns/Rules
- Associations
- Clusters/Groups
- Exceptions/Outliers
- Changes in Patterns Over Time or Location

Clustering
Clustering

Given:
- Data Set D (training set)
- Similarity/distance metric/information

Find:
- Partitioning of data
- Groups of similar/close items

Similarity?

- Groups of similar customers
  - Similar demographics
  - Similar buying behavior
  - Similar health
- Similar products
  - Similar cost
  - Similar function
  - Similar store
  - ...
- Similarity usually is domain/problem specific

Types of Clustering

- Partitioning
  - K-means clustering
  - K-medoids clustering
  - EM (expectation maximization) clustering
- Hierarchical
  - Divisive clustering (top down)
  - Agglomerative clustering (bottom up)
- Density-Based Methods
  - Regions of dense points separated by sparser regions of relatively low density

Types of Clustering

- Hard Clustering:
  - Each object is in one and only one cluster
- Soft Clustering:
  - Each object has a probability of being in each cluster
Two Types of Data/Distance Info

- N-dim vector space representation and distance metric
  - D1: 57, 195, 0, 125, 39, 25, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
  - D2: 78, 160, 1, 130, 100, 37, 40, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
  - Distance (D1,D2) = ?

- Pairwise distances between points (no N-dim space)
  - Similarity/dissimilarity matrix (upper or lower diagonal)
    - Distance: 0 = near, ∞ = far
    - Similarity: 0 = far, ∞ = near

Agglomerative Clustering

- Put each item in its own cluster (641 singletons)
- Find all pairwise distances between clusters
- Merge the two closest clusters
- Repeat until everything is in one cluster
- Hierarchical clustering
- Yields a clustering with each possible # of clusters
- Greedy clustering: not optimal for any cluster size

Merging: Closest Clusters

- Nearest centroids
- Nearest medoids
- Nearest neighbors (shortest link)
- Nearest average distance (average link)
- Smallest greatest distance (maximum link)
- Domain specific similarity measure
  - word frequency, TFIDF, KL-divergence, ...
- Merge clusters that optimize criterion after merge
  - minimum mean_point_happiness

Agglomerative Clustering of Proteins

- Nearest centroids
- Nearest medoids
- Nearest neighbors (shortest link)
- Nearest average distance (average link)
- Smallest greatest distance (maximum link)
- Domain specific similarity measure
  - word frequency, TFIDF, KL-divergence, ...
- Merge clusters that optimize criterion after merge
  - minimum mean_point_happiness
Mean Distance Between Clusters

\[
\text{Mean } _\text{Dist} (c_1, c_2) = \frac{\sum_{i \in c_1} \sum_{j \in c_2} \text{Dist}(i, j)}{|c_1| \cdot |c_2| - 1}
\]

Minimum Distance Between Clusters

\[
\text{Min } _\text{Dist} (c_1, c_2) = \min_{i \in c_1, j \in c_2} \text{Dist}(i, j)
\]

Mean Internal Distance in Cluster

\[
\text{Mean } _\text{Internal } _\text{Dist} (c) = \frac{\sum_{i \in c} \sum_{j \in c, i \neq j} \text{Dist}(i, j)}{|c|(|c| - 1)}
\]

Mean Point Happiness

\[
\mathcal{L}_j = \begin{cases} 
1 & \text{when cluster}(i) = \text{cluster}(j) \\
0 & \text{otherwise}
\end{cases}
\]

\[
\text{Mean } _\text{Happiness} = \frac{\sum_{i \in c} \sum_{j \neq i} \mathcal{L}_j \cdot \text{Dist}(i, j)}{|c| - 1}
\]
Clustering Proteins

Distance Between Helices

- Vector representation of protein data in 3-D space that gives x,y,z coordinates of each atom in helix

- Use a program developed by chemists (fortran) to convert 3-D atom coordinates into average atomic distances in angstroms between aligned helices

- 641 helices = 641 * 640 / 2
  = 205,120 pairwise distances

Agglomerative Clustering of Proteins
**Agglomerative Clustering**

- Greedy clustering
  - once points are merged, never separated
  - suboptimal w.r.t. clustering criterion
- Combine greedy with iterative refinement
  - post processing
  - interleaved refinement

**Computational Cost**
- $O(N^2)$ just to read/calculate pairwise distances
- $N-1$ merges to build complete hierarchy
  - scan pairwise distances to find closest
  - calculate pairwise distances between clusters
  - fewer clusters to scan as clusters get larger
- Overall $O(N^3)$ for simple implementations

**Improvements**
- sampling
- dynamic sampling: add new points while merging
- tricks for updating pairwise distances
K-Means Clustering

- Inputs: data set and k (number of clusters)
- Output: each point assigned to one of k clusters

- K-Means Algorithm:
  - Initialize the k-means
    - assign from randomly selected points
    - randomly or equally distributed in space
  - Assign each point to nearest mean
  - Update means from assigned points
  - Repeat until convergence

K-Means Clustering: Convergence

- Squared-Error Criterion

\[ \text{Squared Error} = \sum_{i \in X} (\text{Dist}(i, \text{mean}(c)))^2 \]

- Converged when SE criterion stops changing
- Increasing K reduces SE - can’t determine K by finding minimum SE
- Instead, plot SE as function of K

K-Means Clustering

- Efficient
  - K << N, so assigning points is O(K*N) < O(N^2)
  - updating means can be done during assignment
  - usually # of iterations << N
  - Overall O(N*K*iterations) closer to O(N) than O(N^2)
- Gets stuck in local minima
  - Sensitive to initialization
- Number of clusters must be pre-specified
- Requires vector space data to calculate means

Soft K-Means Clustering

- Instance of EM (Expectation Maximization)
- Like K-Means, except each point is assigned to each cluster with a probability
- Cluster means updated using weighted average
- Generalizes to Standard_Deviation/Covariance
- Works well if cluster models are known
Soft K-Means Clustering (EM)

- Initialize model parameters:
  - means
  - std devs
  ...
- Assign points probabilistically to each cluster
- Update cluster parameters from weighted points
- Repeat until convergence to local minimum

What do we do if we can’t calculate cluster means?

```
1 2 3 4 5 6 7 8 9 10
1 - d d d d d d d d d
2 - d d d d d d d d d
3 - d d d d d d d d d
4 - d d d d d d d d d
5 - d d d d d d d d d
6 - d d d d d d d d d
7 - d d d d d d d d d
8 - d d d d d d d d d
9 - d d d d d d d d d
```
Cost of K-Means Clustering

- n cases; d dimensions; k centers; i iterations
- compute distance each point to each center: $O(n*d*k)$
- assign each of n cases to closest center: $O(n*k)$
- update centers (means) from assigned points: $O(n*d*k)$
- repeat i times until convergence
- overall: $O(n*d*k*i)$

- much better than $O(n^2)$-$O(n^3)$ for HAC
- sensitive to initialization - run many times
- usually don’t know k - run many times with different k
- requires many passes through data set

Graph-Based Clustering

- 1 scan of database
- early termination, on-line, anytime algorithm yields current best answer

Scaling Clustering to Big Databases

- K-means is still expensive: $O(n*d*k*I)$
- Requires multiple passes through database
- Multiple scans may not be practical when:
  - database doesn’t fit in memory
  - database is very large:
    - $10^4$-$10^8$ (or more) records
    - $>10^2$ attributes
  - expensive join over distributed databases

Goals

- 1 scan of database
- early termination, on-line, anytime algorithm yields current best answer
Scale-Up Clustering?

- Large number of cases (big $n$)
- Large number of attributes (big $d$)
- Large number of clusters (big $c$)