Data Mining
Clustering

Jingpeng Li

Supervised Learning

- F(x): true function (usually not known)
- D: training sample (x, F(x))

57,M,195,0,125,95,39,25,0,1,0,0,0,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0	0
78,M,160,1,130,100,37,40,1,0,0,0,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0	1
69,F,180,0,115,85,40,22,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0	0
18,M,165,0,110,80,41,30,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0	0
54,F,135,0,115,95,39,35,1,1,0,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,1,0,0,0,0	1

- G(x): model learned from D

71,M,160,1,130,105,38,20,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0	?

- Goal: $E[(F(x)-G(x))^2]$ is small (near zero) for future samples

Un-Supervised Learning

Training dataset:

Instance

Input	Output
57,M,195,0,125,95,39,25,0,1,0,0,0,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0	~~0~~
78,M,160,1,130,100,37,40,1,0,0,0,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0	~~1~~
69,F,180,0,115,85,40,22,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0	~~0~~
18,M,165,0,110,80,41,30,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0	~~1~~
54,F,135,0,115,95,39,35,1,1,0,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,1,0,0,0,0	~~1~~
84,F,210,1,135,105,39,24,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0	~~0~~
89,F,135,0,120,95,36,28,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1,0,0	~~1~~
49,M,195,0,115,85,39,32,0,0,0,1,1,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0	~~0~~
40,M,205,0,115,90,37,18,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0	~~0~~
74,M,250,1,130,100,38,26,1,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0	~~1~~
77,F,140,0,125,100,40,30,1,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,1	~~0~~

Test dataset:

~~71,M,160,1,130,105,38,20,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0~~ ?

Un-Supervised Learning

Data set:

57,M,195,0,125,95,39,25,0,1,0,0,0,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0
78,M,160,1,130,100,37,40,1,0,0,0,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0
69,F,180,0,115,85,40,22,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0
18,M,165,0,110,80,41,30,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
54,F,135,0,115,95,39,35,1,1,0,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,1,0,0,0,0
84,F,210,1,135,105,39,24,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0
89,F,135,0,120,95,36,28,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1,0,0
49,M,195,0,115,85,39,32,0,0,0,1,1,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0
40,M,205,0,115,90,37,18,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
74,M,250,1,130,100,38,26,1,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0
77,F,140,0,125,100,40,30,1,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,1

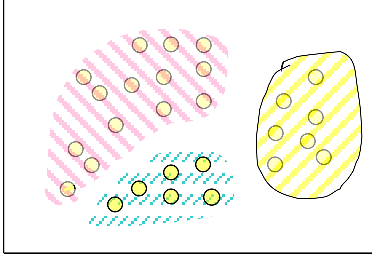
Supervise Vs. Un-Supervised Learning

Supervised
- **y=F(x)**: true function
- **D**: labeled training set
- **D**: $\{x_i, F(x_i)\}$
- **Learn**:
  G(x): model trained to predict labels of new cases
- **Goal**:
  $E[(F(x)-G(x))^2] \approx 0$
- **Well defined criteria**:
  Mean square error

Un-supervised
- **y=?**: no true function
- **D**: unlabeled data set
- **D**: $\{x_i\}$
- **Learn**
  ??????????
- **Goal**:
  ??????????
- **Well defined criteria**:
  ??????????

Clustering (Unsupervised Learning)

- What we have:
  - Data Set D
  - Similarity/distance metric
- What we need to do:
  - Find Partitioning of data, or groups of similar/close items
- An illumination
  - Find "natural" grouping of instances given un-labeled data



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

Similarity: Distance Functions

- Numeric data:
  - Euclidean distance
  - Manhattan distance
- Categorical data (0/1 indicating presence/absence):
  - Hamming distance (# dissimilarity)
  - Jaccard coefficient (% of #similarity in 1s)
- Combined numeric and categorical data:
  - weighted normalized distance

Manhattan & Euclidean Distance

Consider two records $x=(x_1,\dots,x_d)$, $y=(y_1,\dots,y_d)$:

$$d(x,y) = \sqrt[p]{|x_1 - y_1|^p + |x_2 - y_2|^p + \dots + |x_d - y_d|^p}$$

Special cases:

- p=1: Manhattan distance

$$d(x,y) = |x_1 - y_1| + |x_2 - y_2| + \dots + |x_p - y_p|$$

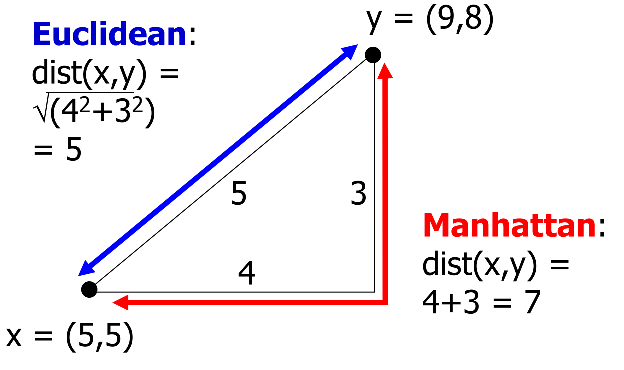
- p=2: Euclidean distance

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \dots + (x_d - y_d)^2}$$

- p=∞: Chebyshev distance

$$d(x,y) = \max(|x_1 - y_1|, |x_2 - y_2|, \dots, |x_p - y_p|)$$

Manhattan & Euclidean Distances: Example



Mean Clustering

- We will look at an approach to clustering numeric data based on picking a number of mean values – one for each cluster
- You hopefully know that the mean (average) of a data set of size S is:

$$\bar{x} = \frac{\sum_{S} x}{S}$$

More Than One Mean

- What if we suspect that our data set is actually a number of data sets mixed together,
- Each one has a mean value of its own
- But we don't know which data point belongs to which set
- Clustering algorithms separate out the data and calculate the means

Mean Clustering Target

- Imagine we think there are 5 clusters in our data
- We want to calculate 5 means: m_1, m_2, m_3, m_4, m_5
- And assign each data point, x_i, to one mean only
- That would lead to 5 data sets, $S_1 \ldots S_5$

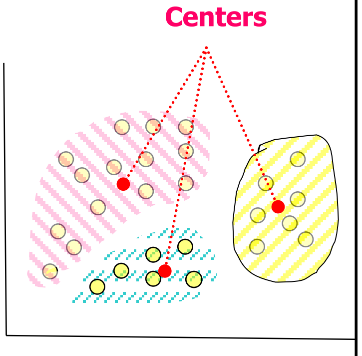
Aim

- Target is to minimise the total distance between the data points and the means to which they are assigned:

$$\arg\min(S) \sum_{i=1}^{k} \sum_{x_j \in s} \left\| x_j - m_i \right\|^2$$

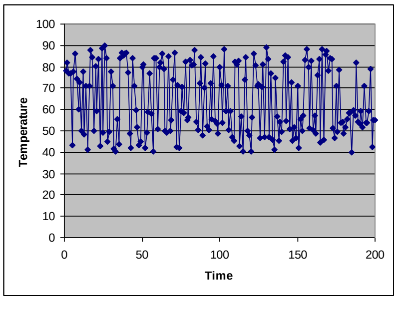
K-Means Clustering Algorithm

- **Goal**: minimize sum of square of distance from all data points to their means
- **Algorithm**:
  - o Pick K different points from the data and assumes they are the cluster centers
    - random, first K, K separated points
  - o Repeat until stabilization:
    - Assign each point to the closest cluster center
    - Generate new cluster centers



K-Means Example

- Imagine a machine that worked in two distinct states, e.g fast and slow
- Mean temperature might be 50 for the slow speed and 80 for the fast speed



K-Means Example

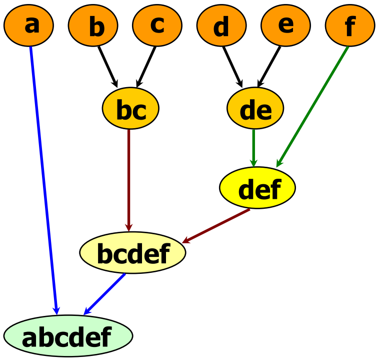
- The machine might have a number of distinct states, all with differing acceptable ranges of temperature, pressure etc
- We don't know what these different states are, nor how many there are of them
- A clustering algorithm will find them – K means does so by finding the middle point of each

K-Means Disadvantages

- Only measures the mean for each cluster – tells you nothing of its shape. You must assume the cluster is round, but they rarely are
- You need to know *K* before you start
- The distance measure, in its simple form, assumes that all ranges are equally important

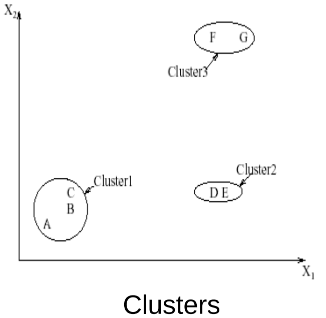
Hierarchical Clustering Algorithm

- **Algorithm:**
  - Start with the same number of clusters as you have data points – every point is a cluster of its own
  - Find the two clusters that are closest together and join them into one.
  - Calculate their new centre
  - Repeat
    - Until you have the desired number of clusters, or
    - everything is in one cluster

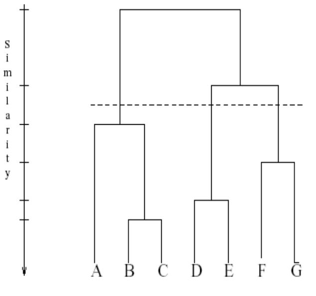


Hierarchical Clustering – Minimum Spanning Tree

- Looks for clusters within clusters
- Cluster 1 (root) is the whole data set
- That splits into a small number of subsets
- Each subset splits into 0 or more subsets etc.



Clusters



Dendrogram

Qualities of a Cluster

- The cluster hierarchy may store other data about its clusters:
  - Population size: how many data points are in that cluster?
  - Variance and range – how far from the centre does most of the data lie

Association Rules

Market Basket Analysis

Consider shopping cart filled with several items. **Market basket analysis** tries to answer the following questions:

- What do customers buy together?
  - 80% of customers purchase items X, Y and Z together (i.e. {X, Y, Z})
- In what pattern do customers purchase items?
  - 60% of customers who purchase X and Y also buy Z (i.e. {X, Y} → {Z})

Association Rule Discovery: Definition

- Giving a set of records, each of which contain some number of items
  - Produce dependency rules, which predict occurrence of some items based on occurrences of other items

Market-basket transactions

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example association rules

{Diaper} → {Beer},
{Beer, Bread} → {Milk},
{Milk, Bread} → {Eggs, Coke}

Frequent Itemset

- **Itemset**
  - A collection of one or more items
    - Example: {Bread, Milk, Diaper}
  - ***k*-itemset**
    - An itemset that contains *k* items
- **Support count (σ)**
  - Frequency of occurrence of an itemset
  - E.g. σ({Bread, Milk, Diaper}) = **2**
- **Support**
  - Fraction of transactions that contain an itemset
  - E.g. s({Milk, Bread, Diaper}) = **2/5**
- **Frequent Itemset**
  - An itemset whose support is greater than or equal to a *minsup* threshold

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Rule Evaluation Metrics

- Association Rule is expressed as X → Y, where X and Y are itemsets. Example: {Milk, Diaper} → {Beer}
- Support (s) = % of transactions that contain both X and Y
  - 40% of customers buy milk, diaper and beer
- Confidence (c) = % of (transactions that contain X) that also contain Y
  - If someone buys milk and diaper, they will buy beer 67% of the time

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example:

$$s = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{|T|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{\sigma(\text{Milk}, \text{Diaper})} = \frac{2}{3} = 0.67$$

Direction

- The direction is important:
- X → Y is not the same as Y → X
- For example,
  - 80% of people who buy a torch buy batteries
  - 5% of people who buy batteries buy a torch

Mining Association Rules

TID	*Items*
1	**Bread, Milk**
2	**Bread, Diaper, Beer, Eggs**
3	**Milk, Diaper, Beer, Coke**
4	**Bread, Milk, Diaper, Beer**
5	**Bread, Milk, Diaper, Coke**

Rules of itemset: {Milk, Diaper, Beer}

{Milk,Diaper} → {Beer} (s=0.4, c=0.67)
{Milk,Beer} → {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} → {Milk} (s=0.4, c=0.67)
{Beer} → {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} → {Milk,Beer} (s=0.4, c=0.5)
{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

Observations:

- Computationally expensive as all the above rules are **binary partitions** of the same itemset: {Milk, Diaper, Beer}
- Rules originating from the same itemset have **identical support** but have different confidence

Finding the Rules

- The apriori algorithm works as follows:
  1. Find all the acceptable itemsets - Support
  2. Use them to generate acceptable rules – Confidence
- So, we find all the itemsets with more than our chosen support and combine them into every possible rule, keeping those with an acceptable confidence

Step 1 – Generate Itemsets

1. Find all the acceptable itemsets of size 1
2. Use the items from step 1 to generate all itemsets of size two and count their support. Keep those that are supported.
3. Repeat for increasingly large itemsets until none of the current size are supported

Example

- With a minimum support of 20%
  - Bread = 40%: Keep
  - Milk = 60%: Keep
  - Porcini = 2%: Discard
- {Bread, Milk} = 30%: Keep
- {Bread, Milk, Sardines} = 15%: Discard
- These are NOT rules yet! Just itemsets

Step 2: Generate Rules

- Generate every combination from the acceptable itemsets:

$$X \rightarrow Y \text{ where } X \cap Y = \text{Empty}$$

- That is, where nothing in X appears in Y, and vice-versa.

Example

- {Bread} $\rightarrow$ {Milk} is good
- {Bread, Milk} $\rightarrow$ {Coffee} is good
- {Bread} $\rightarrow$ {Bread, Milk} is not allowed

Finally

- Discard all the rules that have a confidence score lower than some pre-defined target
- Remember, confidence is the percentage of baskets that contain both parts of the rule