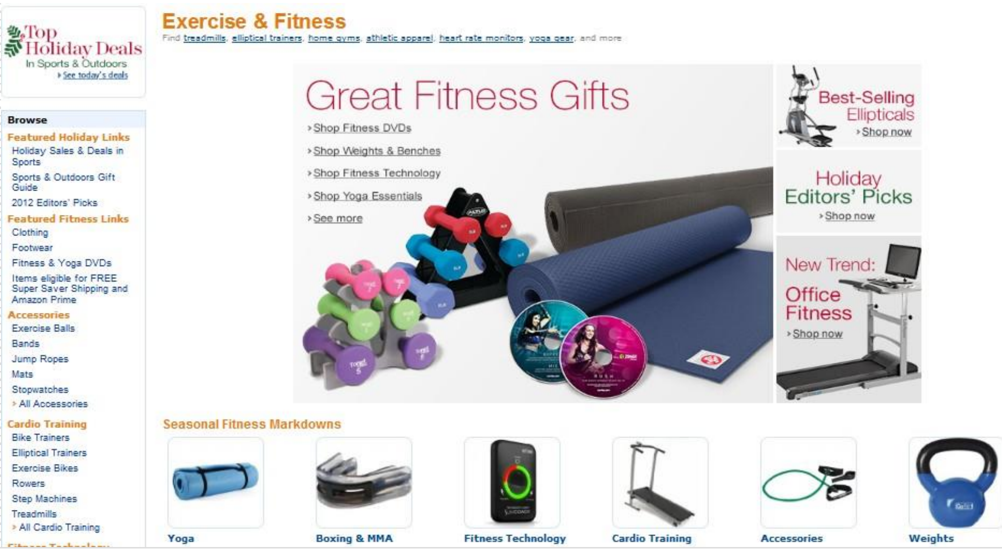
推荐系统技术扩展介绍

Agenda

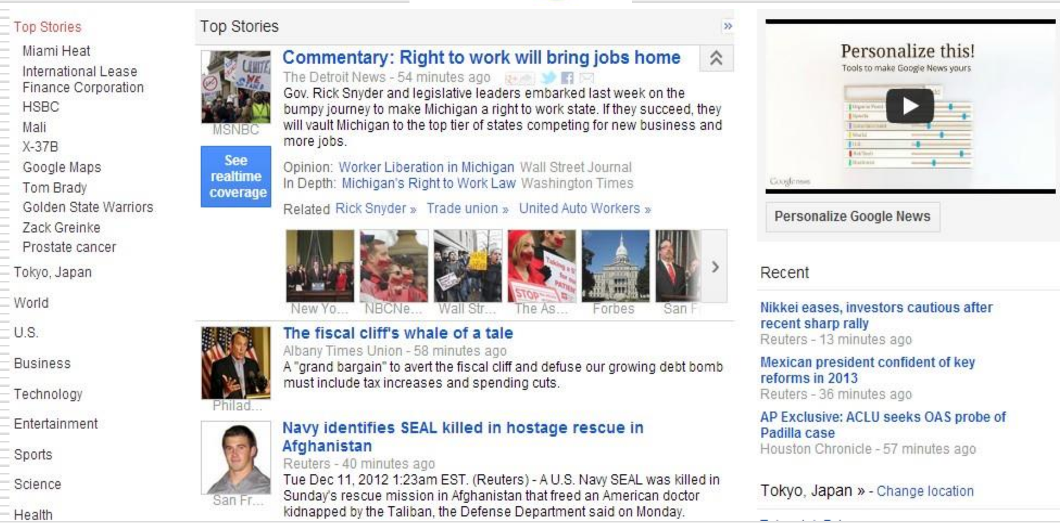
High dim. data	Graph data	Infinite data	Machine learning	Apps
Locality sensitive hashing	PageRank, SimRank	Filtering data streams	SVM	Recommen der systems
Clustering	Community Detection	Web advertising	Decision Trees	Association Rules
Dimensiona lity reduction	Spam Detection	Queries on streams	Perceptron, kNN	Duplicate document detection

Sample Applications

amazon.com

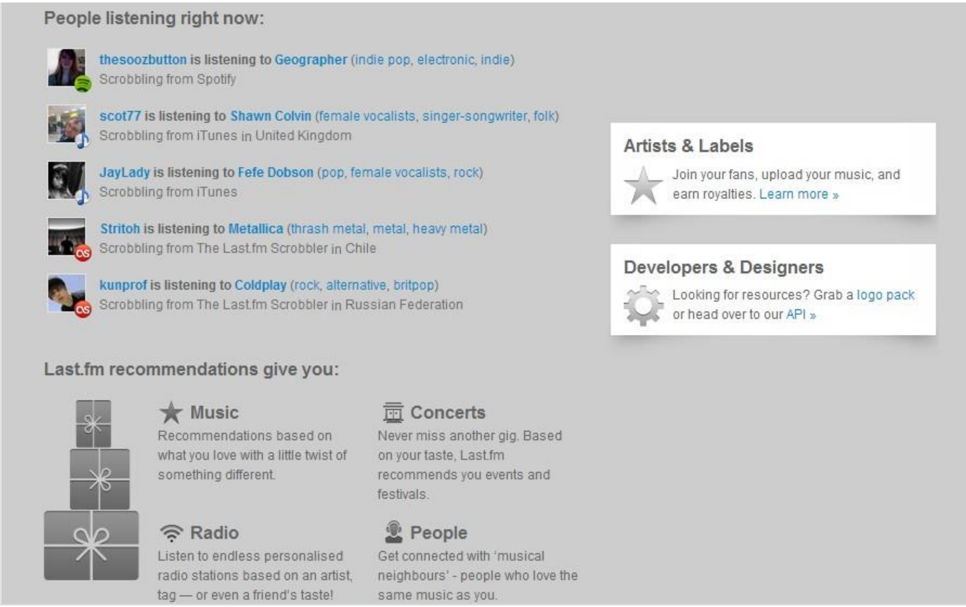


Google

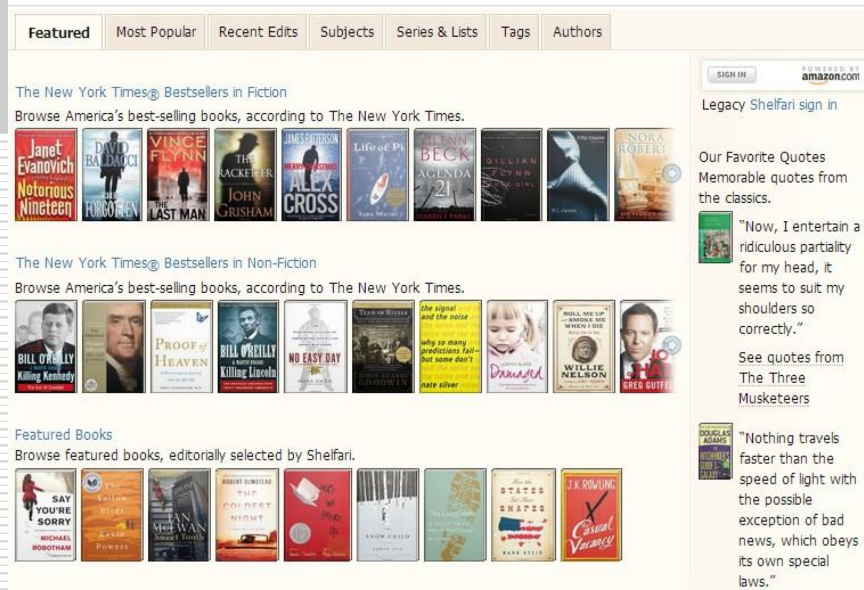


Sample Applications

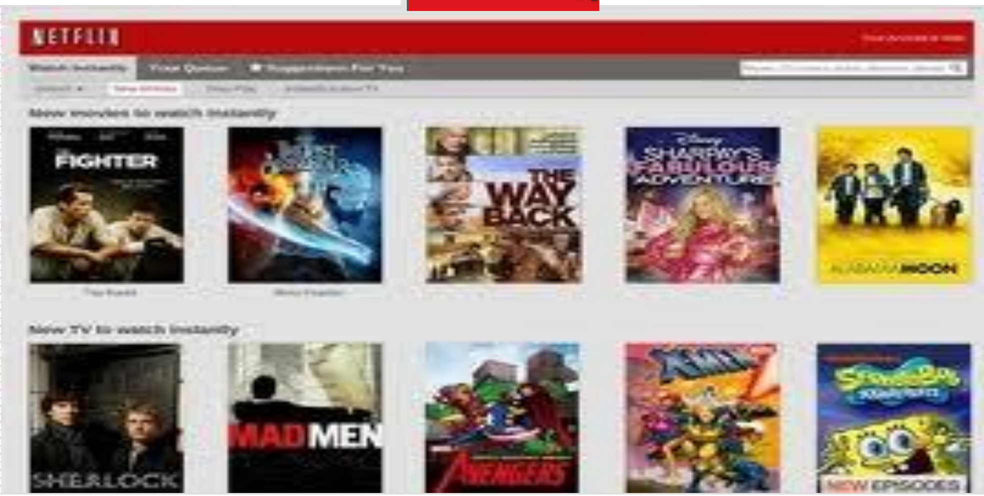
last.fm



shelfari



Sample Applications

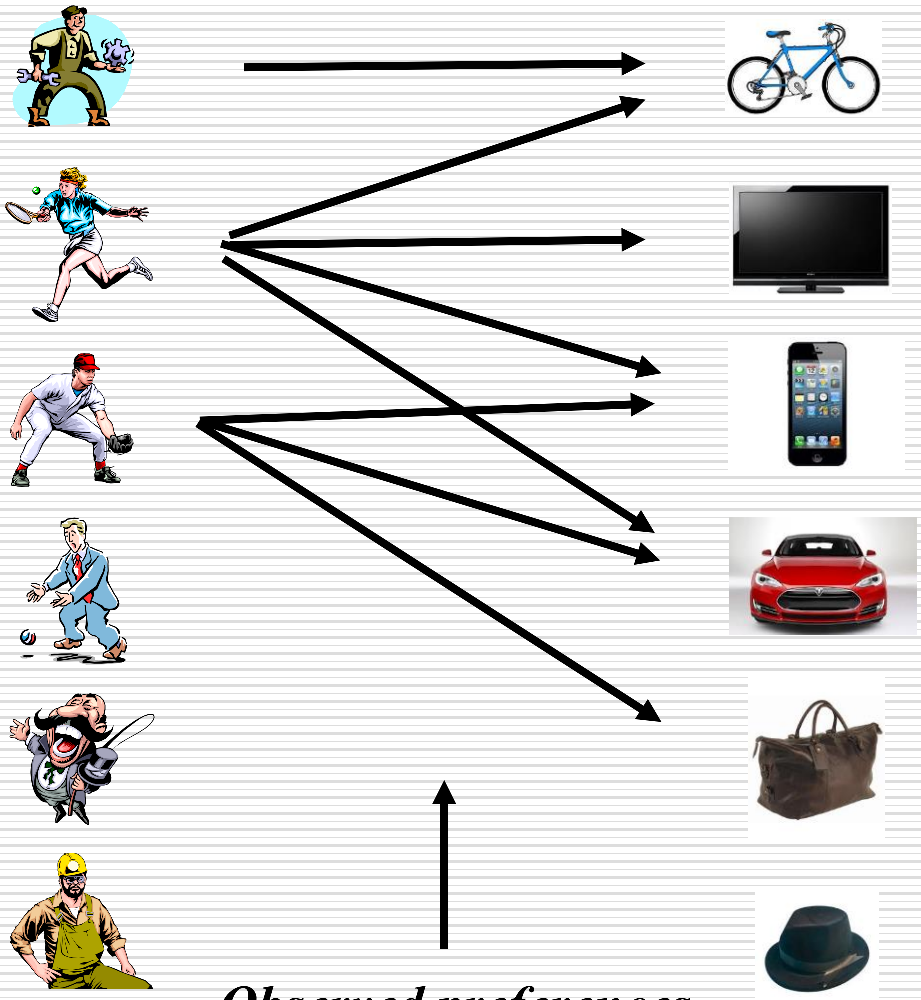


Corporate Intranets

System Inputs

- Interaction data (users ⟷ items)
  - Explicit feedback – rating, comments
  - Implicit feedback – purchase, browsing
- User/Item individual data
  - User side:
    - Structural attribute information
    - Personal description
    - Social network
  - Item side:
    - Structural attribute information
    - Textual description/content information
    - Taxonomy of item (category)

Interaction between Users and Items



Observed preferences

(Purchases, Ratings, page views, bookmarks, etc)

Profiles of Users and Items

User Profile:

(1) Attribute

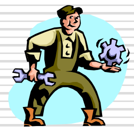
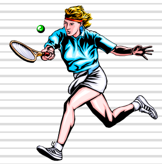
Nationality,Sex,
Age,Hobby,etc

(2) Text

Personal
description

(3) Link

Social network








Item Profile:

(1) Attribute

Price,Weight,Co
lor,Brand,etc

(2) Text

Product
description

(3) link

Taxonomy of
item (category)

All Information about Users and Items

User Profile:

(1) Attribute

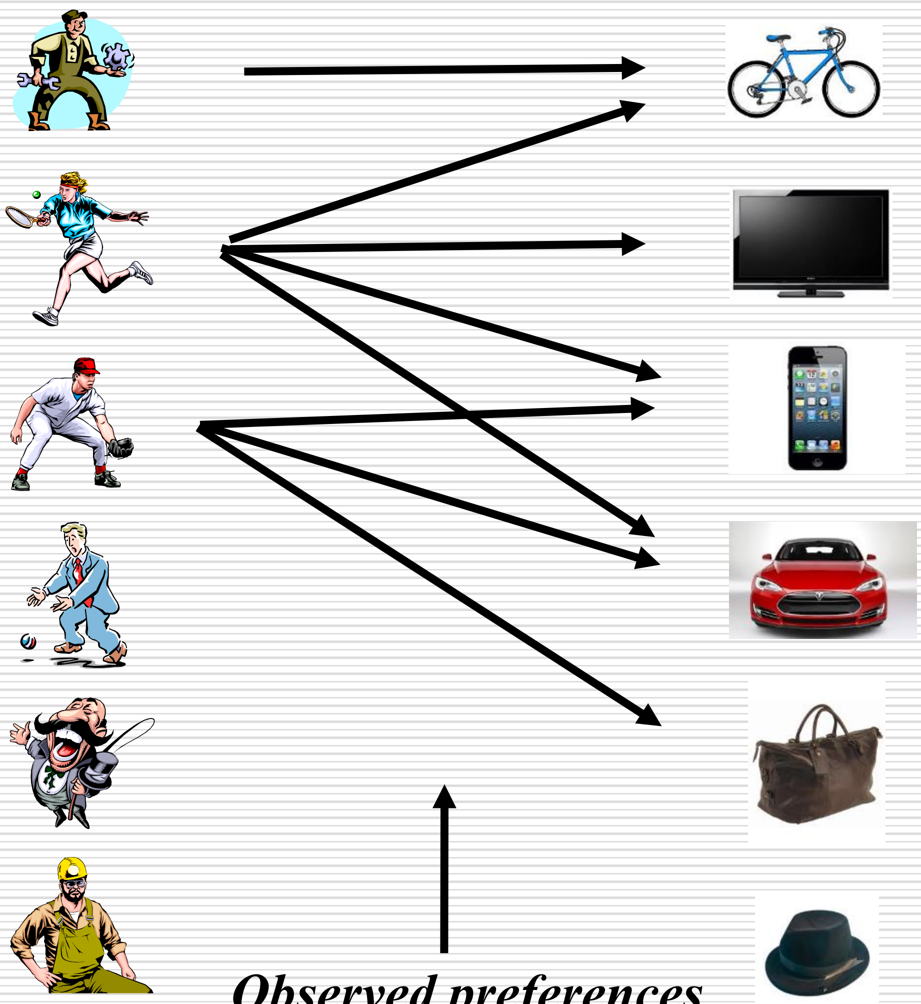
Nationality,Sex, Age,Hobby,etc

(2) Text

Personal description

(3) Link

Social network



Observed preferences

(Purchases, Ratings, page views, bookmarks, etc)

Item Profile:

(1) Attribute

Price,Weight,Color,Brand,etc

(2) Text

Product description

(3) link

Taxonomy of item (category)

Recommendation Approaches

- Collaborative filtering
  - Using **interaction data** (user-item matrix)
  - Process: Identify similar users, extrapolate from their ratings
- Content based strategies
  - Using **profiles of users/items** (features)
  - Process: Generate rules/classifiers that are used to classify new items
- Latent factor based strategies

Recommendation Approaches

- Collaborative filtering
  - Nearest neighbor based
    - User based
    - Item based
- Content based strategies
- Latent factor based strategies

Problems with Collaborative Filtering

- **Cold Start**: There needs to be enough other users already in the system to find a match.
- **Sparsity**: If there are many items to be recommended, even if there are many users, the user/ratings matrix is sparse, and it is hard to find users that have rated the same items.
- **First Rater**: Cannot recommend an item that has not been previously rated.
  - New items
  - Esoteric items
- **Popularity Bias**: Cannot recommend items to someone with unique tastes.
  - Tends to recommend popular items.

Recommendation Approaches

- Collaborative filtering
- Content based strategies
- Latent factor based strategies

Profiles of Users and Items

User Profile:

(1) Attribute

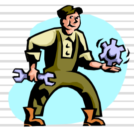
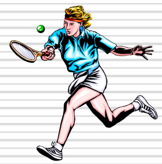
Nationality,Sex, Age,Hobby,etc

(2) Text

Personal description

(3) Link

Social network








Item Profile:

(1) Attribute

Price,Weight,Color,Brand,etc

(2) Text

Product description

(3) link

Taxonomy of item (category)

Advantages of Content-Based Approach

- No need for data on other users.
  - No cold-start or sparsity problems.
- Able to recommend to users with unique tastes.
- Able to recommend new and unpopular items
  - No first-rater problem.
- Can provide explanations of recommended items by listing content-features that caused an item to be recommended.

Recommendation Approaches

- Collaborative filtering
- Content based strategies
  - Text similarity based
  - Clustering
  - Classification
- Latent factor based strategies

Text Similarity based Techniques

- Vector Space Model (VSM)
  - TF-IDF
- Semantic resource based
  - Wordnet
  - Wiki
  - Web

All Information about Users and Items

User Profile:

(1) Attribute

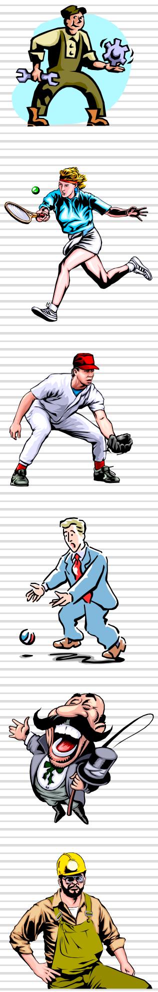
Nationality,Sex, Age,Hobby,etc

(2) Text

Personal description

(3) Link

Social network



Item Profile:

(1) Attribute

Price,Weight,Color,Br and,etc

(2) Text

Product description

(3) link

Associated relation between items (i.e., co-purchased by the same user)

Observed preferences (Purchases, Ratings, page views, play lists, bookmarks, etc)

All Information about Users and Items

User Profile:

(1) Attribute

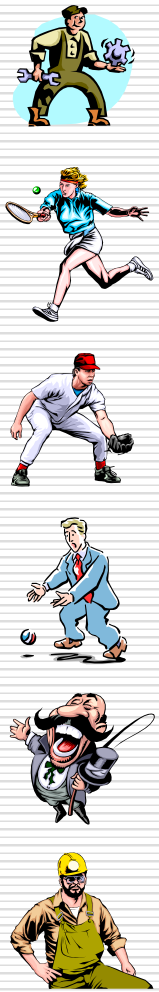
Nationality,Sex, Age,Hobby,etc

(2) Text

Personal description

(3) Link

Social network



I like car, movie, music …

Observed preferences (Purchases, Ratings, page views, play lists, bookmarks, etc)

Item Profile:

(1) Attribute

Price,Weight,Color,Brand,etc

(2) Text

Product description

(3) Link

Associated relation between items (i.e., co-purchased by the same user)

This car is nicely equipped with auto air conditioning …

Profile Representation – Vector Space Model

User Profile

- Structured data
  - attributes: book, car, TV …
- Free text
  - “I like car, movie, music…”

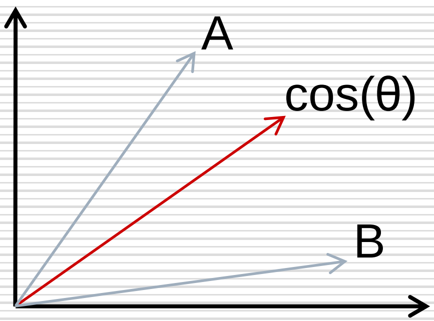
Item Profile

- Structured data
  - attributes: name, color, price …
- Free text
  - “This car is nicely equipped with auto air conditioning…”

	User A	*Item B*
car	1	1
book	0	0
TV	0	0
bike	1	1
…	...	...

Cosine Similarity

$$sim(A,B) = \cos(\theta) = \frac{A \bullet B}{\|A\| \|B\|}$$



Weighted Cosine Similarity

$$sim(A,B) = \frac{\sum_{j=1}^{n} w_{a_j} * w_{b_j}}{\sqrt{\sum_{j=1}^{n} (w_{a_j})^2 * \sum_{j=1}^{n} (w_{b_j})^2}}$$

weight

TF*IDF Weighting

- TF*IDF weighting

$$w(t,d) = tf_{t,d} \times idf_t$$

- Term frequency $tf_{t,d}$ of a term t in a document d

  i.e., $n_{t,d}$ is how many times t is appears in d

$$tf_{t,d} = \frac{n_{t,d}}{\sum_k n_{k,d}}$$

- Inverse document frequency idf_t of a term t

  i.e., df_t how many times t is appears in all documents

$$idf_t = \log\left(\frac{N}{df_t}\right)$$

  where N is the number of all documents

Profile Representation

- Unstructured data
  - e.g., text description or review of the restaurant, or news articles
  - No attribute names with well-defined values
  - Natural language complexity
    - Same word with different meanings
    - Different words with same meaning
- Need to impose structure on free text before it can be used in recommendation algorithm

All Information about Users and Items

User Profile:

(1) Attribute

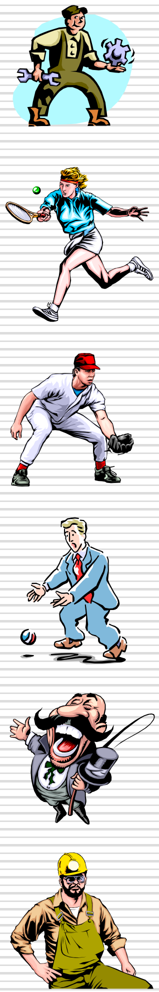
Nationality,Sex, Age,Hobby,etc

(2) Text

Personal description

(3) Link

Social network



Observed preferences (Purchases, Ratings, page views, play lists, bookmarks, etc)

Item Profile:

(1) Attribute

Price,Weight,Color,Brand,etc

(2) Text

Product description

(3) Link

Associated relation between items (i.e., co-purchased by the same user)

I like automobile, movie, music …

This car is nicely equipped with auto air conditioning …

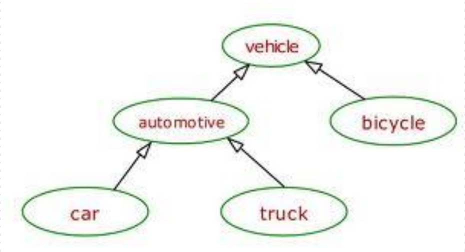
Text Similarity based Techniques

- Vector Space Model (VSM)
  - TF-IDF
- Semantic resource based
  - Wordnet
  - Wiki
  - Web

Knowledge based Similarity

- Knowledge data

  WordNet



- Intuition:

  Two words are similar if they are close to each other

- Measure approach
  - Shortest path based

    [Rada, SMC'89][Wu, ACL'94][Leacock'98]
  - Content based

    [Resnik, IJCAI'95][Jiang, ROLING'97][Lin, ICML'98]

Knowledge-based word semantic similarity

- (Leacock & Chodorow, 1998)

$$sim_{lch} = -\log \frac{len}{2*L}$$

- (Wu & Palmer, 1994)

$$sim_{wup} = \frac{2 * depth(LCS)}{depth(concept_1) + depth(concept_2)}$$

- (Lesk, 1986)
  - Finds the overlap between the dictionary entries of two words

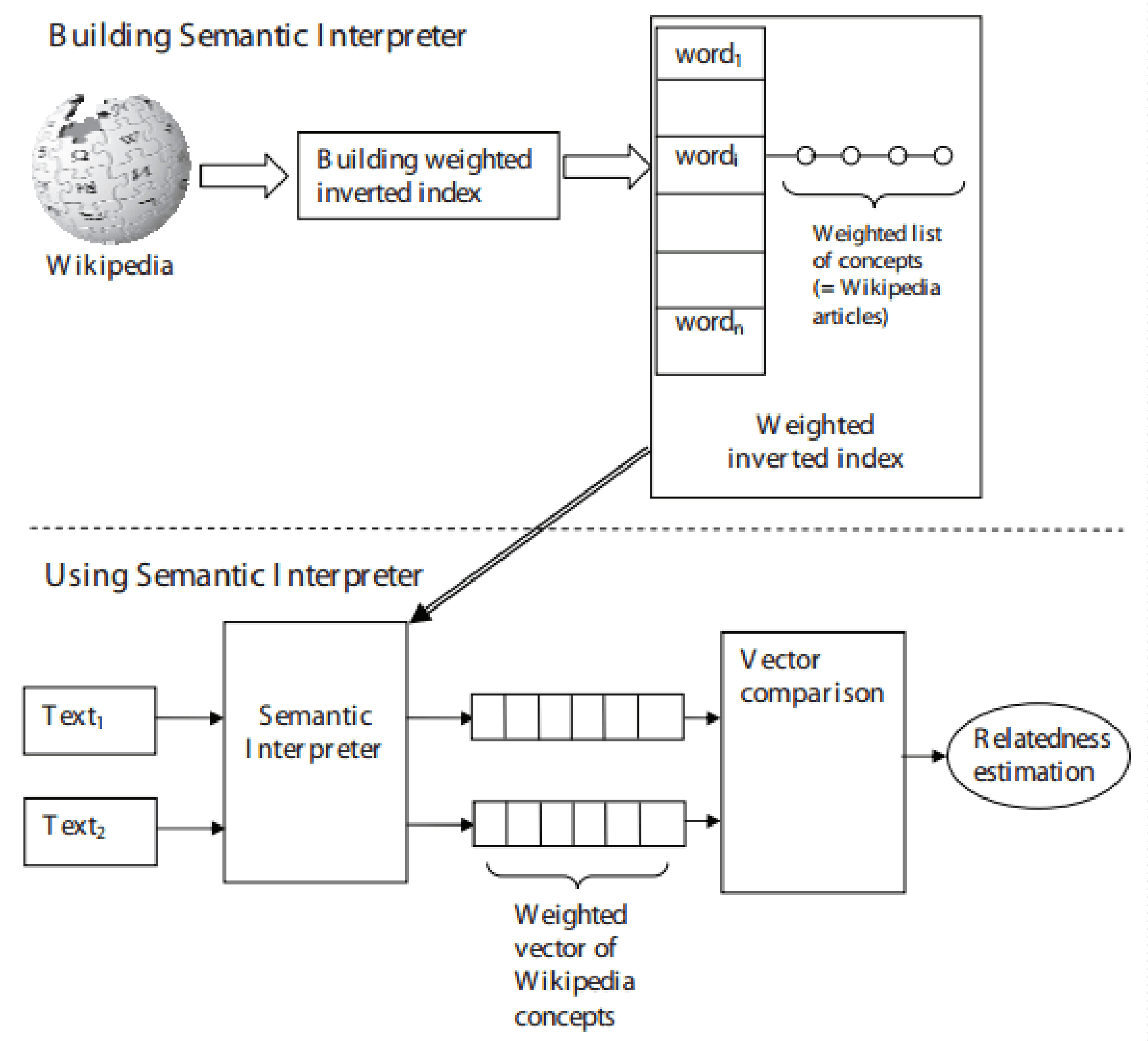
Text Similarity based Techniques

- Vector Space Model (VSM)
  - TF-IDF
- Semantic resource based
  - Wordnet
  - Wiki
  - Web

Explicit Semantic Similarity (ESA)

- Proposed by Gabrilovich [*IJCAI'07*]
- Map text to concepts (i.e., vector) in Wiki
- Calculate ESA score by common vector based measure (i.e., cosine)

ESA Process



This figure is from Gabrilovich IJCAI'07.

ESA Example

- Text1: The dog caught the red ball.
- Text2: A labrador played in the park.

	Glossary of cue sports terms	American Football Strategy	Baseball	Boston Red Sox
T1:	2711	402	487	528
T2:	108	171	107	74

- Similarity Score: 14.38%

This slide is from Rada Mihalcea.

Text Similarity based Techniques

- Vector Space Model (VSM)
  - TF-IDF
- Semantic resource based
  - Wordnet
  - Wiki
  - Web

Corpus based similarity

- Corpus data
  - Web (search engine)

- Intuition:
  - Two words are similar if they frequently occur in the same page
  - PMI-IR [Turney, ECML'01]

PMI-IR

- Pointwise Mutual Information (Church and Hanks'89)

$$PMI(w1, w2) = \log_2 \left(\frac{p(w1 \wedge w2)}{p(w1) * p(w2)} \right)$$

- PMI-IR (Turney'01)

$$PMI - IR(w1, w2) = \log_2 \left(\frac{HitRatio(w1 \wedge w2)}{HitRatio(w1) HitRatio(w2)} \right)$$

$$= \log_2 \left(\frac{\frac{Hit(w1 \wedge w2)}{N}}{\frac{Hit(w1)}{N} * \frac{Hit(w2)}{N}} \right)$$

$$= \log_2 \left(\frac{Hit(w1 \wedge w2) * N}{Hit(w1) * Hit(w2)} \right)$$

where *N* is the number of Web pages

Recommendation Approaches

- Collaborative filtering
- Content based strategies
  - Text similarity based
  - Clustering
  - Classification

All Information about Users and Items

User Profile:

(1) Attribute

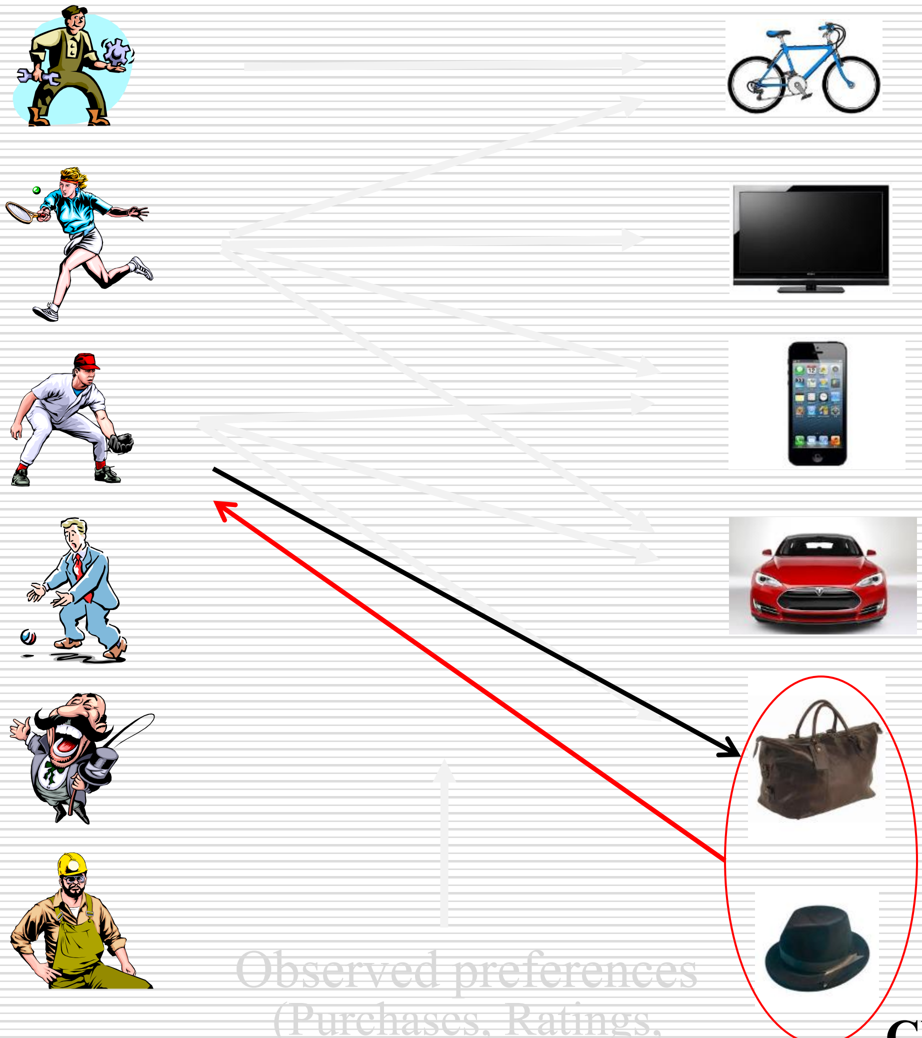
Nationality,Sex, Age,Hobby,etc

(2) Text

Personal description

(3) Link

Social network



Observed preferences (Purchases, Ratings, page views, play lists, bookmarks, etc)

Clustering

Item Profile:

(1) Attribute

Price,Weight,Color,Brand,etc

(2) Text

Product description

(3) link

Associated relation between items (i.e., co-purchased by the same user)

Clustering

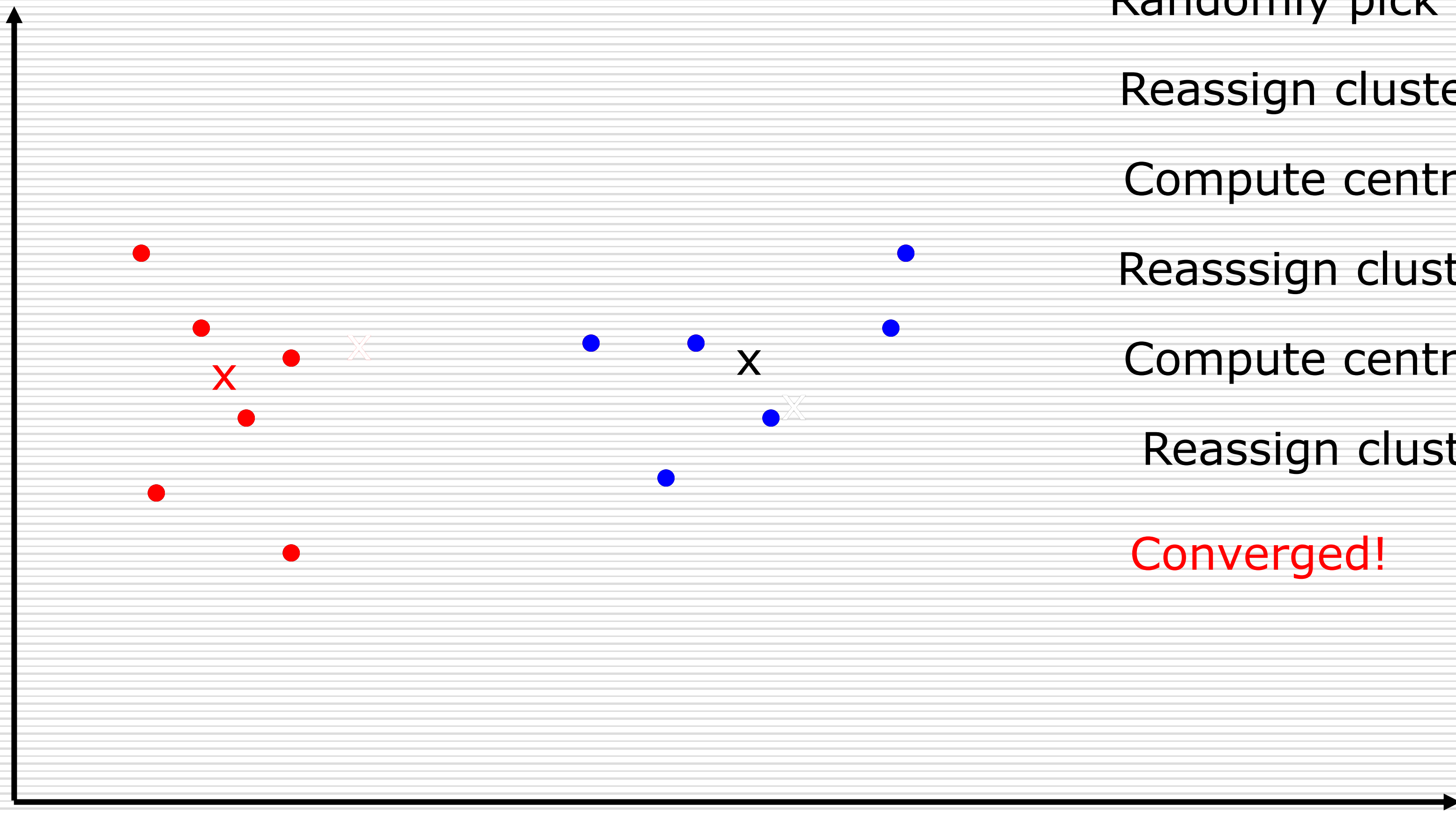
- K-means
- Hierarchical Clustering

K-means

- Introduced by MacQueen, J. B. (1967)
- Works when we know *k*, the number of clusters we want to find
- Idea:
  - Randomly pick *k* points as the "centroids" of the *k* clusters
  - Loop:
    - For each point, put the point in the cluster to whose centroid it is closest
    - Recompute the cluster centroids
    - Repeat loop (until there is no change in clusters between two consecutive iterations.)

Iterative improvement of the objective function:
Sum of the squared distance from each point to the centroid of its cluster

K-means Example (*K*=2)



Randomly pick seeds

Reassign clusters

Compute centroids

Reasssign clusters

Compute centroids

Reassign clusters

Converged!

Clustering

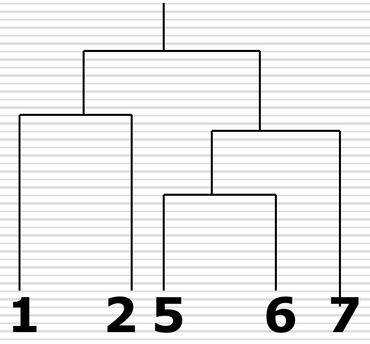
- K-means
- Hierarchical Clustering

Hierarchical Clustering

- Two types:
  - Agglomerative (bottom up)
  - Divisive (top down)
- Agglomerative: two groups are merged if distance between them is less than a threshold
- Divisive: one group is split into two if intergroup distance more than a threshold
- Can be expressed by an excellent graphical representation called dendrogram

Hierarchical Agglomerative Clustering

- *Put every point in a cluster by itself.*
  For I=1 to N-1 do{
  let C_1 and C_2 be the most mergeable pair of clusters
  Create $C_{1,2}$ as parent of C_1 and C_2
  }
- Example: for simplicity, we use 1-dimensional objects.
  - Numerical Objects: 1, 2, 5, 6, 7
- Agglomerative clustering:
  - find two closest objects and merge;
  - => {1,2}, so we have now {1.5,5, 6,7};
  - => {1,2}, {5,6}, so {1.5, 5.5,7};
  - => {1,2}, {{5,6},7}.



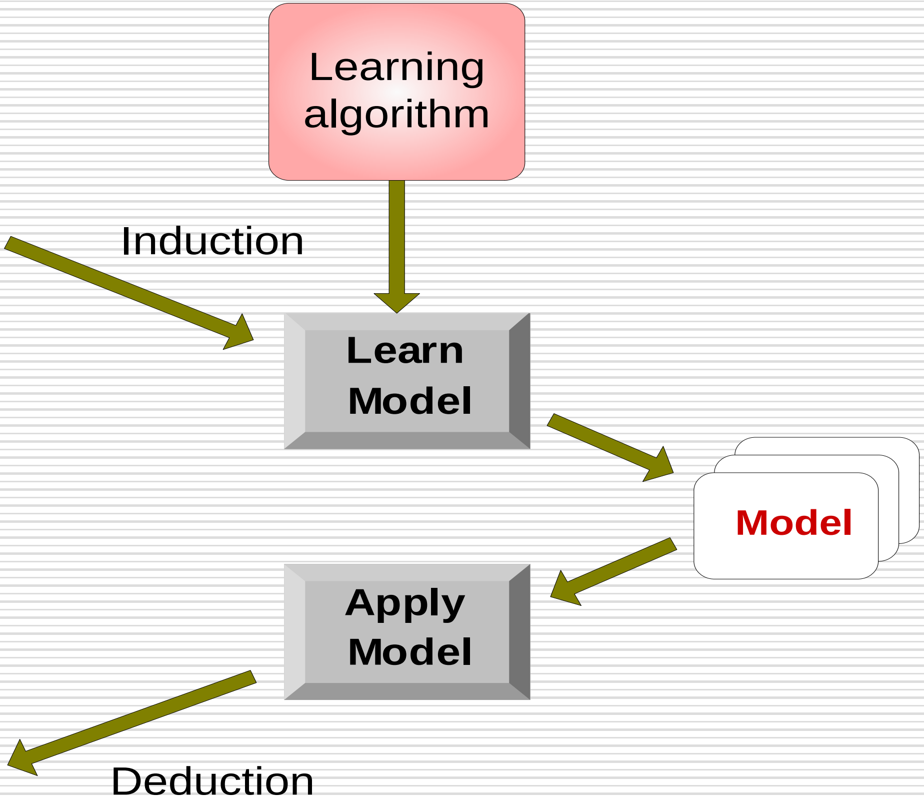
Recommendation Approaches

- Collaborative filtering
- Content based strategies
  - Text similarity based
  - Clustering
  - Classification

Illustrating Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set



Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

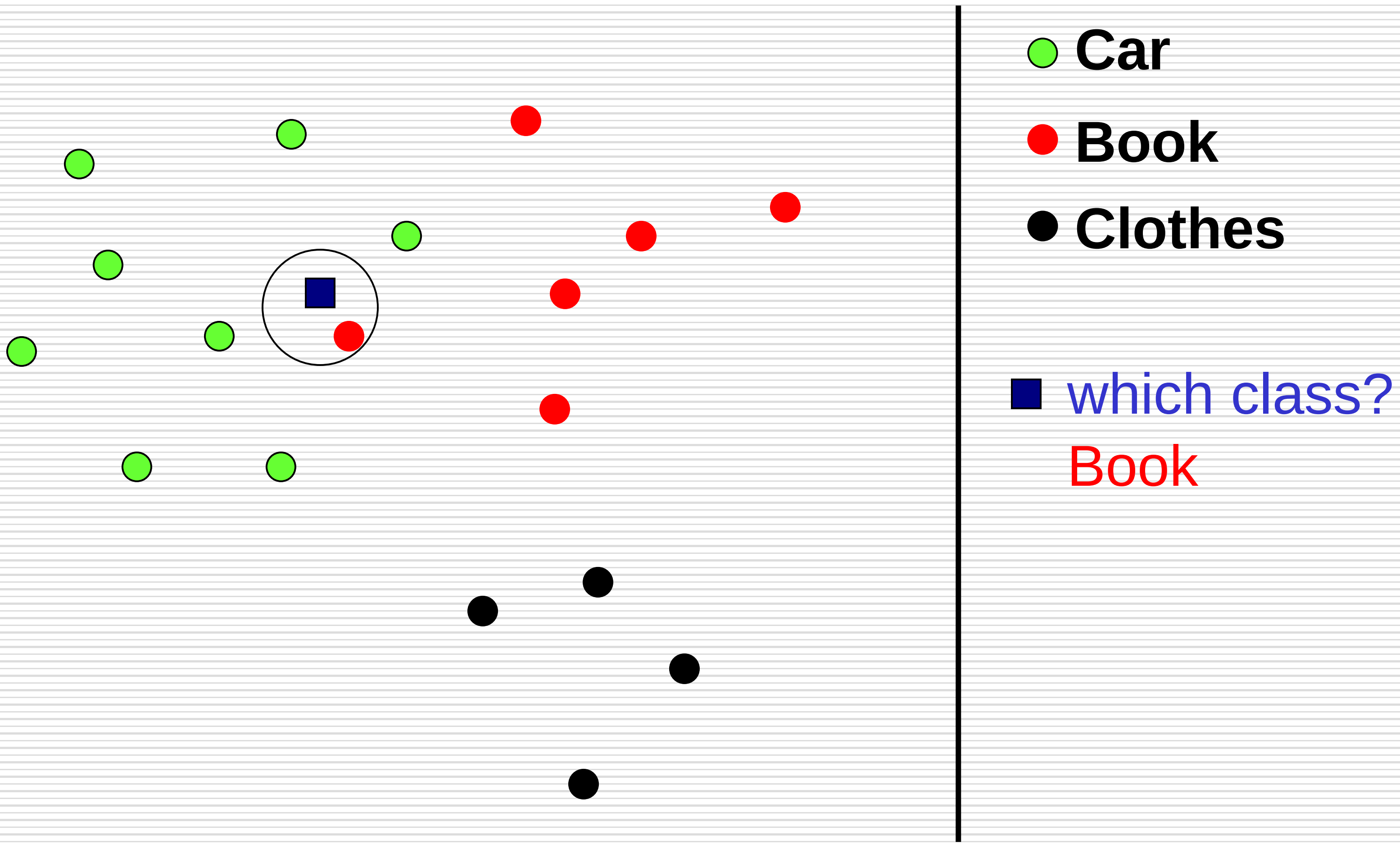
Classification

- k-Nearest Neighbor (kNN)
- Decision Tree
- Naïve Bayesian
- Artificial Neural Network
- Support Vector Machine
- Ensemble methods

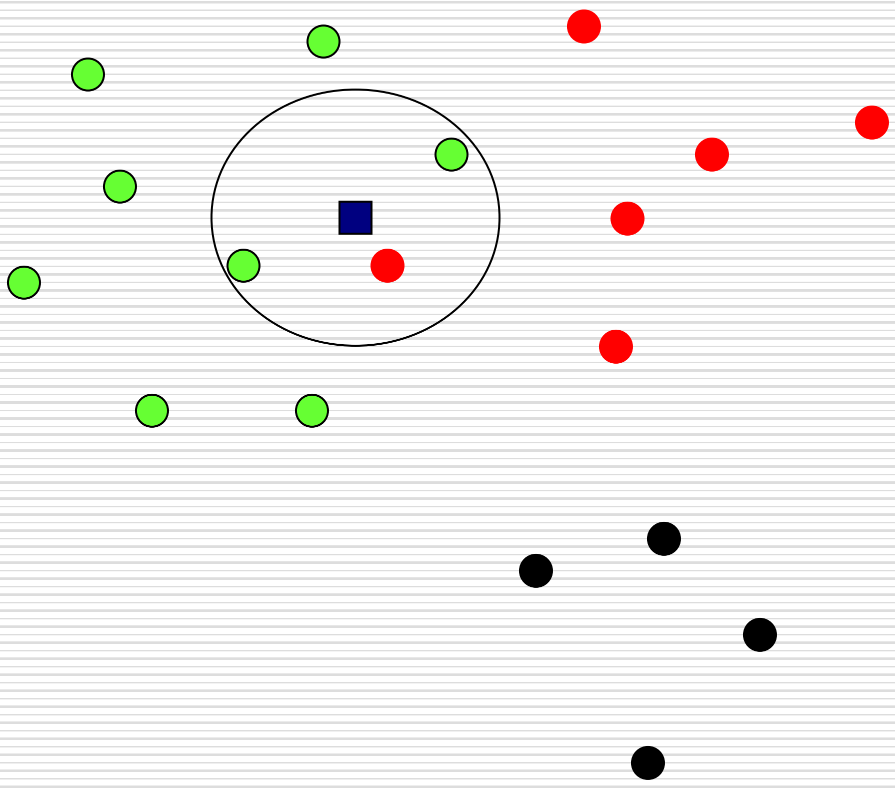
k-Nearest Neighbor Classification (kNN)

- kNN does not build model from the training data.
- Approach
  - To classify a test instance *d*, define *k*-neighborhood *P* as *k* nearest neighbors of *d*
  - Count number *n* of training instances in *P* that belong to class c_j
  - Estimate $\Pr(c_j|d)$ as *n/k (majority vote)*
- No training is needed. Classification time is linear in training set size for each test case.
- *k* is usually chosen empirically via a validation set or cross-validation by trying a range of *k* values.
- Distance function is crucial, but depends on applications.

Example: k=1 (1NN)



Example: k=3 (3NN)



- **Car**
- **Book**
- **Clothes**

which class?

Car

Discussion

- Advantage
  - Nonparametric architecture
  - Simple
  - Powerful
  - Requires no training time
- Disadvantage
  - Memory intensive
  - Classification/estimation is slow
  - Sensitive to *k*

Classification

- k-Nearest Neighbor (kNN)
- Decision Tree
- Naïve Bayesian
- Artificial Neural Network
- Support Vector Machine
- Ensemble methods

Example of a Decision Tree

- Judge the cheat possibility: Yes/No

categorical categorical continuous class

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data

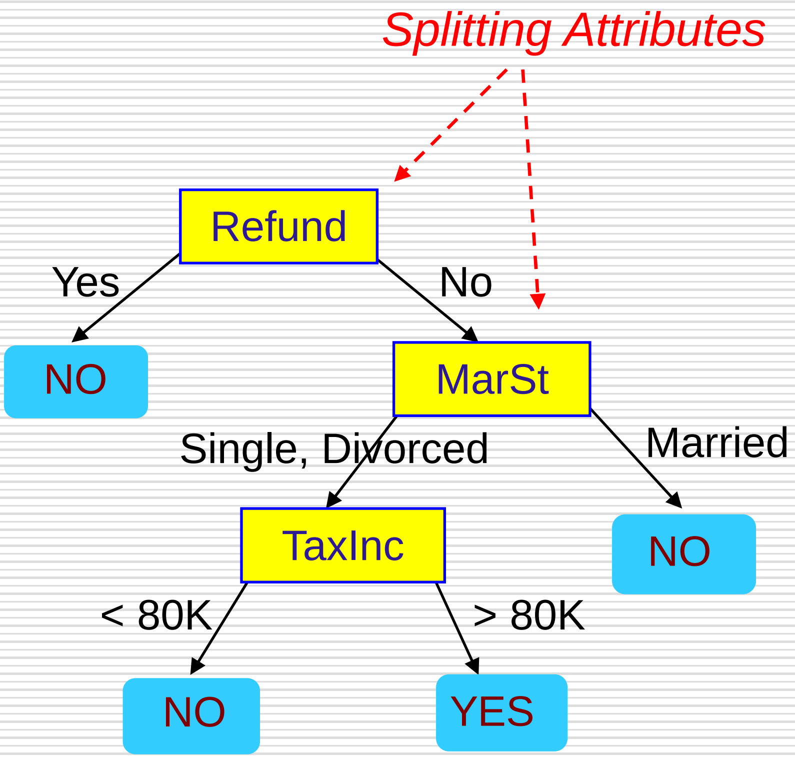
Example of a Decision Tree

- Judge the cheat possibility: Yes/No

categorical categorical continuous class

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data

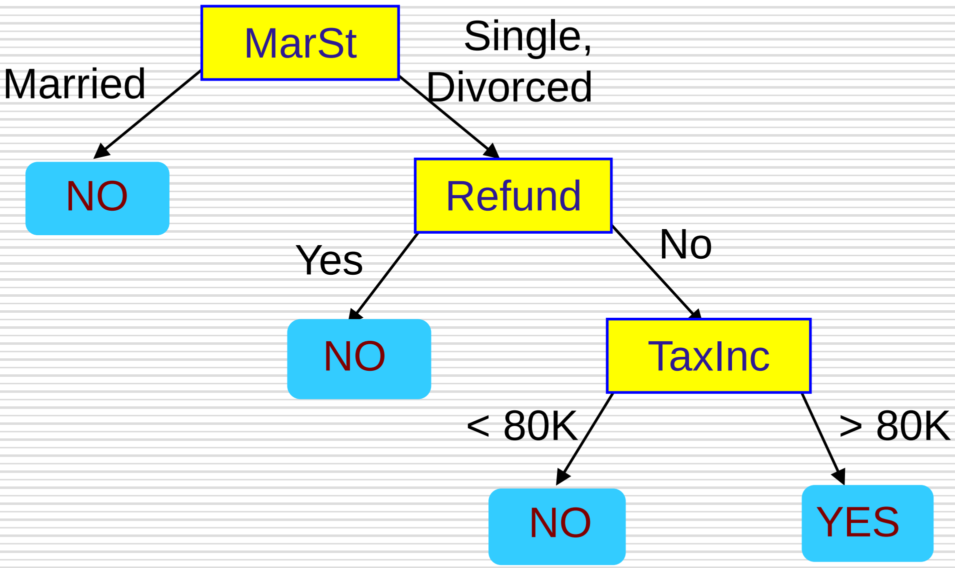


Model: Decision Tree

Another Example of Decision Tree

- Judge the cheat possibility: Yes/No

Tid	Refund	Marital Status	Taxable Income	Cheat
	categorical	categorical	continuous	class
1	Yes	Single	125K	**No**
2	No	Married	100K	**No**
3	No	Single	70K	**No**
4	Yes	Married	120K	**No**
5	No	Divorced	95K	**Yes**
6	No	Married	60K	**No**
7	Yes	Divorced	220K	**No**
8	No	Single	85K	**Yes**
9	No	Married	75K	**No**
10	No	Single	90K	**Yes**



There could be more than one tree that fits the same data!

Decision Tree - Construction

- Creating Decision Trees
  - Manual - Based on expert knowledge
  - Automated - Based on training data (DM)
- Two main issues:
  - Issue #1: Which attribute to take for a split?
  - Issue #2: When to stop splitting?

Classification

- k-Nearest Neighbor (kNN)
- Decision Tree
  - CART
  - C4.5
- Naïve Bayesian
- Artificial Neural Network
- Support Vector Machine
- Ensemble methods

The CART Algorithm

- Classification And Regression Trees
- Developed by Breiman et al. in early 80's.
  - Introduced tree-based modeling into the statistical mainstream
  - Rigorous approach involving cross-validation to select the optimal tree

Key Idea

Recursive Partitioning

- Take all of your data.
- Consider *all* possible values of *all* variables.
- Select the variable/value **($X=t_1$)** that produces the greatest "separation" in the target.
    - **($X=t_1$)** is called a "split".
- If $X < t_1$ then send the data to the "left"; otherwise, send data point to the "right".
- Now repeat same process on these two "nodes"
    - You get a "tree"
    - Note: CART only uses *binary* splits.

Key Idea

- Let Φ(s |t) be a measure of the "goodness" of a candidate split s at node t , where:

$$\Phi(s|t) = 2P_L P_R \sum_{j=1}^{\#\text{ classes}} |P(j|t_L) - P(j|t_R)|$$

$$t_L = \text{left child node of node } t$$

$$t_R = \text{right child node of node } t$$

$$P_L = \frac{\text{number of records at } t_L}{\text{number of records in training set}}$$

$$P_R = \frac{\text{number of records at } t_R}{\text{number of records in training set}}$$

$$P(j|t_L) = \frac{\text{number of class } j \text{ records at } t_L}{\text{number of records at } t}$$

$$P(j|t_R) = \frac{\text{number of class } j \text{ records at } t_R}{\text{number of records at } t}$$

- Then the optimal split maximizes this Φ(s |t) measure over all possible splits at node t .

Key Idea

- $\Phi(s\,|t\,)$ is large when both of its main components are large: $2P_L P_R$ and $\sum_{j=1}^{\#\text{ classes}} |P(j|t_L) - P(j|t_R)|$

1. $2P_L P_R$ - Maximum value if child nodes are equal size (same support)): E.g. 0.5*0.5 = 0.25 and 0.9*0.1= 0.09

2. Q (s |t)= $\sum_{j=1}^{\#\text{ classes}} |P(j|t_L) - P(j|t_R)|$

   - Maximum value if for each class the child nodes are completely uniform (pure)
   - Theoretical maximum value for Q (s|t) is k, where k is the number of classes for the target variable

CART Example

Customer	Savings	Assets	Income ($1000s)	Credit Risk
1	Medium	High	75	**Good**
2	Low	Low	50	**Bad**
3	High	Medium	25	**Bad**
4	Medium	Medium	50	**Good**
5	Low	Medium	100	**Good**
6	High	High	25	**Good**
7	Low	Low	25	**Bad**
8	Medium	Medium	75	**Good**

Training Set of Records for Classifying Credit Risk

CART Example – Candidate Splits

- CART is restricted to binary splits

Candidate Split	Left Child Node, t_L	Right Child Node, t_R
1	Savings = low	Savings={medium, high}
2	Savings = medium	Savings={low, high}
3	Savings = high	Savings={low, medium}
4	Assets = low	Assets={medium, high}
5	Assets = medium	Assets={low, high}
6	Assets = high	Assets={low, medium}
7	Income <=$25,000	Income > $25,000
8	Income <=$50,000	Income > $50,000
9	Income <=$75,000	Income > $75,000

Candidate Splits for t = Root Node

CART Primer

- Split 1. -> Savings=low (L-true, R-false)
  - Right:1,3,4,6,8
  - Left:2,5,7

$$\Phi(s|t) = 2P_L P_R \sum_{j=1}^{\#\text{ classes}} |P(j|t_L) - P(j|t_R)|$$

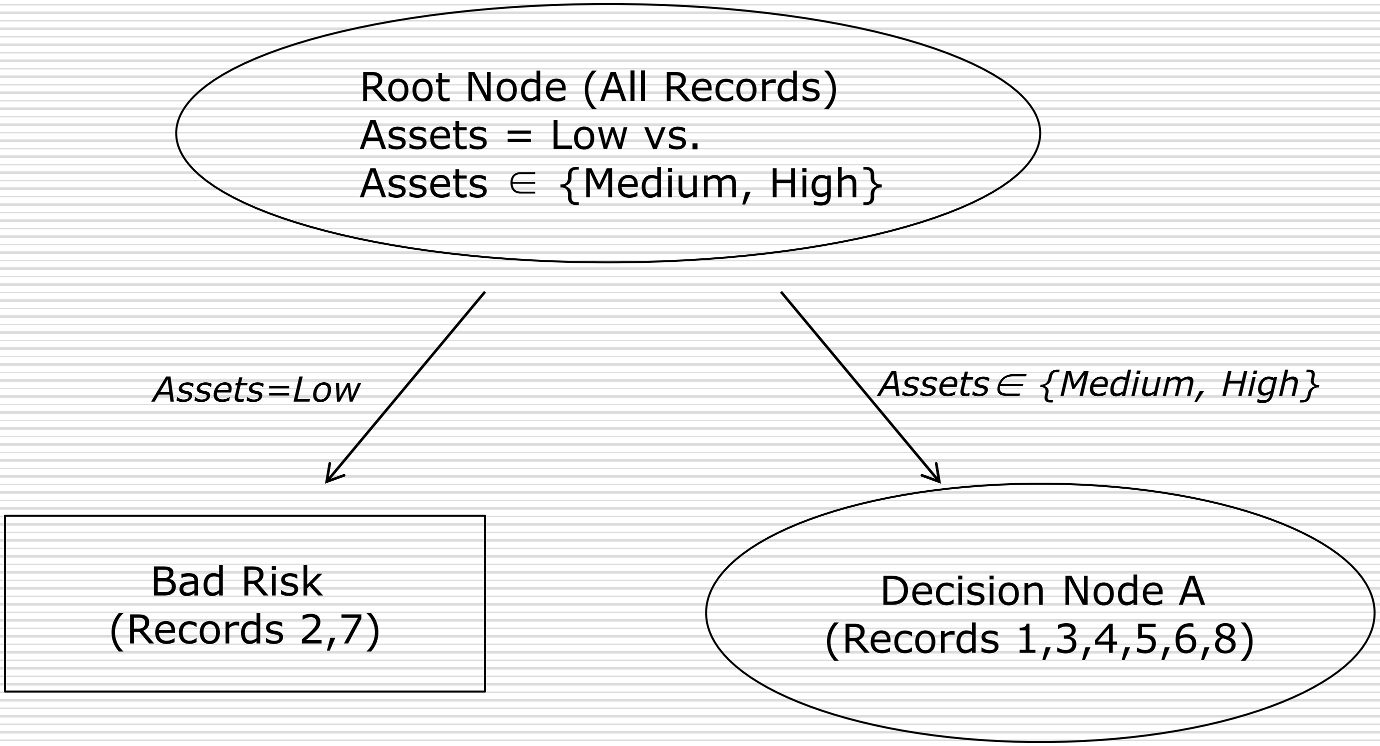
- P_R=5/8 = 0.625 P_L=3/8=0.375 -> 2*$P_L P_R$=15/64=0.46875
- P(j=Bad | t)
  - P(Bad | t_R)= 1/5 = 0.2
  - P(Bad | t_L)= 2/3 = 0.67
- P(j=Good | t)
  - P(Good | t_R)= 4/5 = 0.8
  - P(Good | t_L)= 1/3 = 0.33
- Q(s|t)= |0.67-0.2|+|0.8-0.33| = 0.934

CART Example

Split	P_L	P_R	$P(j\|t_L)$	$P(j\|t_R)$	$2P_LP_R$	$Q(s\|t)$	$\Phi(s\|t)$
1	0.375	0.625	G:0.333 B:0.667	G:0.8 B:0.2	0.46875	0.934	0.4378
2	0.375	0.625	G:1 B:0	G:0.4 B:0.6	0.46875	1.2	0.5625
3	0.25	0.75	G:0.5 B:0.5	G:0.667 B:0.333	0.375	0.334	0.1253
4	0.25	0.75	G:0 B:1	G:0.833 B:0.167	0.375	1.667	0.6248
5	0.5	0.5	G:0.75 B:0.25	G:0.5 B:0.5	0.5	0.5	0.25
6	0.25	0.75	G:1 B:0	G:0.5 B:0.5	0.375	1	0.375
7	0.375	0.625	G:0.333 B:0.667	G:0.8 B:0.2	0.46875	0.934	0.4378
8	0.625	0.375	G:0.4 B:0.6	G:1 B:0	0.46875	1.2	0.5625
9	0.875	0.125	G:0.571 B:0.429	G:1 B:0	0.21875	0.858	0.1877

- For each candidate split, examine the values of the various components of the measure $\Phi(s|t)$

CART Example



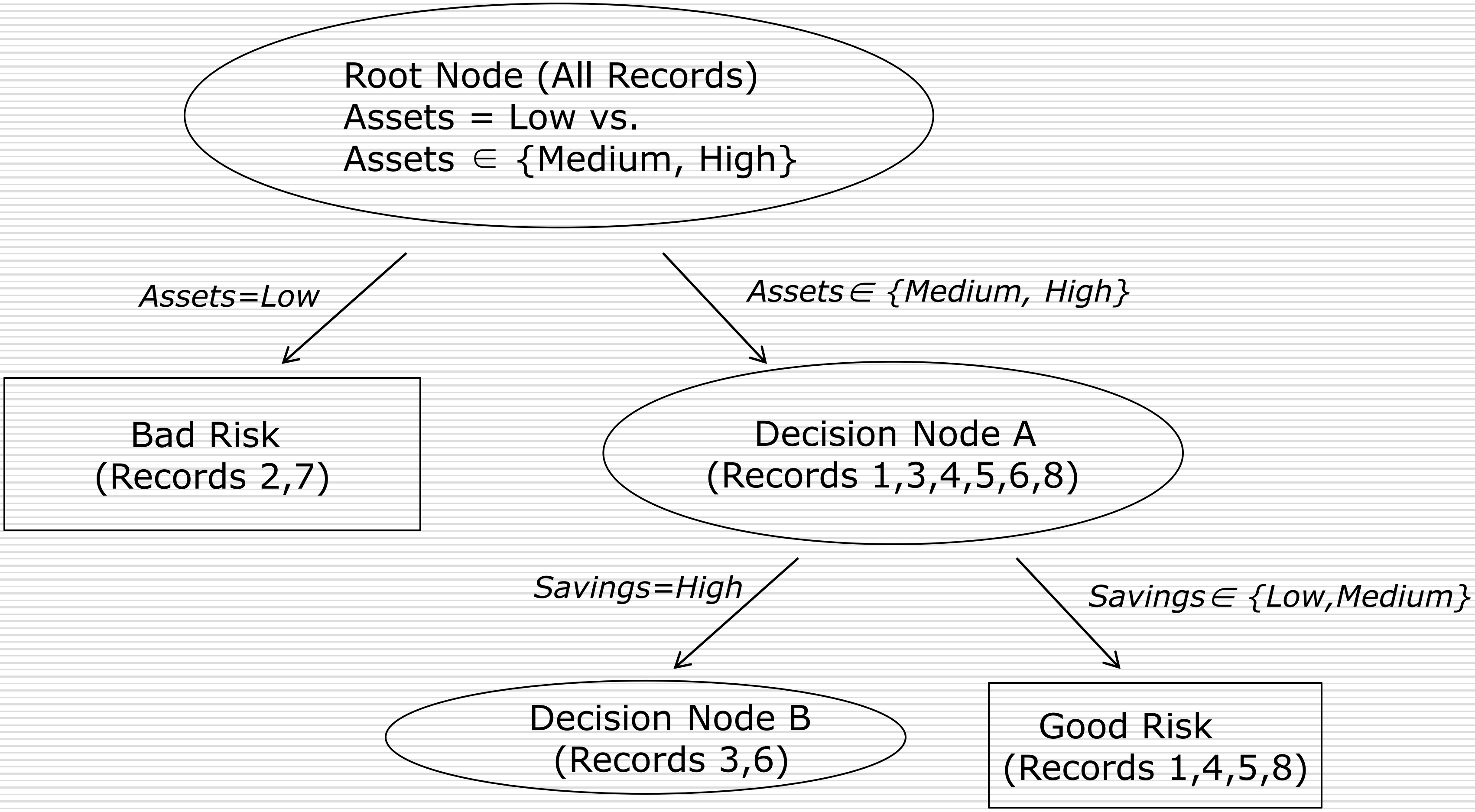
CART decision tree after initial split

CART Example

Split	P_L	P_R	$P(j\|t_L)$	$P(j\|t_R)$	$2P_LP_R$	$Q(s\|t)$	$\Phi(s\|t)$
1	0.167	0.833	G:1 B:0	G:0.8 B:0.2	0.2782	0.4	0.1112
2	0.5	0.5	G:1 B:0	G:0.667 B:0.333	0.5	0.6666	0.3333
3	0.333	0.667	G:0.5 B:0.5	G:1 B:0	0.4444	1	0.4444
5	0.667	0.333	G:0.75 B:0.25	G:1 B:0	0.4444	0.5	0.2222
6	0.333	0.667	G:1 B:0	G:0.75 B:0.25	0.4444	0.5	0.2222
7	0.333	0.667	G:0.5 B:0.5	G:1 B:0	0.4444	1	0.4444
8	0.5	0.5	G:0.667 B:0.333	G:1 B:0	0.5	0.6666	0.3333
9	0.167	0.833	G:0.8 B:0.2	G:1 B:0	0.2782	0.4	0.1112

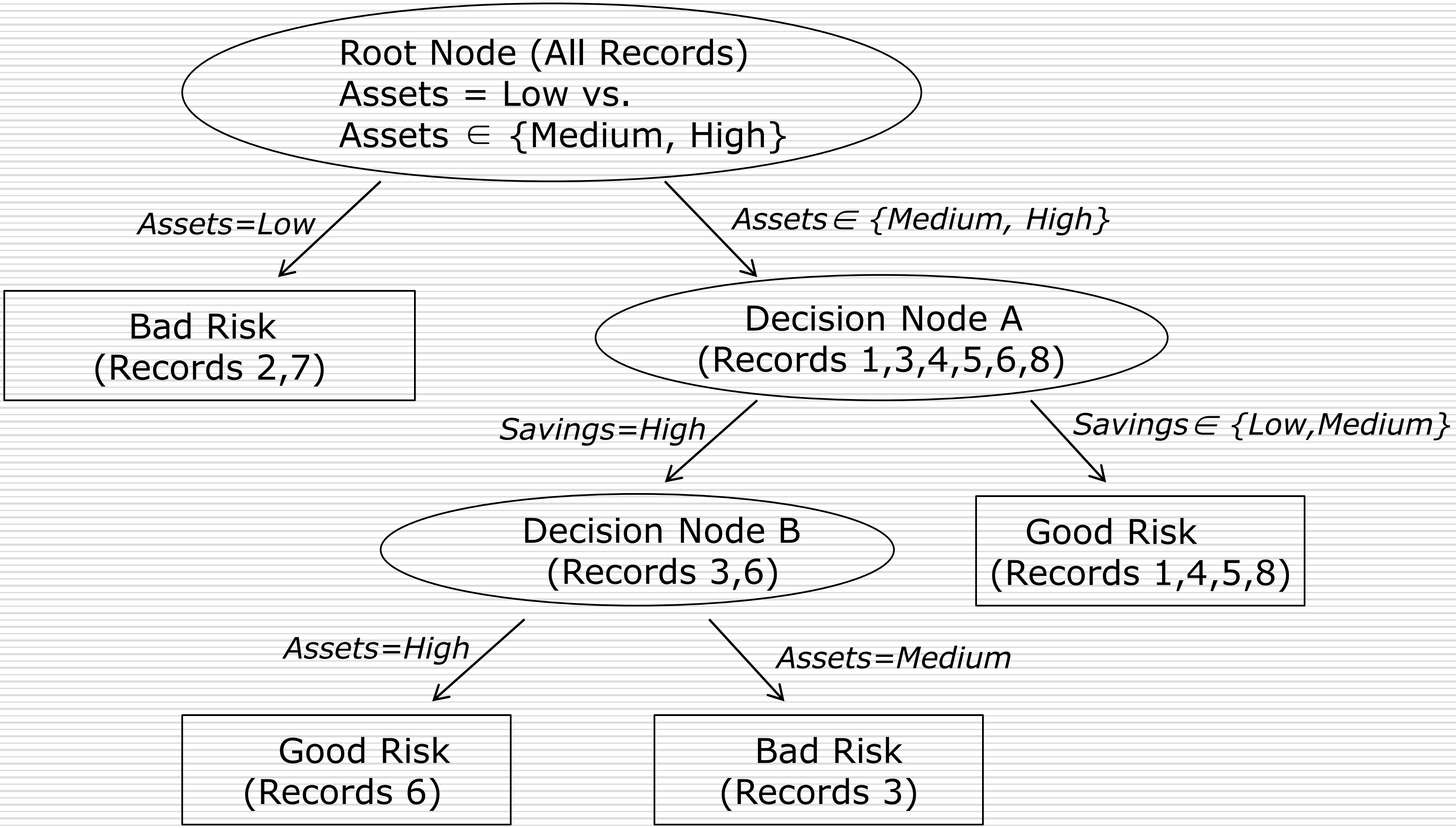
Values of Components of Measure Φ(s|t) for Each Candidate Split on Decision Node A

CART Example



CART decision tree after decision node A split

CART Example



CART decision tree, fully grown form

Classification

- k-Nearest Neighbor (kNN)
- Decision Tree
  - CART
  - C4.5
- Naïve Bayesian
- Artificial Neural Network
- Support Vector Machine
- Ensemble methods

The C4.5 Algorithm

- Proposed by Quinlan in 1993
- An internal node represents a test on an attribute.
- A branch represents an outcome of the test, e.g., Color=red.
- A leaf node represents a class label or class label distribution.
- At each node, one attribute is chosen to split training examples into distinct classes as much as possible
- A new case is classified by following a matching path to a leaf node.

The C4.5 Algorithm

- Differences between CART and C4.5:
  - Unlike CART, the C4.5 algorithm is not restricted to binary splits.
    - It produces a separate branch for each value of the categorical attribute.
  - C4.5 method for measuring node homogeneity is different from the CART.

The C4.5 Algorithm - Measure

- We have a candidate split S, which partitions the training data set T into several subsets, $T_1, T_2, \ldots, T_k$.
- C4.5 uses the concept of entropy reduction to select the optimal split.
- entropy_reduction(S) = H(T)-HS(T), where entropy H(X) is:

$$H(X) = -\sum_j p_j \log_2(p_j)$$

Where Pi represents the proportion of records in subset i .

- The weighted sum of the entropies for the individual subsets $T_1, T_2, \ldots, T_k$

$$H_S(T) = \sum_{i=1}^{k} P_i H_S(T_i)$$

- C4.5 chooses the optimal split - the split with greatest entropy reduction

Classification

- k-Nearest Neighbor (kNN)
- Decision Tree
- Naïve Bayesian
- Artificial Neural Network
- Support Vector Machine
- Ensemble methods

Bayes Rule

- Recommender system question
  - L_i is the class for item *i* (i.e., that the user likes item *i*)
  - A is the set of features associated with item *i*
    - Estimate $p(L_i|A)$
- $p(L_i |A) = p(A| L_i) p(L_i) / p(A)$
- We can always restate a conditional probability in terms of
  - The reverse condition $p(A| L_i)$
  - Two prior probabilities
    - $p(L_i)$
    - $p(A)$
- Often the reverse condition is easier to know
  - We can count how often a feature appears in items the user liked
  - Frequentist assumption

Naive Bayes

- Independence (Naïve Bayes assumption)
  - the features $a_1, a_2, \dots, a_k$ are independent
- For joint probability

$$p(a_1, \cdots, a_k) = \prod_{j=1..k} p(a_j)$$

- For conditional probability

$$p(a_1, \cdots, a_k \mid L_i) = \prod_{j=1..k} p(a_j \mid L_i)$$

- Bayes' Rule

$$p(L_i \mid a_1, a_2, \cdots, a_k) = \frac{p(L_i) \prod_{j=1}^{k} p(a_j \mid L_i)}{\prod_{j=1}^{k} p(a_j)}$$

An Example

Compute all probabilities required for classification

A	B	C
m	b	t
m	s	t
g	q	t
h	s	t
g	q	t
g	q	f
g	s	f
h	b	f
h	q	f
m	b	f

$\Pr(C = t) = 1/2$, $\Pr(C= f) = 1/2$

$\Pr(A=m \mid C=t) = 2/5$
$\Pr(A=m \mid C=f) = 1/5$
$\Pr(B=b \mid C=t) = 1/5$
$\Pr(B=b \mid C=f) = 2/5$

$\Pr(A=g \mid C=t) = 2/5$
$\Pr(A=g \mid C=f) = 2/5$
$\Pr(B=s \mid C=t) = 2/5$
$\Pr(B=s \mid C=f) = 1/5$

$\Pr(A=h \mid C=t) = 1/5$
$\Pr(A=h \mid C=n) =2/5$
$\Pr(B=q \mid C=t) = 2/5$
$\Pr(B=q \mid C=f) = 2/5$

Now we have a test example:

$A = m \quad B = q \quad C = ?$

An Example

- For C = t, we have

$$\Pr(C=t)\prod_{j=1}^{2}\Pr(A_j=a_j\,|\,C=t)=\frac{1}{2}\times\frac{2}{5}\times\frac{2}{5}=\frac{2}{25}$$

- For class C = f, we have

$$\Pr(C=f)\prod_{j=1}^{2}\Pr(A_j=a_j\,|\,C=f)=\frac{1}{2}\times\frac{1}{5}\times\frac{2}{5}=\frac{1}{25}$$

- C = t is more probable. t is the final class.

Naïve Bayesian Classifier

- Advantages:
  - Easy to implement
  - Very efficient
  - Good results obtained in many applications
- Disadvantages
  - Assumption: class conditional independence, therefore loss of accuracy when the assumption is seriously violated (those highly correlated data sets)

Classification

- K-Nearest Neighbor (kNN)
- Decision Tree
- Naïve Bayesian
- Artificial Neural Network
- Support Vector Machine
- Ensemble methods

References for Machine Learning

- T. Mitchell, Machine Learning, McGraw Hill, 1997
- C. M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006
- T. Hastie, R. Tibshirani and J. Friedman, The Elements of Statistical Learning, Springer, 2001.
- V. Vapnik, Statistical Learning Theory, Wiley-Interscience, 1998.
- Y. Kodratoff, R. S. Michalski, Machine Learning: An Artificial Intelligence Approach, Volume III, Morgan Kaufmann, 1990

编程大作业(二)- 推荐系统

- 分组情况与第一次编程大作业Pagerank一样
- 发布作业　2022. 5. 10
- 提交作业　2022. 6. 10

  分组情况和作业要求可在微信群查看
- 相关问题发信给

  bigdatacomputing@163. com