



## DATA ANALYTICS

### Unit 4: Introduction to Recommendation System

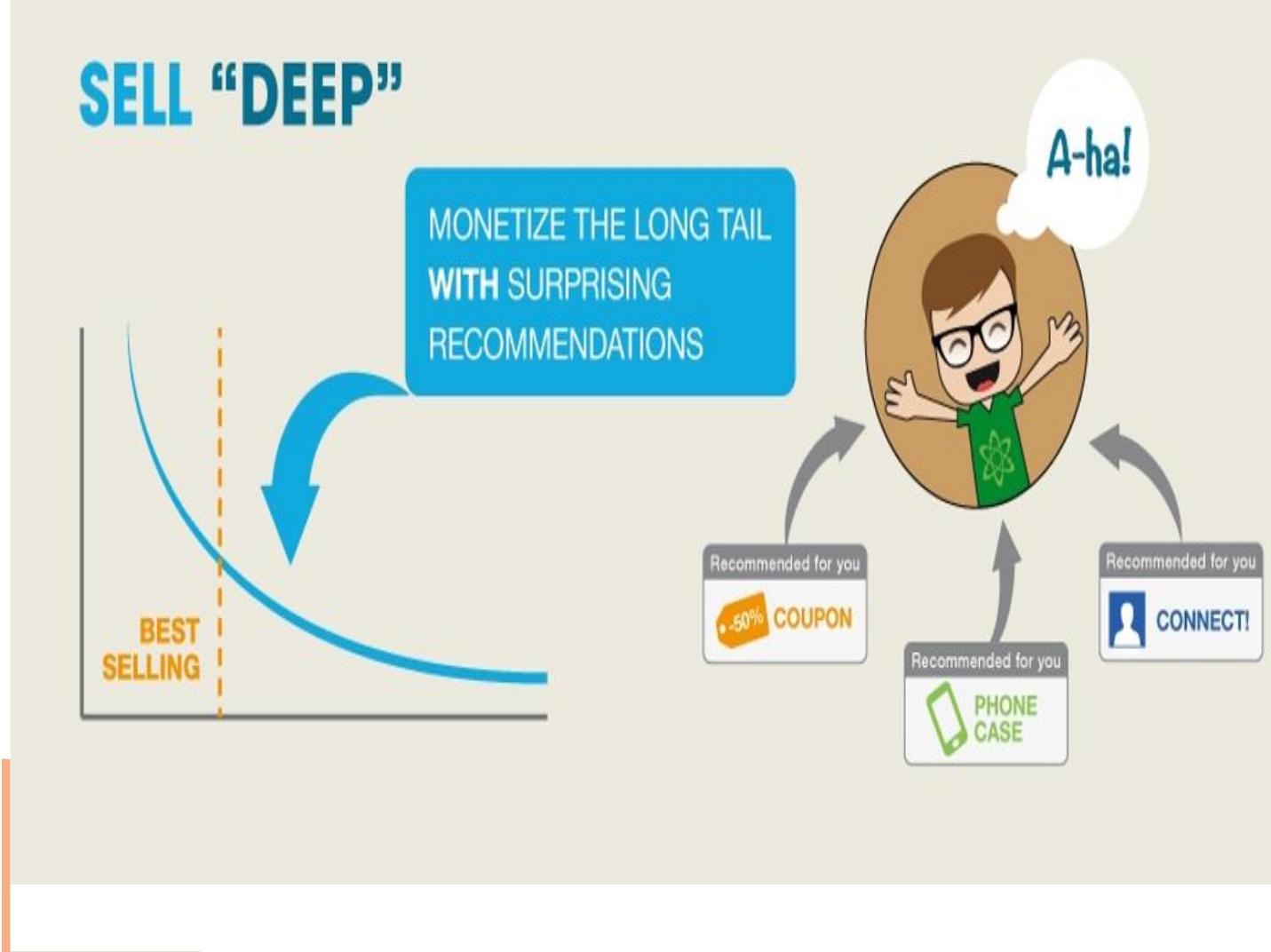
---

Jyothi R.

Department of Computer Science  
and  
Engineering

# DATA ANALYTICS

## Why Recommendation Systems?



"We are leaving the age of information and entering the age of recommendation"

-Chris Anderson in “The Long Tail”

# DATA ANALYTICS

## Age of Recommendation



Search:

User → Items

Recommend:

Items → User

# DATA ANALYTICS

## Amazon A personalized online store

### Frequently Bought Together



Price for both: \$158.15

Add both to Cart

Add both to Wish List

One of these items ships sooner than the other. Show details

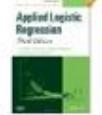
This item: Introduction to Data Mining by Pang-Ning Tan Hardcover \$120.16

Data Science for Business: What you need to know about data mining and data-analytic thinking by Foster Provost Paperback \$37.99

### Customers Who Bought This Item Also Bought

Page 1 of 15



 Data Science for Business: What you need... Foster Provost 4.5 stars 102 <b>#1 Best Seller</b> in Data Mining Paperback \$37.99 	 Data Mining: Practical Machine Learning Tools... Ian H. Witten 4.5 stars 52 Paperback \$40.65 	 Data Mining: Concepts and Techniques, Third... Jiawei Han 4.5 stars 28 Hardcover \$60.22 	 Regression Analysis by Example Samprit Chatterjee 4.5 stars 9 Hardcover \$92.39 	 SAS Statistics by Example Ron Cody 4.5 stars 10 Perfect Paperback \$44.37 	 Applied Logistic Regression David W. Hosmer Jr. 4.5 stars 9 Hardcover \$62.33 	 An Introduction to Statistical Learning... Gareth James 4.5 stars 56 <b>#1 Best Seller</b> in Mathematical & Statistical... Hardcover \$72.79 
---	---	--	--	--	---	---

# DATA ANALYTICS

## Amazon A personalized online store



Introduction to Data Mining

\$120.16 FREE Shipping. | Temporarily out of stock. Order now and we'll deliver when available. We'll e-mail you w

### What Other Items Do Customers Buy After Viewing This Item?



Data Science for Business: What you need to know about data mining and data-analytic thinking Paperback

› Foster Provost

★★★★★ 102

\$37.99 Prime



Introduction to Data Mining Paperback

Pang-ning Tan

★★★★★ 4

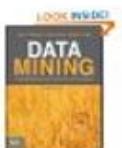


Data Mining: Concepts and Techniques, Third Edition (The Morgan Kaufmann Series in Data Management Sy

› Jiawei Han

★★★★★ 28

\$60.22 Prime



Data Mining: Practical Machine Learning Tools and Techniques, Third Edition (The Morgan Kaufmann Series i

› Ian H. Witten

★★★★★ 52

\$40.65 Prime

# DATA ANALYTICS

## Recommender Problem

---

### A Good recommender

- Show programming titles to a software engineer and baby toys to a new mother
- Don't recommend items, which user already knows or would find anyway.
- Expand User's taste without offending or annoying him/her..

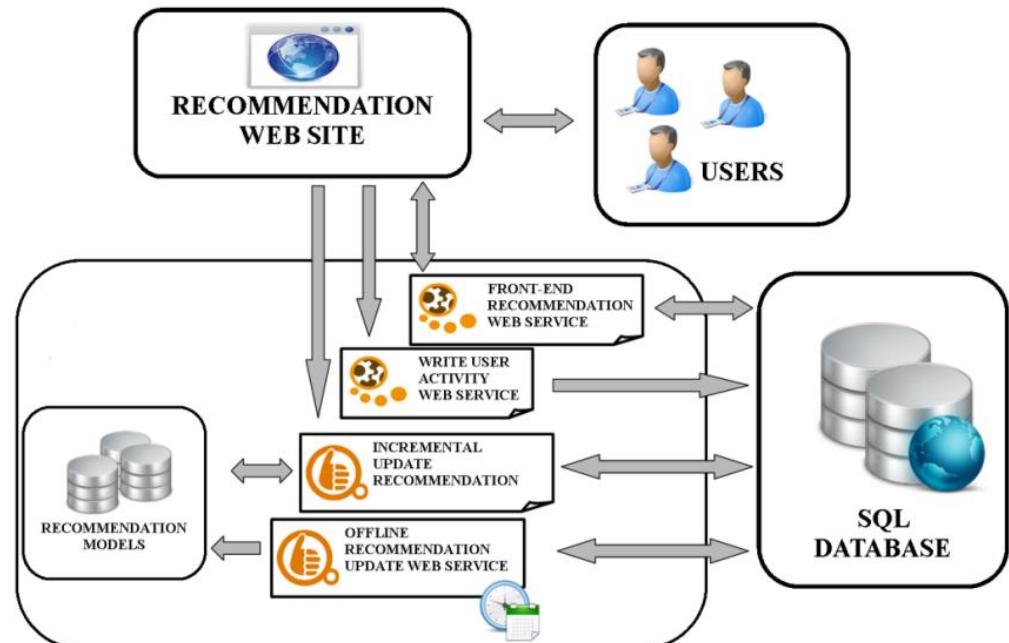
### Challenges

- Huge amounts of data, tens of millions of customers and millions of distinct catalog items.
- Results are required to be returned in real time.
- New customers have limited information.
- Customer data is volatile.

# DATA ANALYTICS

## Amazon's solution

1. Amazon Recommendation Engine: Amazon's model that implements recommendation algorithm. Recommendation algorithm is designed to personalize the online store for each customer.
2. Recommendation Engine Workflow:



## Recommendations:



## Introduction to Recommendation system

---

- Recommendation system - problem of information filtering
- Enhance user experience
  - Assist users in finding information
  - Reduce search and navigation time
- Recommender systems are the most popular applications of data science today, to Increase productivity, Increase credibility.
- They are used to predict the “rating” or “preference” that a user would give to an item.
- Amazon uses it to suggest products to customers.
- YouTube uses it to decide which video to play next on auto play, and,
- Facebook uses it to recommend pages to like and people to follow.
- Most of the companies business model and its success revolves around the potency of their recommendations.

### Goals of Recommender Systems

---

1. **Prediction version of problem:** the first approach is to predict the rating value for a user-item combination. It is assumed that training data is available, indicating user preferences for items. For m users and n items, this corresponds to an incomplete mxn matrix, where the specified values are used for training.
  
2. **Ranking version of problem:** In practice, it is not necessary to predict the ratings of users for specific items in order to make recommendations to users. The determination of the top-k items is more common than the determination of top-k users.

### Goals of Recommender Systems Contd.

---

In order to achieve broader business-centric goal of increasing revenue, the operational and technical goals of recommender systems are as follows

1. **Relevance:** Users are more likely to consume items they find interesting, rating value for a user-item combination.
2. **Novelty:** Recommender systems are truly helpful when the recommended item is something that the user has not seen in the past. For example, Popular movies of a preferred genre would rarely be novel to the user.
3. **Serendipity:** The items recommended are somewhat unexpected, and therefore there is a modest element of lucky discovery. Recommendations are truly surprising to the user. It leads to sales diversity or beginning a new trend of interest in the user.
4. **Increasing Recommendation Diversity:** It has the benefit of ensuring that the user does not get bored by repeated recommendation of similar items.

### Goals of Recommender Systems Contd.

---

- Aside from these concrete goals, a number of soft goals are also met by the recommendation process both from the perspective of the user and merchant.
- The broad diversity of recommender systems that were built either as research prototype, or are available today as commercial systems in various problem settings
  1. GroupLens Recommender System
  2. Amazon.com Recommender System
  3. Netflix Movie Recommender System
  4. Google News Personalization System
  5. Facebook Friend Recommendations

## The Spectrum of Recommendation Applications

---

1. Collaborative Filtering Models
  - i) Memory - based collaborative filtering
    - a) User-Based collaborative filtering
    - b) Item-based collaborative filtering
  - ii) Model-Based Methods
    - a) Types of Ratings: Implicit and Explicit Ratings
    - b) Relationship with missing values.
2. Content-Based Recommender systems
3. Knowledge-Based Recommender Systems
  - i) Constraint-based recommender systems
  - ii) Case-based recommender systems
4. Demographic Recommender systems
5. Hybrid and Ensemble-Based Recommender Systems

## Gathering Ratings

---

### Types of Ratings

- Explicit
- Ask people to rate items
- Doesn't scale: only a small fraction of users leave ratings and reviews
- Implicit
- Learn ratings from user actions
- E.g., purchase implies high rating
- What about low ratings?

## Domain-specific Challenges in Recommender Systems

---

1. Context-Based Recommender Systems: It could include time, location, or social data. For example, the types of clothes recommended by a retailer might depend both on the season and location of the customer. Even particular type of festival or holiday affects the underlying customer activity.
  
2. Time-Sensitive Recommender Systems:
  - i. The rating of an item might evolve with time, as community attitudes evolve and the interests of users change over time. User interests, likes, dislikes, and fashions inevitably evolve with time.
  
  - ii. The rating of an item might be dependent on the specific time of day, day of week, month, or season.
  
  - iii. For example, it makes little sense to recommend winter clothing during the summer, or Raincoats during the dry season.

## Domain-specific Challenges in Recommender Systems

---

### 3. Location-Based Recommender Systems

- i) User-Specific Locality
- ii) Item-specific Locality

### 4. Social Recommender Systems

- i) Structural Recommendation of Nodes and Links
- ii) Product and Content Recommendations with social influence.
- iii) Trustworthy Recommender Systems
- iv) Leveraging Social Tagging Feedback for Recommendations

## References

---

### Text Book:

“Business Analytics, The Science of Data-Driven Decision Making”, U. Dinesh Kumar, Wiley 2017

“Recommender Systems, The text book, Charu C. Aggarwal, Springer 2016  
Section 1.

# DATA ANALYTICS

## Image Courtesy

---

<https://www.scribd.com/presentation/414445910/CS548S15-Showcase-Web-Mining>

[Web Mining](#)

<https://towardsdatascience.com/image-recommendation-engine-leverage-transfert-learning-ec9af32f5239>

<https://www.youtube.com/watch?v=1JRrCEgiyHM>

[www.amazon.com](http://www.amazon.com)

<http://elico.rapid-i.com/recommender-extension.html>



**THANK YOU**

---

**Jyothi R.**  
Assistant Professor,  
Department of Computer Science  
[jyothir@pes.edu](mailto:jyothir@pes.edu)



# DATA ANALYTICS

## Unit 4: Collaborative Filtering System

---

Jyothi R.  
Department of Computer Science  
and Engineering

## Domain-specific Challenges in Recommender Systems

---

1. Context-Based Recommender Systems: It could include time, location, or social data. For example, the types of clothes recommended by a retailer might depend both on the season and location of the customer. Even particular type of festival or holiday affects the underlying customer activity.
  
2. Time-Sensitive Recommender Systems:
  - i. The rating of an item might evolve with time, as community attitudes evolve and the interests of users change over time. User interests, likes, dislikes, and fashions inevitably evolve with time.
  
  - ii. The rating of an item might be dependent on the specific time of day, day of week, month, or season.
  
  - iii. For example, it makes little sense to recommend winter clothing during the summer, or Raincoats during the dry season.

## Domain-specific Challenges in Recommender Systems

---

### 3. Location-Based Recommender Systems

- i) User-Specific Locality
- ii) Item-specific Locality

### 4. Social Recommender Systems

- i) Structural Recommendation of Nodes and Links
- ii) Product and Content Recommendations with social influence.
- iii) Trustworthy Recommender Systems
- iv) Leveraging Social Tagging Feedback for Recommendations

## Advanced Topics and Applications

---

1. The Cold-Start Problem in Recommender Systems: Most people have not rated most items -

### Cold Start:

- New items have no ratings
- New users have no history

2. Attack-Resistant Recommender Systems

3. Group Recommender Systems

4. Multi-Criteria Recommender Systems

5. Active Learning in Recommender Systems

6. Privacy in Recommender Systems

7. Application Domains

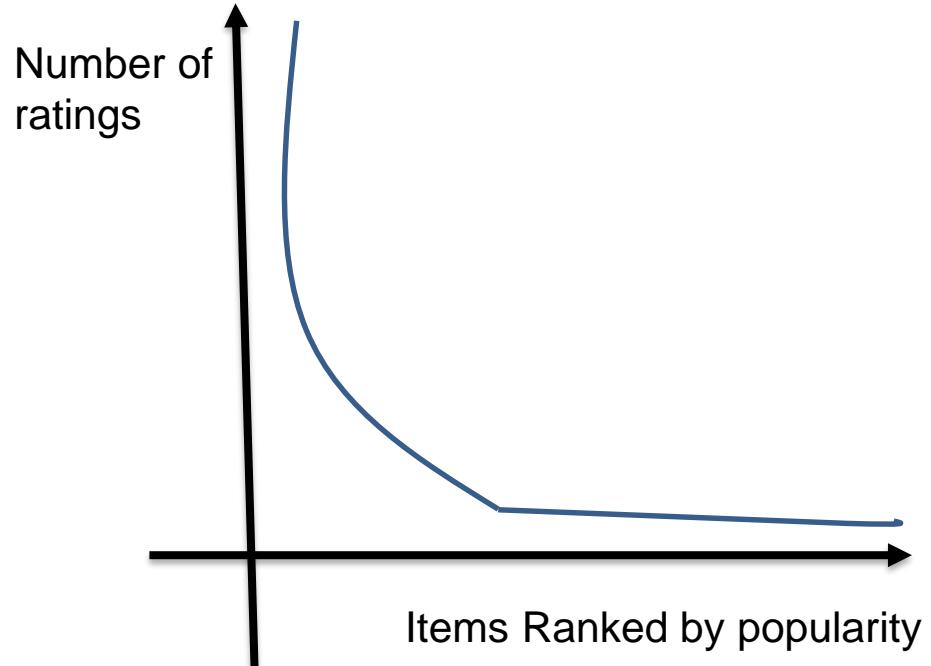
## From Scarcity to Abundance

---

- Shelf space is a scarce commodity for traditional retailers TV networks, Movie Theatres.
- More choice necessitates better filters.
- The web enables near-zero-cost dissemination of information about products.

From scarcity to abundance gives rise to the “Long Tail” phenomenon.

## The long tail:



The long tailed distribution implies that the items, which are frequently rated by users, are fewer in number. This fact has important implications for neighborhood-based collaborative filtering algorithms because the neighborhoods are often defined on the basis of these frequently rated items. In many cases, the ratings of these high frequency items are not representative of the low-frequency items because of the inherent differences in the rating patterns of the two classes of items. As a result, the prediction process may yield misleading results.

The economics of abundance:

Items might be a books, music, videos, or news articles

## Overview of recommendations

---

1. Editorial and hand curated
  - i) List of favourites
  - ii) List of “essential” items
2. Simple aggregates.
  - i) Top 10 Most Popular
  - ii) Most recent uploads
3. Tailored to individual users
  - i) Amazon
  - ii) Netflix
  - iii) Pandora's

Utility Function : A function that looks at every pair of a customer and item and maps it.

$$U: C \times S \rightarrow R$$

- I.     $C$ = set of customers and
- II.    $S$ =Set of items
- III.    $R$ =Set of ratings, it is a totally ordered set eg.-5 stars, real no in  $[0..1]$

# DATA ANALYTICS

## Utility Matrix

	Avatar	KGF	Matrix	Bahubali
Alice	1		0.2	
Bob		0.5		0.3
Carol	0.2		1	
David				0.4

## Key Problems

---

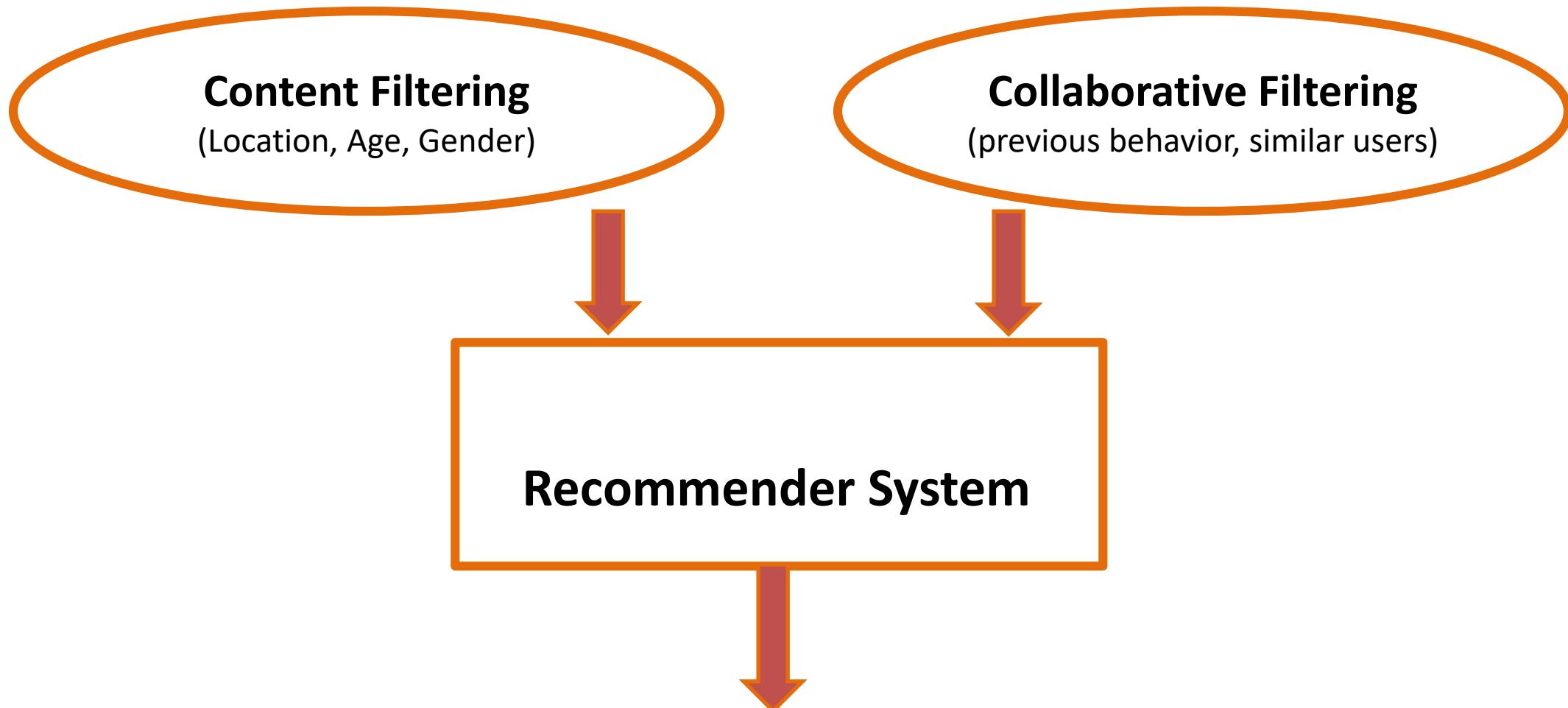
1. Gathering “known” ratings for matrix: How to collect the data in the utility matrix?
2. Extrapolate unknown ratings from the known ones: mainly interested in high unknown ratings. i.e., we are not interested in knowing what you don’t like but what you like
3. Evaluating extrapolation methods: how to measure success/performance of recommendation methods?

## Basic Models of Recommender Systems

---

- The basic models for recommender systems work with two kinds of data, which are
  - i. User-Item Interactions, such as ratings or buying behavior, and
  - ii. The attribute information about the users and items such as textual profiles or relevant keywords.
- Content-based systems also use the ratings matrices in most cases, although the model is usually focused on the ratings of a single user rather than those of all users.

## Types of Recommendation system



## Basic Models of Recommender Systems

---

- In knowledge-based recommender systems, the recommendations are based on explicitly specified user requirements.
- Hybrid systems combine the strengths of various types of recommender systems to create techniques that can perform more robustly in a wide variety of settings.

## Collaborative Filtering Models

---

- Collaborative filtering models use the collaborative power of the ratings by multiple users to make recommendations.
- The main challenge in designing collaborative filtering methods is that the underlying ratings matrices are sparse.

Eg. Movie Recommendations.

## Collaborative Filtering Models

---

- The basic idea of collaborative filtering methods is that these unspecified ratings can be imputed.
- Here the observed ratings are highly correlated across various users and items.
- Most of the models for collaborative filtering focus on leveraging either inter-item correlations or inter-user correlations for the prediction process. Some models also use both types of correlations.

## Collaborative Filtering Models contd.

---

- There are two types of methods that are commonly used in collaborative filtering:
  - a) **Memory-based methods:** they are also referred to as **neighborhood-based collaborative filtering** algorithms. In which the ratings of **user-item combinations** are predicted on the basis of their neighborhoods.

These neighborhoods can be defined in one of two ways -

    - i) **User-based Collaborative filtering:** The ratings provided by the like-minded users of a target user A are used in order to make the recommendations for A
    - ii) **Item-based collaborative filtering:** To make the rating predictions for target item B by user A, the first step is to determine a set S of items that are most similar to target item B.
  - The advantages of memory-based techniques are that they are simple to implement and the resulting recommendations are often easy to explain

## Collaborative Filtering Models contd.

---

b) **Model-based Methods:** Here the machine learning and data mining methods are used in the context of predictive models.

In cases where the model is parameterized, the parameters of this model are learned within the context of an optimization framework.

- Examples: Decision trees, Rule-based models, Bayesian methods and latent factor models.

## Collaborative Filtering Models contd.

---

- Collaborative filtering models are closely related to missing value analysis.
- It can be viewed as a special case of problems in which the data matrix is very large and sparse.
- It can also be viewed as generalizations of classification and regression modeling, here the class/dependent variable can be viewed as an attribute with missing values, other columns are treated as features/independent variables.

## Neighborhood-Based Collaborative Filtering

---

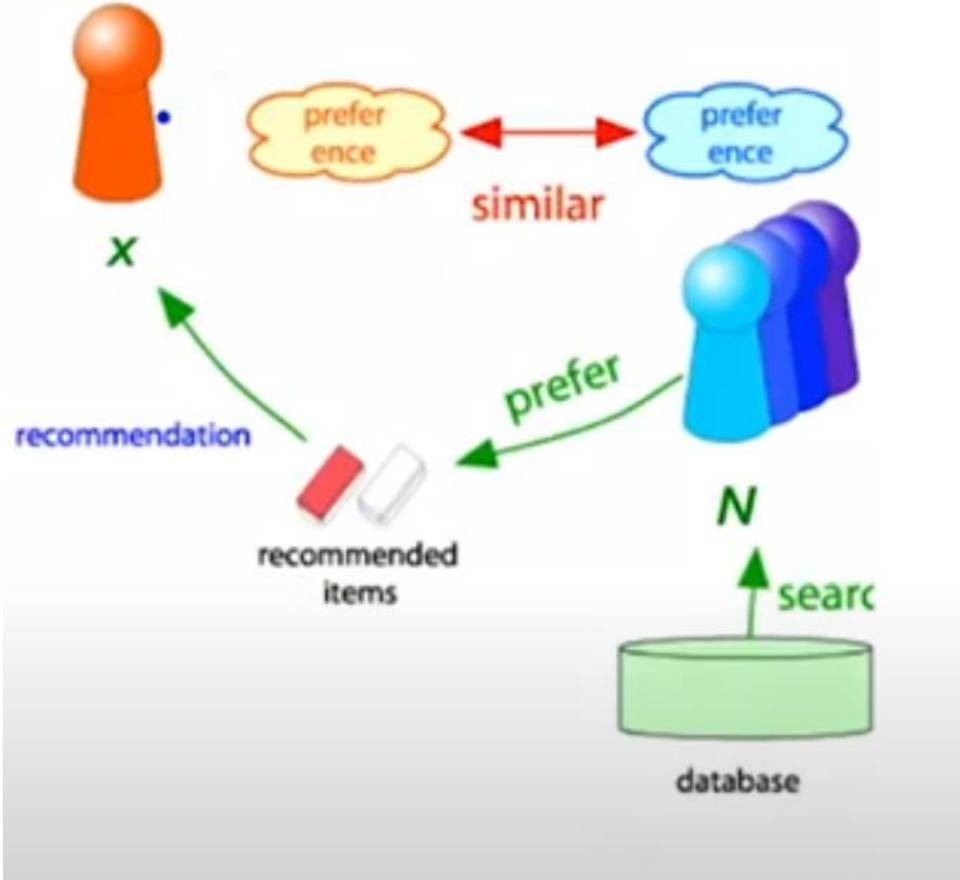
- Neighborhood-based filtering algorithms can be formulated in one of two ways:
  1. Predicting the rating value of a user-item combination: In this case, the missing rating  $r_{uj}$  of the user  $u$  for item  $j$  is predicted.
  2. Determining the top- $k$  items or top- $k$  users: The problem of determine the top- $k$  items is more common than that of finding the top- $k$  users.

## Neighborhood-Based Collaborative Filtering

---

- Item-based methods provide more relevant recommendations because of the fact that a user's own ratings are used to perform the recommendation.
- In item-based methods, similar items are identified to a target item, and the user's own ratings on those items are used to extrapolate the ratings of the target.
- Although item-based recommendations are often more likely to be accurate, the relative accuracy between item-based and user-based methods also depends on the data set at hand.

# Collaborative Filtering



- Consider user x
- Find set N of other users whose ratings are “similar” to x’s ratings
- Estimate x’s ratings based on ratings of users in N.

## Similar Users(1):

	HP1	HP2	HP3	KGF	BB1	BB2	BB3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

- Consider users X and Y with rating vectors  $r_x$  and  $r_y$
- We need a similarity metric  $\text{sim}(x, y)$
- Capture intuition that  $\text{sim}(A,B) \approx \text{sim}(A,C)$

## Option 1: Jaccard Similarity:

	HP1	HP2	HP3	KGF	BB1	BB2	BB3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

- $\text{Sim}(A, B) = |r_A \cap r_B| / |r_A \cup r_B|$
- $\text{Sim}(A, B) < \text{sim}(A, C)$
- Problem: Ignores rating values!

## Option 2: Cosine similarity

	HP1	HP2	HP3	KGF	BB1	BB2	BB3
A	4	0	0	5	1	0	0
B	5	5	4	0	0	0	0
C	0	0	0	2	4	5	
D	0	3	0	0	0	0	3

- $\text{Sim}(A, B) = \cos(r_A, r_B)$
- $\text{Sim}(A, B) = 0.38$ ,  $\text{Sim}(A, C) = 0.32$
- $\text{Sim}(A, B) < \text{sim}(A, C)$ , but not by much
- Problem: treats missing ratings as negative

## Option 3: Centered cosine

- Normalize ratings by subtracting row mean

	HP1	HP2	HP3	KGF	BB1	BB2	BB3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

	HP1	HP2	HP3	KGF	BB1	BB2	BB3
A	$4 - 10/3 = 2/3$			$5/3$	$-7/3$		
B	$1/3$	$1/3$	$-2/3$				
C				$-5/3$	$1/3$	$4/3$	
D		0					0

## Option 3: Centered cosine similarity(2)

	HP1	HP2	HP3	KGF	BB1	BB2	BB3
A	2/3			5/3	-7/3		
B	1/3	1/3	-2/3				
C				-5/3	1/3	4/3	
D		0					0

- $\text{Sim}(A, B) = \cos(r_A, r_B) = 0.09$ ;  $\text{Sim}(A, C) = -0.56$
- $\text{Sim}(A, B) > \text{sim}(A, C)$
- Captures intuition better
- Missing ratings treated as “average”
- Handles “tough raters” and “easy raters”
- Also known as Pearson Correlation

## Item-Item Collaborative Filtering:

---

- So far: User-user Collaborative filtering
- Another view: Item-Item
- For item I, find other similar items
- Estimate rating for item I based on ratings for similar items
- Can use same similarity metrics and prediction functions as in user-user model.

$$r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

- $s_{ij}$  ... Similarity of items I and j
- $r_{xj}$  ... Rating of user x on item j
- $N(i;x)$  ... set items rated by x similar to i

# DATA ANALYTICS

## Item-Item CF( $|N|=2$ )

	Users											
	1	2	3	4	5	6	7	8	9	10	11	
1	1		3		?	5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2				2	5	
6	1		3		3			2			4	



Unknown Rating



Rating between 1 to 5



- Estimate rating of movie 1 by user 5

# DATA ANALYTICS

## Item-Item CF( $|N|=2$ )

	Users												Sim(1,m)	
	1	2	3	4	5	6	7	8	9	10	11			
1	1			3		?	5			5		4		1.00
2				5	4			4			2	1	3	-0.18
3	2	4		1	2			3		4	3	5		0.41
4		2	4		5			4			2			-0.10
5			4	3	4	2					2	5		-0.31
6	1			3		3			2			4		0.59

Here we use Pearson correlation as similarity

- 1) Subtract mean rating m from each movie i

$$M = (1+3+5+5+4)/5 = 3.6$$

Row 1:[-2.6, 0, -0.6, 0, 0, 1.4, 0, 0, 1.4, 0, 0, 0.4, 0]

- 2) Compute cosine similarities between rows

# DATA ANALYTICS

## Item-Item CF(|N|=2)

	Users												
	1	2	3	4	5	6	7	8	9	10	11		
1	1		3		?	5			5		4		
2			5	4			4			2	1	3	
3	2	4		1	2			3		4	3	5	
4		2	4		5			4			2		
5			4	3	4	2					2	5	
6	1		3		3			2			4		

Predict by taking weighted average

$$r_{15} = (0.41*2 + 0.59*3) / (0.41 + 0.59) = 2.6$$

## Item-Item vs. User-User

---

1. In theory, user-user and item-item are dual approaches
2. In practice, item-item outperforms user-user in many use cases.
3. Items are “simpler” than users
  - items belong to a small set of “genres”, users have varied tastes.
  - Item similarity is more meaningful than user similarity

## References

---

“Recommender Systems, The Textbook by Charu C. Aggarwal,  
Springer 2016 – [Sections 1.4 - 2.3.6](#)

# DATA ANALYTICS

## Image Courtesy

---

<http://www.mmds.org/mmds/v2.1/ch09-recsys1.pptx>

[https://www.researchgate.net/publication/287952023\\_Collaborative\\_Filtering\\_Recommender\\_Systems](https://www.researchgate.net/publication/287952023_Collaborative_Filtering_Recommender_Systems)

<http://cs229.stanford.edu/proj2014/Rahul%20Makhijani,%20Saleh%20Samaneh,%20Megh%20Mehta,%20Collaborative%20Filtering%20Recommender%20Systems.pdf>

<https://www.scribd.com/presentation/414445910/CS548S15-Showcase-Web-Mining>

<https://towardsdatascience.com/image-recommendation-engine-leverage-transfert-learning-ec9af32f5239>

<http://elico.rapid-i.com/recommender-extension.html>

<https://www.youtube.com/watch?v=h9gpuJFF-0>



---

## THANK YOU

---

**Jyothi R.**  
Assistant Professor,  
Department of Computer Science  
[jyothir@pes.edu](mailto:jyothir@pes.edu)





# DATA ANALYTICS

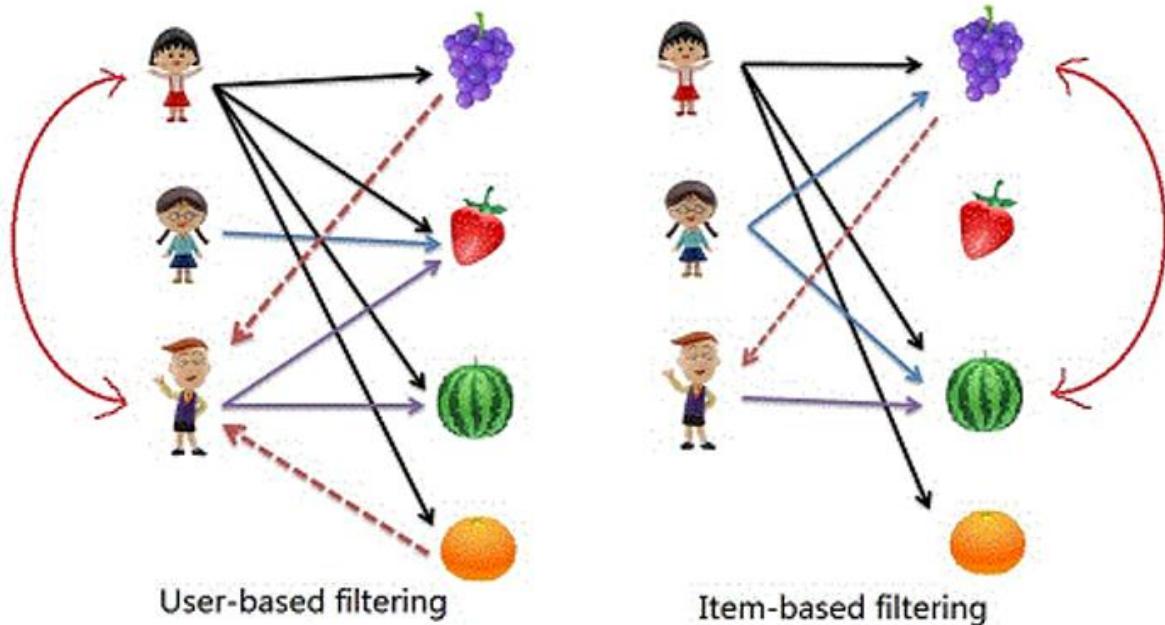
## Unit 4: Recommendation Systems

---

**Jyothi R.**  
Department of Computer Science  
and Engineering

## Amazon's Item-to-Item CF

### Difference with User-to-User CF



# DATA ANALYTICS

## Amazon's Item-to-Item CF

Similarity of user  $i$  with item 17

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
	,1	,3	,6	,1	,3	,4	,3	,3	,2	,6	,2	,5	,4	,5	,5	,3	1	,3	,5	,4	,2	,4	,4	,5

Users

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
a	1		4	5													2								
b		4																5	1		3				
c	5		4		4													3		4		5			
d				3													5		4	2					3
e	3				5													5		1		5	4		
f		4				1		3	5								4		4			3			
g	2	4			4		2		5								4	2	4		5				4
h		2		1		4		3	5								5	4				5			
i	1				3				5								5				4		3		
j		4				4				5							4				4				
k	5			4			2		5								4	2		4			2		
l			3				3			4							4	2	4					3	
m	5	3				5	3		5	4							5	5	3		4				4
n	1		4	5				4	5		1	5		4			3		4	4	4	3			
o		4			4				5		4		5				4	2		5	5				3
p			4			5											5	4	2	4	4	5	4	2	
q			3				3				1	5		4			4		4			4			3
r	4		1	4		2				2		5		4			4			5	4				4
s	2		4		4			5			1		4			2	4		4		5		5		
t	1		4			3				4			5	5		4			4					3	
u	2		1		4		3				1		5	4		2	4			5	4				
v		4	5					4	3	5	5		2					2			2			5	
w		2		2		3		3		5				4	4	5	4	2		3	4				
x	4			5				3	3			4	5			3	3		1						
y			1			3			2	3						3	3		5	4					

Items

## Amazon's Item-to-Item CF

---

- How it works
- Matches each of the user's purchased and rated items to similar items
- Combines those similar items into a recommendation list

An iterative algorithm:

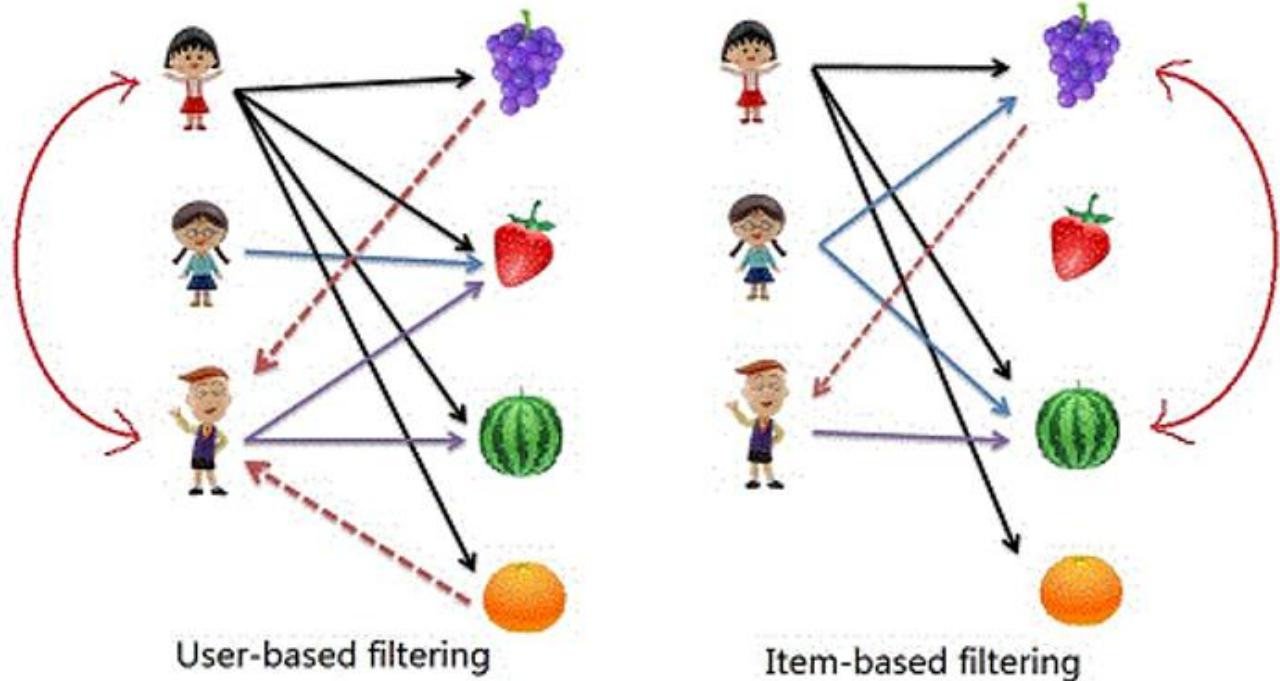
- Builds a similar-items table by finding items that customers tend to purchase together
- Provides a better approach by calculating the similarity between a single product and all related products:

## Amazon's Item-to-Item CF

- The similarity between two items uses the cosine measure
- Each  $M \times 1$  vector corresponds to an item
- A vector's  $M$  dimensions correspond to customers who have purchased that item

```
For each item in product catalog, I1
    For each customer C who purchased I1
        For each item I2 purchased by customer C
            Record that a customer purchased I1 and I2
    For each item I2
        Compute the similarity between I1 and I2
```

## Amazon's Item-to-Item CF vs User-based CF



## Item-Item vs. User-User Scalability and Quality: Comparison

### Item-to-Item collaborative filtering:

- scalability and performance are achieved by creating the expensive similar-items table offline
- online component "looking up similar items" scales independently of the catalog size or the number of customers
- fast for extremely large data sets
- recommendation quality is excellent since it recommends highly correlated similar items
- unlike traditional collaborative filtering,
  - the algorithm performs well with limited user data,
  - producing high-quality recommendations based on as few as two or three items.

### User Based collaborative filtering:

- little or no offline computation
- impractical on large data sets, unless it uses dimensionality reduction, sampling, or partitioning
- dimensionality reduction, sampling, or partitioning reduces recommendation quality

### Cluster models:

- can perform much of the computation offline
- but recommendation quality is relatively poor

## Results:

---

- The MovieLens dataset contains 1 million ratings from 6,040 users on 3,900 movies.
- The best overall results are reached by the item-by-item based approach. It needs 170 seconds to construct the model and 3 seconds to predict 100,021 ratings.

	User Based	Model Based	Item Based
Model Construction Time (sec.)	730	254	170
Prediction Time (sec.)	31	1	3
MAE	0.6688	0.6736	0.6382

## “Core” recommender systems

The conceptual goals of various recommender systems

Approach	Conceptual Goal	Input
Collaborative	Gives us recommendations based on a collaborative approach that leverages the ratings and actions of our peers/myself	User ratings + Community ratings
Content-based	Gives us recommendations based on the content (attributes) we have favored in our past ratings and actions.	User ratings + item attributes + domain knowledge
Knowledge-based	Gives us recommendations based on our explicit specification of the kind of content (attributes) we want	User specification + Item attributes + domain knowledge

## "Core" Recommendation Techniques

Technique	Background	Input	Process
Collaborative	Ratings from U of items in I	Ratings from U of items in I	Identify users in U similar to u, and extrapolate from their ratings of i
Content-based	Features of items in I	U's ratings of items in I	Generate a classifier that fits U's rating behavior and use it on I
Demographic	Demographic information about U and their ratings of items in I	Demographic information about U	Identify users that are demographically similar to U, ad extrapolate from their ratings of i
Utility-based	Features of items in I	A Utility function over items in I that describes U's preferences.	Apply the function to the items and determine I's rank
Knowledge-based	Features of items in I. Knowledge of how these items meet a user's needs.	A description of U's needs or interests.	Infer a match between I and U's need.

## Knowledge-based recommender systems

---

Knowledge-based recommender systems are appropriate in the following situations:

1. Customers want to explicitly **specify their requirements**. Therefore, **interactivity** is a crucial component of such systems. Note that collaborative and content-based systems do not allow this type of detailed feedback.
2. It is **difficult to obtain ratings for a specific type of item** because of the greater complexity of the product domain in terms of the types of items and options available.
3. In some domains, the **ratings may be time-sensitive**. The ratings on an old car or computer are not very useful for recommendations because they evolve with changing product availability and corresponding user requirements.

## Knowledge-based recommender systems types

---

- Knowledge-based recommender systems can be categorized on the basis of user interactive methodology and the corresponding knowledge bases used to facilitate the interaction.
- There are two primary types of knowledge-based recommender systems:
  1. **Constraint-based recommender systems:** In constraint-based systems users typically **specify requirements or constraints** (e.g., lower or upper limits) on the item attributes. Furthermore, **domain-specific rules are used to match the user requirements or attributes to item attributes**. These rules represent the domain-specific knowledge used by the system.
  2. **Case-based recommender systems:** In case-based recommender systems, **specific cases are specified by the user as targets or anchor points**. Similarity metrics are defined on the item attributes to retrieve similar items to these targets. The similarity metrics are often carefully defined in a domain-specific way.

## Knowledge-Based Recommender Systems

---

- The Interaction between user and recommender may take the following forms.
  1. **Conversational Systems:** The user preferences are determined in the context of a feedback loop. The item domain is complex, and the user preferences can be determined only in the context of an iterative conversational system.
  2. **Search-based systems:** User preferences are elicited by using a preset sequence of questions such as the following; "Do you prefer a house in a suburban area or within the city?"
  3. **Navigation-based recommendation:** The user specifies a number of change requests to item being currently recommended. Through an iterative set of change requests, it is possible to arrive at a desirable item.

Eg. " I would like a similar house about 5 miles west of the currently recommended house."

Such recommender systems are also referred to as [critiquing recommender systems](#).

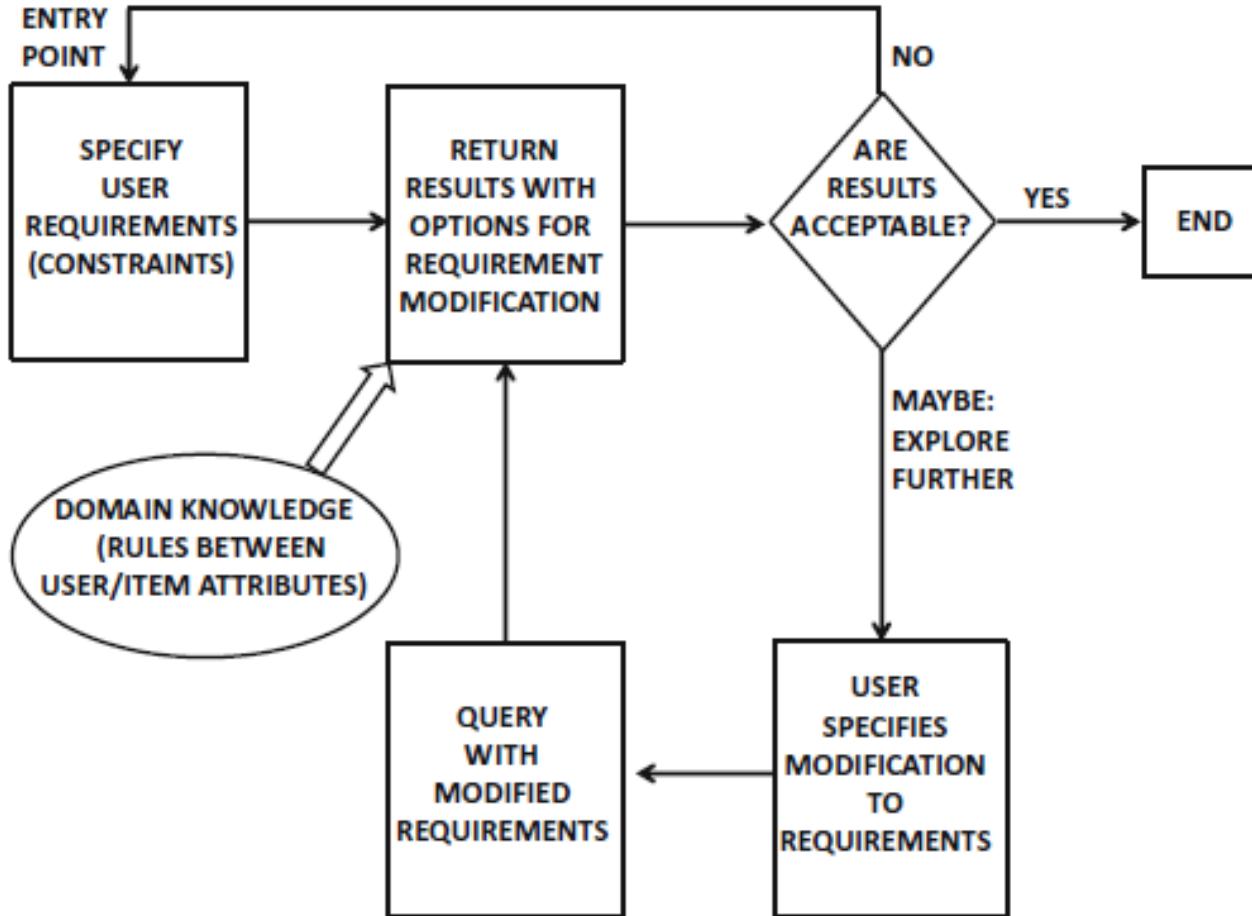
## Knowledge-Based Recommender Systems

---

- Critiquing recommender systems are naturally designed for case-based recommender systems, because one critiques a specific case in order to arrive at the desired outcome.
- A search-based system can be used to set up user requirements for constraint-based recommenders.

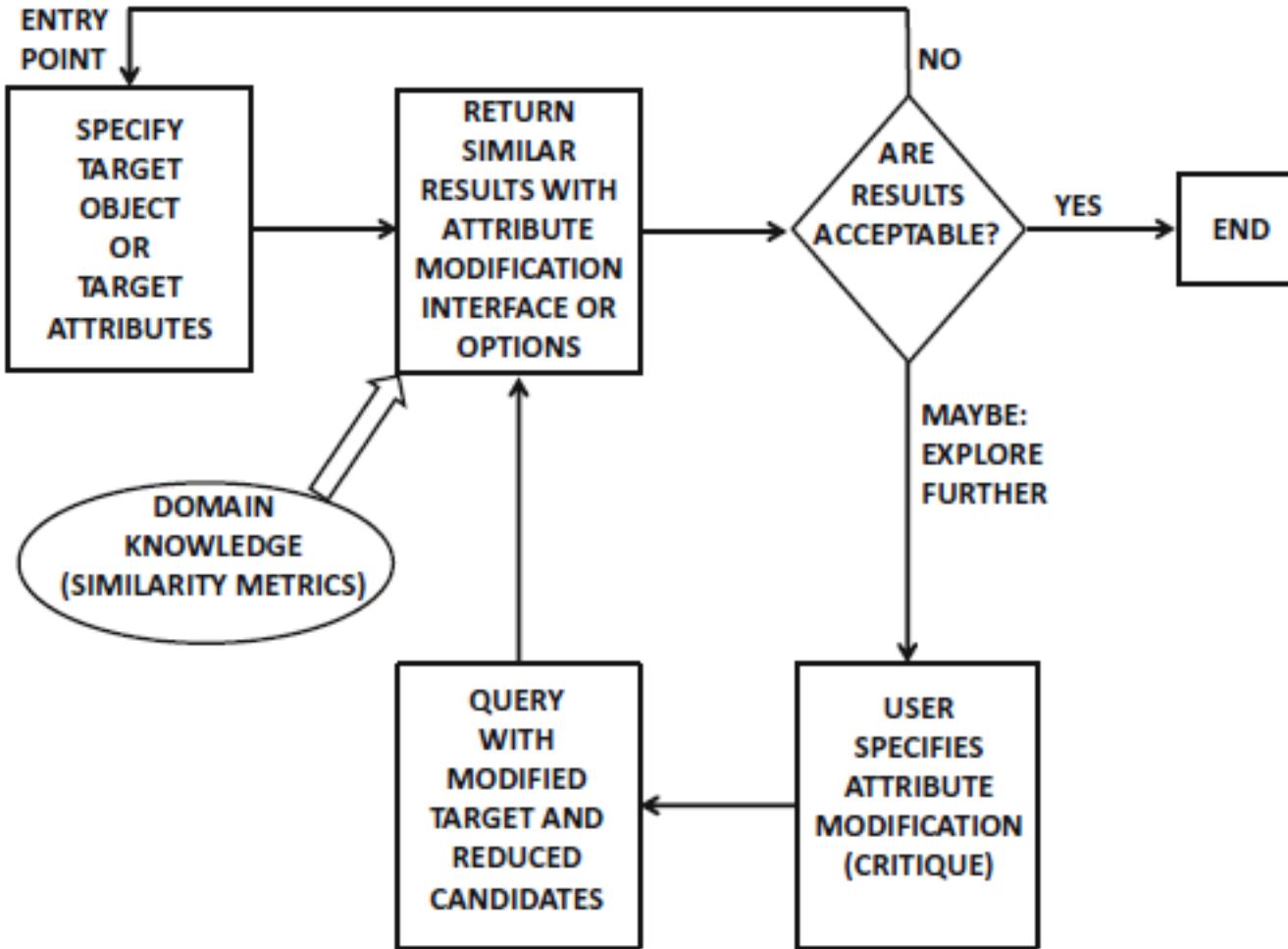
## Knowledge-based Recommendation system

Constraint-based Recommender systems.



## Knowledge-based Recommendation system

Case-based Recommender systems.



## Knowledge-based Recommendation system

### Difference between Constraint-based and Case-based Recommender systems.

- In constraint-based systems, specific requirements or **constraints are specified** by the user.
- The Original query is modified by addition, deletion, modification, or relaxation of the original set of user requirements.
- Users are not in a position to exactly state their requirements up front in a complex product domain, this problem is partially addressed through a knowledge-base of rules, which map user requirements to product attributes.
- In case-based systems, **specific targets or cases are specified**.
- Either the target is modified through user interaction, or the search results are pruned through the use of directional critiques.
- This problem is addressed through a conversational style of critiquing.

## Knowledge-based recommender systems types

---

Examples of attributes in a recommendation application for buying homes.

Item-Id	Beds	Baths	Locality	Type	Floor Area	Price
1	3	2	BTM	Town House	1600	220,000
2	5	2.5	JP	Split-level	3600	973,000
3	4	2	RT	Ranch	2600	630,000
4	2	1.5	MAJESTIC	Condo	1500	220,000
5	4	2	Dollars	Colonial	2700	430,000

## Knowledge-Based Recommender Systems

---

- Suggests products based on **inferences** about a user's needs and preferences
- **Functional knowledge:** about how a particular item meets a particular user need
- The **user model** can be any knowledge structure that supports this inference
- A query, i.e., the set of preferred features for a product
- A case (in a case-based reasoning system)
- An adapted similarity metric (for matching)
- A part of an ontology
- **There is a large use of domain knowledge encoded in a knowledge representation language/approach.**

## Knowledge-Based Recommender Systems

5

### digital camera product advisor

Find by: Product Use | [Product Features](#)

I need photo quality high enough for... [More Info](#)

- 5" x 7" prints (2 megapixels)
- 8" x 10" prints (4 megapixels)
- 11" x 14" prints (6 megapixels)
- No preference

My camera should fit inside a... [More Info](#)

- Shirt pocket
- Backpack
- Waist pack
- No preference

I prefer cameras that have an Epinions.com rating of at least [--select--](#)

--select--

[GET RESULTS](#)

I want to spend... [More Info](#)

From \$  up to \$

I want to zoom in on subjects across a... [More Info](#)

- Small room (8 ft. away)
- Living room (15 ft. away)
- Backyard (35 ft. away)
- No preference

My preferred brands... [More Info](#)

select all that apply

<input type="checkbox"/> Canon	<input type="checkbox"/> Fujifilm	<input type="checkbox"/> Kodak
<input type="checkbox"/> Nikon	<input type="checkbox"/> Olympus	<input type="checkbox"/> Sony

[more brands...](#)

[MORE GUIDANCE](#)

[GET RESULTS](#)

### camcorder product advisor

Find by: Product Use | [Product Features](#)

I need a camcorder for... [More Info](#)

- Occasional & casual recordings
- Home and vacation movies
- Business productions
- No preference

I want to zoom in on subjects across a... [More Info](#)

- Playground (40 ft. away)
- Tennis court (60 ft. away)
- Park (80 ft. away)
- No preference

I prefer camcorders that have an Epinions.com rating of at least [--select--](#)

--select--

[GET RESULTS](#)

I want to spend... [More Info](#)

From \$  up to \$

My camcorder should fit inside a... [More Info](#)

- Shirt pocket
- Backpack
- Waist pack
- No preference

My preferred brands... [More Info](#)

check all -- clear all

<input type="checkbox"/> Canon	<input type="checkbox"/> JVC	<input type="checkbox"/> Panasonic
<input type="checkbox"/> Samsung	<input type="checkbox"/> Sony	

[more brands...](#)

[MORE GUIDANCE](#)

[GET RESULTS](#)

### mp3 player product advisor

Find by: Product Use | [Product Features](#)

My MP3 player (Digital Music Player) needs to be compatible with a... [More Info](#)

select all that apply

<input type="checkbox"/> Windows operating system	<input type="checkbox"/> Mac operating system
---	---

I want my MP3 player to hold... [More Info](#)

- A handful of songs (less than 128 MB)
- A few dozen songs (128 MB - 512 MB)
- Hundreds of songs (512 MB - 5 GB)
- Thousands of songs (5 GB or more)
- No preference

I prefer MP3 players that have an Epinions.com rating of at least [--select--](#)

--select--

[GET RESULTS](#)

I want to spend... [More Info](#)

From \$  up to \$

My preferred brands... [More Info](#)

check all -- clear all

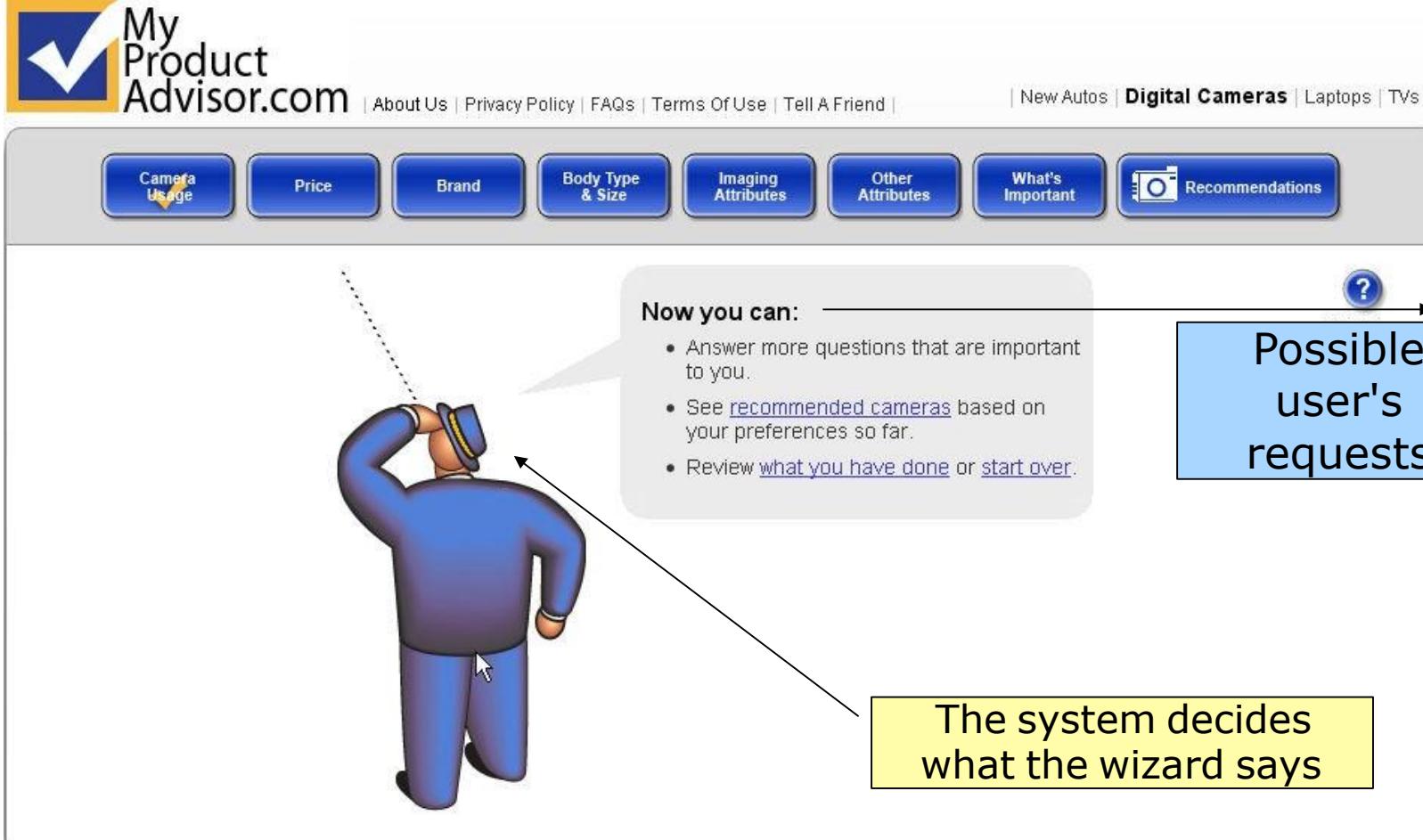
<input type="checkbox"/> Apple/iPod	<input type="checkbox"/> Creative Labs	<input type="checkbox"/> iRiver
<input type="checkbox"/> Lexar	<input type="checkbox"/> RCA	<input type="checkbox"/> Rio

[more brands...](#)

[MORE GUIDANCE](#)

[GET RESULTS](#)

## Knowledge-Based Recommender Systems



## Knowledge-Based Recommender Systems



Welcome to VACATIONCOACH.COM - TRAVEL PLANNING AS UNIQUE AS YOU - Mi... X

File Modifica Visualizza Preferiti Strumenti ?

Indietro ▾ ▶ 🔍 Cerca Preferiti Cronologia ⌂ ⌂ ⌂ ⌂ ⌂

Indirizzo  Vai Collegamenti

BE A MEMBER! : WHY JOIN? : HOW DOES THIS WORK? : ABOUT VACATIONCOACH : FAQs : MEMBERS LOG-IN : HOME

**Someplace Similar.**

Now you can easily find a place that's like a destination you've enjoyed before!

Q1. In which region is the destination you liked?

Europe

Q2. Choose the destination you liked, and we'll find a similar spot.

Paris and Vicinity

let's go!

© VacationCoach Inc., 2000, 2001. All rights reserved.

Operazione completata Internet

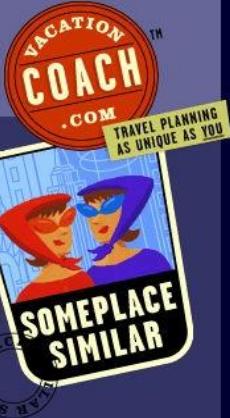
## Knowledge-Based Recommender Systems

Welcome to VACATIONCOACH.COM - TRAVEL PLANNING AS UNIQUE AS YOU - Microsoft Internet Explorer

File Modifica Visualizza Preferiti Strumenti ?  
Indietro ▾ Avanti ▾ Cerca Preferiti Cronologia ▾ ▾ ▾ ▾ ▾

Indirizzo http://www.vacationcoach.com ▾ Vai Collegamenti

BE A MEMBER! WHY JOIN? HOW DOES THIS WORK? ABOUT VACATIONCOACH FAQs MEMBERS LOG-IN HOME

 **SOMEPLACE SIMILAR.** TRAVEL PLANNING AS UNIQUE AS YOU

Now pick a personality type that best describes YOU -- this will help us find similar spots based on things you like.

 <b>CULTURE CREATURE</b> Loves everything cultural - theater, shows, museums... local & historical culture too!	 <b>BEACH BUM</b> Somebody has to lay around on the beach with little umbrellas pitched in their drinks.	 <b>TRAIL TREKKER</b> If it's outdoors - you're there. Hiking, walking... parks, forests, mountains.	 <b>SIGHT SEEKER</b> Always looking for that landmark, event, or attraction.
 <b>CITY SLICKER</b> An urban creature who goes where the action is. Clubs, people... love the pulse of the city.	 <b>AVID ATHLETE</b> Always on the court or the course... always in the game... whatever game it is.	 <b>SHOPPING SHARK</b> Stopped looking for a cure for your shopaholism?	 <b>WINTER WARRIOR</b> Will work for lift ticket. Can become quite abominable if there's no snow on the ground.

{pick one and click!}

Operazione completata Internet

## Knowledge-Based Recommender Systems

Welcome to VACATIONCOACH.COM - TRAVEL PLANNING AS UNIQUE AS YOU - Mic... [Close]

File Modifica Visualizza Preferiti Strumenti ?

Indietro [Back] [Forward] Cerca Preferiti Cronologia [Print] [Help] [Search] [Home]

Indirizzo <http://www.vacationcoach.com> Vai Collegamenti

BE A MEMBER! WHY JOIN? HOW DOES THIS WORK? ABOUT VACATIONCOACH FAQS MEMBERS LOG-IN HOME

If you liked Paris and Vicinity, you'll probably like these destinations as well:

MATCH	DESTINATION	FIND OUT MORE
88%	New York City, NY	<a href="#">more &gt;</a>
87%	Berlin	<a href="#">more &gt;</a>
87%	London	<a href="#">more &gt;</a>
85%	Greater Montreal, QC	<a href="#">more &gt;</a>
85%	Beijing	<a href="#">more &gt;</a>
83%	Washington D.C.	<a href="#">more &gt;</a>
83%	Philadelphia and Lehigh Valley, PA	<a href="#">more &gt;</a>
83%	Chicagoland Region, IL	<a href="#">more &gt;</a>
83%	Hesse (Frankfurt and Vicinity)	<a href="#">more &gt;</a>
82%	Greater Boston, MA	<a href="#">more &gt;</a>

\*BACK TO TOP

Want to try Someplace Similar with a different destination? [Click here](#).

© VacationCoach Inc., 2000, 2001. All rights reserved.

Operazione completata Internet

## Knowledge-Based Recommender Systems

Welcome to VACATIONCOACH.COM - TRAVEL PLANNING AS UNIQUE AS YOU - Micro... x

File Modifica Visualizza Preferiti Strumenti ?

Indietro → Cerca Preferiti Cronologia Vai Collegamenti

Indirizzo http://www.vacationcoach.com

BE A MEMBER! WHY JOIN? HOW DOES THIS WORK? ABOUT VACATIONCOACH FAQs MEMBERS LOG-IN HOME

**SOMEPLACE SIMILAR** 1 FIND ME A PLACE LIKE... 2 ABOUT THIS DESTINATION

**About New York City, NY**

**Recommended for:** Culture Creature **Overall Score:** 99%

**Cost:** (per person/per day) \$364-793  
-- meals and lodging

**why?** Find out why we recommended this place for you.

Go back to [Find Me a Place Like...](#)

**OVERVIEW**  
[YOUR PROS](#)  
[YOUR CONS](#)  
[NOTEWORTHY](#)  
[EVENTS](#)  
[DINING](#)  
[LODGING](#)  
[GETTING THERE](#)

**Overview**  
Where to, Mack? Central Park? You got it. First time to the Big Apple? Well, that's the Manhattan skyline over there -- \$24 in glass beads. The deal of the last millennium, I call it. Then, of course, we have Queens, da Bronx, Staten Island, and Brooklyn, where yours truly was born. In these five boroughs you'll find more landmarks, history, museums, restaurants, shopping, and people than I got problems. Can I name one of each? With my eyes closed. Relax, Mack! It's just a figure of speech. The Empire State Building, The American Museum of Natural History, the Metropolitan Museum of Art, the Carnegie Deli, Bergdorf's, and Sy Glickman. He lives on 86th and Amsterdam. I see you like to be entertained. Well, for you we got theater, nightlife (and I mean all night, Mack), music, and sports. Where? You don't get out much, huh? Ever hear of Broadway, Times Square, Lincoln Center, and the Bronx Bombers? The Yankees. Riight -- I see your meds are kickin' in. We also got the NFL, NHL, NBA, bocci ball in Little Italy, and ping-pong in Chinatown. Say what? You like multiculturalism? You mean who lives here, right? EV-ER-Y-BOD-Y Name a country and you have a little piece of New York. OK, my friend, we're here. That'll be 80 bucks. It seems expensive? Welcome to New York, sweetheart!

**Principal Cities**  
New York City

 **BOOK LODGING**

 **BOOK AIRFARE**

http://www.vacationcoach.com/visitor/request/recommend/aboutdest.jhtml?tc Internet

## Knowledge-Based Recommender Systems

Trip.com - Vacation and Travel Destinations - Microsoft Internet Explorer

Welcome! Tell us what you think about our new site.

Hurry, the London for FREE sale ends January 16th! ►► book now! BRITISH AIRWAYS

Home Flights Lodging Condo Rentals Cars Vacation Packages Cruises Last Minute Trips Travel Resources My Profile My Trips Register/Log In | Help 1.800.TRIp.COM

**Travel Resources**

- > Trip Coach
- > FlightTracker
- > Driving Directions
- > Street Maps
- > Destination Guides
- > Airport Maps
- > Airport Delays
- > Weather
- > Travel Tips
- > Int'l Calling Codes
- > Wireless

**Trip Coach**

People are as different as the trips they take. That's why Trip Coach finds destinations for you based on your travel interests. Select a personality or create your own, and we'll find destinations that are great for you.

Select the personality below that best describes you.

 **WINTER WARRIOR**  
All you need on your trip is snow. Skiing, snow boarding, and hanging out at the lodge mark your final destination.

 **SPORTS ENTHUSIAST**  
Whether spectator or participant, your ideal trip involves anything sports-related □ golf, tennis, baseball, football, and everything in between.

 **SIGHT SEEKER**  
You revel in trips that keep you busy searching for the next tour, attraction, or landmark.

 **SEASONED SHOPPER**  
Your motto is "shop 'til you drop." For you, traveling is all about finding the best shops and bargains in town.

 **OUTDOOR ADVENTURER**  
The great outdoors and all that goes with it - hiking, biking, kayaking, canoeing, skiing, exploring - is your idea of a perfect getaway.

 **FAMILY TRAVELER**  
From amusement parks to festivals to outdoor fun, you love to travel with your children, or you're just a kid at heart. Either way, your trip is usually playful and carefree.

 **CULTURE CONNOISSEUR**  
Your perfect destination offers an abundance of art, architecture, galleries, and theaters.

 **BEACH BUM**  
Your ideal trip revolves around enjoying the latest water sports, sipping tropical drinks, and working on your tan.

If you did not find a personality that fits you,  
 Build your own travel personality.

**CONTINUE**

Trip.com

## Knowledge-Based Recommender Systems

**TripHop Technologies - TripMatcher - Microsoft Internet Explorer**

File Modifica Visualizza Preferiti Strumenti ?  
Indietro ▾ ▷ Cerca Preferiti Cronologia ▶ Vai Collegamenti  
Indirizzo http://www.triplohop.com>Showcase/TripMatcher/searchPa ▾ Vai

Wednesday, September 26, 2001

**TripMatcher™**

Welcome to TripMatcher™, the Web's first vacation advisor. We have researched all the major resorts for you. Answer a few simple questions, and we'll suggest the ski resort that best matches your preferences.

No time to answer? Click here ...

**Activities**

What do you enjoy?  
 Adventure Sports  
 Relaxing  
 Dining Out  
 Leisure Activities  
 Nightlife  
 Shopping  
 Sights & Culture  
 Theme Parks & Zoos  
 Water Sports  
 Winter Sports

**Optional Criteria**

You may refine your search.  
 Avoid Crowded Destinations  
 Avoid Jet Lag  
 Choose Weather Conditions  
 Good Safety Conditions  
 Improve A Foreign Language  
 Select A Specific Environment  
 Set A Budget  
 Specify A Region  
 Traveler Support  
 Traveling With Companions

**Timing**

When are you leaving?  
**Late November**

How long will you be gone?  
**One Week**

**Departure City**

Please choose your gateway.  
**Washington Dc**

Operazione completata

Internet

Trip.com

## Knowledge-Based Recommender Systems

**TripleHop Technologies - TripMatcher - Microsoft Internet Explorer**

File Modifica Visualizza Preferiti Strumenti ?  
Indietro ▾ ➡ Cerca Preferiti Cronologia ⌂ Vai Collegamenti  
Indirizzo http://www.triplehop.com>Showcase/TripMatcher/search

**TRIPLE HOP TECHNOLOGIES**

Home Company Solutions Product Demos Clients Partners Press Room Contact Us

Wednesday, September 26, 2001

**Tell us more !**

Give us a better idea about what you like. Feel free to skip any question, but the more you tell us, the better our recommendation will be.

**Adventure Sports**

Any favorite adventure sports?  
 Children'S Adventure Sports  
 Hiking  
 Mountain Biking  
 Paragliding  
 Rock Climbing  
 Whitewater Rafting

**Relaxing**

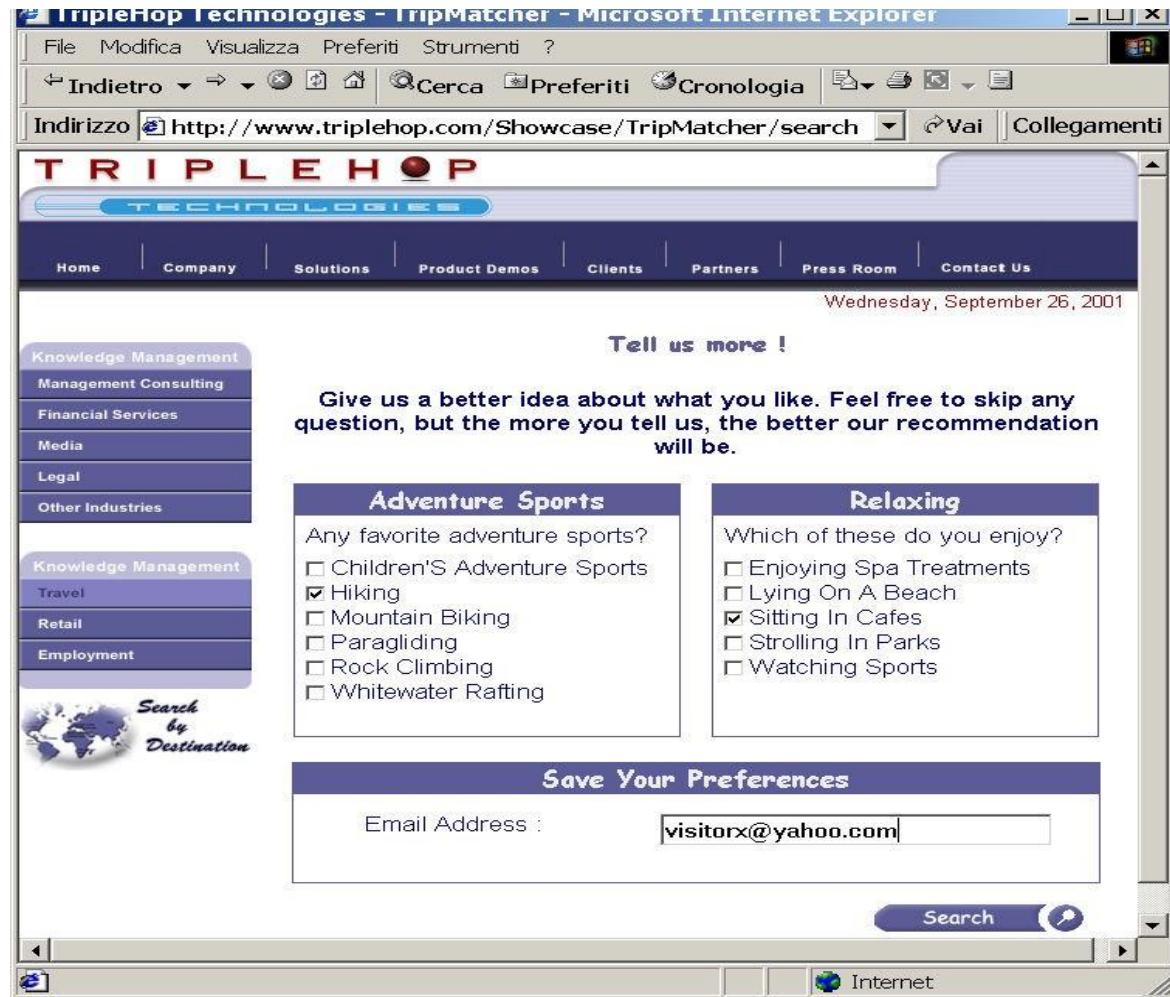
Which of these do you enjoy?  
 Enjoying Spa Treatments  
 Lying On A Beach  
 Sitting In Cafes  
 Strolling In Parks  
 Watching Sports

**Save Your Preferences**

Email Address : visitorx@yahoo.com

Search

Internet



Trip.com

## Knowledge-Based Recommender Systems

**TripHop Technologies - TripMatcher - Microsoft Internet Explorer**  
File Modifica Visualizza Preferiti Strumenti ?  
Indietro Cerca Preferiti Cronologia Vai Collegamenti  
Indirizzo http://www.triphop.com>Showcase/TripMatcher/resultPage.asp

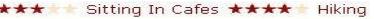
**TRIPLE HOP TECHNOLOGIES**

Home Company Solutions Product Demos Clients Partners Press Room Contact Us

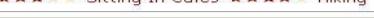
Wednesday, September 26, 2001

We found 15 matches for you. To read more or book a vacation, please click on the destination name or the picture.

New Search 

**1. Monterey Bay - California**  70%  
Like the Pacific Ocean that runs up and down the **Monterey Bay** and Big Sur coastline, it's almost impossible to define and contain this area. There are many towns, each with a distinct flavor. Monterey, with its seal, sea otter, and whale fille ...more  
( Flying time : 5 hours )  
 Sitting In Cafes  Hiking

**2. Salem And The North Shore - Massachusetts**  68%  
Imagine for a moment that you could disintegrate yourself, Willy Wonka or Star Trek style, into little bits, and then transplant yourself whole onto the pages of your favorite New England coffee table book...  
...Right ...more  
( Flying time : 2 hours )  
 Sitting In Cafes  Hiking

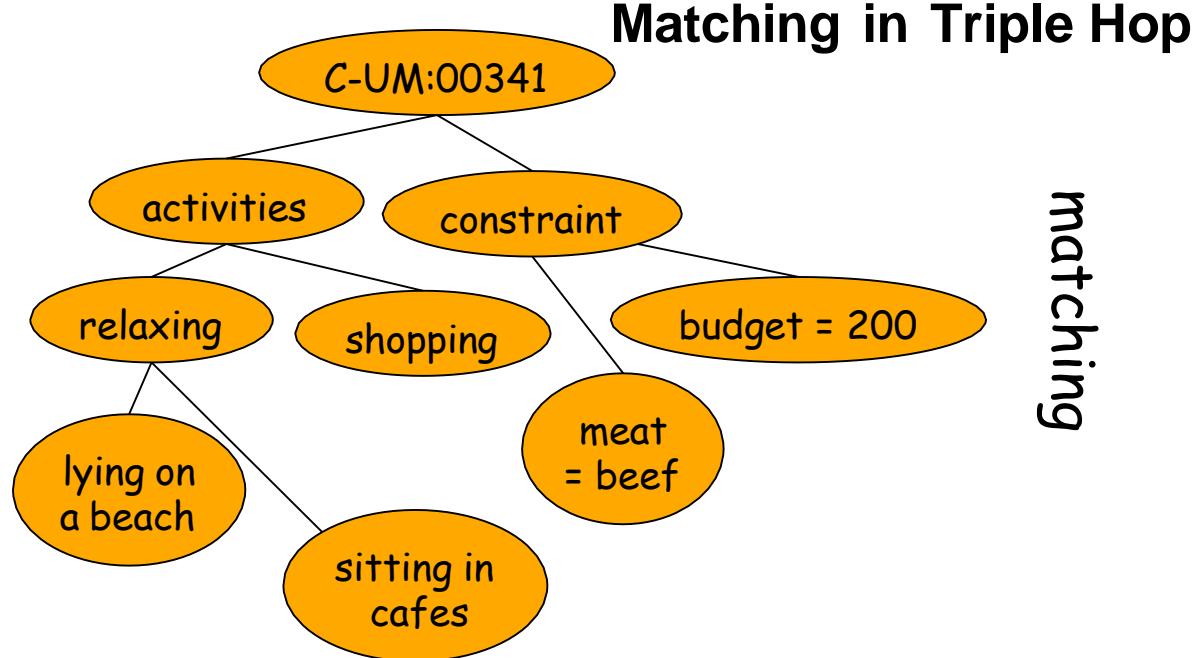
**3. Marin County - California**  67%  
Often dubbed the "Bay Area's Backyard," **Marin County** is an area of recreational and geographic diversity. It is worth visiting for its location alone, as it is bordered by the Pacific Ocean, the Golden Gate Bridge, the San Francisco Bay, and Wine County.  
...more  
( Flying time : 7 hours )  
 Sitting In Cafes  Hiking

Internet

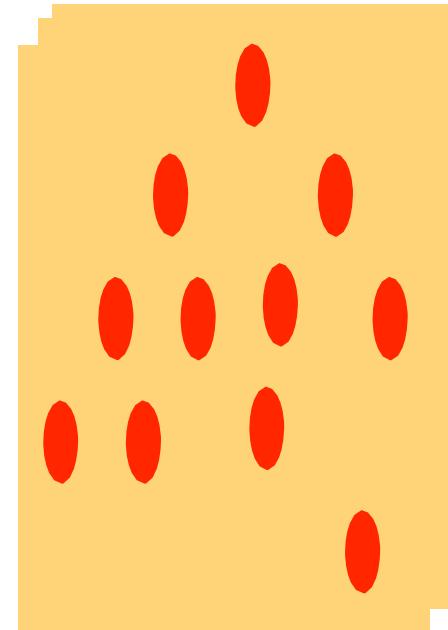
Trip.com

## Knowledge-Based Recommender Systems

Example: TripleHop



Catalogue of Destinations



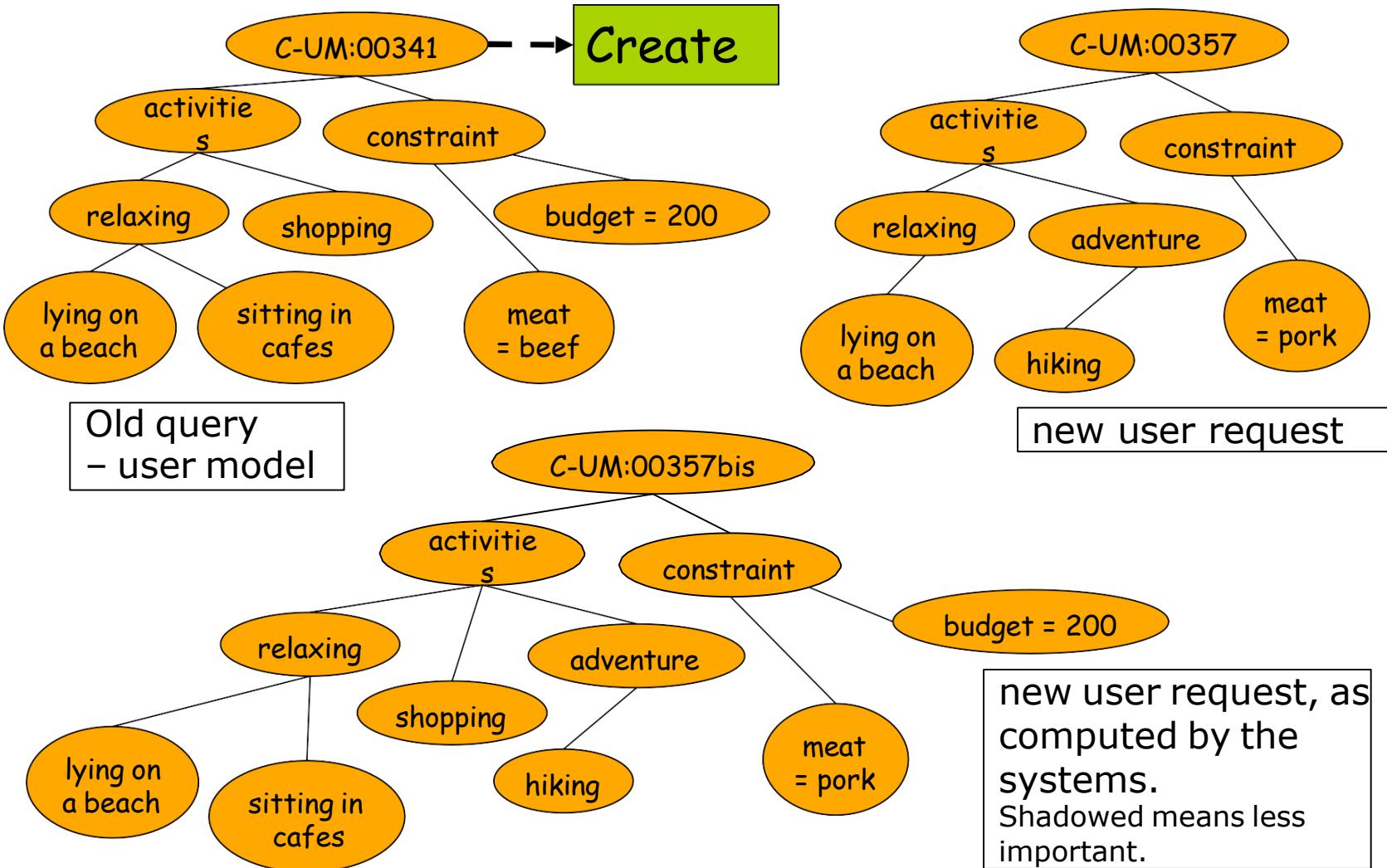
## Knowledge-Based Recommender Systems

---

### TripleHop and Content-Based RS

- The content (destination description) is exploited in the recommendation process
- A classical Content-Based method would have used a “simpler” content model ,e.g., keywords or TF-IDF
- Here a more complex **knowledge structure** – a tree of concepts – is used to model the product (and the query)
- The query is the user model and it is acquired every time the user asks for a new recommendation - (not exactly, more details later)
- Stress on ephemeral needs rather than building a persistent user model
- Typical in Knowledge-Based RS, they are more focused on ephemeral users – because Collaborative Filtering and Content-Based methods cannot cope with that users.

## Knowledge-Based Recommender Systems



**Learning User Profile:  
Query mining**

## Knowledge-Based Recommender Systems

---

### Query Augmentation

- Personalization in search is not only “information **filtering**”
- **Query augmentation:** when a query is entered it can be compared against contextual and individual information to refine the query
  
- Ex1: If the user is searching for a restaurant and enter a keyword “Thai” then the query can be augmented to “Thai food”
- Ex2: If the query “Thai food” does not retrieve any restaurant the query can be refined to “Asian food”
- Ex3: If the query “Asian food” retrieves too many restaurant, and the user searched in the past for “Chinese” food the query can be refined to “Chinese food”.

## Knowledge-Based Recommender Systems

---

### Query Augmentation in TripleHop

1. The current query is **compared** with **previous queries** of the **same user**
2. Preferences expressed in past (similar) queries are identified
3. A new query is built by **combining the short term preferences** contained in the query with the “**inferred**” preferences extracted from the persistent user model (past queries)
4. When the query is matched against an item (destination) if two destinations have the **same degree of matching for the explicit preferences** then the “**inferred**” preferences are used to break the tie
  - This is another example of the **cascade** approach
  - The two combined RS are based on the same knowledge but with two definitions of the user model.

## Knowledge-Based Recommender Systems

---

### What is Case Based Reasoning ?

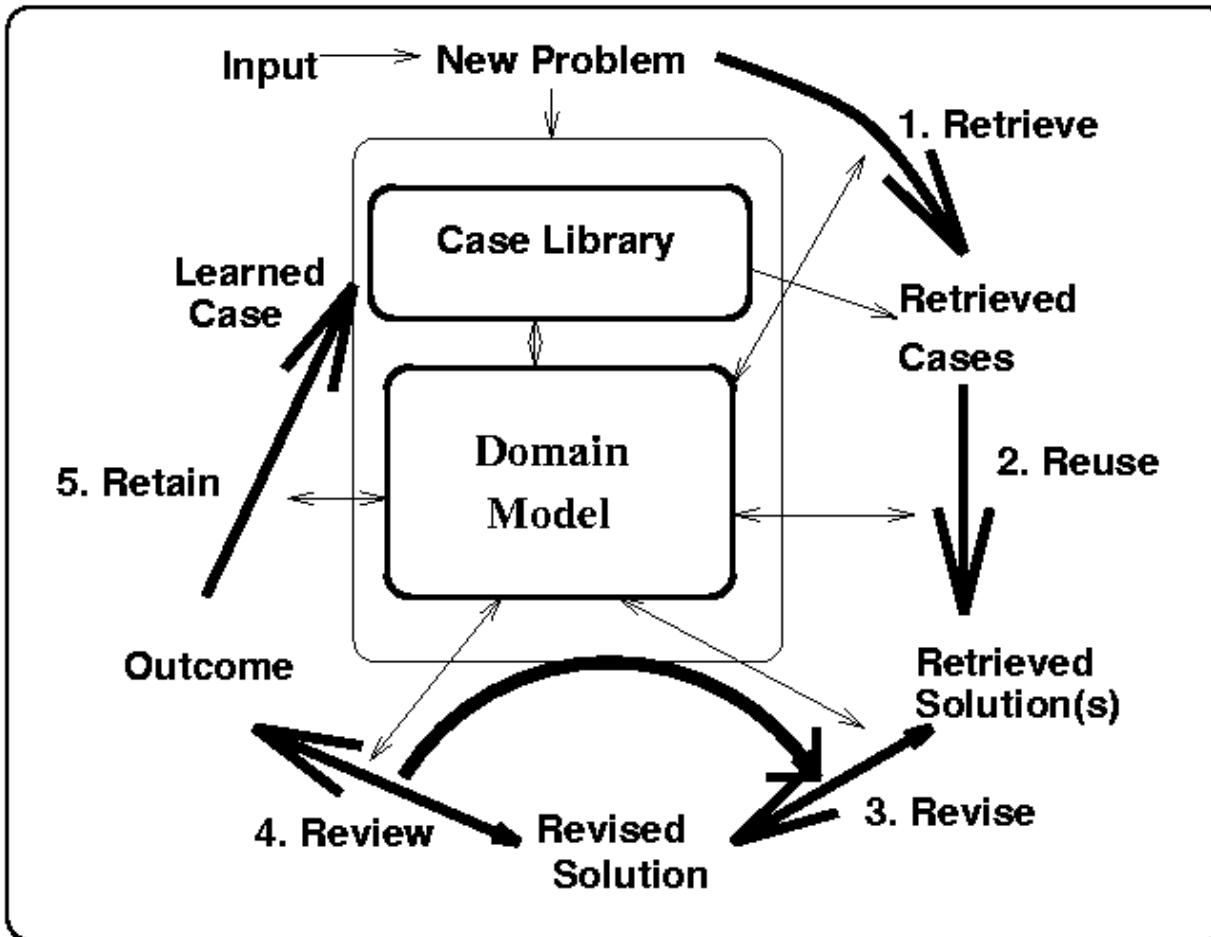
- **A case-based reasoner solves new problems by adapting solutions that were used to solve old problems** (Riesbeck & Shank 1989)

CBR problem solving process:

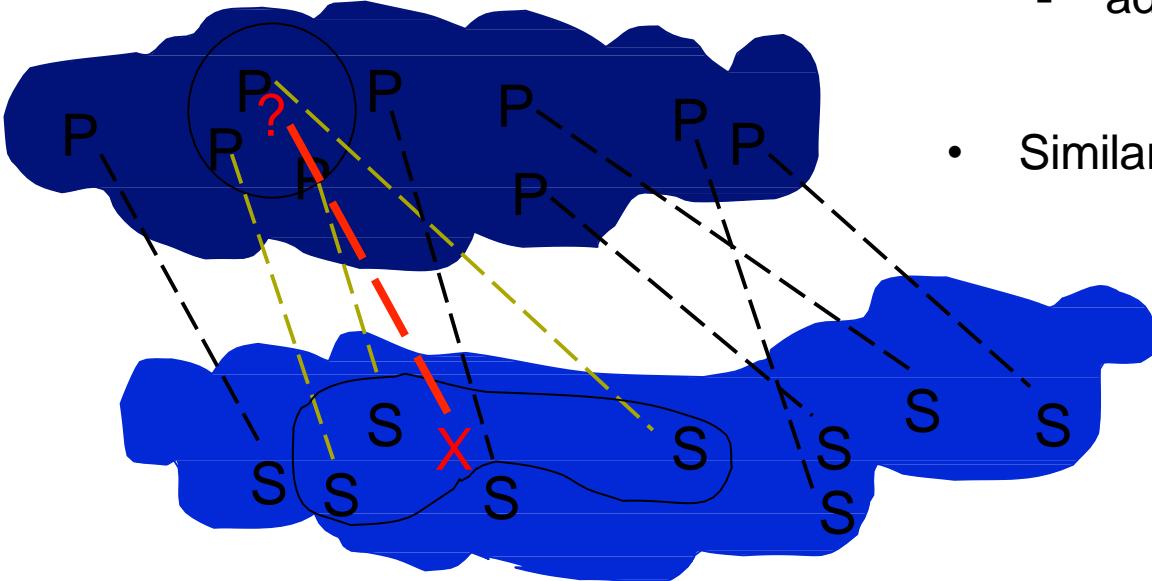
- Store previous experiences (cases) in memory to solve new problems
- Retrieve from the memory similar experience about similar situations
- Reuse the experience in the context of the new situation: complete or partial reuse, or adapt according to differences
- Store new experience in memory (learning)

## Knowledge-Based Recommender Systems

### Case-Based Reasoning



### CBR Assumption



- New problem can be solved by
  - retrieving similar problems
  - adapting retrieved solutions
- Similar problems have similar solutions

## Knowledge-Based Recommender Systems

---

### Examples of CBR

- Classification: “The patient’s ear problems are like this prototypical case of otitis media”
- Compiling solutions: “Patient N’s heart symptoms can be explained in the same way as previous patient D’s”
- Assessing values: My house is like the one that sold down the street for \$250,000 but has a better view
- Justifying with precedents: “This Missouri case should be decided just like Roe v. Wade where the court held that a state’s limitations on abortion are illegal”
- Evaluating options: “If we attack Cuban/Russian missile installations, it would be just like Pearl Harbor”

## Knowledge-Based Recommender Systems

---

### Instance-based learning – Lazy Learning

- One way of solving tasks of approximating discrete or real valued target functions
- Have training examples:  $(x_n, f(x_n))$ ,  $n=1,..$

Key idea:

- Just **store** the training examples
- When a test example is given then **find the closest matches**
- **Use the closest matches to guess** the value of the target function on the test example.

## Knowledge-Based Recommender Systems

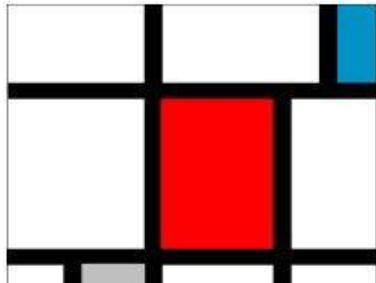
---

### The distance between examples

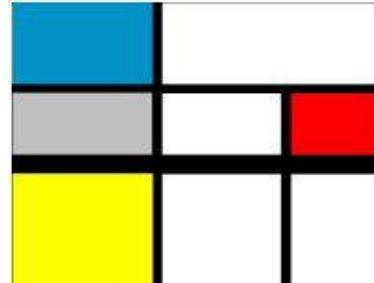
- We need a **measure of distance** (or similarity) in order to know who are the neighbors
- Assume that we have T attributes for the learning problem. Then one example point x has elements  $x_t$ ,  $t=1, \dots, T$
- The distance between two points x and y is often defined as the **Euclidean** distance:

$$d(x, y) = \sqrt{\sum_{t=1}^T [x_t - y_t]^2}$$

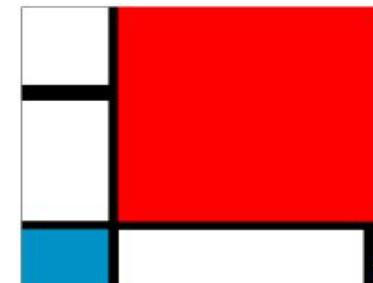
## Knowledge-Based Recommender Systems



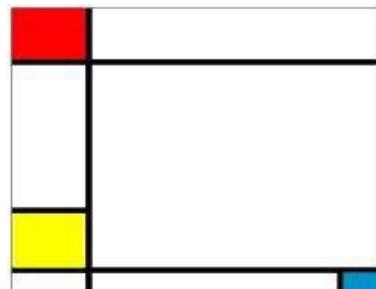
no



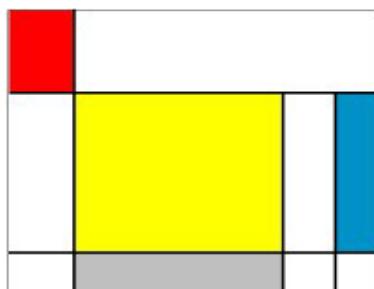
no



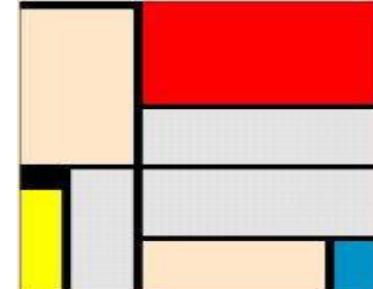
yes



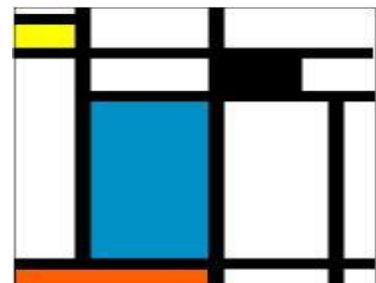
yes



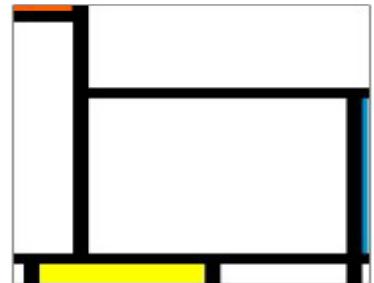
no



yes



no



?

## Knowledge-Based Recommender Systems

### Training data

Number	Lines	Line types	Rectangles	Colours	Mondrian?
1	6	1	10	4	No
2	4	2	8	5	No
3	5	2	7	4	Yes
4	5	1	8	4	Yes
5	5	1	10	5	No
6	6	1	8	6	Yes
7	7	1	14	5	No

### Test instance

Number	Lines	Line types	Rectangles	Colours	Mondrian?
8	7	2	9	4	

## References

---

### Text Book:

“Recommender Systems, The Text Book by Charu C. Aggarwal,  
Springer 2016 Section 1 and Section 2.

# DATA ANALYTICS

## Image Courtesy

---

<http://www.mmds.org/mmds/v2.1/ch09-recsys1.pptx>

[https://www.researchgate.net/publication/287952023\\_Collaborative\\_Filtering\\_Recommender\\_Systems](https://www.researchgate.net/publication/287952023_Collaborative_Filtering_Recommender_Systems)

<http://cs229.stanford.edu/proj2014/Rahul%20Makhijani,%20Saleh%20Samaneh,%20Megh%20Mehta,%20Collaborative%20Filtering%20Recommender%20Systems.pdf>

<https://www.scribd.com/presentation/414445910/CS548S15-Showcase-Web-Mining>

<https://towardsdatascience.com/image-recommendation-engine-leverage-transfert-learning-ec9af32f5239>

<http://elico.rapid-i.com/recommender-extension.html>

<https://www.youtube.com/watch?v=h9gpuJFF-0>



---

## THANK YOU

---

**Jyothi R.**  
Assistant Professor,  
Department of Computer Science  
[jyothir@pes.edu](mailto:jyothir@pes.edu)





# DATA ANALYTICS

## Unit 4: Recommendation Systems

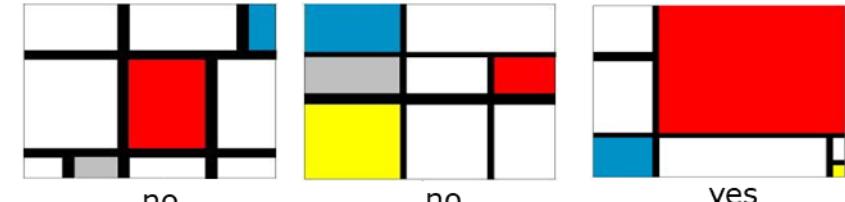
---

**Jyothi R.**  
Department of Computer Science  
and Engineering

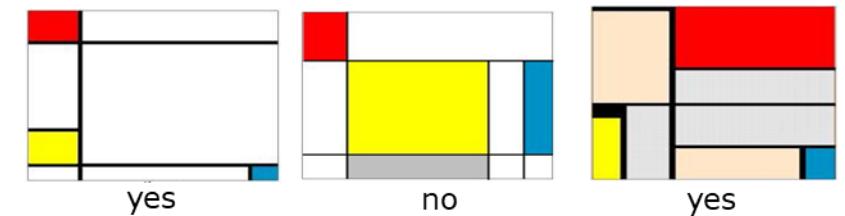
## Knowledge-Based Recommender Systems

	Lines	Lines Types	Rect	Colors	Class	Distance to test
Train1	4	2	8	5	no	3,32
Train2	5	2	7	4	yes	2,83
Train3	5	1	8	4	yes	2,45
Train4	5	1	10	5	no	2,65
Train5	6	1	8	6	yes	2,65
Train6	7	1	14	5	no	5,20
test	7	2	9	4		
Train1	-0,32	0,32	-0,11	0,06	no	0,80
Train2	-0,08	0,32	-0,21	-0,28	yes	0,52
Train3	-0,08	-0,16	-0,11	-0,28	yes	0,69
Train4	-0,08	-0,16	0,08	0,06	no	0,77
Train5	0,16	-0,16	-0,11	0,39	yes	0,86
Train6	0,40	-0,16	0,47	0,06	no	0,76
test	0,40	0,32	-0,02	-0,28		

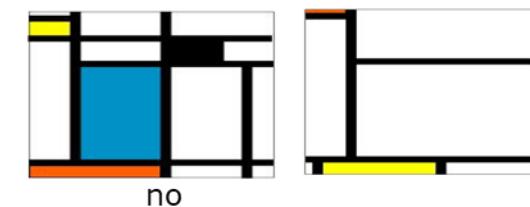
Feature values are not normalized



Feature values are normalized



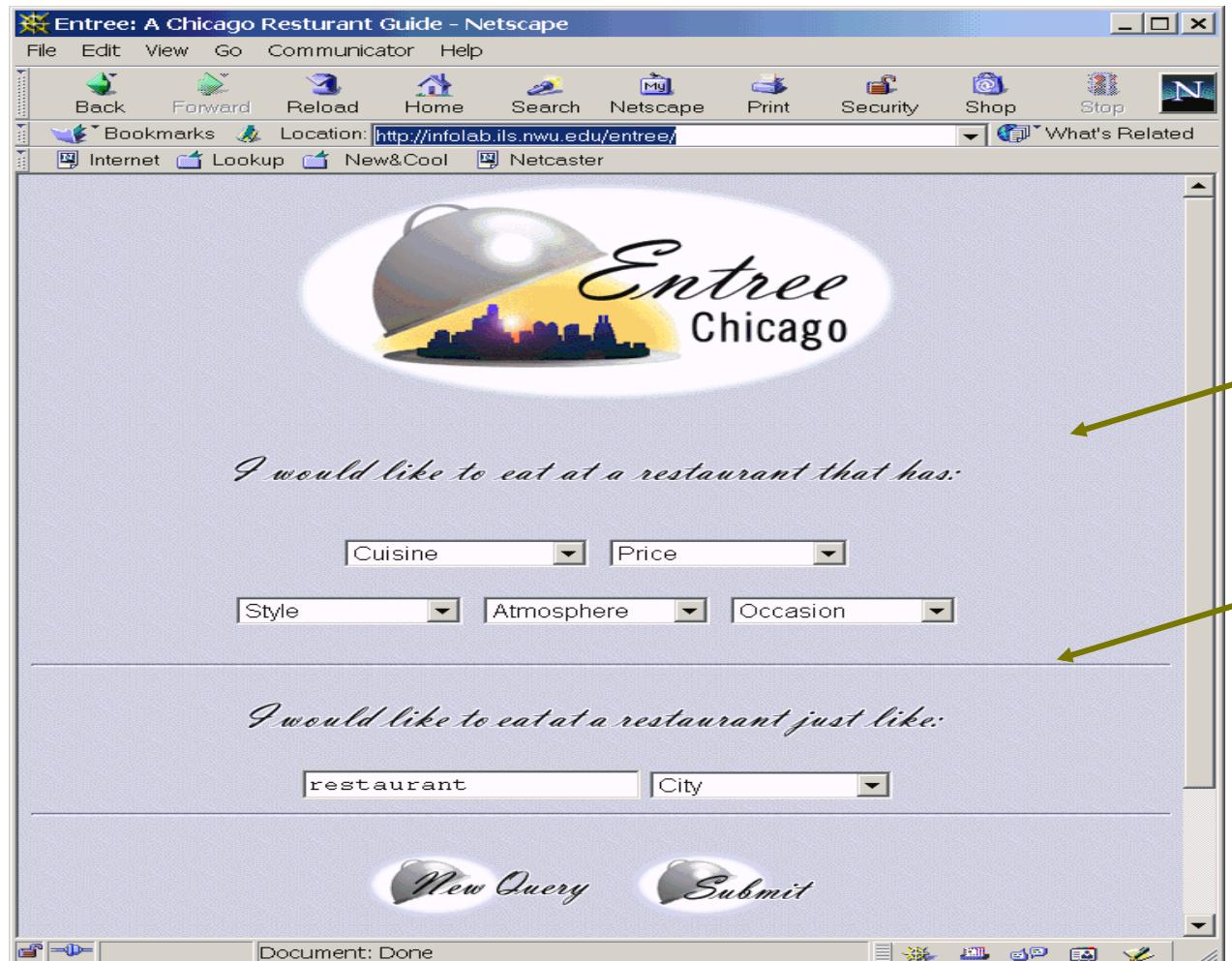
What is the difference between this feature value normalization and vector Normalization in IR?



$$x' = (x - \text{avg}(X)) / 4 * \text{stdev}(X), \text{ where } x \text{ is a feature value of the feature } X$$

## Knowledge-Based Recommender Systems

### Example of CBR Recommender System



- Entree is a restaurant recommender system – it finds restaurants:
1. matching some user goals (case features)
  2. or similar to restaurants the user knows and likes

## Knowledge-Based Recommender Systems

---

### The Product is the Case

- In Entrée a case is a restaurant – **the case is the product**
- The **problem** component is the description of the restaurant given by the user
- The user will input a partial description of it – this is the only difficulty
- The **solution** part of the case is the restaurant itself – i.e. the name of the restaurant
- The assumption is that the needs of the user can be modeled as the features of the product description ....

## Knowledge-Based Recommender Systems

### Partial Match

The screenshot shows a Netscape browser window with the title "Entree: A Chicago Restaurant Guide - Netscape". The URL in the address bar is <http://infolab.ils.nwu.edu/cgi-bin/entree/query.pl>. The main content area displays "Entree Results" with a logo. Below it, a section titled "We recommend:" lists "Dave's Italian Kitchen" with a map link. The restaurant's address is 906 Church St. (bet. Ridge & Sherman Aves.), Evanston, 708-864-6000. It is categorized as "Italian" and "below \$15". Descriptions include "Fair Decor, Excellent Service, Excellent Food, No Reservations, Weekend Brunch, Carry in Wine and Beer, Wheelchair Access, Long Drive". Below this, there are two rows of circular icons with text labels: "less \$\$", "nicer", "cuisine", "traditional", "creative", "livelier", and "quieter". At the bottom, under "For other suggestions, select:", there is a grid of restaurant names: Dave's Italian Kitchen, Dancing Noodles Cafe, Anna Maria Pasteria, Gusto Italiano, La Sorella di Francesca, Mia Francesca, Carlucci, Village, Rosebud, Spavone's Seven Hills, and Salvatore's.

In general, only a subset of the preferences will be matched in the recommended restaurant.

## Knowledge-Based Recommender Systems

### Nearest Neighbor

Entree: A Chicago Restaurant Guide - Netscape

File Edit View Go Communicator Help

Back Forward Reload Home Search Netscape Print Security Shop Stop

Bookmarks Location: <http://infolab.ils.nwu.edu/cgi-bin/entree/query.pl> What's Related

Internet Lookup New&Cool Netcaster

 *Entree Results*

The Washington DC restaurant you chose is:

**Parioli**  
4800 Elm St. (Wisconsin Ave.), Bethesda, MD, 301-951-8600  
Italian \$15-\$30  
Excellent Decor, Excellent Service, Extraordinary Food, Authentic, Catering for Special Events, Takeout Available, Delivery Available, Health Conscious Menus, Dining Outdoors, Parking/Valet, Private Rooms Available, Private Parties, No Smoking Allowed, Weekend Dining, Wheelchair Access

We recommend:

**Stefani's** ([map](#))  
1418 W. Fullerton Ave. (Southport Ave.), Chicago, 312-348-0111 601 Skokie Blvd. (bet. Dundee & Lake Cook Rds.), Northbrook, 708-564-3950  
Pizza, Italian \$15-\$30  
Excellent Decor, Excellent Service, Excellent Food, Dining Outdoors, Private Rooms Available, Private Parties, Weekend Brunch, Parking/Valet

Document: Done

## Knowledge-Based Recommender Systems

---

### Recommendation in Entre

- The system first selects from the database the set of all restaurants that satisfy the largest number of logical constraints generated by considering the input features type and value
- If necessary, implicitly relaxes the lowest important constraints until some restaurants could be retrieved
- Typically the relaxation of constraints will produce many restaurants in the result set
- Sorts the retrieved cases using a similarity metric
  - this takes into account all the input features.

## Knowledge-Based Recommender Systems

---

### Similarity in Entree

- This similarity metric assumes that the user goals, corresponding to the input features (or the features of the source case), could be sorted to reflect the importance of such goals from the user point of view
- Hence the global similarity metric (algorithm) sorts the products first with respect to the most important goal and then iteratively with respect to the remaining goals (multi-level sort)
- Note: it does not work as a maximization of a Utility-Similarity defined as the sum of local utilities.

## Knowledge-Based Recommender Systems

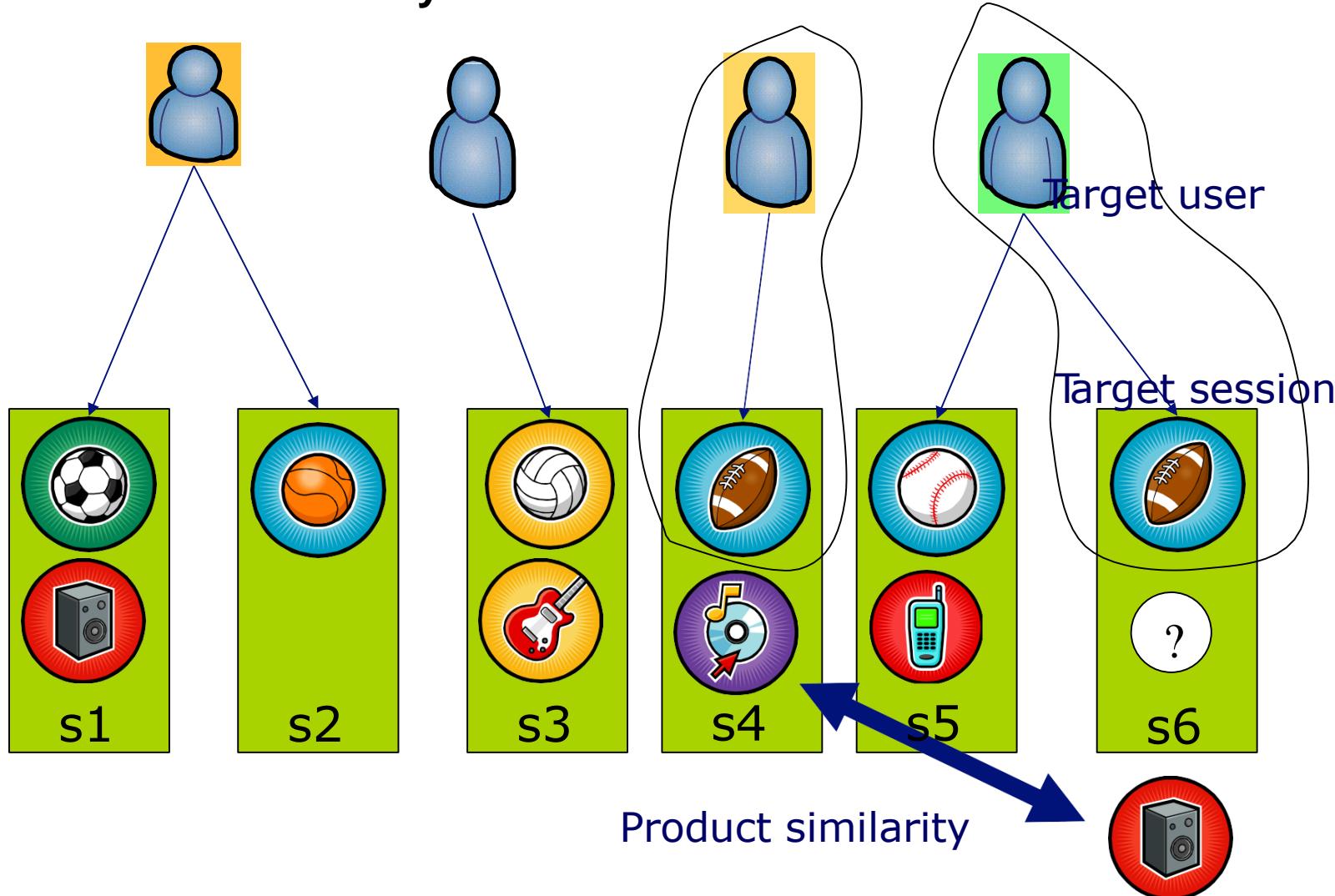
### Example

Restaurant	Price	Cuisine	Atmosphere
Woodys	10	A	A
Gabbana	12	B	B

- If the user query q is: **price=9 AND cuisine=B AND Atm=B**
- And the weights (importance) of the features is: 0.5 price, 0.3 Cuisine, and 0.2 Atmosphere
- The Entrée will suggest Woodys first (and then Gabbana)
- A more traditional CBR system will suggest Gabbana because the similarities are (30 is the price range):
- $\text{Sim}(q, \text{Woodys}) = 0.5 * (1 - 1/30) + 0.3 * 0 + 0.2 * 0 = 0.48$
- $\text{Sim}(q, \text{Gabbana}) = 0.5 * (1 - 3/30) + 0.3 * 1 + 0.2 * 1 = 0.45 + 0.3 + 0.2 = 0.95$

## Knowledge-Based Recommender Systems

### Two-fold Similarity



## Knowledge-Based Recommender Systems

---

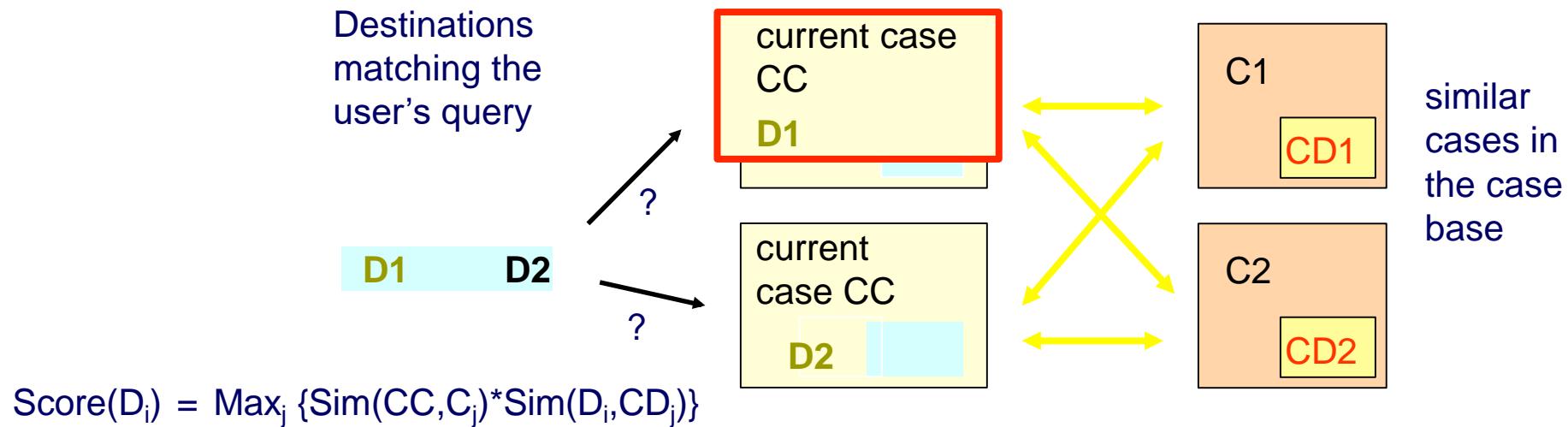
### Rank using Two-Fold Similarity

Given the current session case  $c$  and a set of retrieved products  $R$  (using the interactive query management facility - IQM)

1. retrieve 10 cases ( $c_1, \dots, c_{10}$ ) from the repository of stored cases (recommendation sessions managed by the system) that are most **similar** to  $c$  with respect to the collaborative features
2. extract products ( $p_1, \dots, p_{10}$ ) from cases ( $c_1, \dots, c_{10}$ ) of the same type as those in  $R$
3. For each product  $r$  in  $R$  compute the  $\text{Score}(r)$  as the maximum of the product of a) the similarity of  $r$  with  $p_i$ , the similarity of the current case  $c$  and the retrieved case  $c_i$  containing  $p_i$
4. sort and display products in  $R$  according to the  $\text{Score}(r)$ .

## Knowledge-Based Recommender Systems

## Example: Scoring Two Destinations



Sim(CC,C1)	0.2
Sim(CC,C2)	0.6

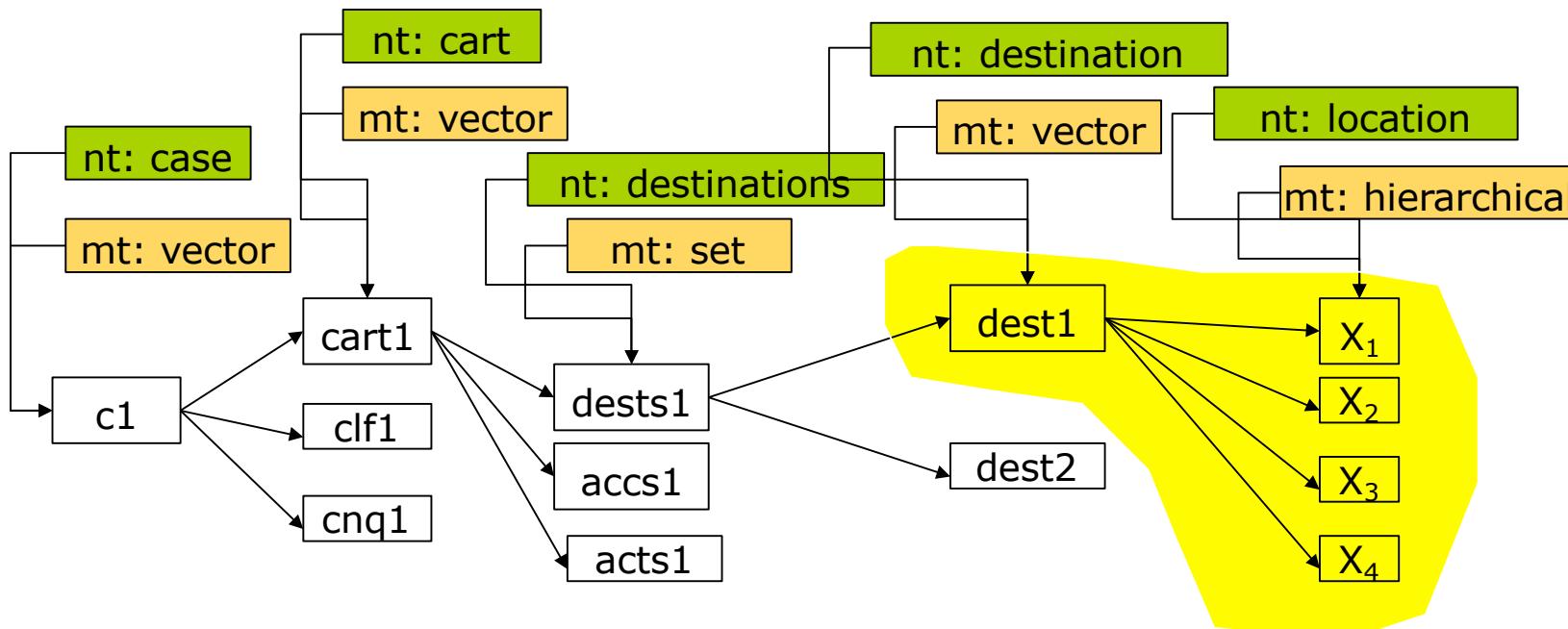
Sim(D1, CD1)	0.4
Sim(D1, CD2)	0.7
Sim(D2, CD1)	0.5
Sim(D2, CD2)	0.3

$$\text{Score}(D1)=\text{Max}\{0.2*0.4, \underline{0.6*0.7}\}=0.42$$

$$\text{Score}(D2)=\text{Max}\{0.2*0.5, 0.6*0.3\}=0.18$$

## Tree-based Case Representation

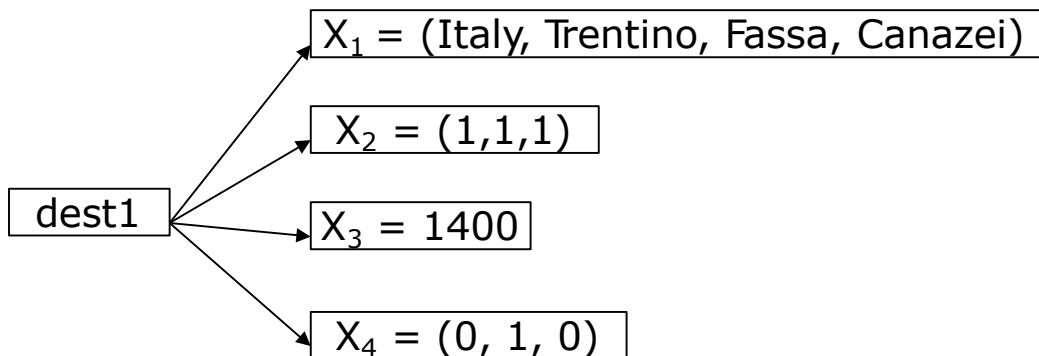
- A case is a rooted tree and each node has a:  
**node-type:** similarity between two nodes in two cases is defined only for nodes with the same node-type
- **Metric type:** node content structure - how to measure the node similarity with another node in a second case



## Knowledge-Based Recommender Systems

### Item Representation

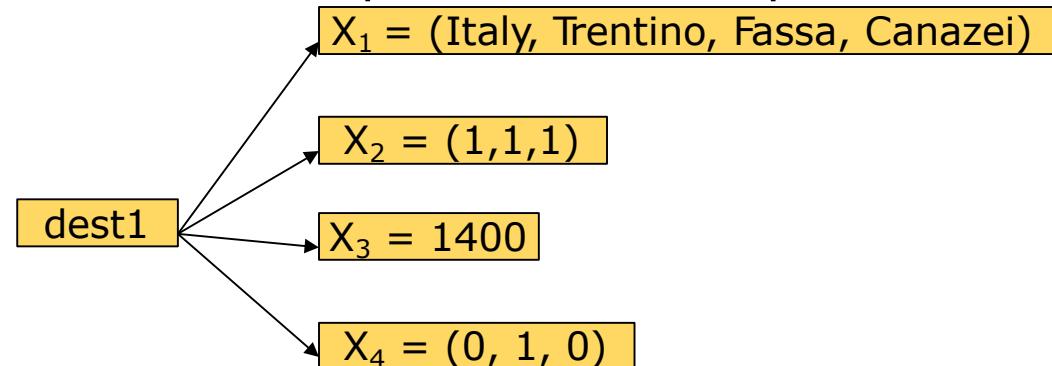
	Node Type	Metric Type	Example: Canazei
X <sub>1</sub>	LOCATION	Set of hierarchical related symbols	Country=ITALY, Region=TRENTINO, TouristArea=FASSA, Village=CANAZEI
X <sub>2</sub>	INTERESTS	Array of Booleans	Hiking=1, Trekking=1, Biking=1
X <sub>3</sub>	ALTITUDE	Numeric	1400
X <sub>4</sub>	LOCTYPE	Array of Booleans	Urban=0, Mountain=1, Rivereside=0



## Knowledge-Based Recommender Systems

## Item Query Language

- For querying purposes items x are represented as simple vector features  $x=(x_1, \dots, x_n)$



(Italy, Trentino, Fassa, Canazei, 1, 1, 1, 1400, 0, 1, 0)

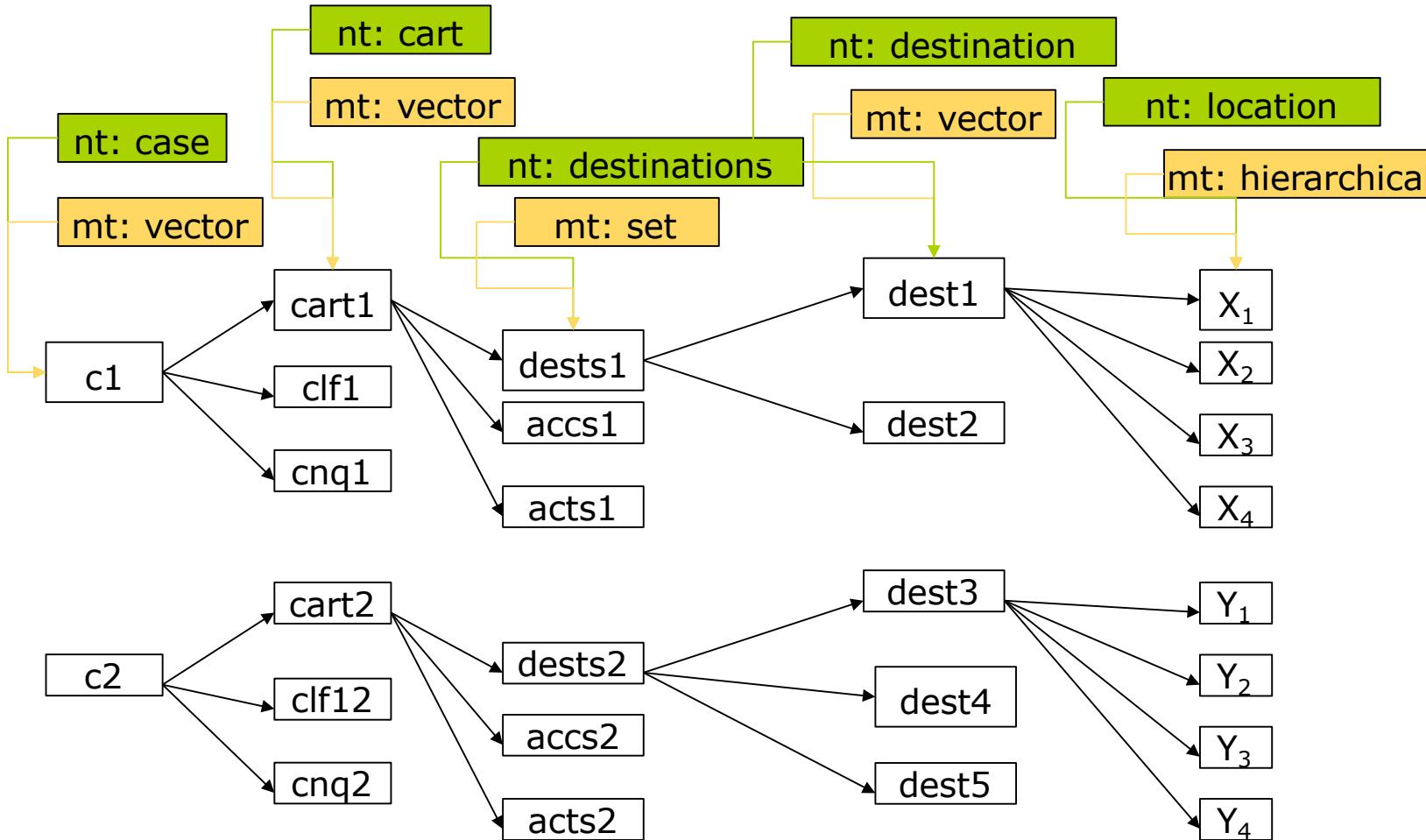
- A query is a conjunction of constraints over features:

$$q=c_1 \wedge c_2 \wedge \dots \wedge c_m \quad \text{where } m \leq n \text{ and}$$

$$c_k = \begin{cases} x_{i_k} = \text{true} & \text{if } x_{i_k} \text{ is boolean} \\ x_{i_k} = v & \text{if } x_{i_k} \text{ is nominal} \\ x_{i_k} \in [l, u] & \text{if } x_{i_k} \text{ is numerical} \end{cases}$$

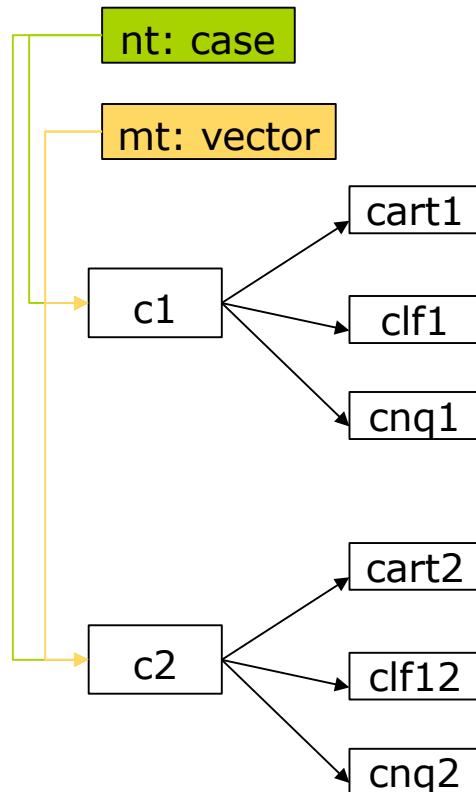
## Knowledge-Based Recommender Systems

### Case Distance



## Case Distance

$$d(c_1, c_2) = \frac{1}{\sqrt{\sum_{i=1}^3 W_i}} \sqrt{W_1 d(cart_1, cart_2)^2 + W_2 d(clf_1, clf_2)^2 + W_3 d(cnq_1, cnq_2)^2}$$



## Knowledge-Based Recommender Systems

---

### CBR Knowledge Containers

1. CBR is a knowledge-based approach to problem solving
2. The knowledge is “contained” into four **containers**
3. **Cases:** the instances belonging to our case base
4. **Case representation language:** the representation language that we decided to use to represent cases
5. **Retrieval knowledge:** the knowledge encoded in the similarity metric and in the retrieval algorithm
6. **Adaptation knowledge:** how to reuse a retrieved solution to solve the current problem.

## Knowledge-Based Recommender Systems

---

### Conclusions

- Knowledge-based systems exploits knowledge to map a user to the products she likes
- KB systems uses a variety of techniques
- Knowledge-based systems requires a big effort in term of knowledge extraction, representation and system design
- Many KB recommender systems are rooted in Case-Based Reasoning
- Similarity of complex data objects is required often required in KB RSs.
- NutKing is a hybrid case-based recommender system
- The case is the recommendation session.

## Knowledge-Based Recommender Systems

---

### Questions

1. What are the main differences between a CF recommender system and a KB RS (such as activebuyers.com or Entree)?
2. What is the role of query augmentation?
3. What is the basic rationale of a CBR recommender system?
4. What is a case in a CBR recommender system such as Entree?
5. How a CBR recommender system learns to recommend?
6. What are the knowledge containers in a CBR RS?
7. What are the main differences between a “classical” CBR recommender system such as Entrée and Nutking?
8. What are the motivations for the introduction of the double- similarity ranking method?
9. What are the types of local similarity metrics used in Nutking?

## References

---

### Text Book:

“Recommender Systems, The text book, Charu C. Aggarwal, Springer  
2016 Section 1.and Section 2.

# DATA ANALYTICS

## Image Courtesy

---

<http://www.mmds.org/mmds/v2.1/ch09-recsys1.pptx>

[https://www.researchgate.net/publication/287952023\\_Collaborative\\_Filtering\\_Recommender\\_Systems](https://www.researchgate.net/publication/287952023_Collaborative_Filtering_Recommender_Systems)

<http://cs229.stanford.edu/proj2014/Rahul%20Makhijani,%20Saleh%20Samaneh,%20Megh%20Mehta,%20Collaborative%20Filtering%20Recommender%20Systems.pdf>

<https://www.scribd.com/presentation/414445910/CS548S15-Showcase-Web-Mining>

<https://towardsdatascience.com/image-recommendation-engine-leverage-transfert-learning-ec9af32f5239>

<http://elico.rapid-i.com/recommender-extension.html>

<https://www.youtube.com/watch?v=h9gpuJFF-0>



---

## THANK YOU

---

**Jyothi R.**  
Assistant Professor,  
Department of Computer Science  
[jyothir@pes.edu](mailto:jyothir@pes.edu)





---

# DATA ANALYTICS

## Unit 4: Decision trees- CART

---

**Jyothi R.**  
**Assistant Professor**  
Department of Computer Science  
and Engineering

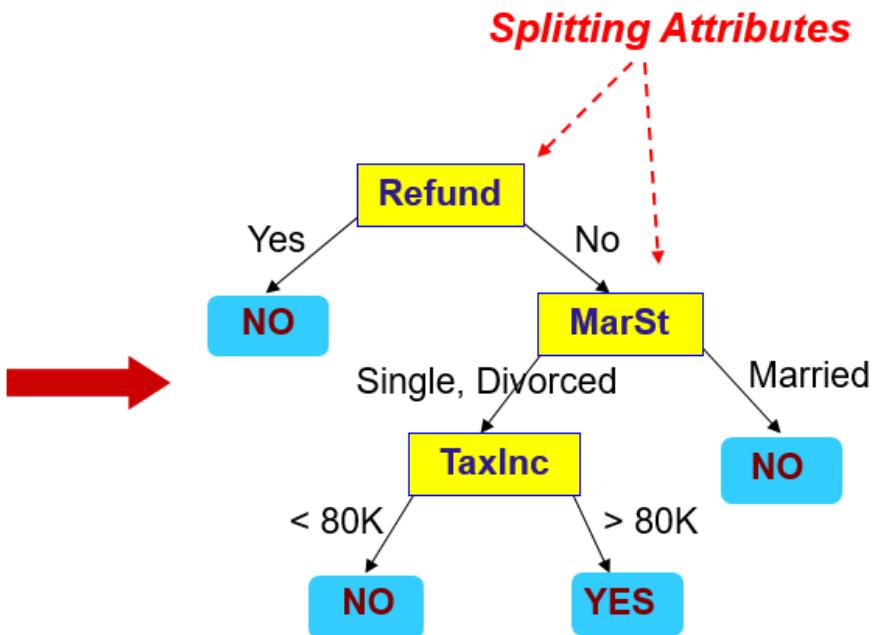
- Decision Trees are collection of divide-conquer problem-solving strategies that use tree-like structure to predict the outcome of a variable.
- Decision trees are a collection of predictive analytics techniques that use tree-like graphs for predicting the value of a response variable or target variable based on the values of explanatory variables or predictors.
- It is one of the supervised learning algorithms used for predicting both the discrete and the continuous dependent variable.
- Decision trees are effective for solving classification problems in which the response variable or target variable takes discrete values.

## Decision Tree

### Example1 of an Decision Tree

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				

Training Data

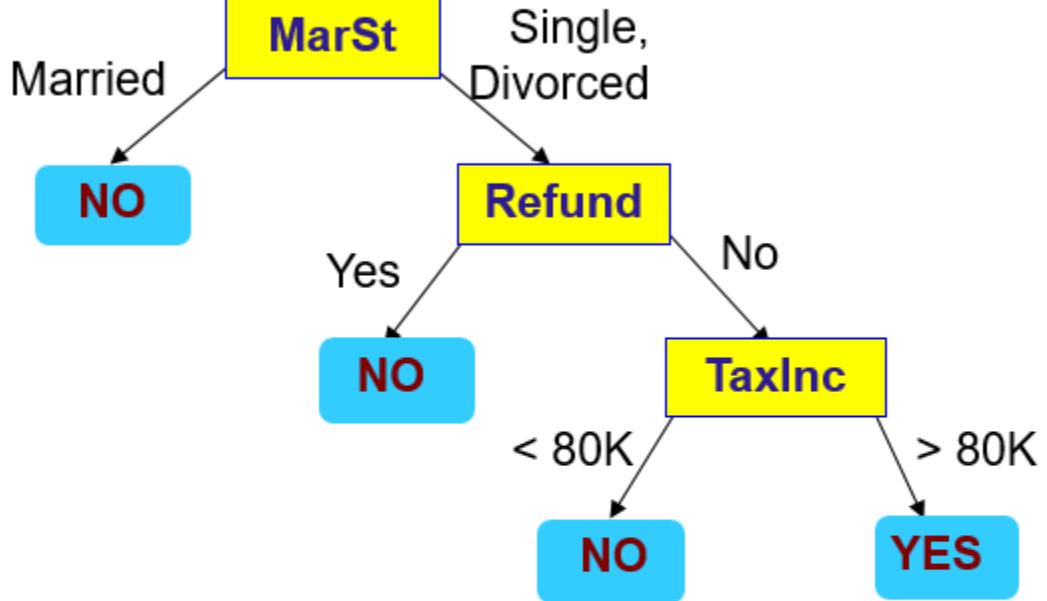


Model: Decision Tree

## Decision Tree

### Example 2 of an Decision Tree

		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	



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

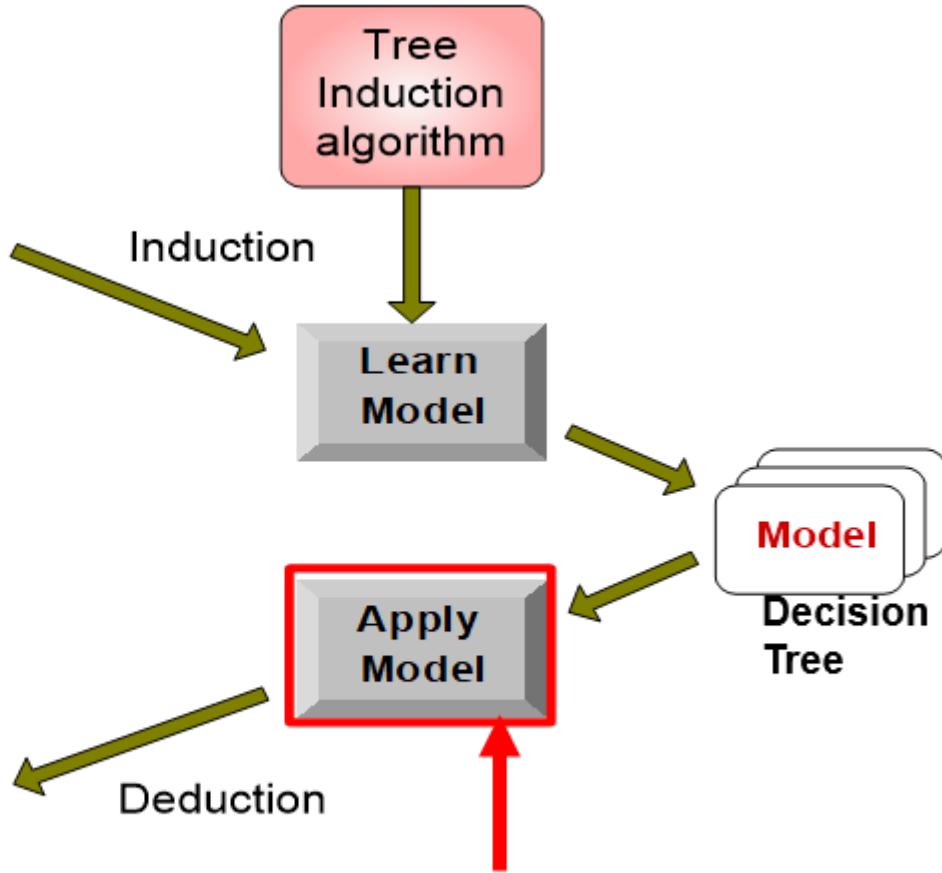
## Decision Tree Classification Task

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

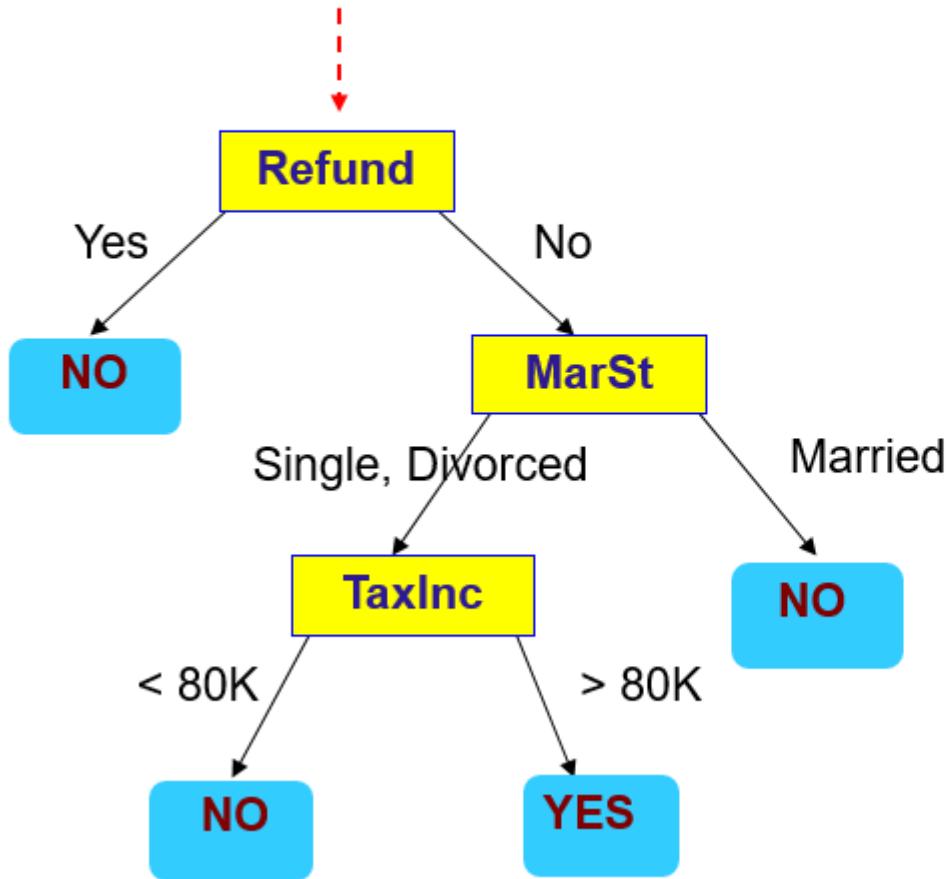
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



## Apply Model to Test Data

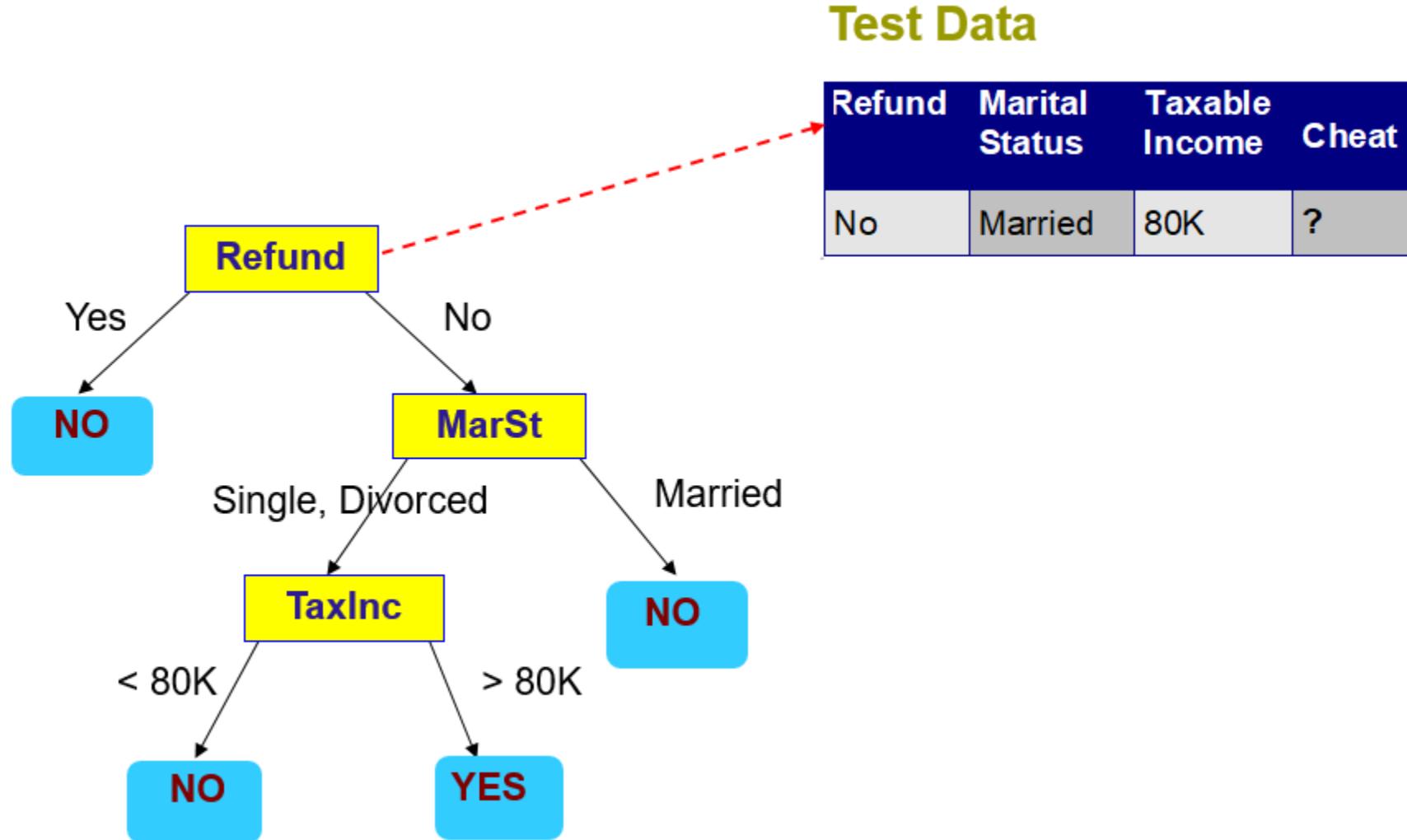
Start from the root of tree.



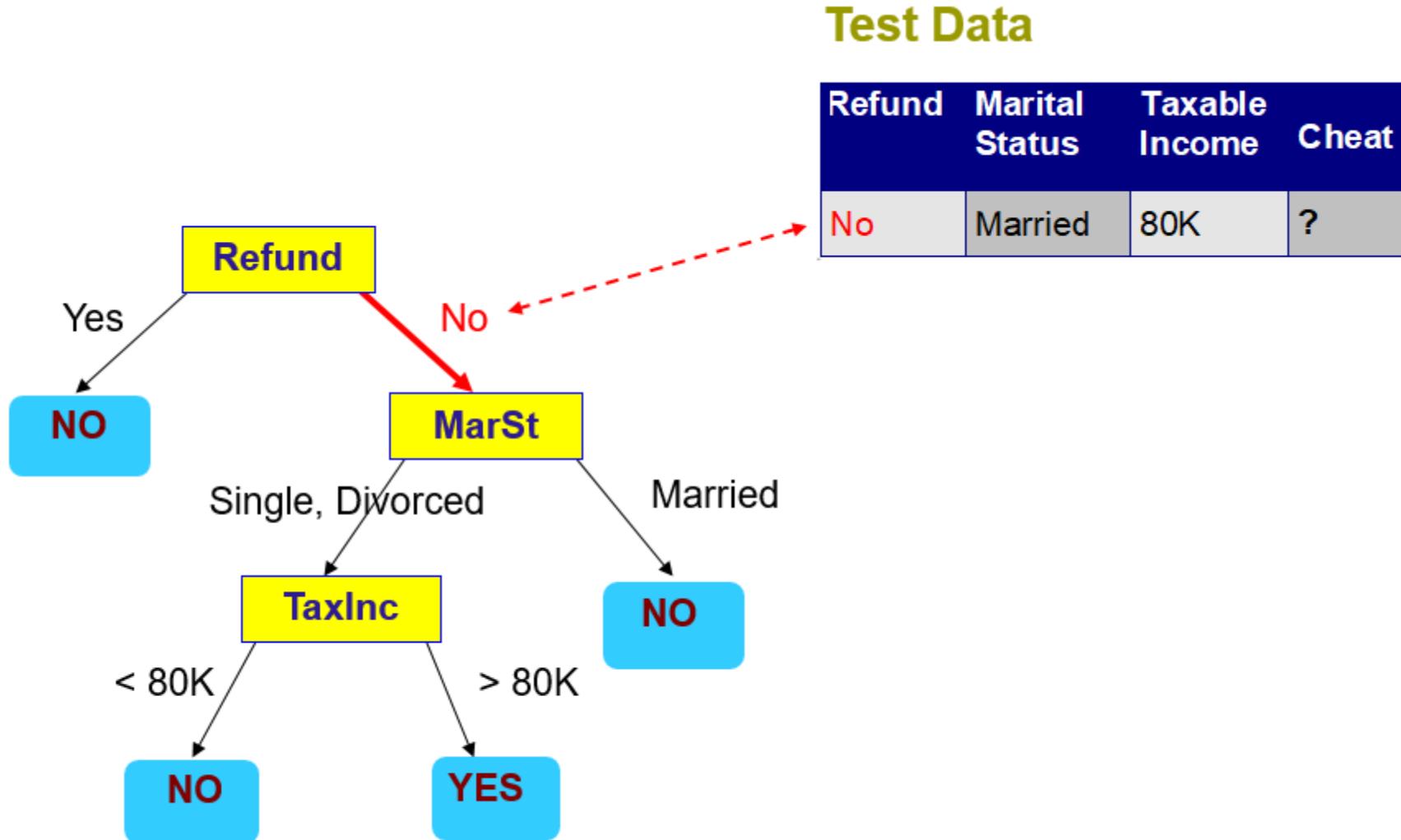
### Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

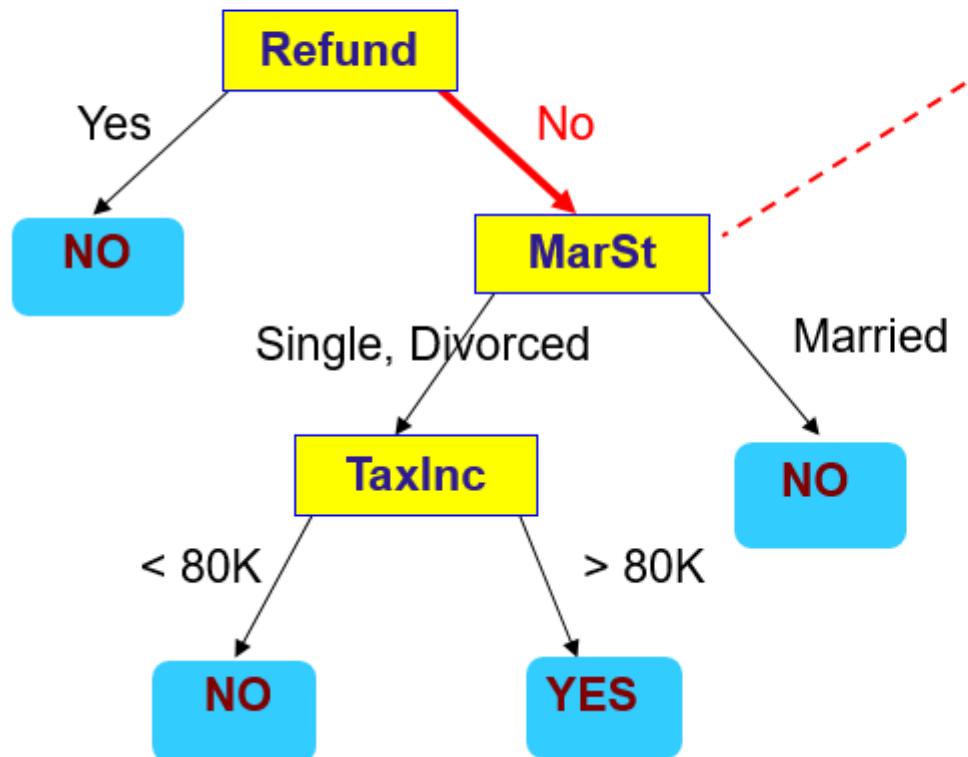
## Apply Model to Test Data



## Apply Model to Test Data



## Apply Model to Test Data



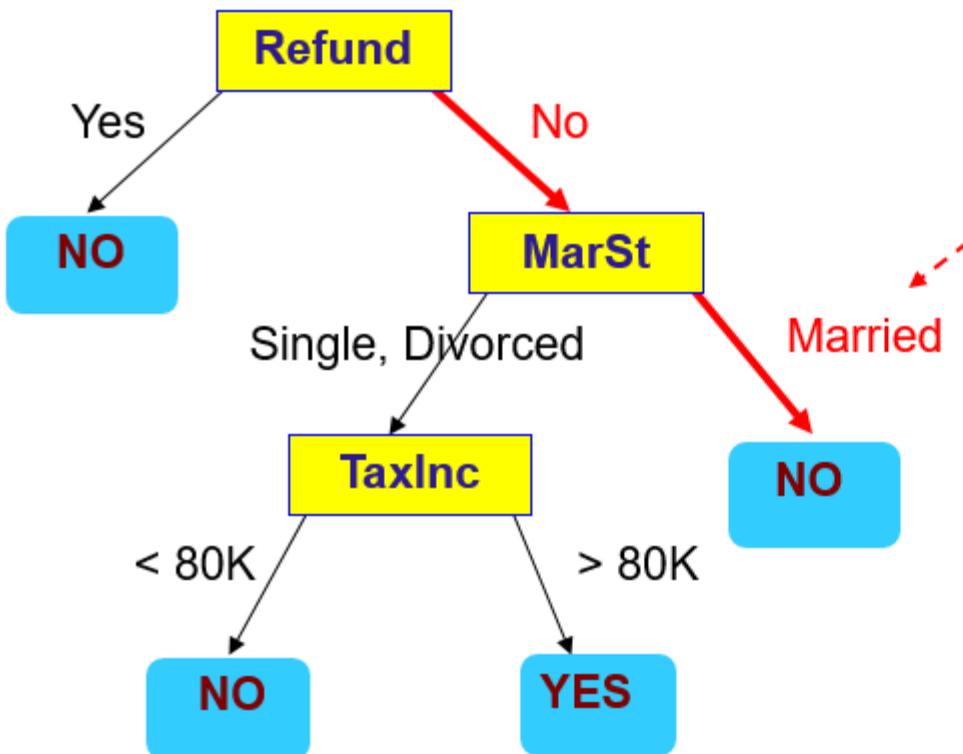
### Test Data

Refund	Marital Status	Taxable Income	Cheat
N9	Married	80K	?

## Apply Model to Test Data

### Test Data

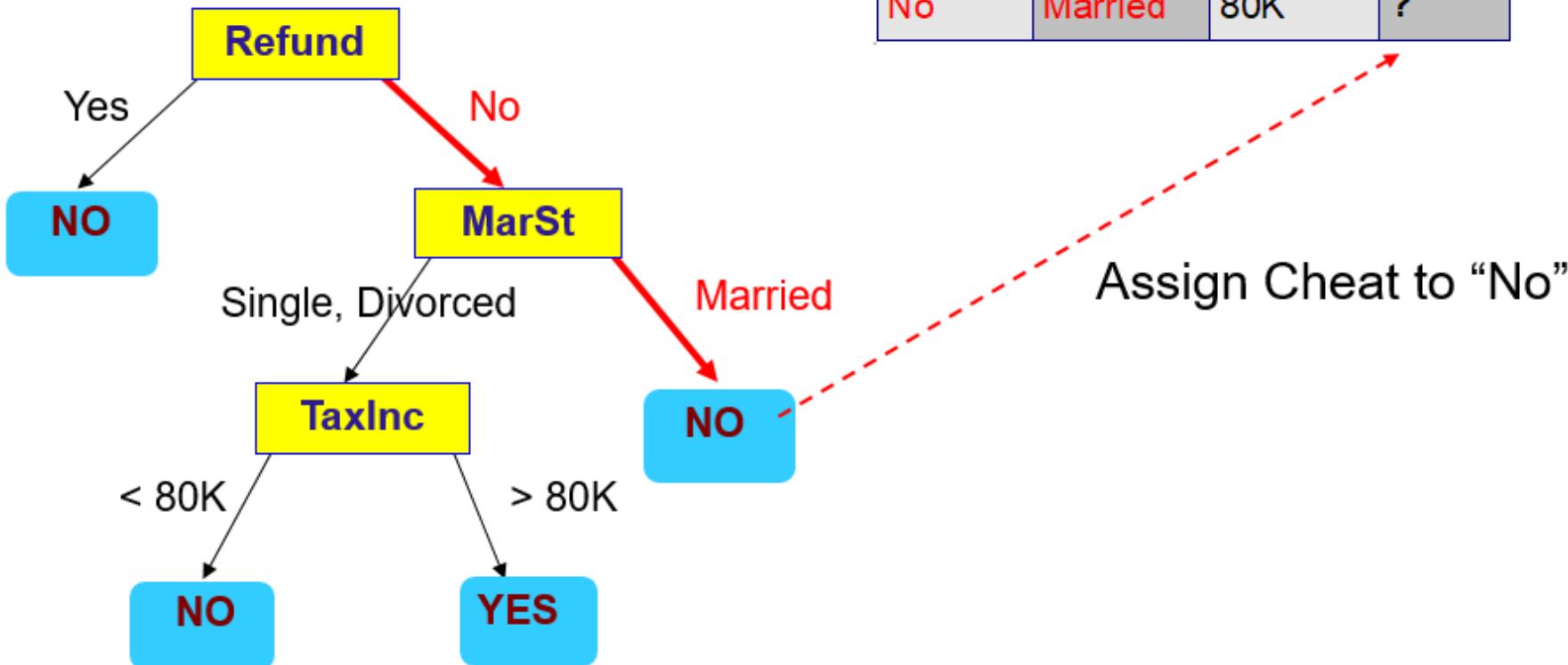
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



## Apply Model to Test Data

### Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



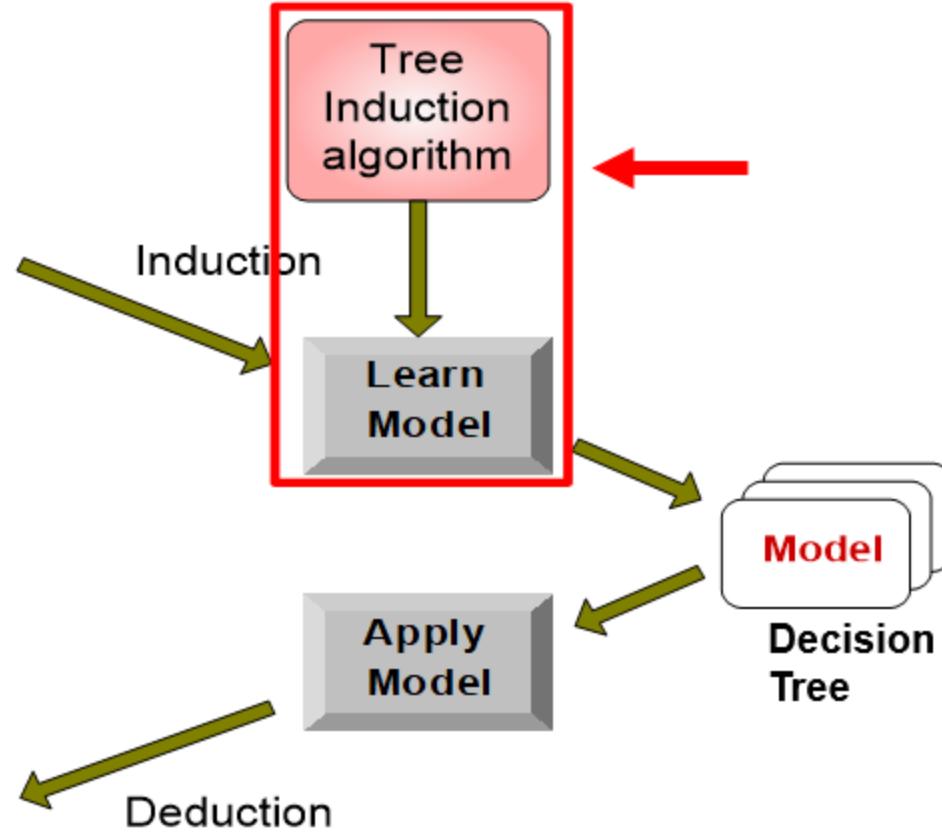
## Decision Tree Classification Task

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



## Decision Tree Induction

---

- Decision trees use the following criteria to develop the tree:
  1. Splitting Criteria: Splitting criteria are used to split a node i.e. set of data into subsets.
  2. Merging Criteria: When the predictor variable is categorical with n categories, it is possible that all n categories may be statistically significant. Thus, few categories may be merged to create a compound or aggregate category.
  3. Stopping Criteria: Stopping criteria is used for pruning the tree (stopping the tree from further branching) to reduce the complexity associated with business rules generated from the tree. Usually levels (depth) from root node (where each level corresponds to adding a predictor variable), minimum number of observation in a node for splitting are used as stopping criteria.

## Decision Tree

---

- The following steps are used for generating decision trees:
  1. Start with the root node in which all the data is present.
  2. Decide on a splitting criterion and stopping criteria: The root node is then split into two or more subsets leading to tree branches (called edges) using the splitting criterion. Nodes thus created are known as internal nodes. Each internal node has exactly one incoming edge.
  3. Further divide each internal node until no further splitting is possible or the stopping criterion is met. The terminal nodes (aka leaf nodes) will not have any outgoing edges.
  4. Terminal nodes are used for generating business rules.
  5. Tree pruning (a process for restricting the size of the tree) is used to avoid large trees and overfitting the data. Tree pruning is achieved through different stopping criteria.

## Classification and Regression Tree (CART)

---

- CART is used for a Classification Tree when the dependent variable is discrete and - a Regression Tree when the dependent variable is continuous.
- Classification tree uses various impurity measures such as the **Gini Impurity Index** and **Entropy** to split the nodes.
- Regression Tree splits the node that minimizes the **Sum of Squared Errors (SSE)**. CART is a binary tree wherein every node is split into only two branches.

## Classification and Regression Tree (CART)

- The following steps are used to generate a classification and a regression tree:

1. Start with the complete training data in the root node.

2. Decide on the measure of impurity (usually Gini impurity index or Entropy).

Choose a predictor variable that minimizes the impurity when the parent node is split into children nodes. This happens when the original data is divided into two subsets using a predictor variable such that it results in the maximum reduction in the impurity in the case of discrete dependent variable or the maximum reduction in SSE in the case of a continuous dependent variable.

3. Repeat step 2 for each subset of the data for each internal node using the independent variables until:

- (a) All the dependent variables are exhausted

- (b) The stopping criteria are met. Few stopping criteria used are number of levels of tree from the root node, minimum number of observations in parent/child node (e.g. 10% of the training data), and minimum reduction in impurity index.

4. Generate business rules for the leaf (terminal) nodes of the tree.

## Classification and Regression Tree (CART)

---

- The following steps are used to generate a classification and a regression tree:
  1. In Classification and regression tree(CART) impurity measures such as
  2. Gini impurity index or entropy are used as splitting criteria when the dependent variable is categorical and
  3. Sum of squared errors (SSE) is used when the dependent variable is continuous.

## References

---

### Text Book:

“Business Analytics, The Science of Data-Driven Making”, U. Dinesh Kumar, Wiley 2017 (Chapter 14)

“Recommender Systems, The text book, Charu C. Aggarwal, Springer 2016 Section 1 and Section 2

# DATA ANALYTICS

## Image Courtesy

---

<http://webcache.googleusercontent.com/search?q=cache:vW5NL8dqVQkJ:www.cs.kent.edu/~jin/DM07/ClassificationDecisionTree.ppt+&cd=5&hl=en&ct=clnk&gl=in>

[https://cmci.colorado.edu/classes/INFO-4604/fa17/files/slides-16\\_ensemble.pdf](https://cmci.colorado.edu/classes/INFO-4604/fa17/files/slides-16_ensemble.pdf)



---

## THANK YOU

---

**Jyothi R.**  
Assistant Professor,  
Department of Computer Science  
[jyothir@pes.edu](mailto:jyothir@pes.edu)





---

# DATA ANALYTICS

## Unit 4: Recommender Systems

---

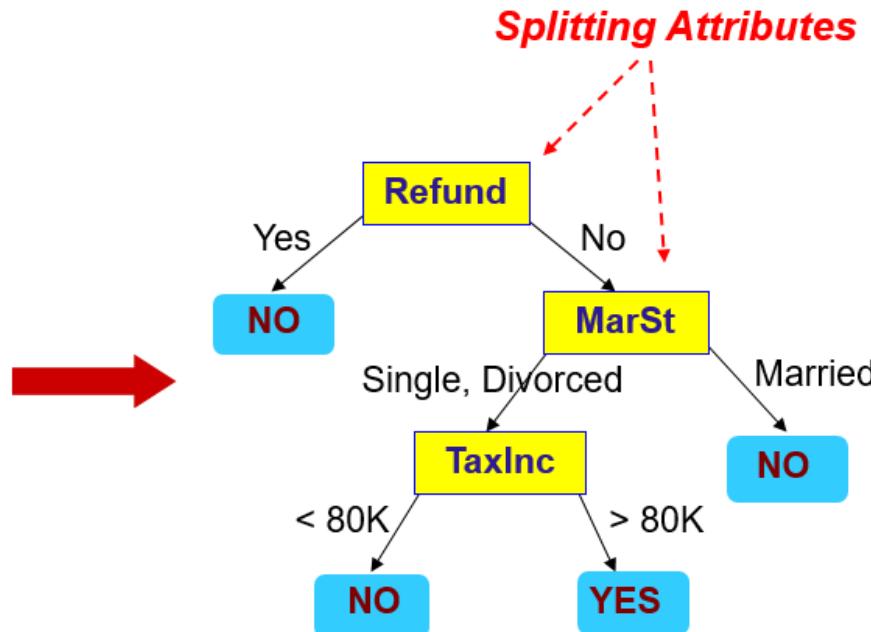
**Jyothi R.**  
**Assistant Professor**  
Department of Computer Science  
and Engineering



## Decision Tree

### Example of a Decision Tree

		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	



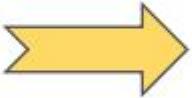
Training Data

Model: Decision Tree

Using the following condition  
 $85k < \text{Taxable Income} \leq 95k$ ?  
causes a multiway split.  
We have three branches:  
(1) Income < 85k  
(2) Income between 85k-95k  
(3) Income > 95k

## Ensemble Methods

- General Idea Combine multiple classifiers usually with different strengths to build a bigger, better classifier



## Ensemble Methods and Boosting

---

- The ensemble method is a machine-learning-algorithm that generates several classifiers using different sampling strategies such as bootstrap aggregating.
- A majority-voting-approach may be used for classifying a new observation using the multiple classifiers that are part of the ensemble method.
- In the ensemble method, several techniques such as logistic regression, CHAID, CART etc., are used.
- For a new observation, its class is identified using all the classifiers that are part of the ensemble method
- Different classifiers are likely to classify a new observation into different categories
- The final class of a new observation is decided based on a majority vote in which each classifier is given equal weightage
- **Adaboosting:** Boosting algorithms assign weights to each classifier based on their accuracy

## Ensemble methods

---

Typical application: classification

Ensemble of classifiers is a set of classifiers whose individual decisions combined in some way to classify new examples.

1. Generate multiple classifiers
2. Each votes on test instance
3. Take majority as classification

Classifiers different due to different sampling of training data, or randomized parameters within the classification algorithm

Aim: Take a simple, mediocre algorithm and transform it into a super classifier without requiring any new or fancy algorithm.

## Ensemble methods: Summary

---

Differ in training strategy, and combination method

1. Parallel training with different training sets: bagging
2. Sequential training, iteratively re-weighting training examples so current classifier focuses on hard examples: boosting
3. Parallel training with objective encouraging division of labor: mixture of experts

Notes:

- Also known as meta-learning
- Typically applied to weak models, such as decision stumps (single-node decision trees), or linear classifiers

## Variance-bias tradeoff?

---

Minimize two sets of errors:

1. Variance: Error from sensitivity to small fluctuations in the training set
2. Bias: Erroneous assumptions in the model

Variance-bias decomposition: is a way of analyzing the generalization error as a sum of 3 terms: variance, bias and irreducible error ( resulting from the problem itself)

## Why do ensemble methods work?

---

Based on one of two basic observations:

1. Variance reduction: if the training sets are completely independent, it will always help to average an ensemble because this will reduce variance without affecting bias (e.g., bagging) -- reduce sensitivity to individual data points.
2. Bias reduction: for simple models, average of models has much greater capacity than single model (e.g., hyperplane classifiers, Gaussian densities).

Averaging models can reduce bias substantially by increasing capacity, and control variance by fitting one component at a time (e.g., boosting).

## Ensemble methods: Justification

---

Ensemble methods more accurate than any individual members:

- Accurate (better than guessing)
- Diverse (different errors on new examples)

Independent errors: prob k of N classifiers

## Ensemble methods: Netflix

---

- Clear demonstration of the power of ensemble methods
- Original prize winner (BellKor) used an ensemble of 107 models!
- “Our experience is that most efforts should be concentrated in deriving substantially different approaches, rather than refining a simple technique.”
- “We strongly believe that the success of an ensemble approach depends on the ability of its various predictors to expose different complementing aspects of the data. Experience shows that this is very different than optimizing the accuracy of each individual predictor.”

## Bootstrap estimation

---

- Repeatedly draw  $n$  samples from  $D$
- For each set of samples, estimate a statistic
- The bootstrap estimate is the mean of the individual estimates
- Used to estimate a statistic parameter and its variance
- Bagging: bootstrap aggregation

## Bagging

---

- Simple idea: generate  $M$  bootstrap samples from your original training set. Train on each one to get  $y$ , and average them
- For regression: average predictions
- For classification: average class probabilities  
(or take the majority vote if only hard outputs available)
- Bagging approximates the Bayesian posterior mean. The more bootstraps the better, so use as many as you have time for. Each bootstrap sample is drawn with replacement, so each one contains some duplicates of certain training points and leaves out other training points completely

## Cross-validated committees

---

- Bagging works well for unstable algorithms
  - can change dramatically with small changes in training data
- But can hurt a stable algorithm: a Bayes optimal algorithm may leave out some training examples in every bootstrap
- Alternative method based on different training examples:  
**cross-validated committees:**
- Here k disjoint subsets of data are left out of training sets
- Again uses majority for combination

## Boosting

---

- Also works by manipulating training set, but **classifiers trained sequentially**
- Each classifier trained given knowledge of the performance of previously trained classifiers: focus on hard examples
- Final classifier: **weighted sum of component classifiers**

## References

---

### Text Book:

“Business Analytics, The Science of Data-Driven Making”, U. Dinesh Kumar, Wiley 2017 (Chapter 14)

“Recommender Systems, The text book, Charu C. Aggarwal, Springer 2016 Section 1 and Section 2.

# DATA ANALYTICS

## Image Courtesy

---

[http://webcache.googleusercontent.com/search?q=cache:vW5N  
L8dqVQkJ:www.cs.kent.edu/~jin/DM07/ClassificationDecisionTr  
ee.ppt+&cd=5&hl=en&ct=clnk&gl=in](http://webcache.googleusercontent.com/search?q=cache:vW5NL8dqVQkJ:www.cs.kent.edu/~jin/DM07/ClassificationDecisionTree.ppt+&cd=5&hl=en&ct=clnk&gl=in)

[https://cmci.colorado.edu/classes/INFO-  
4604/fa17/files/slides-16\\_ensemble.pdf](https://cmci.colorado.edu/classes/INFO-4604/fa17/files/slides-16_ensemble.pdf)



---

## THANK YOU

---

**Jyothi R.**  
Assistant Professor,  
Department of Computer Science  
[jyothir@pes.edu](mailto:jyothir@pes.edu)





## DATA ANALYTICS

### Unit 4: Brief review of other classifiers: SVM, ANN and Data Driven Approaches

---

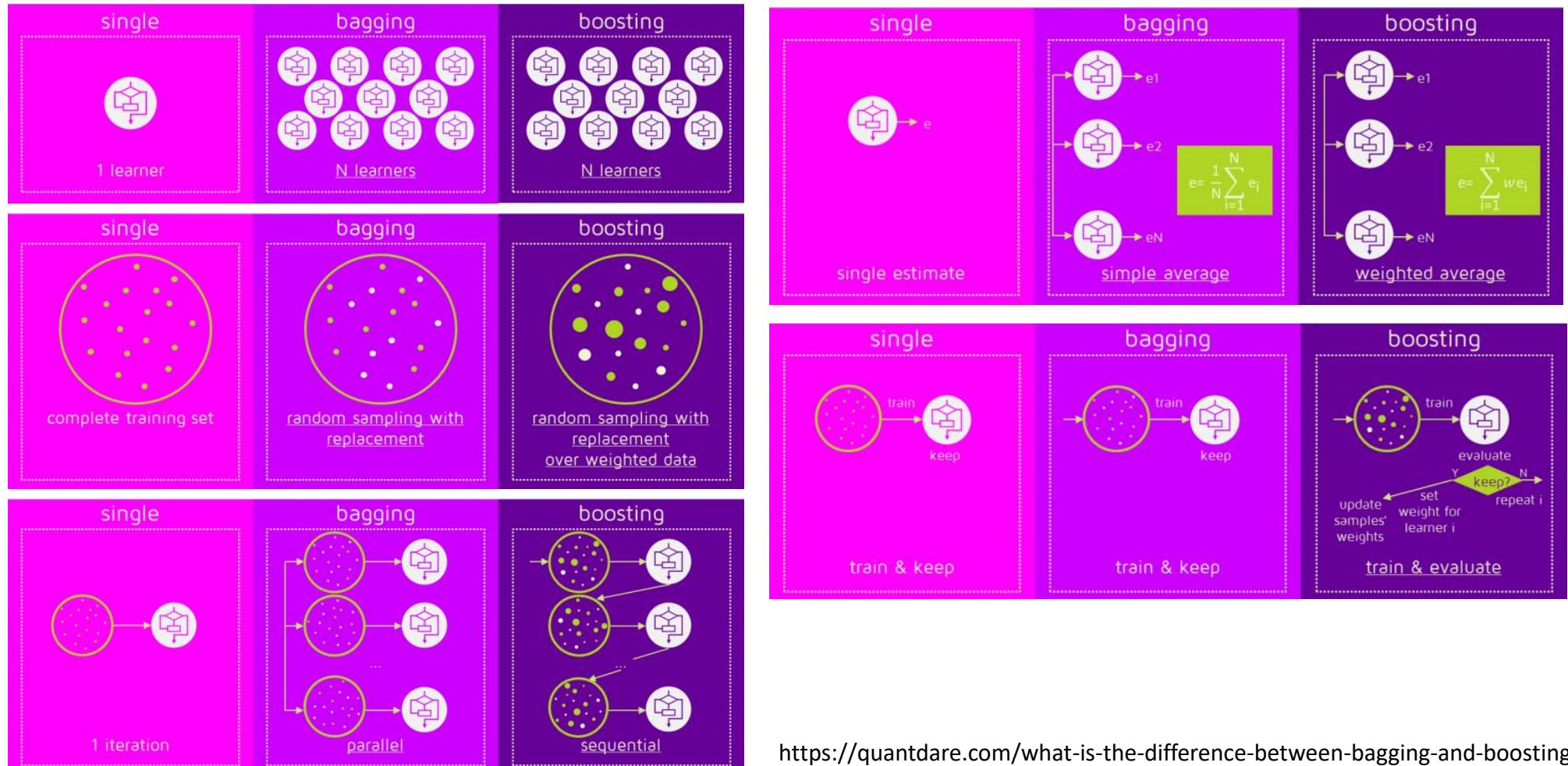
Jyothi R.

Department of Computer Science  
and Engineering

# DATA ANALYTICS



# Bagging vs Boosting



<https://quantdare.com/what-is-the-difference-between-bagging-and-boosting/>

## Bagging

vs

## Boosting

- **Partitioning of data** – random
- **Goal to achieve** – minimum variance
- **Methods used** – random subspace
- **Combining the models** – simple average
- **Example** – random forest

- higher vote to misclassified samples
- maximum accuracy
- gradient descent
- weighted majority vote
- Adaboost

### Advantages

- Reduces over-fitting of the model.
- Handles higher dimensionality data very well.
- Maintains accuracy for missing data.

### Disadvantages:

- Since final prediction is based on the mean predictions from subset trees, it won't give precise values for the classification and regression model.

### Advantages

- Supports different loss function ('binary logistic' can be used for example).
- Works well with interactions.

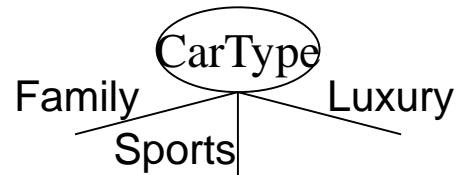
### Disadvantages:

- Prone to over-fitting.
- Requires careful tuning of different hyper-parameters.

## Splitting a node

Depends on attribute types

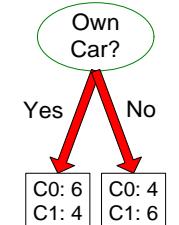
Nominal



Multiway split

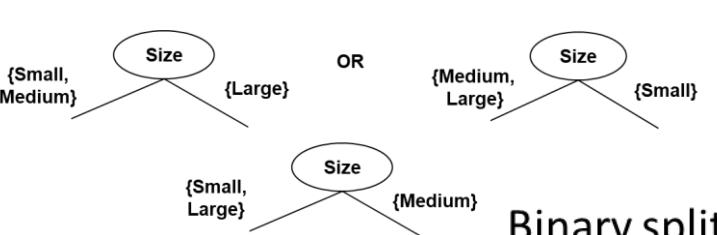


Binary split

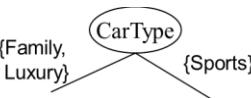


Which is the best split?

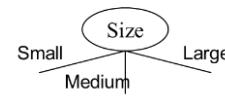
Ordinal



Binary split



Multiway split



Continuous

Different ways of handling

Discretization to form an ordinal categorical attribute

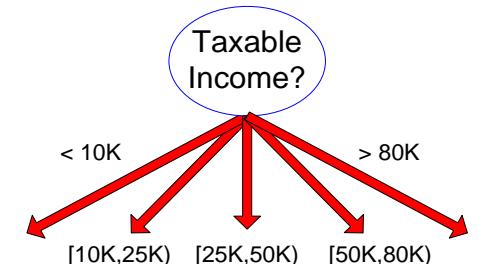
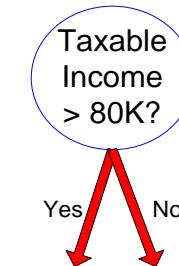
Static – discretize once at the beginning

Dynamic – ranges can be found by equal interval

bucketing, equal frequency bucketing (percentiles), or clustering.

Binary Decision:  $(A < v)$  or  $(A \geq v)$

- consider all possible splits and find the best cut
- can be compute intensive



(i) Binary split

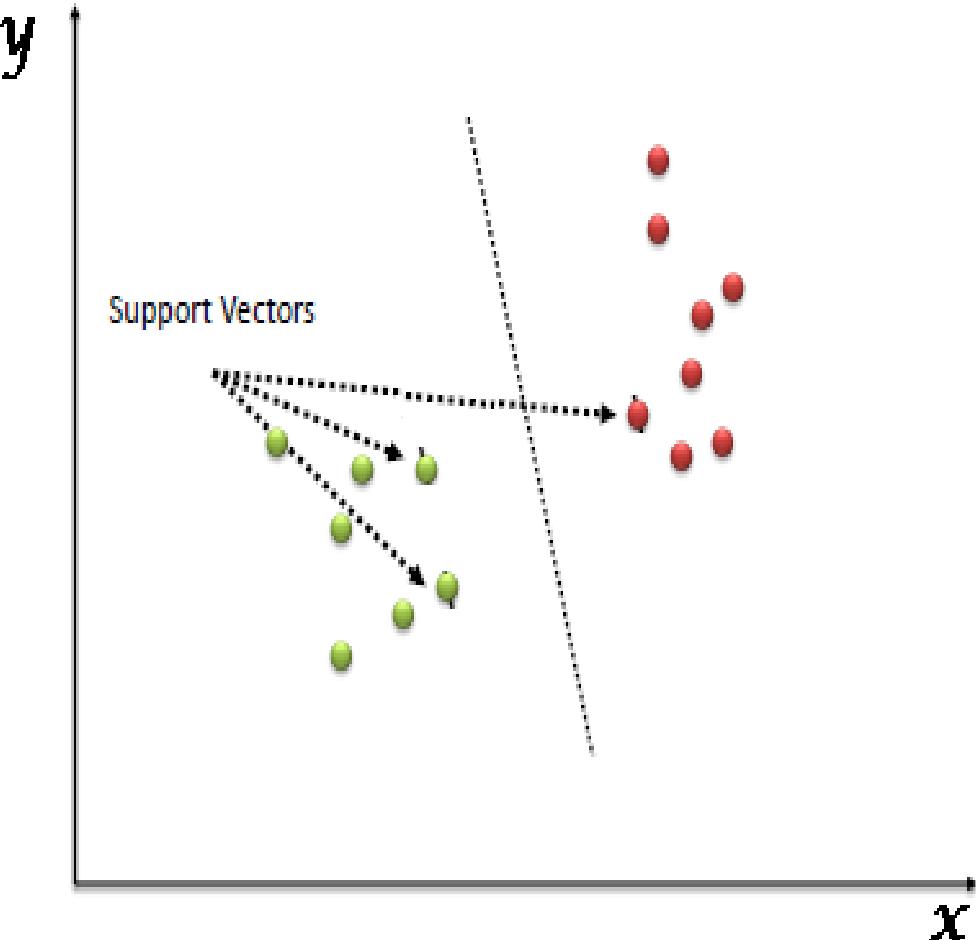
(ii) Multi-way split

## Introduction

---

- Support vector machine(SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges.
- In the SVM algorithm, we plot each data item as a point in n-dimensional space with the value of each feature being the value of a particular coordinate.
- Then, we perform classification by finding the hyper-plane that differentiates the two classes very well.

- Support Vectors are simply the coordinates of individual observation.
- The SVM classifier is a frontier which best segregates the two classes (hyper-plane/ line).

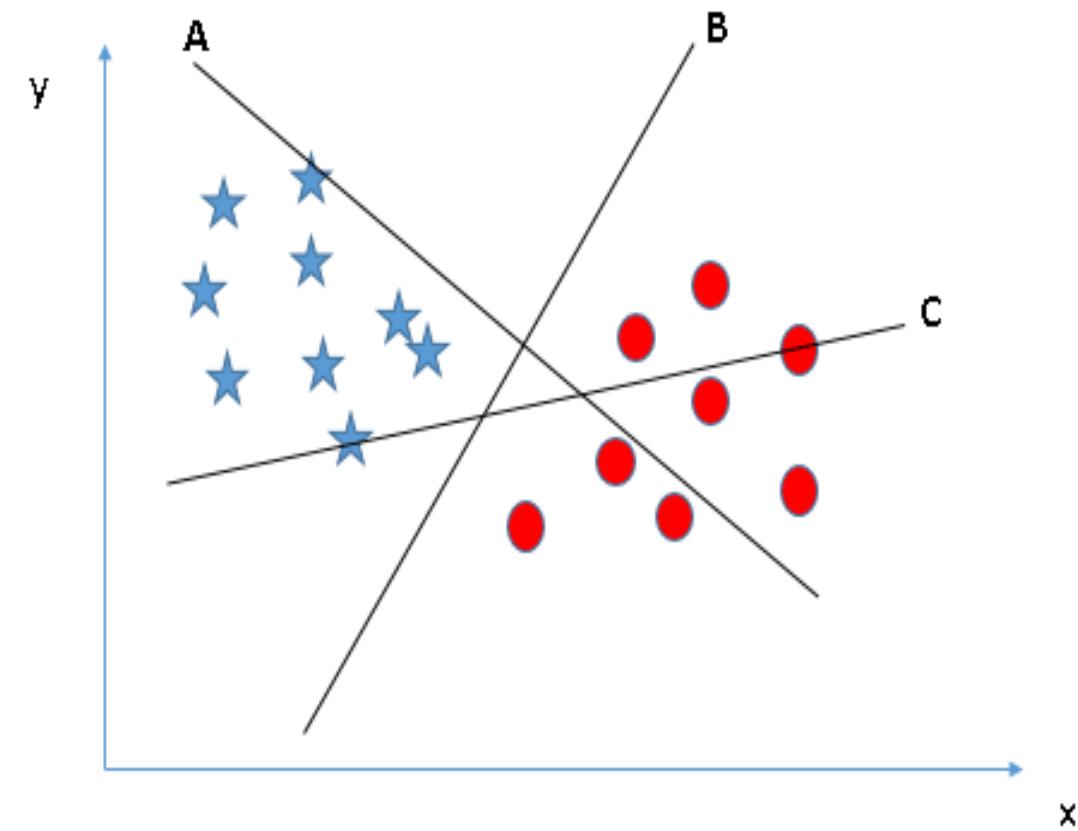


## How does it work?

- **Identify the right hyper-plane (Scenario-1):**

Here, we have three hyper-planes (A, B and C)

- Now, identify the right hyper-plane to classify star and circle.
- We need to remember a thumb rule to identify the right hyper-plane: “Select the hyper-plane which segregates the two classes better”.
- In this scenario, hyper-plane “B” has excellently performed this job

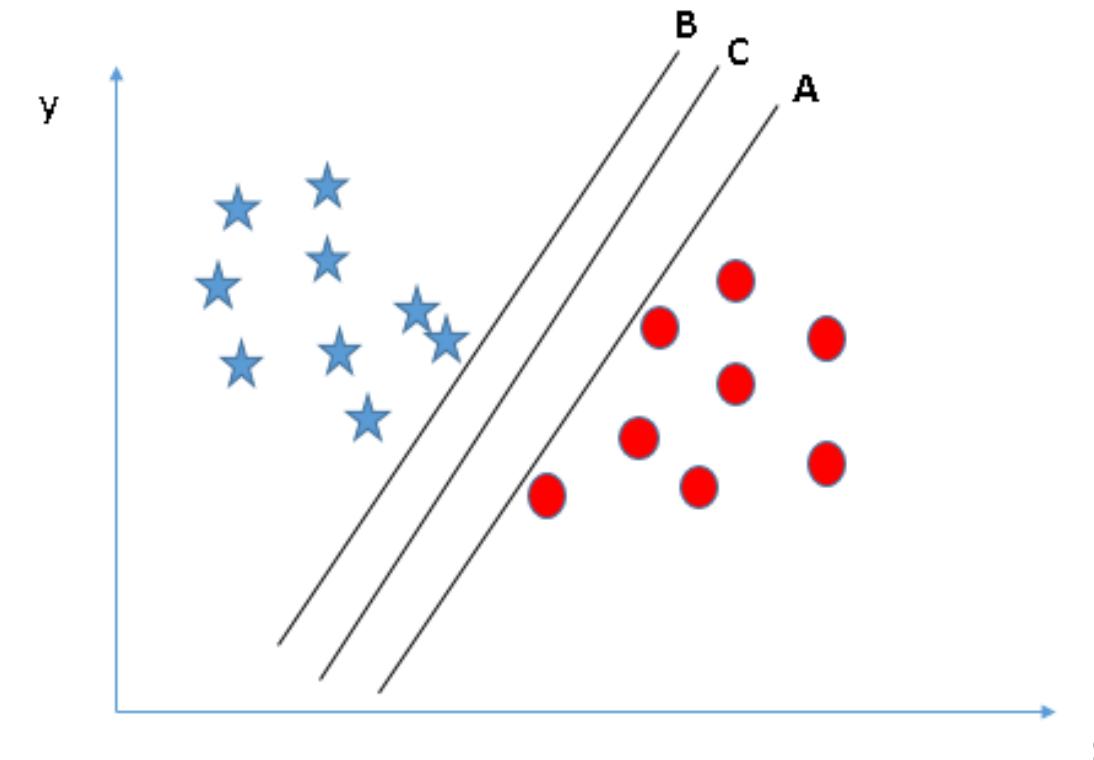


## How does it work?

- **Identify the right hyper-plane (Scenario-2):**

Here, we have three hyper-planes (A, B and C) and all are segregating the classes well. Now, How can we identify the right hyper-plane?

- Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as Margin



## How does it work?

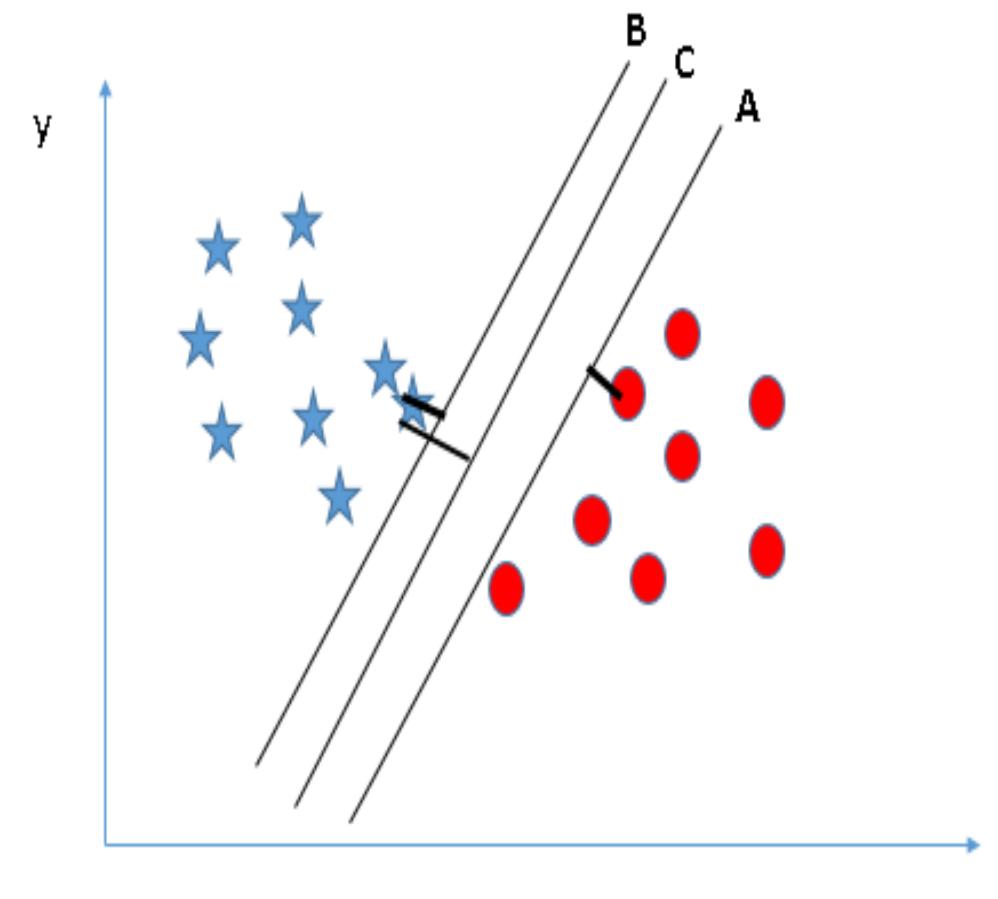
- **Identify the right hyper-plane (Scenario-2):**

Above, you can see that the margin for hyper-plane C is high as compared to both A and B.

- Hence, we name the right hyper-plane as C.

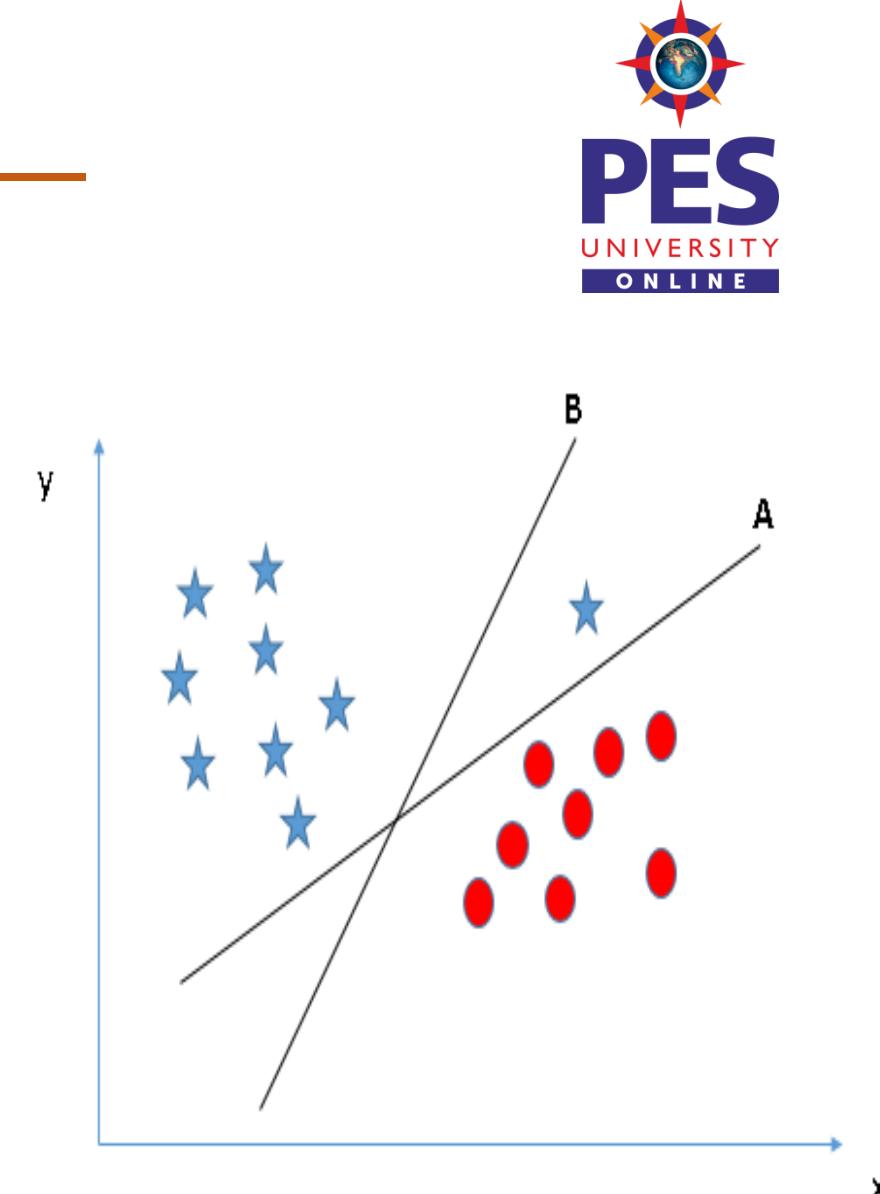
Another lightning reason for selecting the hyper-plane with higher margin is robustness.

- If we select a hyper-plane having low margin then there is high chance of miss-classification.



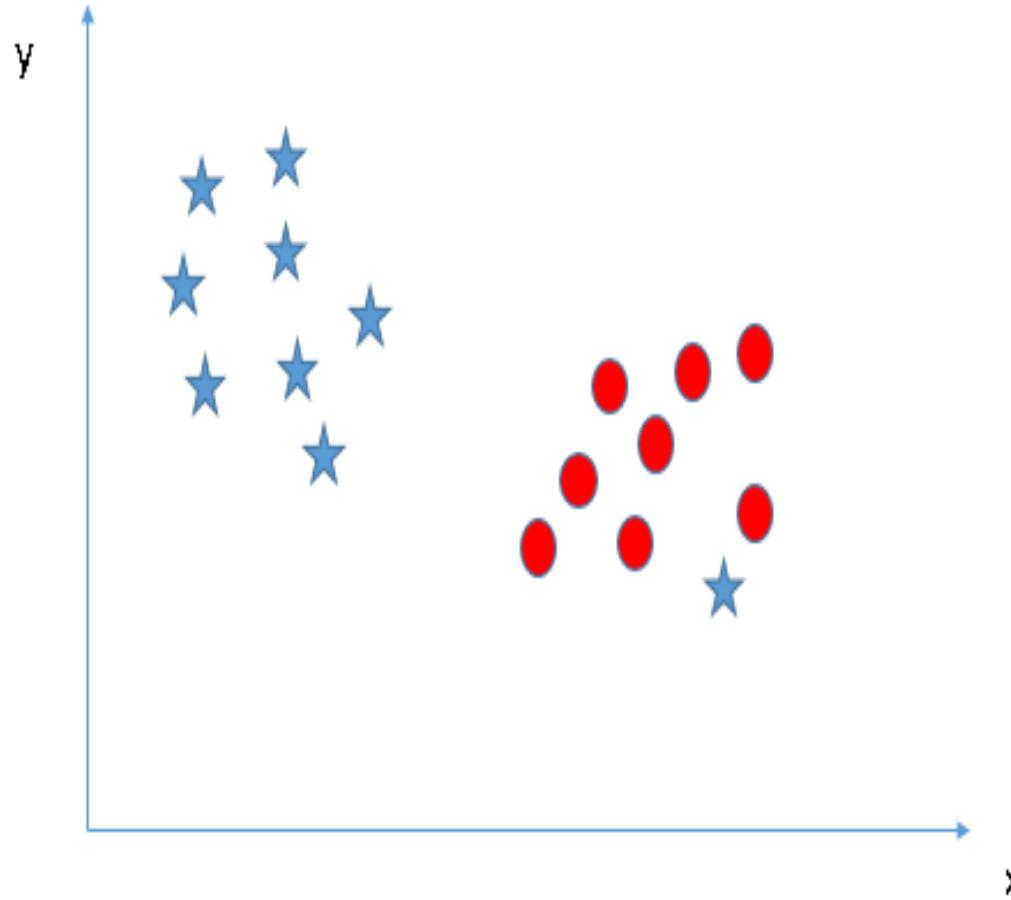
## How does it work?

- **Identify the right hyper-plane (Scenario-3):**
- Use the rules as discussed in previous section to identify the right hyper-plane.
- Some of you may have selected the hyper-plane B as has higher margin compared to A.
- But, here is the catch, SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin.
- Here, hyper-plane B has a classification error and A has classified all correctly. **Therefore, the right hyper-plane is A.**

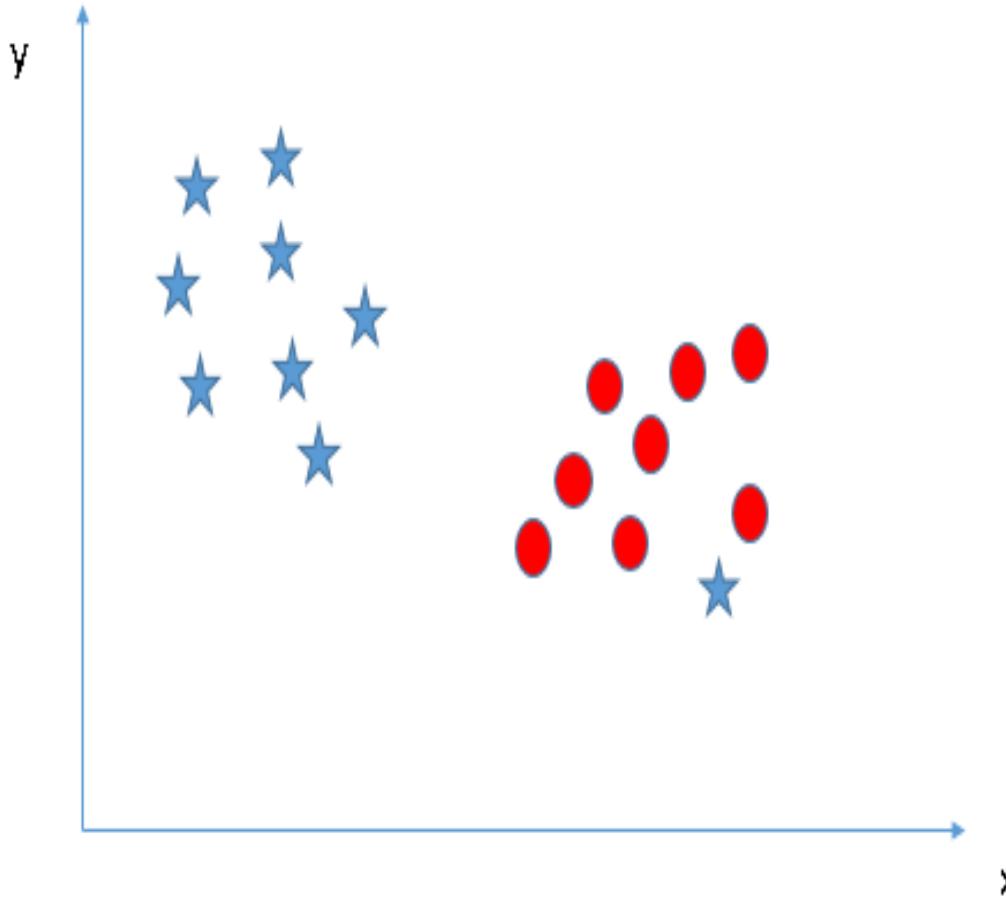


## How does it work?

- **Can we classify two classes (Scenario-4)?:**
- It is unable to segregate the two classes using a straight line, as one of the stars lies in the territory of other(circle) class as an outlier.



- **Can we classify two classes (Scenario-4)?:**
- As we already mentioned, one star at other end is like an outlier for star class. The SVM algorithm has a feature to ignore outliers and find the hyper-plane that has the maximum margin. Hence, we can say, SVM classification is robust to outliers.

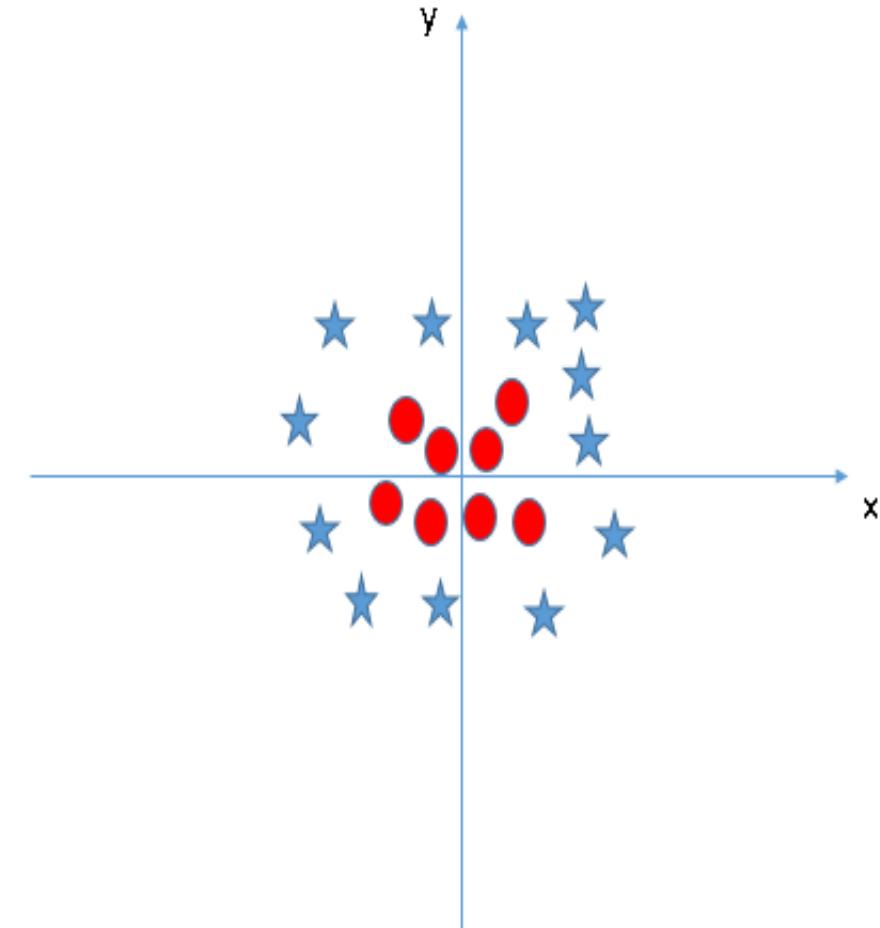


## How does it work?

- **Find the hyper-plane to segregate to classes**

**(Scenario-5):**

- In the scenario below, we can't have linear hyper-plane between the two classes, so how does SVM classify these two classes? Till now, we have only looked at the linear hyper-plane.
- SVM can solve this problem. Easily! It solves this problem by introducing additional feature. Here, we will add a new feature  $z=x^2+y^2$ . Now, let's plot the data points on axis x and z:

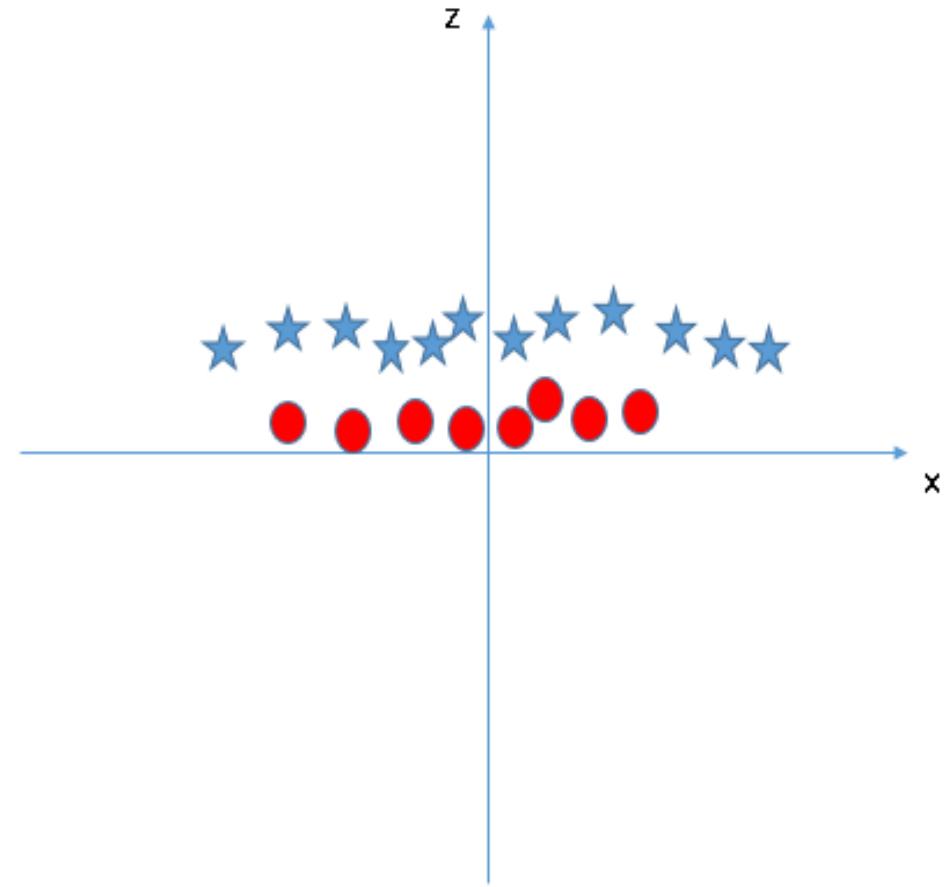


## How does it work?

- **Find the hyper-plane to segregate to classes**

**(Scenario-5):**

- In above plot, points to consider are:
- All values for  $z$  would be positive always because  $z$  is the squared sum of both  $x$  and  $y$
- In the original plot, red circles appear close to the origin of  $x$  and  $y$  axes, leading to lower value of  $z$  and star relatively away from the origin result to higher value of  $z$ .



## How does it work?

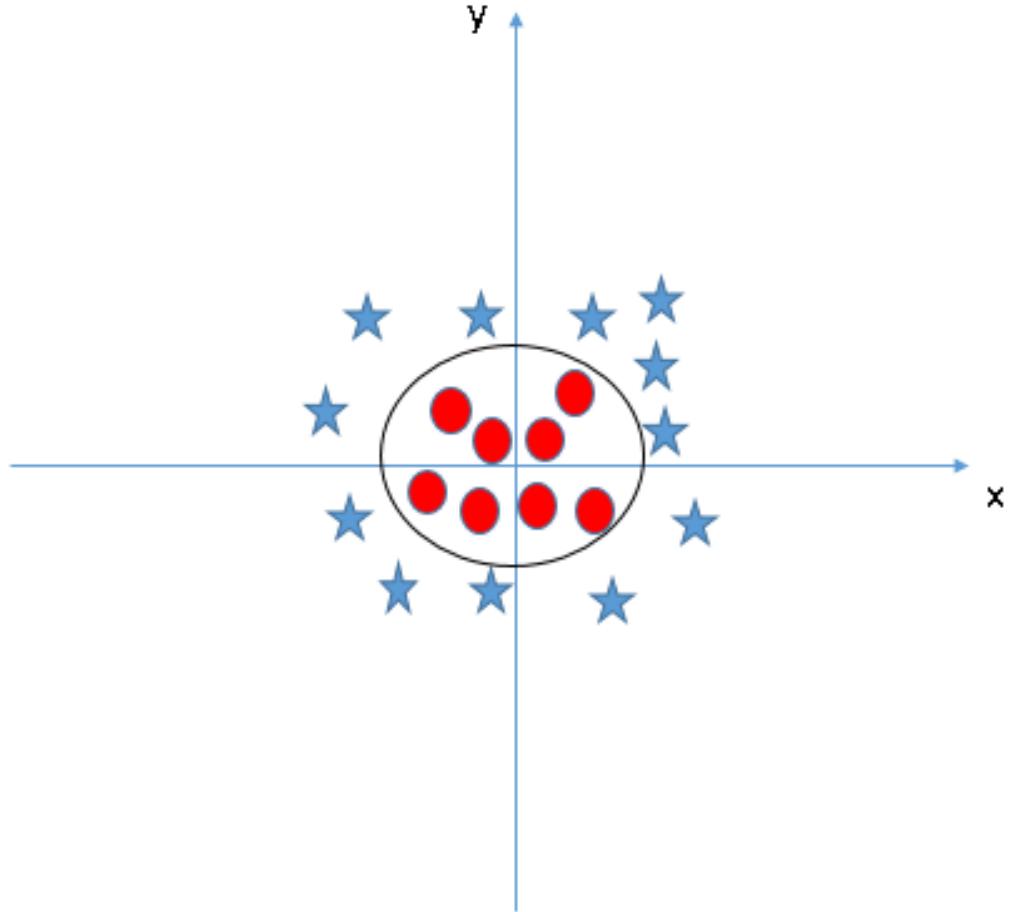
---

- In the SVM classifier, it is easy to have a linear hyper-plane between these two classes. But, another burning question which arises is, should we need to add this feature manually to have a hyper-plane.
- No, the SVM algorithm has a technique called the kernel trick. The SVM kernel is a function that takes low dimensional input space and transforms it to a higher dimensional space i.e. it converts not separable problem to separable problem.

## How does it work?

---

- It is mostly useful in non-linear separation problem.
- Simply put, it does some extremely complex data transformations, then finds out the process to separate the data based on the labels or outputs you've defined.
- When we look at the hyper-plane in original input space it looks like a circle:



## How to implement SVM in Python and R?

---

- In Python, scikit-learn is a widely used library for implementing machine learning algorithms.
- SVM is also available in the scikit-learn library and we follow the same structure for using it (Import library, object creation, fitting model and prediction).

## Support Vector Machine(SVM) code in R

---

- The e1071 package in R is used to create Support Vector Machines with ease.
- It has helper functions as well as code for the Naive Bayes Classifier.

## Pros and Cons associated with SVM

---

- Pros:
- It works really well with a clear margin of separation
- It is effective in high dimensional spaces.
- It is effective in cases where the number of dimensions is greater than the number of samples.
- It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.

## Pros and Cons associated with SVM

---

- Cons:
- It doesn't perform well when we have large data set because the required training time is higher
- It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping
- SVM doesn't directly provide probability estimates, these are calculated using an expensive five-fold cross-validation. It is included in the related SVC method of Python scikit-learn library.

## Introduction to Artificial Neural Network

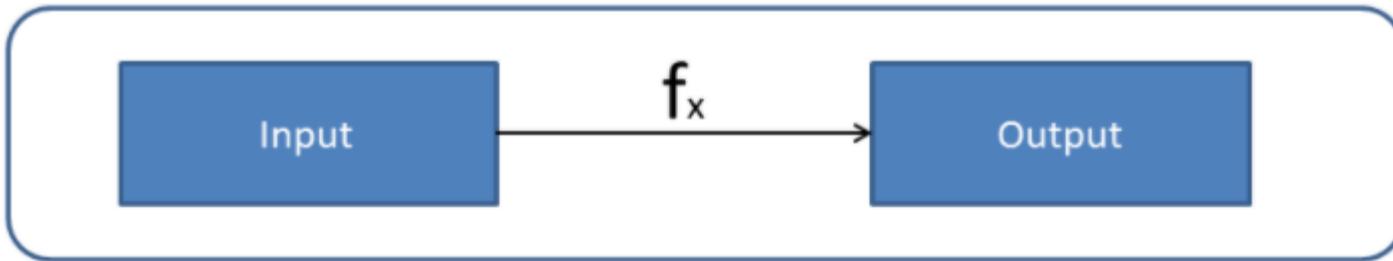
---

- A simple machine is a set of algorithm, which converts input(s) to output(s)
- In this scenario, the same input will always lead to the same output.
- Human brain, on the other hand, has a unique characteristic of creating transient states through neurons in between the sensory organs and the brain (decision taking unit).
- Hence, the probabilistic interim state brings out a factor of randomness, which brings out what we call “Creativity”.

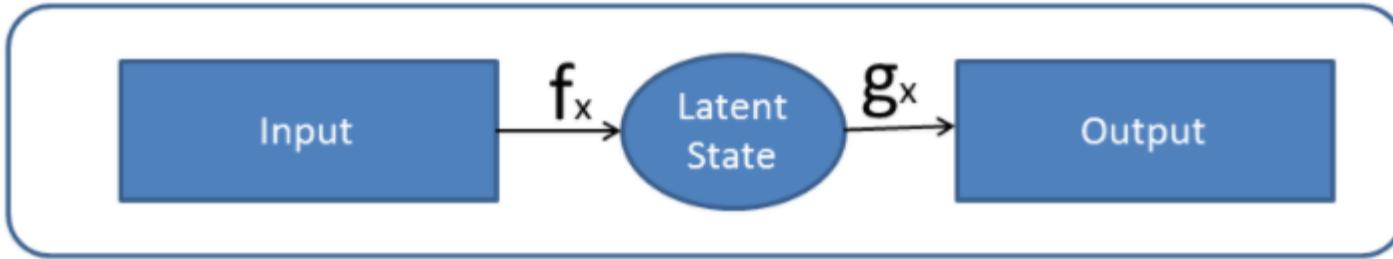
## Introduction to Artificial Neural Network

- In ANN (Artificial neural network) or rather all machine learning algorithm, we build some kind of transient states, which allows the machine to learn in a more sophisticated manner.
- The objective here is to bring out the framework of ANN algorithm in parallel to the functionality of human brain.
- A single perceptron (or neuron) can be imagined as a Logistic Regression. Artificial Neural Network, or ANN, is a group of multiple perceptron's/ neurons at each layer.
- ANN is also known as a Feed-Forward Neural network because inputs are processed only in the forward direction:

## Introduction to Artificial Neural Network

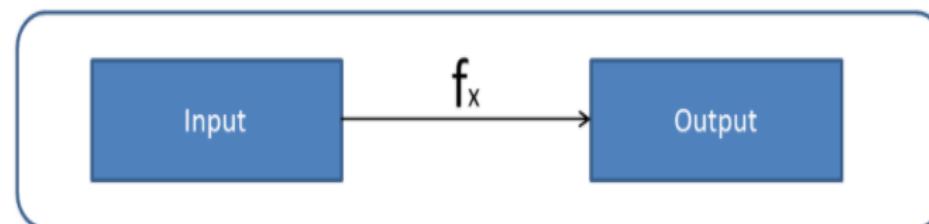
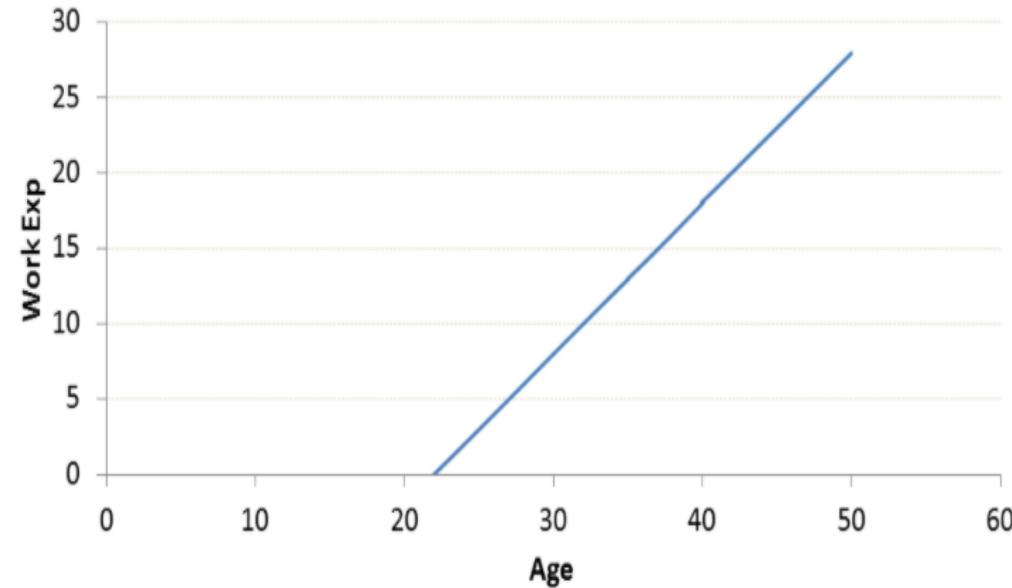


*From brainless box to machine learning*



## How does a simple predictive algorithm work?

- A simple predictive algorithm tries to mimic the relationship between the Input and the output variables.
- The function derived in such routines is a direct linear or non-linear function between input and output variables.
- For instance, if we try to predict the total work experience of a person using his age, following is the kind of relationship we will observe:



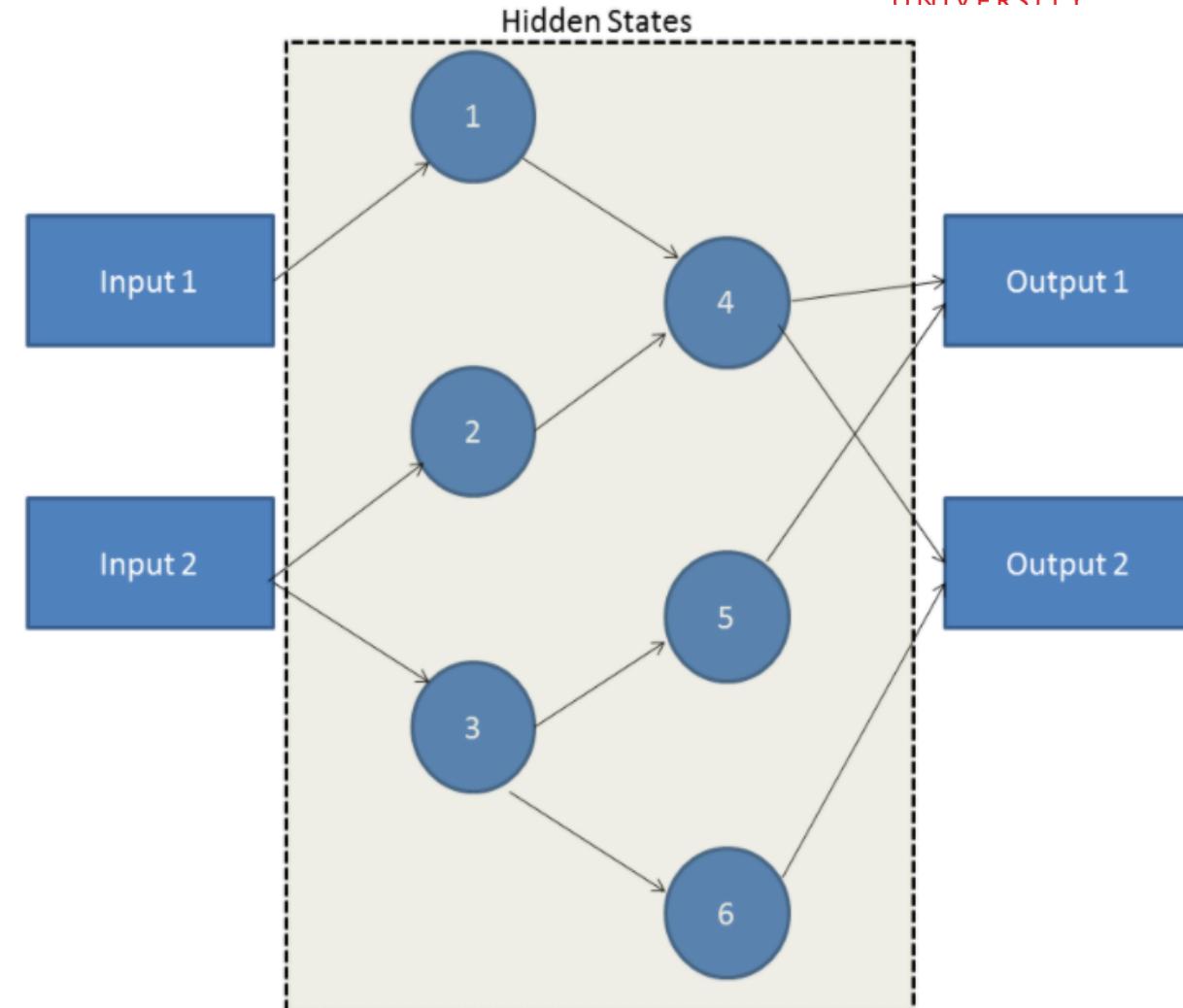
## How does a simple predictive algorithm work?

---

- Relationships can easily be predicted using simple regression algorithms.
- But it becomes difficult to make predictions in case of complex non-linear relationships and significant covariate terms.
- In such cases, we need more sophisticated machine learning tools.
- To make such predictions, we have two options – either predict a complex non linear function or break this problem into multiple steps and solve for each step.
- The later can be achieved easily using an artificial neural network (ANN).

## How does ANN work?

- It is truly said that the working of ANN takes its roots from the neural network residing in human brain.
- ANN operates on something referred to as Hidden State. These hidden states are similar to neurons. Each of these hidden state is a transient form which has a probabilistic behavior. A grid of such hidden state act as a bridge between the input and the output.



## How does ANN work?

---

- Let's try to understand what the diagram actually means.
- We have a vector of three inputs and we intend to find the probability that the output event will fall into class 1 or class 2.
- For this prediction we need to predict a series of hidden classes in between (the bridge). The vector of the three inputs in some combination predicts the probability of activation of hidden nodes from 1 – 4.
- The probabilistic combination of hidden state 1-4 are then used to predict the activation rate of hidden nodes 5-8. These hidden nodes 5-8 in turn are used to predict hidden nodes 9-12, which finally predicts the outcome.
- The intermediate latent states allows the algorithm to learn from every prediction.

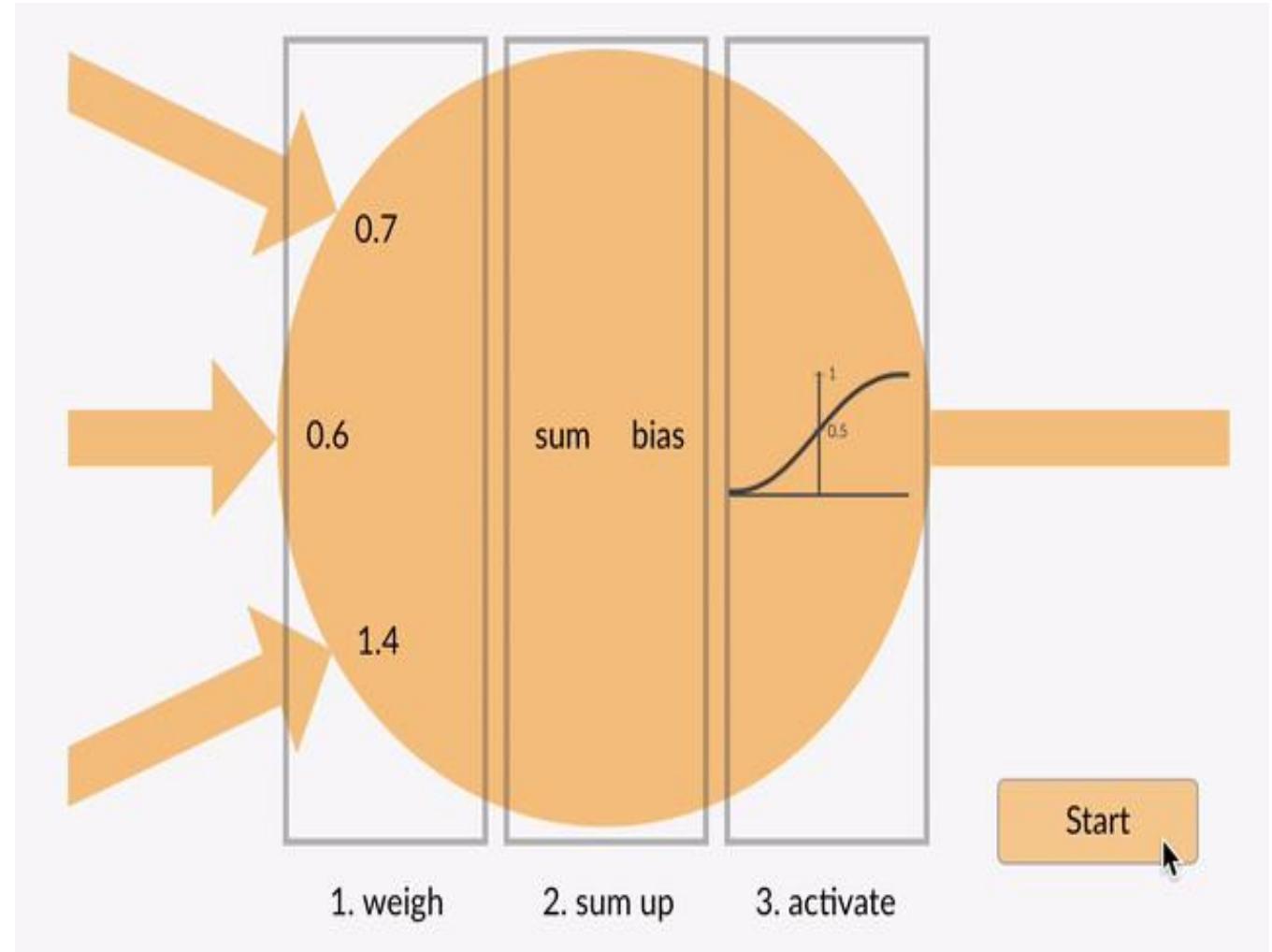
## Advantages of Artificial Neural Network (ANN)

---

- Artificial Neural Network is capable of learning any nonlinear function.
- Hence, these networks are popularly known as Universal Function Approximators. ANNs have the capacity to learn weights that map any input to the output.
- One of the main reasons behind universal approximation is the activation function. Activation functions introduce nonlinear properties to the network.
- This helps the network learn any complex relationship between input and output.

## Advantages of Artificial Neural Network (ANN)

- Here, the output at each neuron is the activation of a weighted sum of inputs.
- what happens if there is no activation function? The network only learns the linear function and can never learn complex relationships.
- An activation function is a powerhouse of ANN!



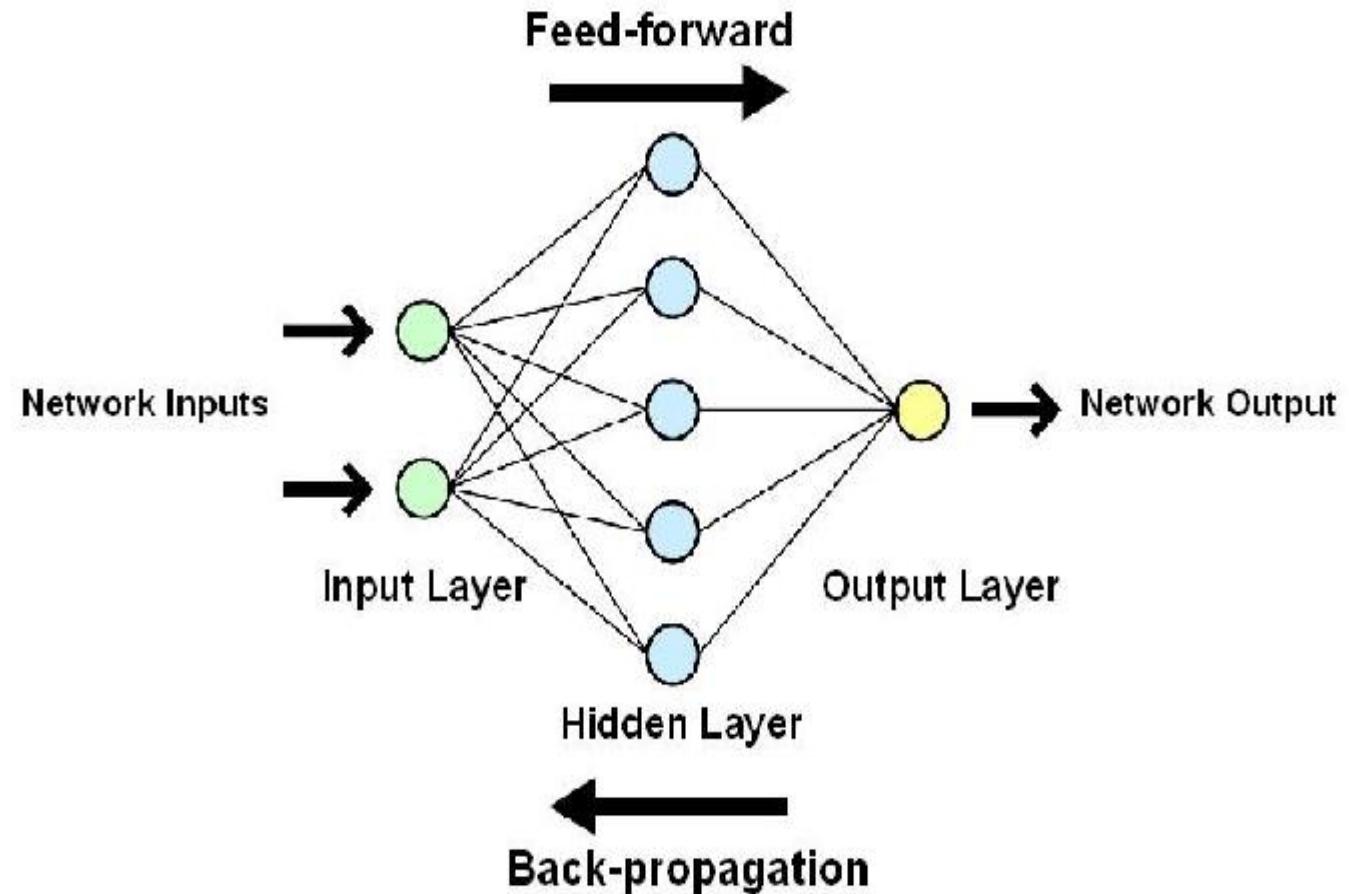
## Challenges with Artificial Neural Network (ANN)

---

- While solving an image classification problem using ANN, the first step is to convert a 2-dimensional image into a 1-dimensional vector prior to training the model. This has two drawbacks:
- The number of trainable parameters increases drastically with an increase in the size of the image
- One common problem in all these neural networks is the Vanishing and exploding gradient.
- This problem is associated with the backpropagation algorithm.

## Challenges with Artificial Neural Network (ANN)

- The weights of a neural network are updated through this backpropagation algorithm by finding the gradients:
- So, in the case of a very deep neural network (network with a large number of hidden layers), the gradient vanishes or explodes as it propagates backward which leads to vanishing and exploding gradient.
- ANN cannot capture sequential information in the input data which is required for dealing with sequence data



## Data-Driven Approach

---

- If we want to consider abandoning heuristics and best practices and take on a data-driven approach to algorithm selection:
- Feed the data 'as is' to a neural network with more than 10 layers: **deep learning!**
- The Model will figure out the features
- Cons: Why is it working? More importantly, why is it failing?  
If we cannot answer these questions: how do we improve the approach??  
Parameter tuning is a nightmare with many millions of knobs to turn
- What are people interested in Machine Learning/ Artificial Intelligence Research today?  
FATE in AI... that is, ensuring:
  - fairness
  - accountability
  - transparency
  - ethics

## Which classifier and what parameters?

---

**Deciding the best algorithm and tuning parameters (an automated solution approach):**

- Rather than picking your favorite algorithm, try 10 or 20 algorithms
- Double down on those that show signs of being better in performance, robustness, speed or whatever concerns interest you most
- Rather than picking the common parameters, grid search tens, hundreds or thousands of combinations of parameters.
- Become the objective scientist, leave behind anecdotes and study the intersection of complex learning systems and data observations from your problem domain.

## Automated Solution-Approach in Action

---

- This is a powerful approach that requires less up-front knowledge, but a lot more back-end computation and experimentation.
- As such, it is likely you will be required to work with a smaller sample of your dataset so that you can get results quickly
- We can have a test harness that we can have complete faith in.
- Note: how can you have complete trust in your test harness?
- You develop trust by selecting the test options in a data-driven manner that gives you objective confidence that your chosen configuration is reliable.
- The type of estimation method (split, boosting, k-fold cross validation, etc.) and its configuration (size of k, etc.).

## Leverage Automation

---

- The automated solution approach is a problem of [search](#)
- Leverage automation to answer: Which is the best algorithm?  
What are the best parameter values?
- You can write re-usable scripts to search for the most reliable test harness for our problem before we begin. No more ad hoc guessing.
- We can write a reusable script to try automatically 10, 20, 100 algorithms across a variety of libraries and implementations. No more favorite algorithms or libraries.
- The line between different algorithms is gone and a new parameter configuration is a new algorithm. we can write re-usable scripts to grid or random search each algorithm to truly sample its capability.
- Add feature engineering on the front so that each “view” on the data is a new problem for algorithms to be challenged against.
- Bolt-on ensembles at the end to combine some or all results (meta-algorithms).

## Summary on an Automated Solution Approach

---

- In the traditional Machine Learning (model-driven approach) we look at the common heuristic and best-practice approach for algorithm selection and parameter tuning
- Pros: Excellent rationale for the solution approach, clear interpretability
  - We can go back and analyze when something fails and fix it
- Cons: This approach requires human intervention, an understanding of the domain, is time consuming and possibly limit what the machine can do.
- We yearn for silver bullet general purpose best algorithms and best algorithm configurations, when no such things exist
- **There is no best general purpose machine learning algorithm**
- **There are no best general purpose machine learning algorithm parameters**
- The transferability of capability for an algorithm from one problem to another is questionable (look-up: **transfer learning**)
- The solution: **study the working of multiple algorithms on your problem**
- Automated approach: spot check algorithms, grid search for parameters and quickly find methods that yield good results, reliably (empirical justification for the choice of the learning algorithm and specific instantiation of parameters)

### Text Book:

“Business Analytics, The Science of Data-Driven Making”, U. Dinesh Kumar, Wiley 2017

“Recommender Systems, The text book, Charu C. Aggarwal, Springer 2016 Section 1.and Section 2.

# DATA ANALYTICS

## Image Courtesy

---

<https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/>

<https://www.analyticsvidhya.com/blog/2014/10/introduction-neural-network-simplified/>



---

## THANK YOU

---

**Jyothi R.**  
Assistant Professor,  
Department of Computer Science  
[jyothir@pes.edu](mailto:jyothir@pes.edu)





## DATA ANALYTICS

### Unit 4: Brief Review of Unsupervised Learning Algorithms (k-means, Hierarchical agglomerative clustering)

Jyothi R.

Department of Computer Science  
and Engineering

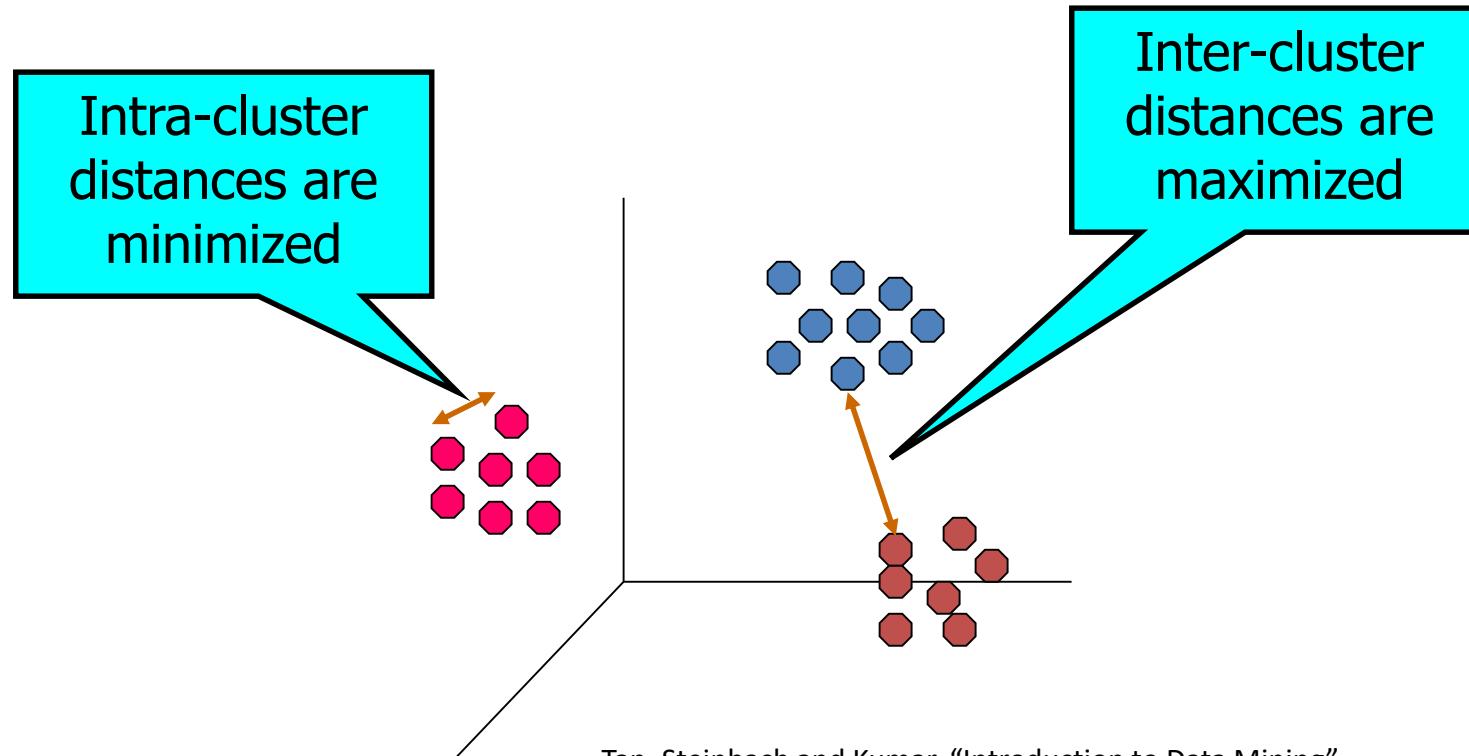
## Introduction

---

- Mastering unsupervised learning opens up a broad range of avenues for a data scientist.
- Clustering has wide application across domains and industries
- Examples: Uber's route optimization, Amazon's recommendation system, Netflix's customer segmentation, and so on...
- Why is it important?
  - Annotating data for supervised learning is time consuming, expensive and not always practical
  - We can use clustering to group similar data points for sampling
  - It provides a technique for describing the data and possibly getting insights (such as discovering subgroups within a known class, etc.)
- Basic clustering algorithms include K-Means clustering, hierarchical clustering, and the DBSCAN algorithm

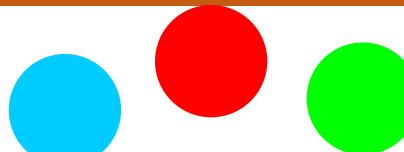
## What is clustering?

Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups

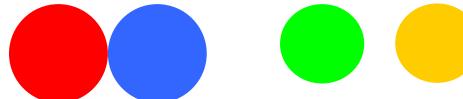


## Types of clusters

- Well-separated clusters



- Center-based clusters

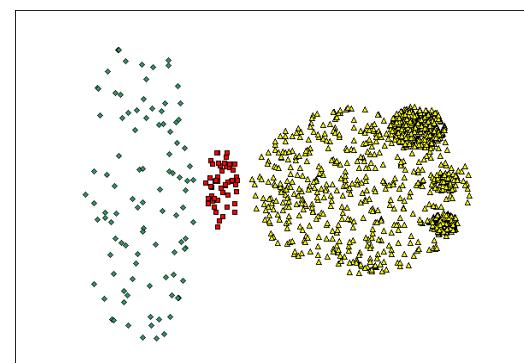


Mean or medoid is the 'prototype'

- Contiguous clusters



- Density-based clusters



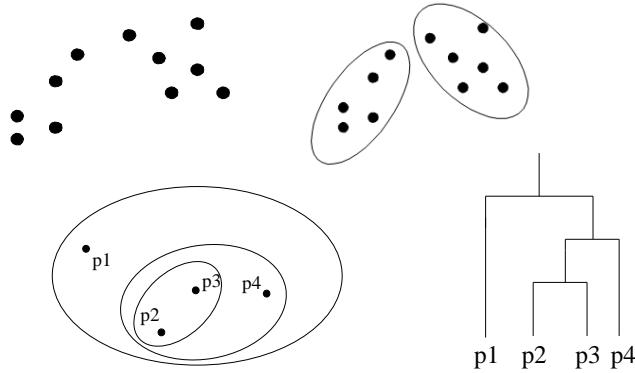
- Property or Conceptual

- Described by an Objective Function

Minimize the edge weight between clusters  
and maximize the edge weight within clusters

## Types of clustering techniques

- Partitional



- Hierarchical

- Exclusive (vs nonexclusive) – points can belong to multiple clusters
- Fuzzy vs nonfuzzy – points belong to every cluster with a weight in  $(0,1)$
- Property or Conceptual - we only want to cluster some of the data
- Described by an Objective Function – cluster of widely different sizes, shapes, and densities

## k-means clustering

---

Partitional clustering approach

Each cluster is associated with a **centroid** (center point)

Each point is assigned to the cluster with the closest centroid

Number of clusters,  $K$ , must be specified

The basic algorithm is very simple

- 
- 1: Select  $K$  points as the initial centroids.
  - 2: **repeat**
  - 3:   Form  $K$  clusters by assigning all points to the closest centroid.
  - 4:   Recompute the centroid of each cluster.
  - 5: **until** The centroids don't change
-

## k-means parameters

---

- Initial centroids are often chosen randomly.
  - Clusters produced vary from one run to another.
- The centroid is (typically) the mean of the points in the cluster.
- ‘Closeness’ is measured by Euclidean distance, cosine similarity, correlation, etc.
- K-means will converge for common similarity measures mentioned above.
- Most of the convergence happens in the first few iterations.
  - Often the stopping condition is changed to ‘Until relatively few points change clusters’
- Complexity is  $O( n * K * I * d )$ 
  - $n$  = number of points,  $K$  = number of clusters,  
 $I$  = number of iterations,  $d$  = number of attributes

## Evaluating k-means clusters

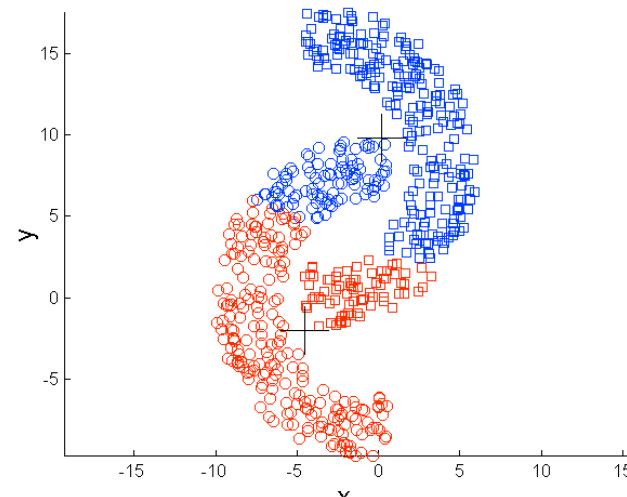
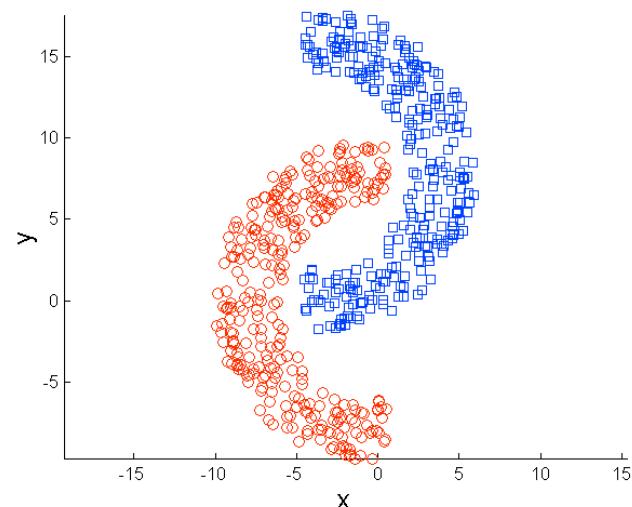
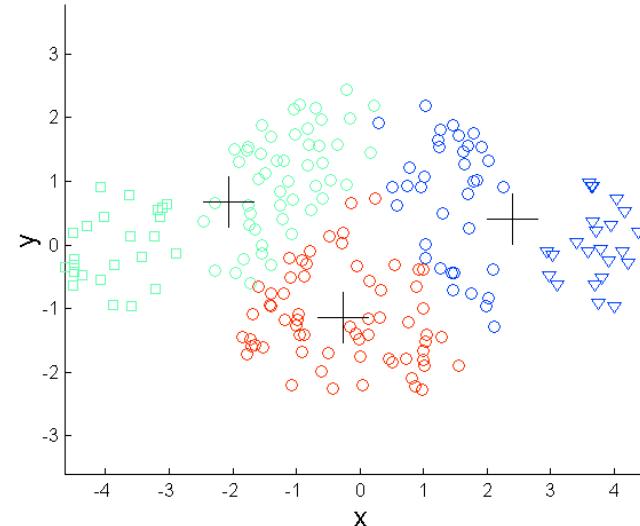
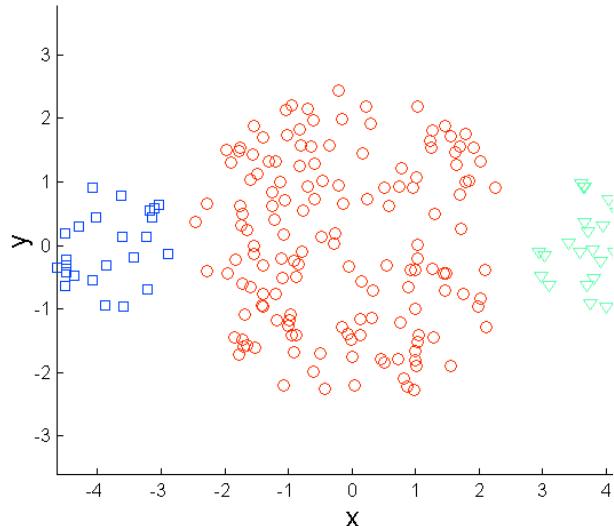
---

- Most common measure is Sum of Squared Error (SSE)
  - For each point, the error is the distance to the nearest cluster
  - To get SSE, we square these errors and sum them.

$$SSE = \sum_{i=1}^K \sum_{x \in C_i} dist^2(m_i, x)$$

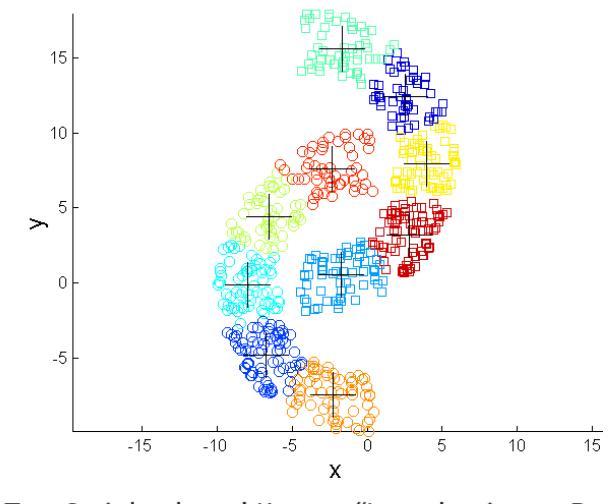
- $x$  is a data point in cluster  $C_i$  and  $m_i$  is the representative point for cluster  $C_i$ 
  - ◆ can show that  $m_i$  corresponds to the center (mean) of the cluster
- Given two clusters, we can choose the one with the smallest error
- One easy way to reduce SSE is to increase  $K$ , the number of clusters
  - ◆ A good clustering with smaller  $K$  can have a lower SSE than a poor clustering with higher  $K$

## k-means clustering – some cons



- Different configurations for each run due to random initialization
- Works well only for similarly shaped (or sized clusters)
- Does not work well for inherently nonglobular clusters

Try: (1) Choosing  $k \gg$  no. of clusters  
 (2) Run kmeans multiple times;  
 select the best configuration



## Quality and optimal number of clusters

- Number of clusters and recommended index (Calinski and Harabasz)

$$CH(k) = \frac{B(k)/k-1}{W(k)/(n-k)}$$

- Hartigan statistic

$$H(k) = \{(W(k)/W(k-1)) - 1\}/(n-k-1)$$

- Silhouette statistic

Which can be also written as:

$$S(i) = (b(i)-a(i))/\max\{a(i), b(i)\}$$
$$s(i) = \begin{cases} 1 - a(i)/b(i), & \text{if } a(i) < b(i) \\ 0, & \text{if } a(i) = b(i) \\ b(i)/a(i) - 1, & \text{if } a(i) > b(i) \end{cases}$$

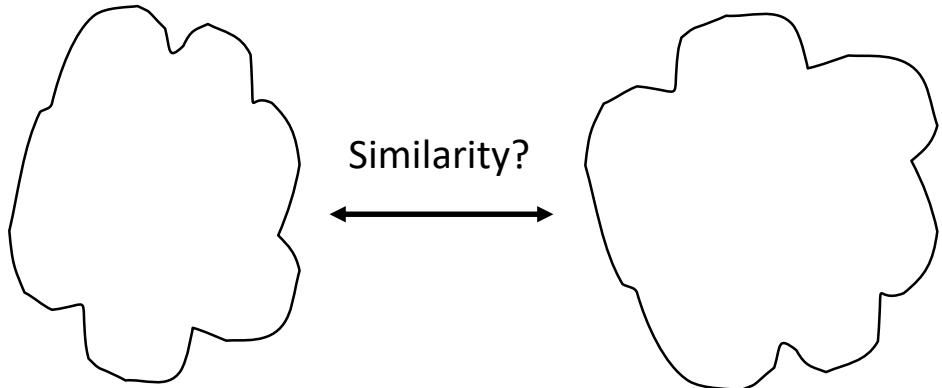
From the above definition it is clear that

$$-1 \leq s(i) \leq 1$$

For data point  $i \in C_i$  (data point  $i$  in the cluster  $C_i$ ), let

$$a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i, i \neq j} d(i, j) \quad b(i) = \min_{k \neq i} \frac{1}{|C_k|} \sum_{j \in C_k} d(i, j)$$

## Agglomerative hierarchical clustering



	p1	p2	p3	p4	p5	...
p1						
.						

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

### Text Book:

“Business Analytics, The Science of Data-Driven Making”, U. Dinesh Kumar, Wiley 2017 ([Chapter 14.1-14.2.6, 14.3-14.6](#))

“Recommender Systems, The text book, Charu C. Aggarwal, Springer 2016 Section 1.and Section 2.

# DATA ANALYTICS

Image Courtesy

---



<https://www.analyticsvidhya.com/blog/2020/09/how-dbscan-clustering-works/>



---

## THANK YOU

---

**Jyothi R.**  
Assistant Professor,  
Department of Computer Science  
[jyothir@pes.edu](mailto:jyothir@pes.edu)





# DATA ANALYTICS

## Unit 4: Clustering Algorithms - DBSCAN

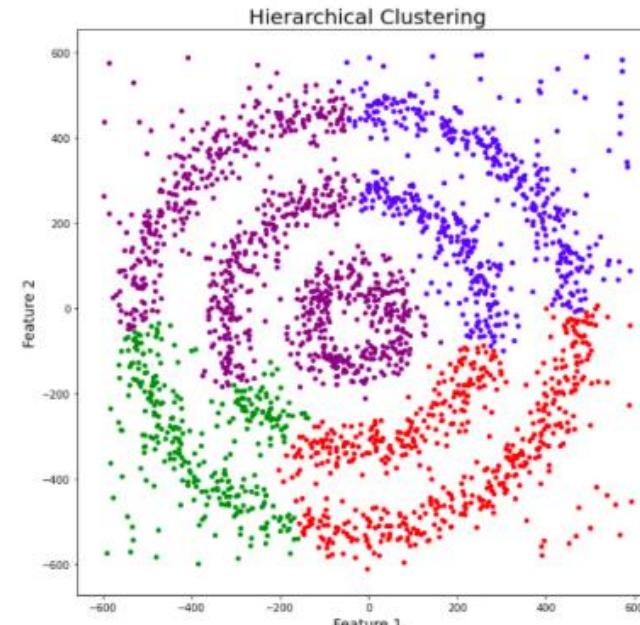
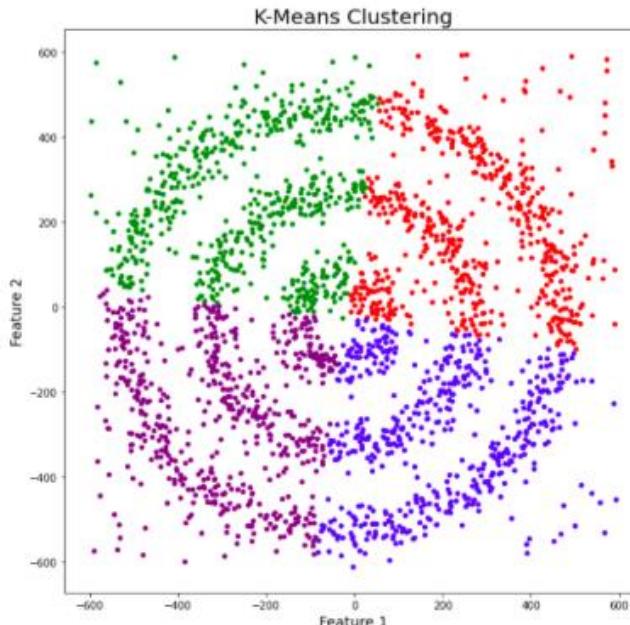
---

**Jyothi R.**

Department of Computer Science  
and Engineering

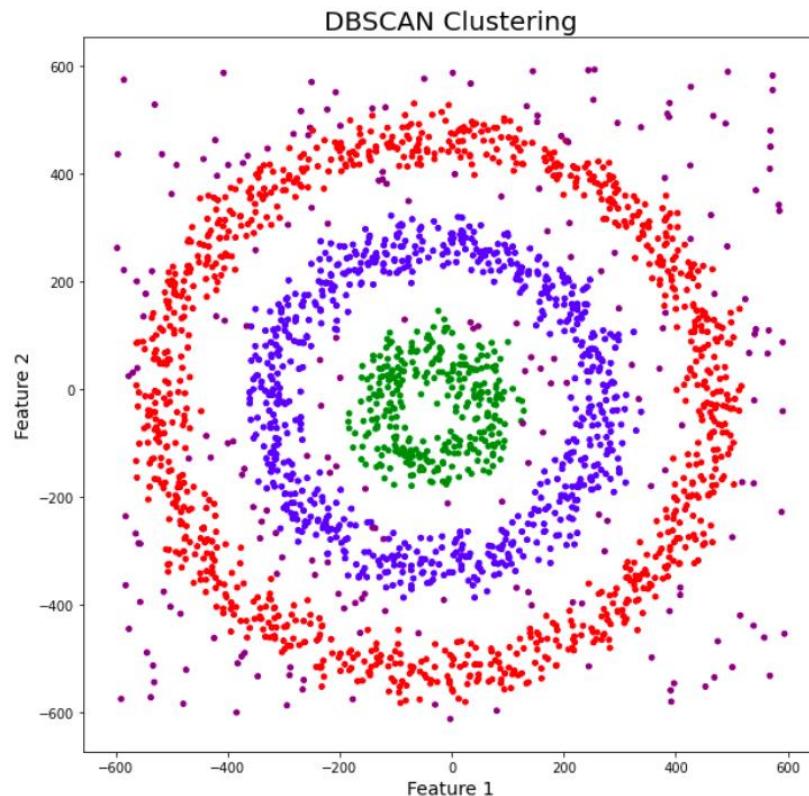
## Why do we need DBSCAN Clustering?

- Three different dense clusters in the form of concentric circles with some noise here.
- Now, let's run K-Means and Hierarchical clustering algorithms and see how they cluster these data points.
- There are four colors in the graph. Noise is considered as a different cluster which is represented by the purple color.
- Sadly, both of them failed to cluster the data points. Also, they were not able to properly detect the noise present in the dataset.



## Why do we need DBSCAN Clustering?

- Now, let's take a look at the results from DBSCAN clustering.
- DBSCAN is not just able to cluster the data points correctly, but it also perfectly detects noise in the dataset.



## What Exactly is DBSCAN Clustering?

---

- DBSCAN stands for Density-Based Spatial Clustering of Applications with Noise.
- It was proposed by Martin Ester et al. in 1996. DBSCAN is a density-based clustering algorithm that works on the assumption that clusters are dense regions in space separated by regions of lower density.
- It can identify clusters in large spatial datasets by looking at the local density of the data points.

## DBSCAN Clustering

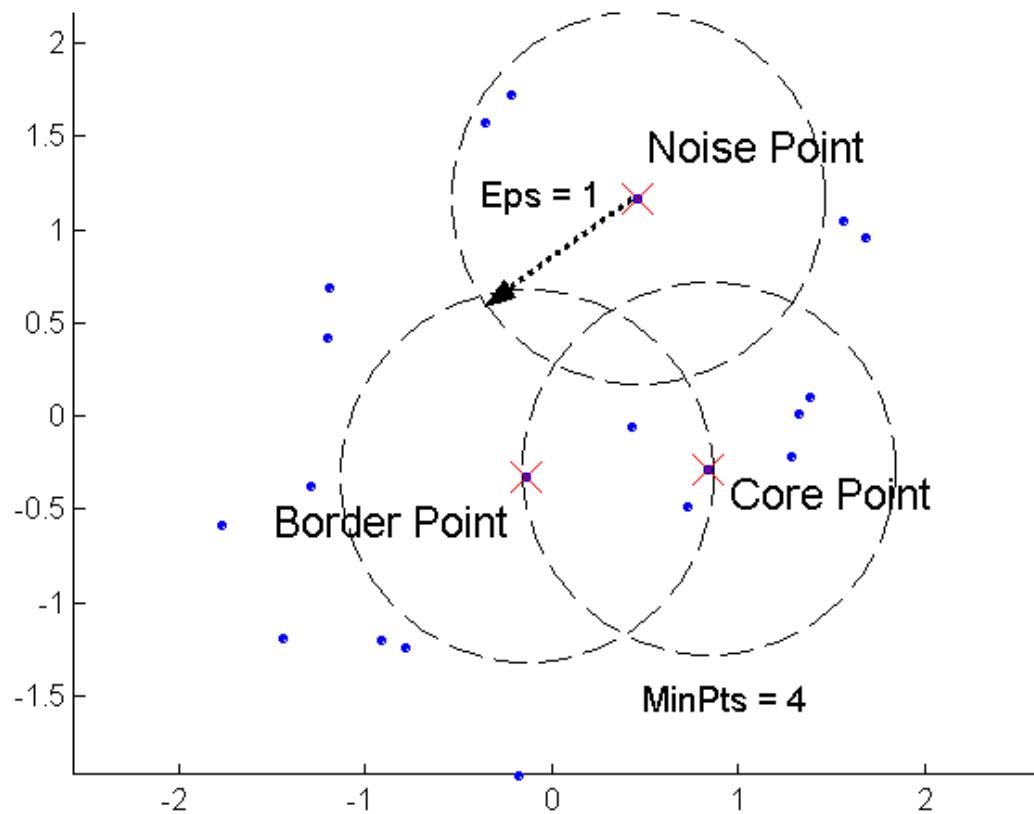
Density = number of points within a specified radius ( $\text{Eps}$ )

A point is a **core point** if it has more than a specified number of points ( $\text{MinPts}$ ) within  $\text{Eps}$

These are points that are at the interior of a cluster

A **border point** has fewer than  $\text{MinPts}$  within  $\text{Eps}$ , but is in the neighborhood of a core point

A **noise point** is any point that is not a core point or a border point.



## DBSCAN Algorithm – Labeling Clusters

---

- Eliminate noise points
- Perform clustering on the remaining points

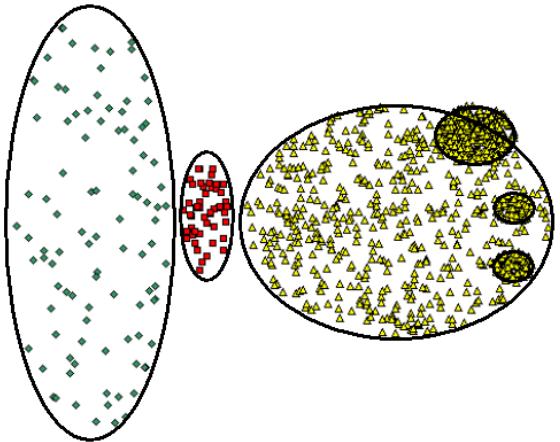
```
current_cluster_label ← 1
for all core points do
    if the core point has no cluster label then
        current_cluster_label ← current_cluster_label + 1
        Label the current core point with cluster label current_cluster_label
    end if
    for all points in the  $Eps$ -neighborhood, except  $i^{th}$  the point itself do
        if the point does not have a cluster label then
            Label the point with cluster label current_cluster_label
        end if
    end for
end for
```

## What Exactly is DBSCAN Clustering?

---

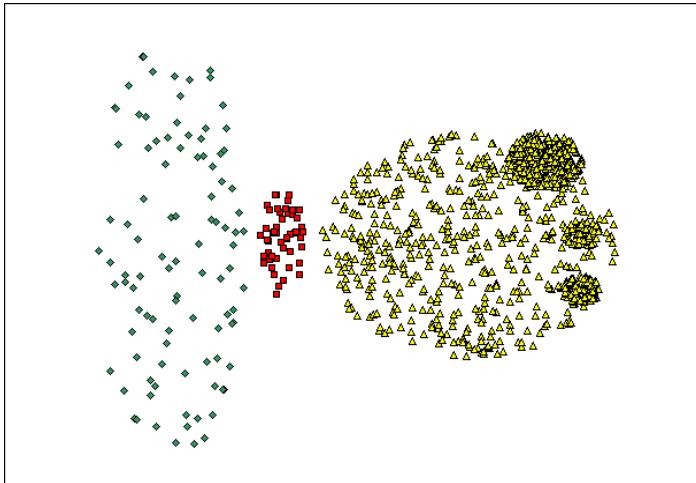
- The most exciting feature of DBSCAN clustering is that it is robust to outliers
- It also does not require the number of clusters to be set beforehand, unlike k-means, where we have to specify the number of centroids
- DBSCAN requires only two parameters: epsilon and minPoints
- Epsilon is the radius of the circle to be created around each data point to check the density and
- minPoints is the minimum number of data points required inside that circle for that data point to be classified as a Core point

## DBSCAN Clustering – Some Results

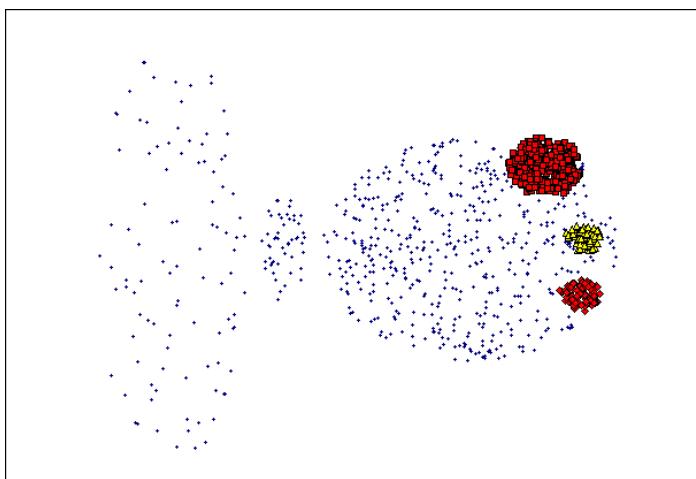


Original Points

- Varying densities
- High-dimensional data



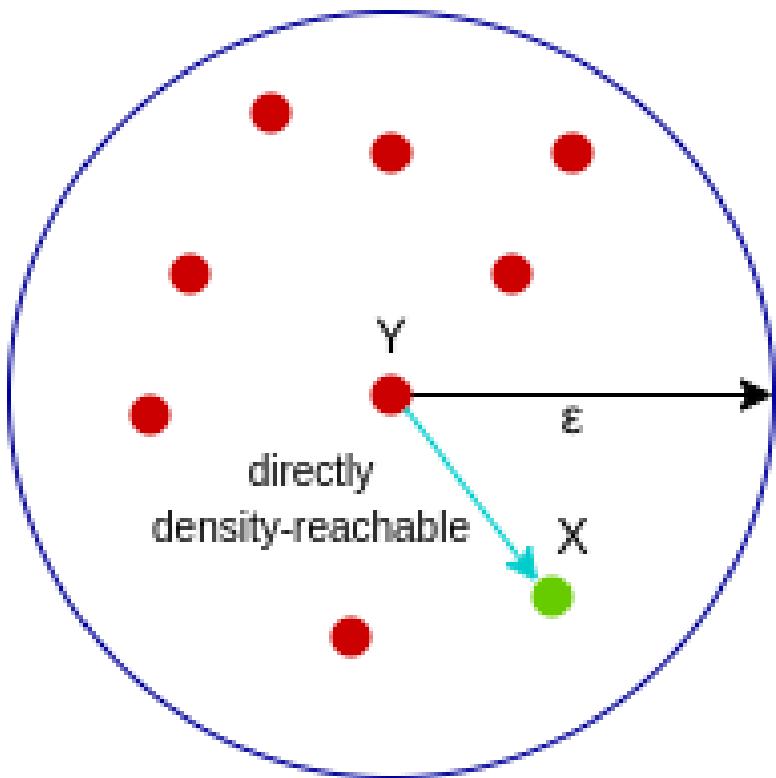
(MinPts=4, Eps=9.75).



(MinPts=4, Eps=9.92)

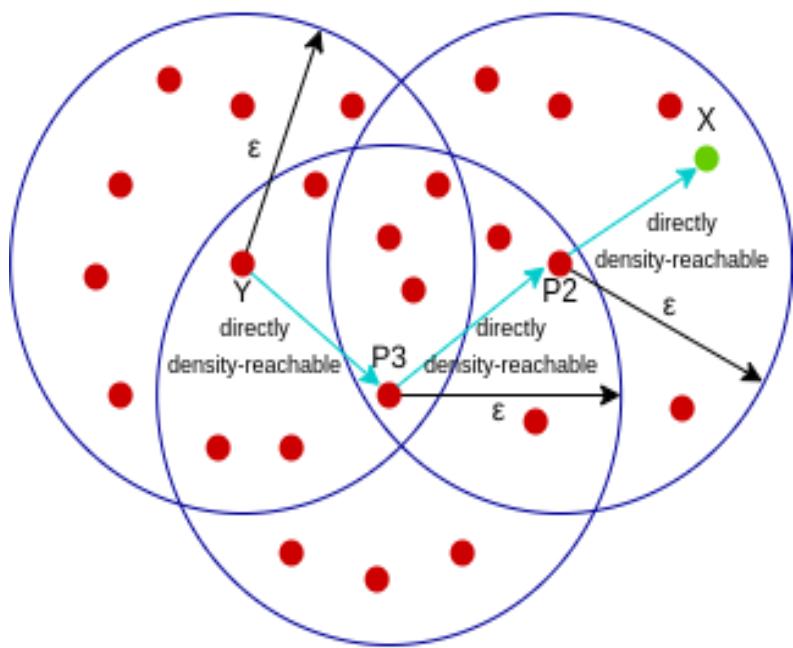
## Reachability and Connectivity

- X belongs to the neighborhood of Y, i.e,  $\text{dist}(X, Y) \leq \text{epsilon}$
- Y is a core point
- Here, X is directly density-reachable from Y, but vice versa is not valid.



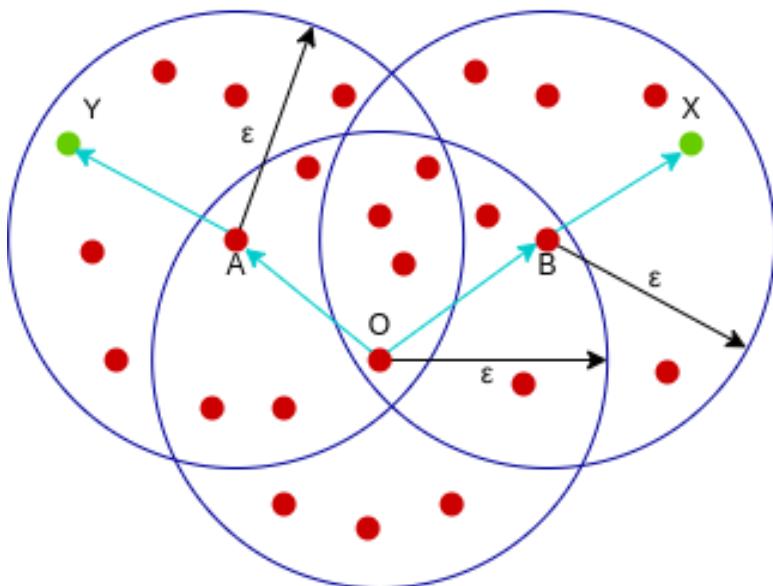
## Reachability and Connectivity

- A point X is density-reachable from point Y w.r.t epsilon, minPoints,
- if there is a chain of points p1, p2, p3, ..., pn and p1=X and pn=Y such that pi+1 is directly density-reachable from pi.
- Here, X is density-reachable from Y with X being directly density-reachable from P2, P2 from P3, and P3 from Y. But, the inverse of this is not valid.



## Reachability and Connectivity

- A point X is density-connected from point Y w.r.t epsilon and minPoints.
- if there exists a point O such that both X and Y are density-reachable from O w.r.t to epsilon and minPoints.
- Here, both X and Y are density-reachable from O, therefore, we can say that X is density-connected from Y.



## Parameter Selection in DBSCAN Clustering

---

- DBSCAN is very sensitive to the values of epsilon and minPoints.
- Therefore, it is very important to understand how to select the values of epsilon and minPoints.
- A slight variation in these values can significantly change the results produced by the DBSCAN algorithm.
- The value of minPoints should be at least one greater than the number of dimensions of the dataset, i.e.,  
 $\text{minPoints} \geq \text{Dimensions} + 1$ .
- It does not make sense to take minPoints as 1 because it will result in each point being a separate cluster.
- Therefore, it must be at least 3.

## Parameter Selection in DBSCAN Clustering

---

- Generally, MinPts is set to twice the dimensions.  
But domain knowledge also decides its value.
- The value of epsilon can be decided from the K-distance graph.
- The point of maximum curvature (elbow) in this graph tells us about the value of epsilon.
- If the value of epsilon chosen is too small then a higher number of clusters will be created, and more data points will be taken as noise.
- Whereas, if chosen too big then various small clusters will merge into a big cluster, and we will lose details.

## Cluster validation

---

**Cluster Cohesion or compactness:** Measures how closely related are objects in a cluster

Example: SSE

**Cluster Separation:** Measure how distinct or well-separated a cluster is from other clusters

Example: Squared Error

Cohesion is measured by the **within cluster sum of squares (WSS)**

$$WSS = \sum_i \sum_{x \in C_i} (x - m_i)^2$$

Separation is measured by the **between cluster sum of squares (BSS)**

$$BSS = \sum_i |C_i| (m - m_i)^2$$

Where  $|C_i|$  is the size of cluster i

## References

---

Chapter 7.6 of R1 Han, Kamber and Pei (2nd Edn)

OR

Chapter 10.4 of R1 Han, Kamber and Pei (3rd Edn)

T1: Business Analytics, The Science of Data-Driven Making”, U. Dinesh Kumar, Wiley 2017 ([Chapter 14.4.4](#))

<https://www.analyticsvidhya.com/blog/2020/09/how-dbscan-clustering-works/>



**PES**  
UNIVERSITY  
ONLINE

# DATA ANALYTICS

## **Unit 4: Content Based Analysis- Dealing with Textual Data**

**Jyothi R.**

Department of Computer Science  
and Engineering

## Introduction

---

1. Basic Components of Content-Based Systems
2. Preprocessing and Feature Extraction
3. Examples

## Introduction

---

- Content-based systems are designed to exploit scenarios in which items can be described with a descriptive sets of attributes (user profile, product features, etc.).
- And in such cases where a user's own ratings and actions on other movies are insufficient to discover meaningful recommendations.
- This approach is particularly useful when the item is new, and there are few ratings available for that item (cold start problem of collaborative filtering)

## Introduction

---

- Content-based recommender systems try to match users to items that are similar to what they have liked in the past.
- This similarity is not necessarily based on rating correlations across users but on the basis of the attributes of the objects liked by the user.
- Unlike collaborative systems, which explicitly leverage the ratings of other users in addition to that of the target user, **content-based systems largely focus on the target user's own ratings and the attributes of the items liked by the user.**
- Therefore, the **other users have little, if any, role to play** in content-based systems.
- In other words, the content-based methodology leverages a different source of data for the recommendation process.

## Content-Based Recommender Systems

---

- At the most basic level, content-based systems are dependent on two sources of data:
  1. The first source of data is a **description of various items** in terms of content-centric attributes.  
Example of such a representation could be the text description of an item by the manufacturer.
  2. The second source of data is a **user profile**, which is **generated from user feedback about various items**.
- The user **feedback might be explicit or implicit**. Explicit feedback may correspond to ratings, whereas implicit feedback may correspond to user actions.

## Content-Based Recommender Systems

---

- In a Content-based systems the ratings are collected in a way similar to collaborative systems.
- The user profile relates the attributes of the various items to user interests
- A very basic example of a user profile might simply be a set of labeled training documents of item descriptions, the user ratings as the labels, and a classification or regression model relating the item attributes to the user ratings.
- The specific user profile is heavily dependent on the methodology at hand. For example, explicit ratings might be used in one setting, and implicit feedback might be used in another.
- It is also possible for the user to specify her own profile in terms of keywords of interest,

## Content-Based Recommender Systems

---

- Content-based systems are largely used in scenarios in which a significant amount of attribute information is available at hand.
- In many cases, these attributes are keywords, which are extracted from the product descriptions.
- In fact, the vast majority of content based systems extract text attributes from the underlying objects.
- Content-based systems are, therefore, particularly well suited to giving recommendations in text-rich and unstructured domains.
- A classical example of the use of such systems is in the recommendation of Web pages.
- For example, the previous browsing behavior of a user can be utilized to create a content-based recommender system.

## Content-Based Recommender Systems

---

- The use of such systems is not restricted only to the Web domain.
- Keywords from product descriptions are used to create item and user profiles for the purposes of recommendations in other e-commerce settings.
- In other settings, relational attributes such as manufacturer, genre, and price, may be used in addition to keywords.
- Such attributes can be used to create structured representations, which can be stored in a relational database.
- In these cases, it is necessary to combine the structured and unstructured attributes in a single structured representation.
- The basic principles of content-based systems, however, remain invariant to whether a structured or unstructured representation is used.

## How do Content Based Recommender Systems work?

---

- A content based recommender works with data that the user provides, either explicitly (rating) or implicitly (clicking on a link).
- Based on that data, a user profile is generated, which is then used to make suggestions to the user. As the user provides more inputs or takes actions on the recommendations, the engine becomes more and more accurate.

## How do Content Based Recommender Systems work?

---

**1. Preprocessing and feature extraction:** Content-based systems are used in a wide variety of domains, such as Web pages, product descriptions, news, music features, and so on. In most cases, features are extracted from these various sources to convert them into a keyword-based vector-space representation. This is the first step of any content based recommendation system, and it is highly domain-specific. However, the proper extraction of the most informative features is essential for the effective functioning of any content-based recommender system.

**2. Content-based learning of user profiles:** As discussed earlier, a content-based model is specific to a given user. Therefore, a user-specific *model* is constructed to predict user interests in items, based on their past history of either buying or rating items. In order to achieve this goal, user feedback is leveraged, which may be manifested in the form of previously specified ratings (explicit feedback) or user activity (implicit feedback). Such feedbacks are used in conjunction with the attributes of the items in order to construct the training data. A learning model is constructed on this training data. This stage is often not very different from classification or regression modeling, depending on whether the feedback is categorical (e.g., binary act of selecting an item), or whether the feedback is numerical (e.g., ratings or buying frequency). The resulting model is referred to as the *user profile* because it conceptually relates user interests (ratings) to item attributes.

**3. Filtering and recommendation:** In this step, the learned model from the previous step is used to make recommendations on items for specific users. It is important for this step to be very efficient because the predictions need to be performed in real time.

# DATA ANALYTICS

## Example: Movie Recommendation

---

Consider a movie recommendation site such as IMDb, that provides personalized recommendations for movies. Each movie is usually associated with a description of the movie such as its synopsis, the director, actors, genre, and so on. A short description of *Shrek* at the IMDb Website is as follows:

“After his swamp is filled with magical creatures, an ogre agrees to rescue a princess for a villainous lord in order to get his land back.”

Many other attributes, such as [user tags](#), are also available, which can be [treated as content centric keywords](#). In the case of *Shrek*, one might simply [concatenate all the keywords in the various fields to create a text description](#). The main problem is that the various keywords may not have equal importance in the recommendation process. For example, a particular actor might have greater importance in the recommendation than a word from the synopsis. This can be achieved in two ways:

1. [Domain-specific knowledge](#) can be used to decide the relative importance of keywords.

For example, the title of the movie and the primary actor may be given more weight than the words in the description. In many cases, this process is done in a heuristic way with trial and error.

2. In many cases, it may be possible to [learn the relative importance of various features in an automated way](#). This process is referred to as feature weighting, which is closely related to feature selection. Both feature weighting and feature selection are described in a later section.

## How do we convert text to numbers for content based recommendation?

---

- The concepts of Term Frequency (TF) and Inverse Document Frequency (IDF) are used in information retrieval systems and also content based filtering mechanisms (such as a content based recommender).
- They are used to determine the relative importance of a document / article / news item / movie etc.

## How do we convert text to numbers for content based recommendation?

---

### Term Frequency (TF) and Inverse Document Frequency (IDF)

- TF is simply the frequency of a word in a document.
- IDF is the inverse of the document frequency among the whole corpus of documents.
- TF-IDF is used mainly because of two reasons: Suppose we search for “the rise of analytics” on Google.
- It is certain that “the” will occur more frequently than “analytics” but the relative importance of analytics is higher than the search query point of view.  
In such cases, TF-IDF weighting negates the effect of high frequency words in determining the importance of an item (document).

## How do we convert text to numbers for content based recommendation?

### Term Frequency (TF) and Inverse Document Frequency (IDF)

- But while calculating TF-IDF, log is used to dampen the effect of high frequency words.
- For example: TF = 3 vs TF = 4 is vastly different from TF = 10 vs TF = 1000.
- In other words the relevance of a word in a document cannot be measured as a simple raw count and hence may incorporate weights as shown below:

$$w_{t,d} = \begin{cases} 1 + \log_{10} tf_{t,d}, & \text{if } tf_{t,d} > 0 \\ 0, & \text{otherwise} \end{cases}$$

Term Frequency	Weighted Term Frequency
0	0
10	2
1000	4

### Inverse document frequency (IDF)

$\log_{10}(n/n_i)$  where n is the total number of documents and  
 $n_i$  the number of documents in which the term  $i^{\text{th}}$  term appears

**TF-IDF is a product of the term frequency and inverse document frequency**

## How do we convert text to numbers for content based recommendation?

---

- It can be seen that the effect of high frequency words is damped and these values are more comparable to each other as opposed to the original raw term frequency.
- Hence the computation of TF-IDF is preceded by
  - Stop word removal (eliminating articles (a, an, the), prepositions, conjunctions, and pronouns)
  - Stemming (hoping, hoped, hope -> the root word 'hop')  
(this can be detrimental as hope and hop (jump) can be confused with each other)
  - Phrase extraction ("cross examine" can mean something different from 'cross' or 'examine')
- After calculating TF-IDF scores, how do we determine which items are closer to each other, rather closer to the user profile?
- This is accomplished using the Vector Space Model which computes the proximity based on the angle between the vectors.

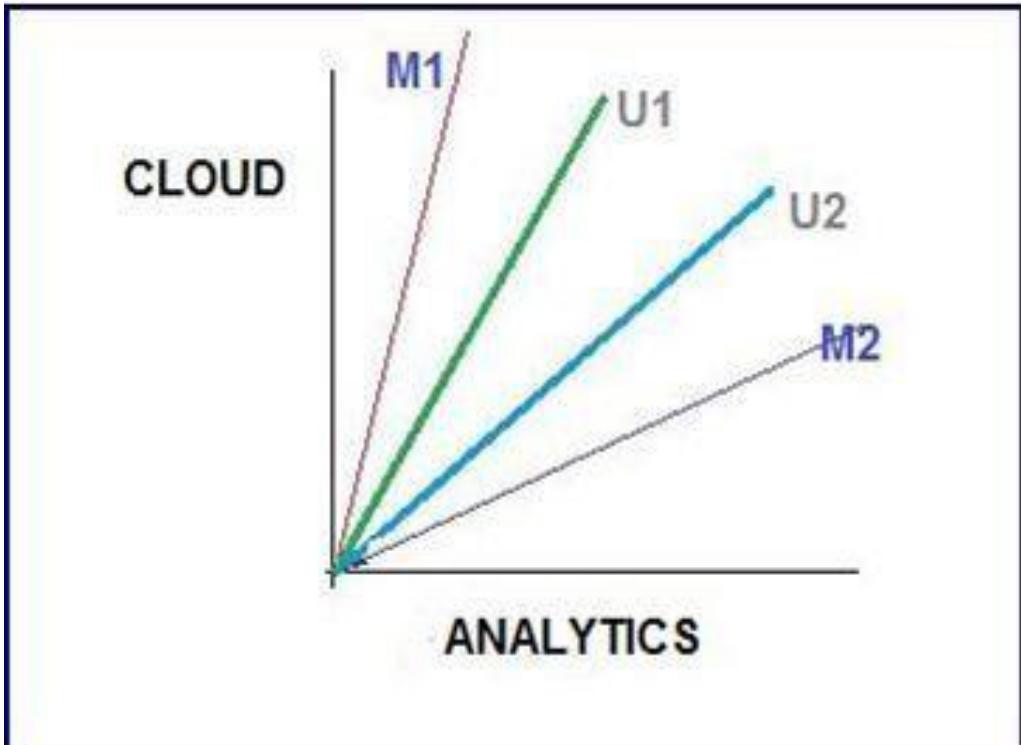
## How does a Vector Space Model work?

---

- In this model, each item is stored as a vector of its attributes (which are also vectors) in an n-dimensional space and the angles between the vectors are calculated to determine the similarity between the vectors.
- Next, the user profile vectors are also created based on his actions on previous attributes of items and the similarity between an item and a user is also determined in a similar way.

## How does a Vector Space Model work?

Lets try to understand this with an example.



- Shown above is a 2-D representation of a two attributes, Cloud and Analytics.
- M1 & M2 are documents.
- U1 & U2 are users.
- The document M2 is more about Analytics than cloud whereas M1 is more about cloud than Analytics.
- User U1, likes articles on the topic 'cloud' more than the ones on 'analytics' and vice-versa for user U2.
- The method of calculating the user's likes / dislikes / measures is calculated by taking the cosine of the angle between the user profile vector( $U_i$ ) and the document vector.

## Case study – How do we calculate TF – IDF ?

- Let's understand this with an example. Suppose, we search for "IoT and analytics" on Google and the top 5 links that appear have some frequency count of certain words as shown below:

Articles	Analytics	Data	Cloud	Smart	Insight
<u>Article 1</u>	21	24	0	2	2
<u>Article 2</u>	24	59	2	1	0
<u>Article 3</u>	40	115	8	10	19
<u>Article 4</u>	4	28	5	0	1
<u>Article 5</u>	8	48	4	3	4
<u>Article 6</u>	17	49	8	0	5
DF	5,000	50,000	10,000	5,00,000	7000

- Among the corpus of documents / blogs which are used to search for the articles, 5000 contains the word analytics, 50,000 contain data and similar count goes for other words. Let us assume that the total corpus of docs is 1 million( $10^6$ ).

## Term Frequency (TF)

- As seen in the image below, for article 1, the term Analytics has a TF of  $1 + \log_{10} 21 = 2.322$ . In this way, TF is calculated for other attributes of each of the articles. These values make up the attribute vector for each of the articles.

Articles	Analytics	Data	Cloud	Smart	Insight	Length of Vector
<u>Article 1</u>	2.322219295	2.380211242	0	1.301029996	1.301029996	3.800456039
<u>Article 2</u>	2.380211242	2.770852012	1.301029996	1	0	4.004460697
<u>Article 3</u>	2.602059991	3.06069784	1.903089987	2	2.278753601	5.380804488
<u>Article 4</u>	1.602059991	2.447158031	1.698970004	0	1	3.527276247
<u>Article 5</u>	1.903089987	2.681241237	1.602059991	1.477121255	1.602059991	4.257450611
<u>Article 6</u>	2.230448921	2.69019608	1.903089987	0	1.698970004	4.326697114

## How does Vector Space Model works?

---

- The reason behind using cosine is that the value of cosine will increase with decreasing value of the angle between which signifies more similarity.
- The vectors are **length normalized** after which they become vectors of length 1 and then the cosine calculation is simply the sum-product of vectors.

## References

---

### Text Book:

“Business Analytics, The Science of Data-Driven Making”, U. Dinesh Kumar, Wiley 2017

“Recommender Systems, The text book, Charu C. Aggarwal, Springer 2016 [Chapter 4](#)

# DATA ANALYTICS

## Image Courtesy

---



<https://towardsdatascience.com/introduction-to-two-approaches-of-content-based-recommendation-system-fc797460c18c>

<https://www.kaggle.com/ashish95arora/content-based-recommender-using-text-mining>



---

# THANK YOU

---

**Jyothi R.**  
Assistant Professor,  
Department of Computer Science  
[jyothir@pes.edu](mailto:jyothir@pes.edu)





# DATA ANALYTICS

## Unit 4:Text Classification and Clustering

---

**Jyothi R., Gowri Srinivasa**  
Department of Computer Science  
and Engineering

## Introduction to Text Classification

---

- The problem of text classification is closely related to that of classification of records with set-valued features.
- However, this model assumes that only information about the presence or absence of words is used in a document.
- In reality, the frequency of words also plays a helpful role in the classification process, and the typical domain-size of text data (the entire lexicon size) is much greater than a typical set-valued classification problem.

## Introduction to Text Classification

---

- The problem of text classification finds applications in a wide variety of domains in text mining. Some examples of domains in which text classification is commonly used are as follows:
- **News filtering and Organization:** Most of the news services today are electronic in nature in which a large volume of news articles is created every day. It is difficult to organize these news articles manually. Therefore, automated methods can be very useful for news categorization in a variety of web portals. This application is also referred to as text filtering.
- **Document Organization and Retrieval:** A variety of supervised methods may be used for document organization in many domains. These include large digital libraries of documents, web collections, scientific literature or even social feeds. Hierarchically organized document collections can be particularly useful for browsing and retrieval.
- **Opinion Mining:** Customer reviews or opinions are often short text documents which can be mined to determine useful information from the review.
- **Email Classification and Spam Filtering:** It is often desirable to classify email in order to determine either the subject or to determine irrelevant or junk email in an automated way. This is also referred to as spam filtering or email filtering.

## What makes text classification different?

---

- A wide variety of techniques have been designed for text classification.
- Note that these classes of techniques also generally exist for other data domains such as quantitative or categorical data.
- Since text may be modeled as quantitative data with frequencies on the word attributes, it is possible to use most of the methods for quantitative data directly on text.
- However, text is a particular kind of data in which the **word attributes are sparse, and high dimensional, with low frequencies for most of the words.**
- Therefore, it is critical to design classification methods which effectively account for these characteristics of text.

## Methods Used for Text Classification

---

- Some methods, which are commonly used for text classification are as follows:
- **Decision Trees:** Decision trees are designed with the use of a hierarchical division of the underlying data space with the use of different text features.
  - The hierarchical division of the data space is designed in order to create class partitions which are more skewed in terms of their class distribution.
  - For a given text instance, we determine the partition that it is most likely to belong to, and use it for the purposes of classification.
- **Pattern (Rule)-based Classifiers:** In rule-based classifiers we determine the word patterns which are most likely to be related to the different classes.
  - We construct a set of rules, in which the left-hand side corresponds to a word pattern, and the right-hand side corresponds to a class label.
- **SVM Classifiers:** SVM Classifiers attempt to partition the data space with the use of linear or non-linear delineations between the different classes.
- The key in such classifiers is to determine the optimal boundaries between the different classes

## Methods Used for Text Classification

---

- **Neural Network Classifiers:** Neural networks are used in a wide variety of domains for the purposes of classification.
  - In the context of text data, the main difference for neural network classifiers is to adapt these classifiers with the use of word features.
  - We note that neural network classifiers are related to SVM classifiers; indeed, they both are in the category of discriminative classifiers, which are in contrast with the generative classifiers.
- **Bayesian (Generative) Classifiers:** We attempt to build a probabilistic classifier based on modeling the underlying word features in different classes.
  - The idea is then to classify text based on the posterior probability of the documents belonging to the different classes on the basis of the word presence in the documents.
- **Other Classifiers:** Almost all classifiers can be adapted to the case of text data, such as nearest neighbor classifiers, and genetic algorithm-based classifiers.

## Feature Selection for Text Classification

---

- Before any classification task, one of the most fundamental tasks that needs to be accomplished is that of document representation and feature selection.
- While feature selection is also desirable in other classification tasks, it is especially important in text classification due to the high dimensionality of text features and the existence of irrelevant (noisy) features.
- In general, text can be represented in two separate ways.
- The first is as a [bag of words](#), in which a document is represented as a set of words, together with their associated frequency in the document. Such a representation is essentially independent of the sequence of words in the collection.
- The second method is to represent text directly as strings, in which each document is a sequence of words ([n-grams – unigram, bigram \(a sequence of two words\), trigram \(three words\), etc.](#)).
- Most text classification methods use the bag-of-words representation because of its simplicity for classification purposes.

## Feature Selection for Text Classification

---

- The most common feature selection which is used in both supervised and unsupervised applications is that of stop-word removal and stemming.
- In stop-word removal, we determine the common words in the documents which are not specific or discriminatory to the different classes.
- In stemming, different forms of the same word are consolidated into a single word.
- For example, singular, plural and different tenses are consolidated into a single word.
- We note that these methods are not specific to the case of the classification problem, and are often used in a variety of unsupervised applications such as clustering and indexing.
- In the case of the classification problem, it makes sense to supervise the feature selection process with the use of the class labels.
- This kind of selection process ensures that those features which are highly skewed towards the presence of a particular class label are picked for the learning process.

## Typical tasks with processing text data

---

- Sentence segmentation (and tokenization) – how many words? where does a sentence begin/ end?
- Stemming and Lemmatization – running to run or am, are, is, etc., mapped to the root ‘be’
- Parts of Speech (PoS) tagging – noun, verb, article, adjective, etc. (role in the sentence)
- Named entity recognition – person, location, organization, etc.
- Relation extraction – “[Tim Cook](#) is the [CEO](#) of [Apple](#)”
- Sentiment analysis – positive, negative or neutral? How positive or negative or neutral?
- Semantic analysis – meaning/ event/ intent extraction from sentences
- Topic classification/ key word extraction
- Information retrieval/ text summarization
- Automated response generation

## Interaction of Feature Selection with Classification

---

- Since the classification and feature selection processes are dependent upon one another, it is interesting to test how the feature selection process interacts with the underlying classification algorithms.
- In this context, two questions are relevant:
- Can the feature-specific insights obtained from the intermediate results of some of the classification algorithms be used for creating feature selection methods that can be used more generally by other classification algorithms?
- Do the different feature selection methods work better or worse with different kinds of classifiers?

## Interaction of Feature Selection with Classification

---

- In regard to the first question, it was shown in that feature selection which was derived from linear classifiers, provided very effective results.
- In regard to the second question, it was shown in that the sophistication of the feature selection process itself was more important than the specific pairing between the feature selection process and the classifier.

## Interaction of Feature Selection with Classification

---

- Linear Classifiers are those for which the output of the linear predictor is defined to be  $p = \overline{A} \cdot \overline{X} + b$ ,
- where  $\overline{X} = (x_1 \dots x_n)$  is the normalized document word frequency vector,  $\overline{A} = (a_1 \dots a_n)$  is a vector of linear coefficients with the same dimensionality as the feature space, and  $b$  is a scalar.

## Interaction of Feature Selection with Classification

---

- Both the basic neural network and basic SVM classifiers belong to this category.
- The idea here is that if the coefficient  $a$  is close to zero, then the corresponding feature does not have a significant effect on the classification process.
- On the other hand, since large absolute values of  $a_j$  may significantly influence the classification process, such features should be selected for classification.

## Interaction of Feature Selection with Classification

---

- In the context of the SVM method, which attempts to determine linear planes of separation between the different classes, the vector  $A$  is essentially the normal vector to the corresponding plane of separation between the different classes.
- This intuitively explains the choice of selecting features with large values of  $|aj|$ .
- This class of feature selection methods was quite robust, and performed well even for classifiers such as the Naive Bayes method, which were unrelated to the linear classifiers from which these features were derived.

## Probabilistic and Naive Bayes Classifiers

---

- Probabilistic classifiers are designed to use an implicit mixture model for generation of the underlying documents.
- This mixture model typically assumes that each class is a component of the mixture. Each mixture component is essentially a generative model, which provides the probability of sampling a particular term for that component or class.
- This is why this kind of classifiers are often also called generative classifier.

## Probabilistic and Naive Bayes Classifiers

---

- The naive Bayes classifier is perhaps the simplest and also the most commonly used generative classifiers.
- It models the distribution of the documents in each class using a probabilistic model with independence assumptions about the distributions of different terms.

## Background for Naïve Bayes Classification

---

The United Nations  
Security Council today

Manchester United  
beat Barca to reach

**Can you tell which is about Politics and which is  
about Sports?**

## Naïve Bayes Classification: Posterior Probability

---

## Naïve Bayes: Learn Multinomial Probabilities

The United Nations

Politics

The United States and

Manchester United

Sports

Manchester and Barca

$$P(\text{and}/\text{Politics}) = 1/7$$

$$P(\text{The}/\text{Politics}) = 2/7$$

$$P(\text{and}/\text{Sports}) = 1/5$$

$$P(\text{United}/\text{Politics}) = 2/7$$

$$P(\text{Manchester}/\text{Sports}) = 2/5$$

$$P(\text{Nations}/\text{Politics}) = 1/7$$

$$P(\text{United}/\text{Sports}) = 1/5$$

$$P(\text{States}/\text{Politics}) = 1/7$$

$$P(\text{Barca}/\text{Sports}) = 1/5$$

$$P(\text{Politics}) = 7/12$$

$$P(\text{Sports}) = 5/12$$

## Naïve Bayes: Classify ‘United Nations’

---

# United Nations

$P(\text{Sports} | \text{United Nations})$

$$= P(\text{United} | S) * P(\text{Nations} | \text{Sports}) * P(\text{Sports})$$

$$= (1/5) * (0) * (5/12) = 0$$

$P(\text{Politics} | \text{United Nations})$

$$= P(\text{United} | P) * P(\text{Nations} | \text{Politics}) * P(\text{Politics})$$

$$= (2/7) * (1/7) * (7/12) = 1/(7*6)$$

# United Nations

$P(\text{Politics} | \text{United Nations})$

>

$P(\text{Sports} | \text{United Nations})$

So, the classifier has returned the category **POLITICS**

## Probabilistic and Naive Bayes Classifiers

---

- Two classes of models are commonly used for naive Bayes classification. Both models essentially compute the posterior probability of a class, based on the distribution of the words in the document.
- These models ignore the actual position of the words in the document, and work with the “bag of words” assumption. The major difference between these two models is the assumption in terms of taking (or not taking) word frequencies into account, and the corresponding approach for sampling the probability space:
- **Multivariate Bernoulli Model:** In this model, we use the presence or absence of words in a text document as features to represent a document. Thus, the frequencies of the words are not used for the modeling a document, and the word features in the text are assumed to be binary, with the two values indicating presence or absence of a word in text. Since the features to be modeled are binary, the model for documents in each class is a multivariate Bernoulli model.
- **Multinomial Model:** In this model, we capture the frequencies of terms in a document by representing a document with a bag of words.
- The documents in each class can then be modeled as samples drawn from a multinomial word distribution. As a result, the conditional probability of a document given a class is simply a product of the probability of each observed word in the corresponding class.

## Probabilistic and Naive Bayes Classifiers

---

- No matter how we model the documents in each class (be it a multivariate Bernoulli model or a multinomial model), the component class models (i.e., generative models for documents in each class) can be used in conjunction with the Bayes rule to compute the posterior probability of the class for a given document, and the class with the highest posterior probability can then be assigned to the document.

## Mixture Modeling for Text Classification

---

- We note that the afore-mentioned Bayes methods simply assume that each component of the mixture corresponds to the documents belonging to a class.
- A more general interpretation is one in which the components of the mixture are created by a clustering process, and the class membership probabilities are modeled in terms of this mixture.
- Mixture modeling is typically used for unsupervised (probabilistic) clustering or topic modeling, though the use of clustering can also help in enhancing the effectiveness of probabilistic classifiers.

## Mixture Modeling for Text Classification

---

- These methods are particularly useful in cases where the amount of training data is limited.
- In particular, clustering can help in the following ways:
  - The Bayes method implicitly estimates the word probabilities
  - $P(t_i \in T | C^T = i)$  of a large number of terms in terms of their fractional presence in the corresponding component.
- This is clearly noisy.
- By treating the clusters as separate entities from the classes, we now only need to relate (a much smaller number of) cluster membership probabilities to class probabilities.
- This reduces the number of parameters and greatly improves classification accuracy.

## Mixture Modeling for Text Classification

---

- The use of clustering can help in incorporating unlabeled documents into the training data for classification.
- The premise is that unlabeled data is much more copiously available than labeled data, and when labeled data is sparse, it should be used in order to assist the classification process.
- While such unlabeled documents do not contain class-specific information, they do contain a lot of information about the clustering behavior of the underlying data.
- This can be very useful for more robust modeling, when the amount of training data is low. This general approach is also referred to as co-training.
- The common characteristic of both the methods is that they both use a form of supervised clustering for the classification process.
- While the goal is quite similar (limited training data), the approach used for this purpose is quite different.

## References

---

Text Processing - Cohan Sujay Carlos, AiaiooLabs

<http://aiaioo.com/courses/nlp/Text%20Analytics%20Course%20-%202%20Day.pptx>

[https://link.springer.com/chapter/10.1007/978-1-4614-3223-4\\_6](https://link.springer.com/chapter/10.1007/978-1-4614-3223-4_6)



---

## THANK YOU

---

**Jyothi R.**  
Assistant Professor,  
Department of Computer Science  
[jyothir@pes.edu](mailto:jyothir@pes.edu)





**PES**  
UNIVERSITY  
ONLINE

## DATA ANALYTICS

### **Unit 4: Market Basket Analysis (Apriori Algorithm)**

---

**Jyothi R.**

Department of Computer Science  
and Engineering

## Market Basket Analysis

---

- The market-basket model of data is used to describe a common form of many- many relationship between two kinds of objects.
- On the one hand, we have items, and on the other we have baskets, sometimes called “transactions.”
- Each basket consists of a set of items (an itemset), and usually we assume that the number of items in a basket is small – much smaller than the total number of items.
- The number of baskets is usually assumed to be very large, bigger than what can fit in main memory.
- The data is assumed to be represented in a file consisting of a sequence of baskets.
- In terms of the distributed file system the baskets are the objects of the file, and each basket is of type “set of items.”

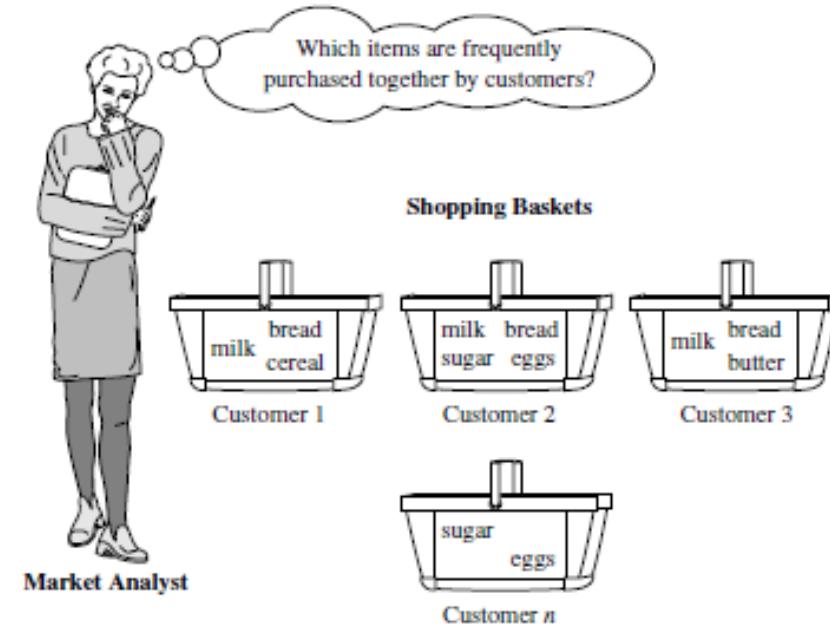
## Market Basket Analysis

---

- Frequent itemset mining leads to the discovery of associations and correlations among items in large transactional or relational data sets.
- With massive amounts of data continuously being collected and stored, many industries are becoming interested in mining such patterns from their databases.
- The discovery of interesting correlation relationships among huge amounts of business transaction records can help in many business decision-making processes such as catalog design, cross-marketing, and customer shopping behavior analysis.

## Market Basket Analysis

- A typical example of frequent itemset mining is market basket analysis.
- This process analyzes customer buying habits by finding associations between the different items that
- customers place in their “shopping baskets” as shown in Figure : **Market Basket Analysis**



## Market Basket Analysis

---

- The discovery of these associations can help retailers develop marketing strategies by gaining insight into which items are frequently purchased together by customers.
- For instance, if customers are buying milk, how likely are they to also buy bread (and what kind of bread) on the same trip to the supermarket?
- This information can lead to increased sales by helping retailers do selective marketing and plan their shelf space.

## Market Basket Analysis

---

- Example: Suppose, as manager of an AllElectronics branch, you would like to learn more about the buying habits of your customers.
- Specifically, you wonder, “Which groups or sets of items are customers likely to purchase on a given trip to the store?”
- To answer your question, market basket analysis may be performed on the retail data of customer transactions at your store.
- You can then use the results to plan marketing or advertising strategies, or in the design of a new catalog.

## Market Basket Analysis

---

- Market basket analysis may help you design different store layouts. In one strategy, items that are frequently purchased together can be placed in proximity to further encourage the combined sale of such items.
- If customers who purchase computers also tend to buy antivirus software at the same time, then placing the hardware display close to the software display may help increase the sales of both items.

## Market Basket Analysis

---

- In an alternative strategy, placing hardware and software at opposite ends of the store may entice customers who purchase such items to pick up other items along the way.
- For instance, after deciding on an expensive computer, a customer may observe security systems for sale while heading toward the software display to purchase antivirus software, and may decide to purchase a home security system as well.
- Market basket analysis can also help retailers plan which items to put on sale at reduced prices.
- If customers tend to purchase computers and printers together, then having a sale on printers may encourage the sale of printers as well as computers.

### Itemset

A collection of one or more items

Example: {Milk, Bread, Diaper}

### k-itemset

An itemset that contains k items

### Support count ( $\sigma$ )

Frequency of occurrence of an itemset

E.g.  $\sigma(\{\text{Milk, Bread, Diaper}\}) = 2$

### Support

Fraction of transactions that contain an itemset

E.g.  $s(\{\text{Milk, Bread, Diaper}\}) = 2/5$

### 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

## Association Rule Mining

- **Association Rule**

- An implication expression of the form  $X \rightarrow Y$ , where X and Y are itemsets
- Example:  
 $\{\text{Milk, Diaper}\} \rightarrow \{\text{Beer}\}$

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**

- Support (s)
  - ◆ Fraction of transactions that contain both X and Y
- Confidence (c)
  - ◆ Measures how often items in Y appear in transactions that contain X

Example:

$$\{\text{Milk, Diaper}\} \Rightarrow \text{Beer}$$

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

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

## Association Rule Mining

---

Two-step approach:

1. Frequent Itemset Generation
  - Generate all itemsets whose support  $\geq \text{minsup}$
2. Rule Generation
  - Generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset

Frequent itemset generation is still computationally expensive

## Apriori Principle

---

**If an itemset is frequent, then all of its subsets must also be frequent**

Apriori principle holds due to the following property of the support measure:

$$\forall X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$

Support of an itemset never exceeds the support of its subsets

This is known as the **anti-monotone** property of support

## Generating frequent itemsets

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Items (1-itemsets)

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

## Generating frequent itemsets (given minsup)

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Minimum Support = 3

If every subset is considered,  
 ${}^6C_1 + {}^6C_2 + {}^6C_3 = 41$   
 With support-based pruning,  
 $6 + 6 + 1 = 13$

Items (1-itemsets)

Itemset	Count
{Bread,Milk}	3
{Bread,Beer}	2
{Bread,Diaper}	3
{Milk,Beer}	2
{Milk,Diaper}	3
{Beer,Diaper}	3

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Triplets (3-itemsets)

Itemset	Count
{Bread,Milk,Diaper}	2

...

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

## Apriori Algorithm for Frequent Itemset Generation

---

Method:

- Let  $k=1$
- Generate frequent itemsets of length 1
- Repeat until no new frequent itemsets are identified
  - Generate length  $(k+1)$  candidate itemsets from length  $k$  frequent itemsets
  - Prune candidate itemsets containing subsets of length  $k$  that are infrequent
  - Count the support of each candidate by scanning the DB
  - Eliminate candidates that are infrequent, leaving only those that are frequent

## Minimum support (minsup)

---

- Note that the itemset support defined is sometimes referred to as *relative support*, whereas the occurrence frequency is called the **absolute support**.
- If the relative support of an itemset  $I$  satisfies a prespecified **minimum support threshold** (i.e., the absolute support of  $I$  satisfies the corresponding **minimum support count threshold**), then  $I$  is a **frequent** itemset.
- The set of frequent  $k$ -itemsets is commonly denoted by  $L_k$

## Applying multiple minimum support

How to apply multiple minimum support?

MS(i): minimum support for item i

e.g.: MS(Milk)=5%, MS(Coke) = 3%,

MS(Broccoli)=0.1%, MS(Salmon)=0.5%

$$\begin{aligned} \text{MS}\{\text{Milk, Broccoli}\} &= \min (\text{MS}(\text{Milk}), \text{MS}(\text{Broccoli})) \\ &= 0.1\% \end{aligned}$$

Challenge: Support is no longer anti-monotone

Suppose: Support(Milk, Coke) = 1.5% and

Support(Milk, Coke, Broccoli) = 0.5%

{Milk,Coke} is infrequent but {Milk,Coke,Broccoli} is frequent

Order the items according to their minimum support (in ascending order)

$$\begin{aligned} \text{e.g.: MS}(\text{Milk}) &= 5\%, \quad \text{MS}(\text{Coke}) = 3\%, \\ \text{MS}(\text{Broccoli}) &= 0.1\%, \quad \text{MS}(\text{Salmon}) = 0.5\% \end{aligned}$$

Ordering: Broccoli, Salmon, Coke, Milk

Need to modify Apriori such that:

$L_1$  : set of frequent items

$F_1$  : set of items whose support is  $\geq \text{MS}(1)$   
where  $\text{MS}(1)$  is  $\min_i (\text{MS}(i))$

$C_2$  : candidate itemsets of size 2 is generated from  $F_1$  instead of  $L_1$

## Multiple minimum support and modified Apriori

Order the items according to their minimum support (in ascending order)

e.g.: MS(Milk)=5%, MS(Coke) = 3%,  
MS(Broccoli)=0.1%, MS(Salmon)=0.5%

Ordering: Broccoli, Salmon, Coke, Milk

Need to modify Apriori such that:

$L_1$  : set of frequent items

$F_1$  : set of items whose support is  $\geq MS(1)$   
where  $MS(1)$  is  $\min_i(MS(i))$

$C_2$  : candidate itemsets of size 2 is generated from  $F_1$   
instead of  $L_1$

Modifications to Apriori: In traditional Apriori, A candidate  $(k+1)$ -itemset is generated by merging two frequent itemsets of size  $k$

The candidate is pruned if it contains any infrequent subsets of size  $k$

Pruning step has to be modified:

Prune only if subset contains the first item

e.g.: Candidate={Broccoli, Coke, Milk} (ordered according to minimum support)

{Broccoli, Coke} and {Broccoli, Milk} are frequent but {Coke, Milk} is infrequent

Candidate is not pruned because {Coke, Milk} does not contain the first item, i.e., Broccoli.

## Rule Generation

How to efficiently generate rules from frequent itemsets?

In general, confidence does not have an anti-monotone property  
 $c(ABC \rightarrow D)$  can be larger or smaller than  $c(AB \rightarrow D)$

But confidence of rules generated from the same itemset has an anti-monotone property e.g.,  $L = \{A, B, C, D\}$ :

$$c(ABC \rightarrow D) \geq c(AB \rightarrow CD) \geq c(A \rightarrow BCD)$$

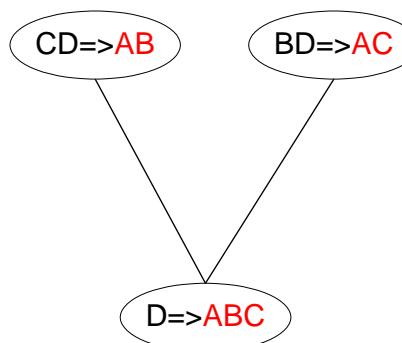
Confidence is anti-monotone w.r.t. number of items on the RHS of the rule

Candidate rule is generated by merging two rules that share the same prefix in the rule consequent

join( $CD \Rightarrow AB$ ,  $BD \Rightarrow AC$ )

would produce the candidate rule  $D \Rightarrow ABC$

Prune rule  $D \Rightarrow ABC$  if its subset  $AD \Rightarrow BC$  does not have high confidence



## Support and Confidence

---

$$\text{support}(A \Rightarrow B) = P(A \cup B)$$

$$\text{confidence}(A \Rightarrow B) = P(B|A).$$

$$\text{confidence}(A \Rightarrow B) = P(B|A) = \frac{\text{support}(A \cup B)}{\text{support}(A)} = \frac{\text{support\_count}(A \cup B)}{\text{support\_count}(A)}.$$

## Computing Confidence

---

- In confidence of rule equation  $A \Rightarrow B$  can be easily derived from the support counts of  $A$  and  $A \cup B$ .
- That is, once the support counts of  $A$ ,  $B$ , and  $A \cup B$  are found, it is straightforward to derive the corresponding association rules  $A \Rightarrow B$  and  $B \Rightarrow A$  and check whether they are strong.
- Thus, the problem of mining association rules can be reduced to that of mining frequent itemsets.

## Evaluation of an association rule

Contingency table for  $X \rightarrow Y$

	Y	$\bar{Y}$	
X	$f_{11}$	$f_{10}$	$f_{1+}$
$\bar{X}$	$f_{01}$	$f_{00}$	$f_{0+}$
	$f_{+1}$	$f_{+0}$	$ T $

- $f_{11}$ : support of X and Y
- $f_{10}$ : support of X and  $\bar{Y}$
- $f_{01}$ : support of  $\bar{X}$  and Y
- $f_{00}$ : support of  $\bar{X}$  and  $\bar{Y}$

$$Lift = \frac{P(Y | X)}{P(Y)}$$

$$Interest = \frac{P(X, Y)}{P(X)P(Y)}$$

$$PS = P(X, Y) - P(X)P(Y)$$

$$\phi-coefficient = \frac{P(X, Y) - P(X)P(Y)}{\sqrt{P(X)[1 - P(X)]P(Y)[1 - P(Y)]}}$$

## Limitation of Confidence

	Coffee	Not Coffee	
Tea	15	5	20
Not Tea	75	5	80
	90	10	100

Association Rule: Tea  $\rightarrow$  Coffee

Confidence=  $P(\text{Coffee}|\text{Tea}) = 0.75$  (75% of those who drink tea also drink coffee)

but  $P(\text{Coffee}) = 0.9$  (90% of the people in our sample drink coffee (most of them do!))

$\Rightarrow$  Although confidence is high, rule is misleading

$\Rightarrow P(\text{Coffee}|\text{NotTea}) = 0.9375$  (more interesting/ meaningful that nearly 94% of those who do not drink tea, drink coffee)

$\Rightarrow$  One is more likely to drink coffee if they do not drink tea (than if they do drink tea)

## Additional References

---

R1 Data Mining: Concepts and Techniques by Han, Kamber and Pei  
(Morgan Kaufman)

Introduction to Data Mining by Tan, Steinbach and Kumar (Pearson – First Edition) Chapters 6 and 7



---

## THANK YOU

---

**Jyothi R.**  
Assistant Professor,  
Department of Computer Science  
[jyothir@pes.edu](mailto:jyothir@pes.edu)





# DATA ANALYTICS

**Unit 4: Rule Generation  
(Apriori Algorithm) +  
Evaluation of Recommender Systems**

---

**Gowri Srinivasa**

Department of Computer Science and Engineering

## Rule Generation

How to efficiently generate rules from frequent itemsets?

In general, confidence does not have an anti-monotone property  
 $c(ABC \rightarrow D)$  can be larger or smaller than  $c(AB \rightarrow D)$

But confidence of rules generated from the same itemset has an anti-monotone property e.g.,  $L = \{A, B, C, D\}$ :

$$c(ABC \rightarrow D) \geq c(AB \rightarrow CD) \geq c(A \rightarrow BCD)$$

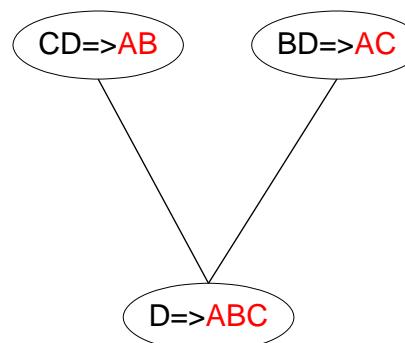
Confidence is anti-monotone w.r.t. number of items on the RHS of the rule

Candidate rule is generated by merging two rules that share the same prefix in the rule consequent

join( $CD \Rightarrow AB$ ,  $BD \Rightarrow AC$ )

would produce the candidate rule  $D \Rightarrow ABC$

Prune rule  $D \Rightarrow ABC$  if its subset  $AD \Rightarrow BC$  does not have high confidence



## Rule Generation

---

Given a frequent itemset  $L$ , find all non-empty subsets  $f \subset L$  such that  $f \rightarrow L - f$  satisfies the minimum confidence requirement

If  $\{A, B, C, D\}$  is a frequent itemset, candidate rules:

$$\begin{array}{llll} ABC \rightarrow D, & ABD \rightarrow C, & ACD \rightarrow B, & BCD \rightarrow A, \\ A \rightarrow BCD, & B \rightarrow ACD, & C \rightarrow ABD, & D \rightarrow ABC \\ AB \rightarrow CD, & AC \rightarrow BD, & AD \rightarrow BC, & BC \rightarrow AD, \\ BD \rightarrow AC, & CD \rightarrow AB \end{array}$$

If  $|L| = k$ , then there are  $2^k - 2$  candidate association rules  
(ignoring  $L \rightarrow \emptyset$  and  $\emptyset \rightarrow L$ )

## Support and Confidence

---

$$\text{support}(A \Rightarrow B) = P(A \cup B)$$

$$\text{confidence}(A \Rightarrow B) = P(B|A).$$

$$\text{confidence}(A \Rightarrow B) = P(B|A) = \frac{\text{support}(A \cup B)}{\text{support}(A)} = \frac{\text{support\_count}(A \cup B)}{\text{support\_count}(A)}.$$

## Computing Confidence

---

- In confidence of rule equation  $A \Rightarrow B$  can be easily derived from the support counts of  $A$  and  $A \cup B$ .
- That is, once the support counts of  $A$ ,  $B$ , and  $A \cup B$  are found, it is straightforward to derive the corresponding association rules  $A \Rightarrow B$  and  $B \Rightarrow A$  and check whether they are strong.
- Thus, the problem of mining association rules can be reduced to that of mining frequent itemsets.

## Evaluation of an association rule

Contingency table for  $X \rightarrow Y$

	Y	$\bar{Y}$	
X	$f_{11}$	$f_{10}$	$f_{1+}$
$\bar{X}$	$f_{01}$	$f_{00}$	$f_{0+}$
	$f_{+1}$	$f_{+0}$	$ T $

- $f_{11}$ : support of X and Y
- $f_{10}$ : support of X and  $\bar{Y}$
- $f_{01}$ : support of  $\bar{X}$  and Y
- $f_{00}$ : support of  $\bar{X}$  and  $\bar{Y}$

$$Lift = \frac{P(Y | X)}{P(Y)}$$

$$Interest = \frac{P(X, Y)}{P(X)P(Y)}$$

$$PS = P(X, Y) - P(X)P(Y)$$

$$\phi-coefficient = \frac{P(X, Y) - P(X)P(Y)}{\sqrt{P(X)[1 - P(X)]P(Y)[1 - P(Y)]}}$$

## Limitation of Confidence

	Coffee	Not Coffee	
Tea	15	5	20
Not Tea	75	5	80
	90	10	100

Association Rule: Tea  $\rightarrow$  Coffee

Confidence=  $P(\text{Coffee}|\text{Tea}) = 0.75$  (75% of those who drink tea also drink coffee)

but  $P(\text{Coffee}) = 0.9$  (90% of the people in our sample drink coffee (most of them do!))

$\Rightarrow$  Although confidence is high, rule is misleading

$\Rightarrow P(\text{Coffee}|\text{NotTea}) = 0.9375$  (more interesting/ meaningful that nearly 94% of those who do not drink tea, drink coffee)

$\Rightarrow$  One is more likely to drink coffee if they do not drink tea (than if they do drink tea)

## Continuous and Categorical Attributes

How to apply association analysis formulation to non-asymmetric binary variables?

Session Id	Country	Session Length (sec)	Number of Web Pages viewed	Gender	Browser Type	Buy
1	USA	982	8	Male	IE	No
2	China	811	10	Female	Netscape	No
3	USA	2125	45	Female	Mozilla	Yes
4	Germany	596	4	Male	IE	Yes
5	Australia	123	9	Male	Mozilla	No
...	...	...	...	...	...	...

Example of Association Rule:

$$\{\text{Number of Pages } \in [5,10] \wedge (\text{Browser}=\text{Mozilla})\} \rightarrow \{\text{Buy} = \text{No}\}$$

Transform categorical attribute into asymmetric binary variables

Introduce a new “item” for each distinct attribute-value pair

Example: replace Browser Type attribute with

Browser Type = Internet Explorer

Browser Type = Mozilla

Browser Type = Mozilla

### Potential Issues

**What if an attribute has many possible values?**

Example: attribute country has more than 200 possible values

Many of the attribute values may have very low support

**Potential solution:** Aggregate the low-support attribute values

**What if distribution of attribute values is highly skewed?**

Example: 95% of the visitors have Buy = No

Most of the items will be associated with (Buy=No) item

**Potential solution:** drop the highly frequent items

Multiple minimum support also comes in handy in both cases

Different kinds of rules:

$\text{Age} \in [21, 35] \wedge \text{Salary} \in [70k, 120k] \rightarrow \text{Buy}$

$\text{Salary} \in [70k, 120k] \wedge \text{Buy} \rightarrow \text{Age: } \mu=28, \sigma=4$

Different methods:

Discretization-based

Statistics-based (mean, median, standard deviation, etc.)

Non-discretization based minApriori (concept hierarchy)

Discretization-based

Unsupervised:

Equal-width binning

Equal-depth binning

Clustering

Supervised:

Attribute values,  $v$

Class	$v_1$	$v_2$	$v_3$	$v_4$	$v_5$	$v_6$	$v_7$	$v_8$	$v_9$
Anomalous	0	0	20	10	20	0	0	0	0
Normal	150	100	0	0	0	100	100	150	100

bin<sub>1</sub>
bin<sub>2</sub>
bin<sub>3</sub>

## Evaluation – objective measures

#	Measure	Formula
1	phi-coefficient	$\frac{P(A,B) - P(A)P(B)}{\sqrt{P(A)P(B)(1-P(A))(1-P(B))}}$
2	Goodman-Kruskal's ( $\lambda$ )	$\frac{\sum_j \max_k P(A_j, B_k) + \sum_k \max_j P(A_j, B_k) - \max_j P(A_j) - \max_k P(B_k)}{2 - \max_j P(A_j) - \max_k P(B_k)}$
3	Odds ratio ( $\alpha$ )	$\frac{P(A,B)P(\bar{A},\bar{B})}{P(\bar{A},B)P(\bar{A},\bar{B})}$
4	Yule's $Q$	$\frac{P(A,B)P(\bar{A}\bar{B}) - P(A,\bar{B})P(\bar{A},B)}{P(A,B)P(\bar{A}\bar{B}) + P(A,\bar{B})P(\bar{A},B)} = \frac{\alpha-1}{\alpha+1}$
5	Yule's $Y$	$\frac{\sqrt{P(A,B)P(\bar{A}\bar{B})} - \sqrt{P(A,\bar{B})P(\bar{A},B)}}{\sqrt{P(A,B)P(\bar{A}\bar{B})} + \sqrt{P(A,\bar{B})P(\bar{A},B)}} = \frac{\sqrt{\alpha}-1}{\sqrt{\alpha}+1}$
6	Kappa ( $\kappa$ )	$\frac{P(A,B) + P(\bar{A},\bar{B}) - P(A)P(B) - P(\bar{A})P(\bar{B})}{1 - P(A)P(B) - P(\bar{A})P(\bar{B})}$
7	Mutual Information ( $M$ )	$\sum_i \sum_j P(A_i, B_j) \log \frac{P(A_i, B_j)}{P(A_i)P(B_j)}$
8	J-Measure ( $J$ )	$\max \left( P(A,B) \log \left( \frac{P(B A)}{P(B)} \right) + P(\bar{A}\bar{B}) \log \left( \frac{P(\bar{B} A)}{P(\bar{B})} \right), P(A,B) \log \left( \frac{P(\bar{A} B)}{P(\bar{A})} \right) + P(\bar{A}\bar{B}) \log \left( \frac{P(\bar{A} B)}{P(\bar{B})} \right) \right)$
9	Gini index ( $G$ )	$\max \left( P(A)[P(B A)^2 + P(\bar{B} A)^2] + P(\bar{A})[P(B \bar{A})^2 + P(\bar{B} \bar{A})^2] - P(B)^2 - P(\bar{B})^2, P(B)[P(A B)^2 + P(\bar{A} B)^2] + P(\bar{B})[P(A \bar{B})^2 + P(\bar{A} \bar{B})^2] - P(A)^2 - P(\bar{A})^2 \right)$
10	Support ( $s$ )	$P(A,B)$
11	Confidence ( $c$ )	$\max(P(B A), P(A B))$
12	Laplace ( $L$ )	$\max \left( \frac{NP(A,B)+1}{NP(A)+2}, \frac{NP(A,B)+1}{NP(B)+2} \right)$
13	Conviction ( $V$ )	$\max \left( \frac{P(A)P(\bar{B})}{P(AB)}, \frac{P(B)P(\bar{A})}{P(B\bar{A})} \right)$
14	Interest ( $I$ )	$\frac{P(A,B)}{\frac{P(A)P(B)}{P(A,B)}}$
15	cosine ( $IS$ )	$\frac{P(A,B)}{\sqrt{P(A)P(B)}}$
16	Piatetsky-Shapiro's ( $PS$ )	$P(A,B) - P(A)P(B)$
17	Certainty factor ( $F$ )	$\max \left( \frac{P(B A)-P(B)}{1-P(B)}, \frac{P(A B)-P(A)}{1-P(A)} \right)$
18	Added Value ( $AV$ )	$\max(P(B A) - P(B), P(A B) - P(A))$
19	Collective strength ( $S$ )	$\frac{P(A,B)+P(\bar{A}\bar{B})}{P(A)P(B)+P(\bar{A})P(\bar{B})} \times \frac{1-P(A)P(B)-P(\bar{A})P(\bar{B})}{1-P(A,B)-P(\bar{A}\bar{B})}$
20	Jaccard ( $\zeta$ )	$\frac{P(A,B)}{P(A)+P(B)-P(A,B)}$
21	Klosgen ( $K$ )	$\sqrt{P(A,B)} \max(P(B A) - P(B), P(A B) - P(A))$

It is sufficient if we understand the idea behind the measures and are able to use some of these, such as, support, confidence, lift (or interest), phi-coefficient to evaluate a confidence rule or test for independence of (or correlation) between itemsets

## Evaluation – subjective measures

---

Objective measure:

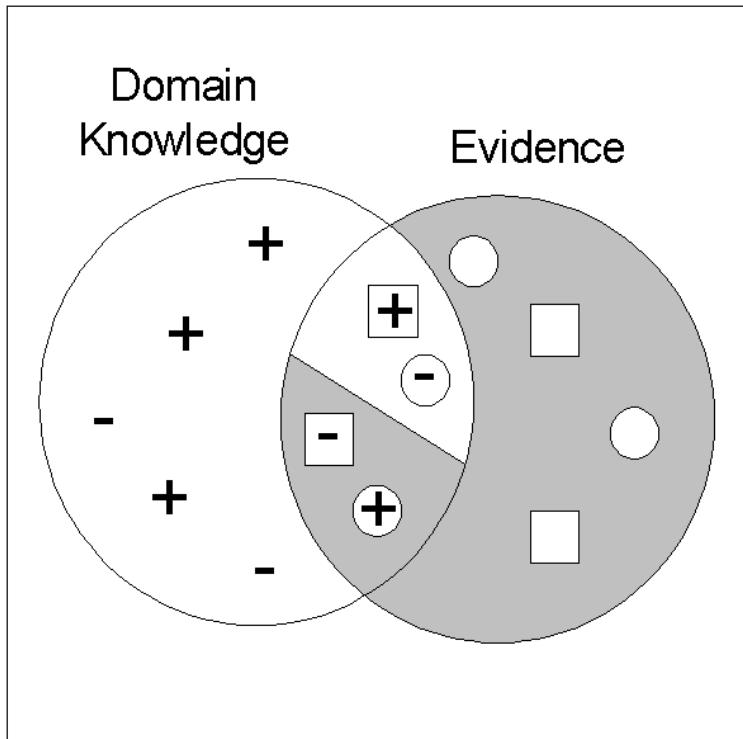
Rank patterns based on statistics computed from data  
e.g., 21 measures of association (support, confidence,  
Laplace, Gini, mutual information, Jaccard, etc).

Subjective measure:

Rank patterns according to user's interpretation  
A pattern is subjectively interesting if it contradicts the  
expectation of a user (Silberschatz & Tuzhilin)  
A pattern is subjectively interesting if it is actionable  
(Silberschatz & Tuzhilin)

## Interestingness via unexpectedness

Need to model expectation of users (domain knowledge)



- ⊕ Pattern expected to be frequent
- ⊖ Pattern expected to be infrequent
- Pattern found to be frequent
- Pattern found to be infrequent
- ⊕ ⊖ Expected Patterns
- ⊖ ⊕ Unexpected Patterns

Need to combine expectation of users with evidence from data  
(i.e., extracted patterns)

## Evaluation of Recommender Systems

---

### Paradigms

1. **User Studies:** test subjects are actively recruited, and they are asked to interact with the recommender system to perform specific tasks (likes and dislikes inferred). Feedback can be collected from the user before and after the interaction.  
Results from user evaluations cannot be fully trusted.
2. **Online Evaluation:** A/B Testing (coming up in Unit 5)
3. **Offline Evaluation:** With historical datasets

## Evaluation of Recommender Systems

---

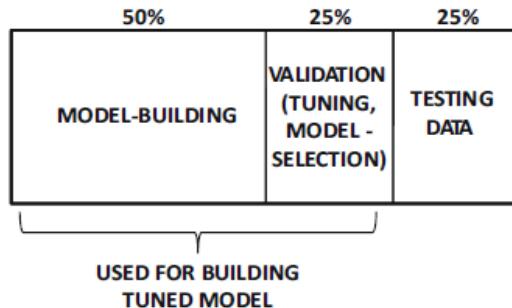
### Measures

1. **Accuracy:** MSE, RMSE, etc.
2. **Coverage:** The fraction of users for which at least  $k$  ratings may be predicted (user-space coverage); the fraction of items for which the ratings of at least  $k$  users can be predicted (item-space coverage); the fraction of items that are recommended to at least one user (catalog coverage)
3. **Confidence and Trust:** confidence measures the system's faith in the recommendation, trust measures the user's faith in the evaluation
4. **Novelty:** Likelihood of a system to recommend an item the user was not aware of (A *differential* accuracy between future and past predictions can be used to quantify this.)
5. **Serendipity** – ‘lucky discovery’ or surprise factor (Ethiopian restaurant recommended to someone who likes Indian food is serendipitous; all that is novel is not serendipitous!)
6. **Diversity:** If three movies are recommended, they must not all be of the same genre; the changes a user will select one of them will then be higher (measured using content-centric similarity between pairs of items)
7. **Robustness and stability:** not significantly affected in the presence of attacks such as fake ratings or when the patterns in the data evolve significantly over time
8. **Scalability:** perform effectively and efficiently in the presence of large amounts of data (quantified based on training time, prediction time and memory requirements)

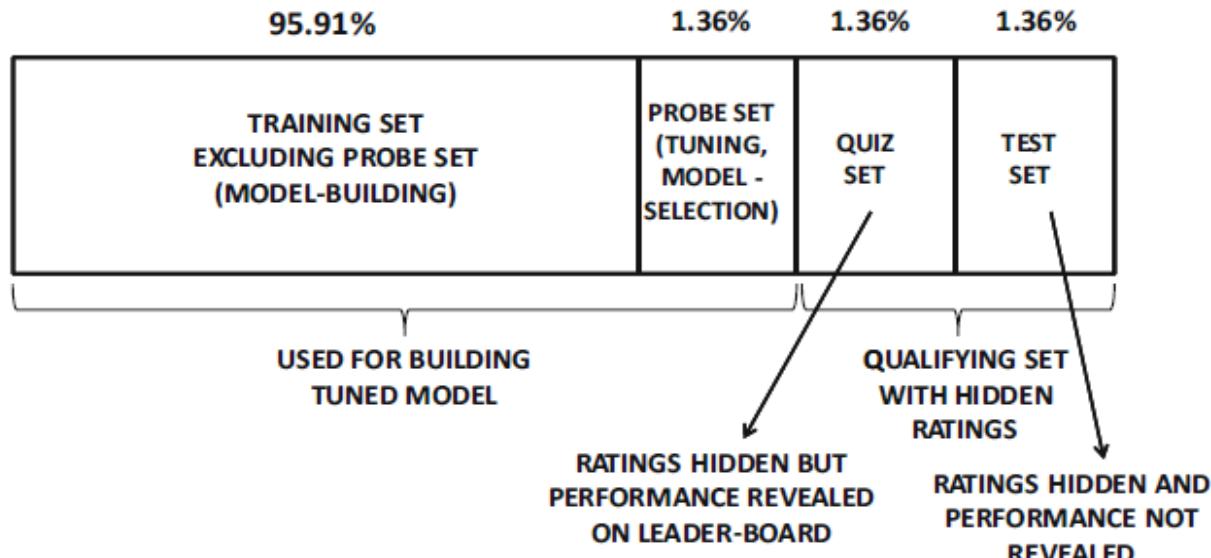
## Case Study: The Netflix Prize - 1

The competition began on October 2, 2006: Prize awarded to the best collaborative filtering algorithm to predict user ratings based on training data <user, movie, date of grade, grade>

Usual partitioning of data in problems



Partitioning of Data for the Netflix Prize Challenge to penalize overfitting



## Case Study: The Netflix Prize - 2

---

On September 21, 2009 “[BellKor's Pragmatic Chaos](#)” won the \$1 million for their algorithm

This team used an ensemble of models; specifically, they used Gradient Boosted Decision Trees to combine 500 models! (Previous solutions had used linear regression for the combination).

Briefly, gradient boosted decision trees work by sequentially fitting a series of decision trees to the data; each tree is asked to predict the error made by the previous trees, and is often trained on slightly perturbed versions of the data.

Since GBDTs have a built-in ability to apply different methods to different slices of the data, we can add in some predictors:

- Number of movies each user rated
- Number of users that rated each movie
- Factor vectors of users and movies
- Hidden units of a restricted Boltzmann Machine

that help the trees make useful clusterings (For example, one thing that Bell and Koren found (when using an earlier ensemble method) was that RBMs are more useful when the movie or the user has a low number of ratings, and that matrix factorization methods are more useful when the movie or user has a high number of ratings.)

For the interested student: read more about this [here](#) (a summary) and [here](#) (links to the top papers). Crowd sourcing problem solving led to [founding Kaggle](#) and other similar organizations!

## A few other application domains

---

1. *Query recommendation*: How can web logs can be used to recommend queries to users?

Typically *session-specific* (i.e., dependent on the history of user behavior in a short session) and do not use *long-term* user behavior. This is because queries are often issued in scenarios in which user re-identification mechanisms are not available over multiple sessions.

2. *Portal content and news personalization*: Many online portals have strong user identification mechanisms by which returning users can be identified. In such cases, the content served to the user can be personalized. This approach is also used by news personalization engines, such as Google News, in which Gmail accounts are used for user identification. News personalization is usually based on implicit feedback containing user behavior (clicks), rather than explicit ratings.

3. *Computational advertising*: A form of recommendation, because it is desirable for companies to be able to identify advertisements for users based on a relevant context (Web page or search query). Therefore, many ideas from recommendation systems are directly used in the area of computational advertising.

4. *Reciprocal recommender systems*: In these cases, both the users and items have preferences (and not just the users). For example, in an online dating application, both parties have preferences, and a successful recommendation can be created only by satisfying both parties.

## Additional References

---

R1 Data Mining: Concepts and Techniques by Han, Kamber and Pei  
(Morgan Kaufman)

Introduction to Data Mining by Tan, Steinbach and Kumar (Pearson – First Edition) Chapters 6 and 7

Recommender Systems – The Textbook by Charu C. Agarwal (Chapter 7)



---

# THANK YOU

---

**Gowri Srinivasa**  
Professor,  
Department of Computer Science  
[gsrinivasa@pes.edu](mailto:gsrinivasa@pes.edu)

