DATA ANALYTICS

Unit 4: Recommendation Systems

Jyothi R.

Department of Computer Science and Engineering

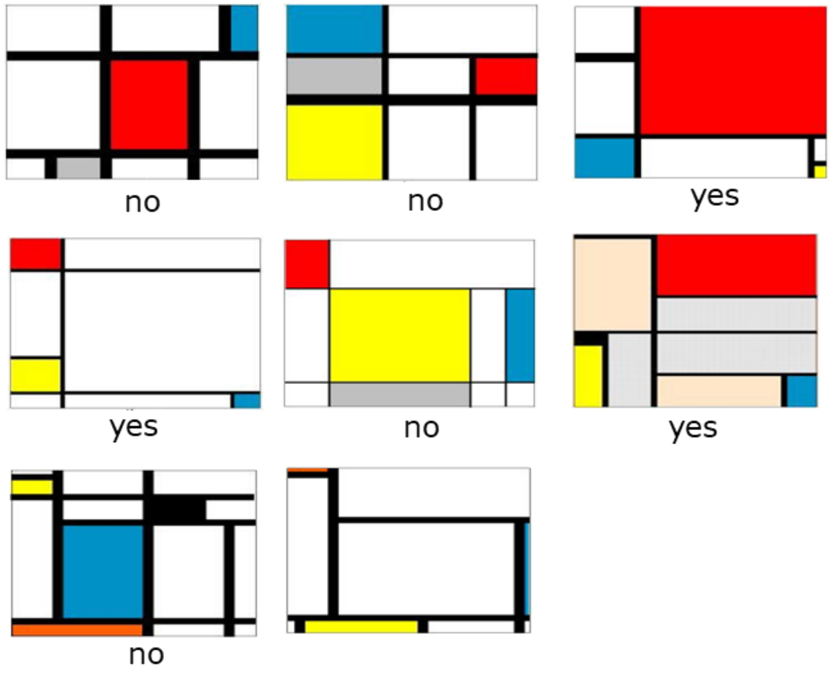
Knowledge-Based Recommender Systems

	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

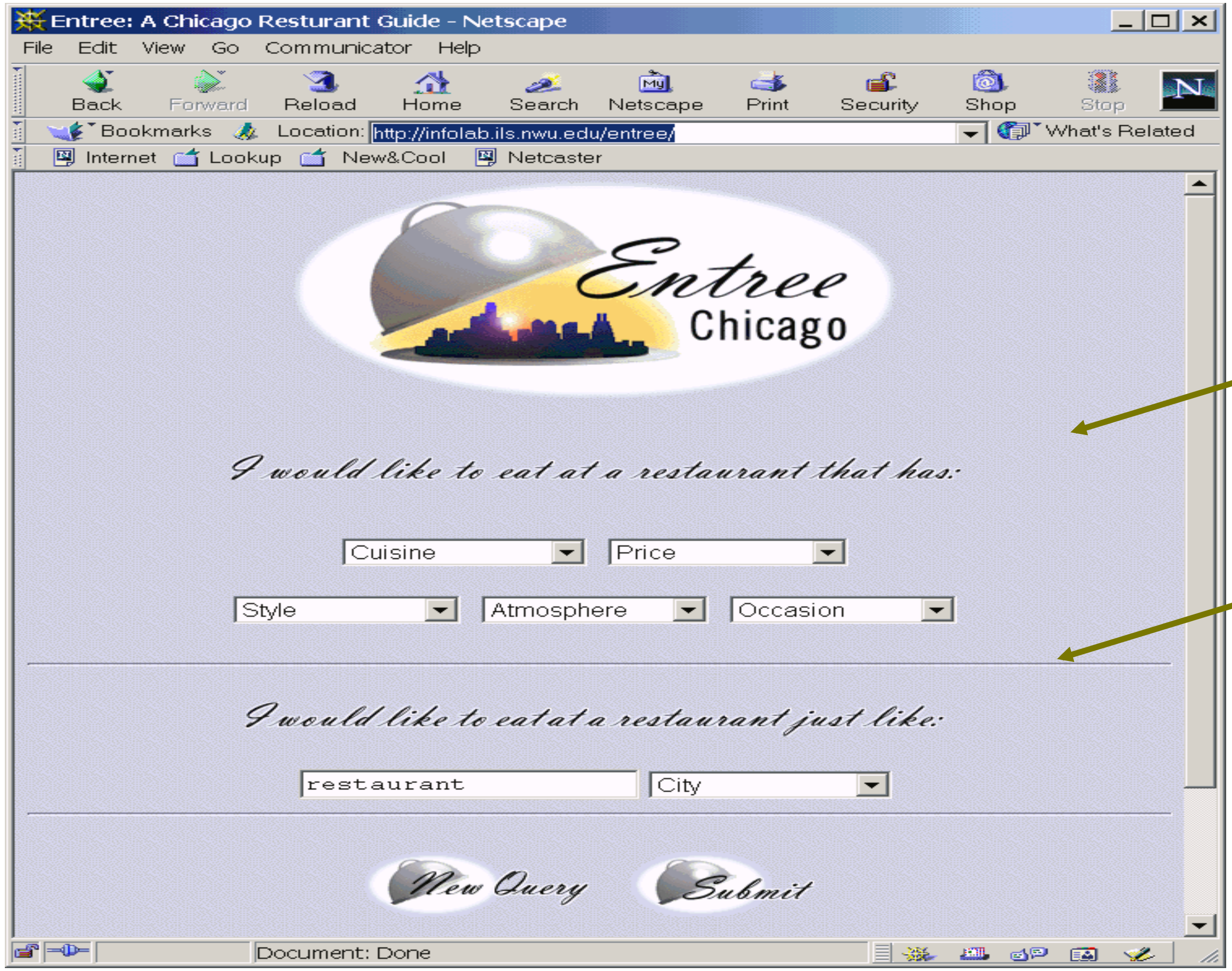
What is the difference between this feature value normalization and vector Normalization in IR?



x' = (x −avg(X))/4*stdev(X)), where x 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

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 ….

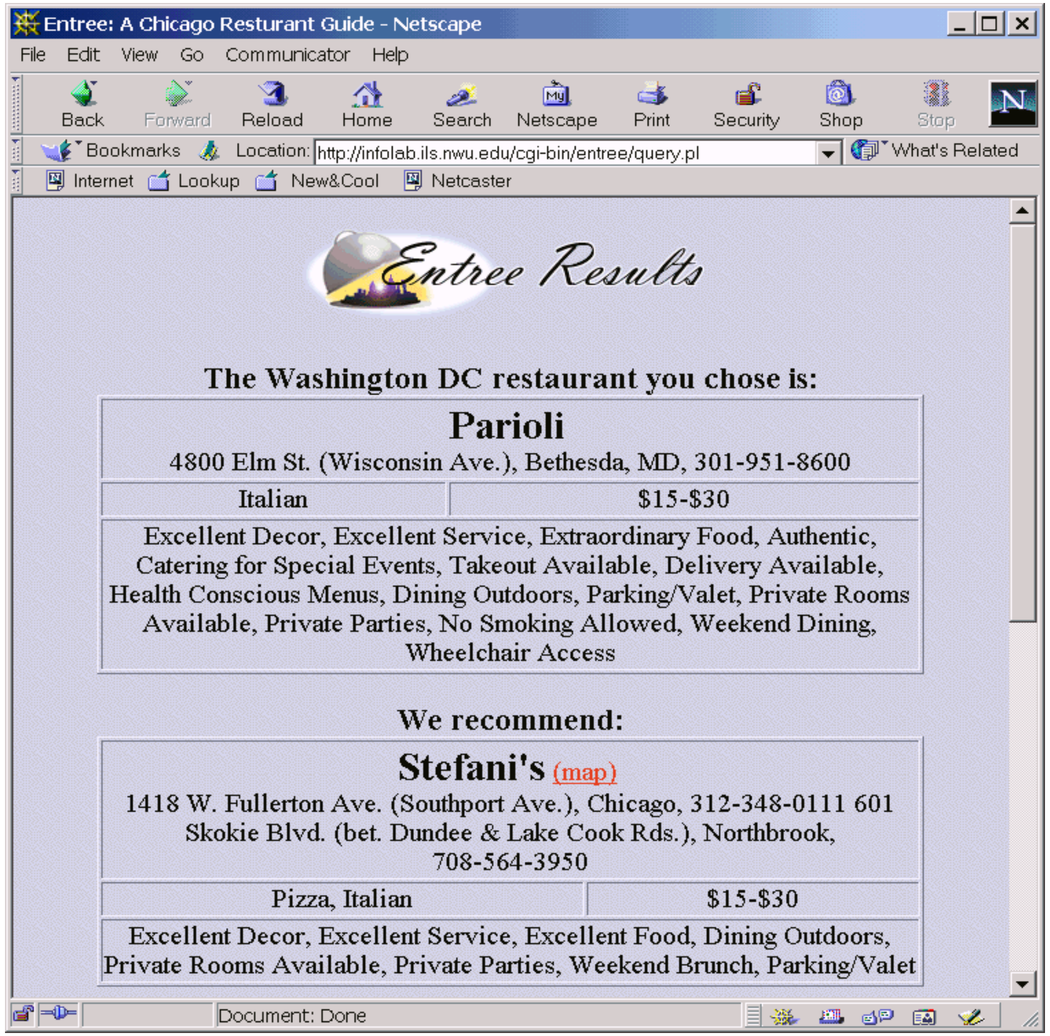
Knowledge-Based Recommender Systems

Partial Match

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

Knowledge-Based Recommender Systems

Nearest Neighbor



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 the most important goal and then iteratively with respect to the remaining goals (multi-level sort)
- Note: it does not works as a maximization of a Utility-Similarity defined as the sum of local utilities.

Knowledge-Based Recommender Systems

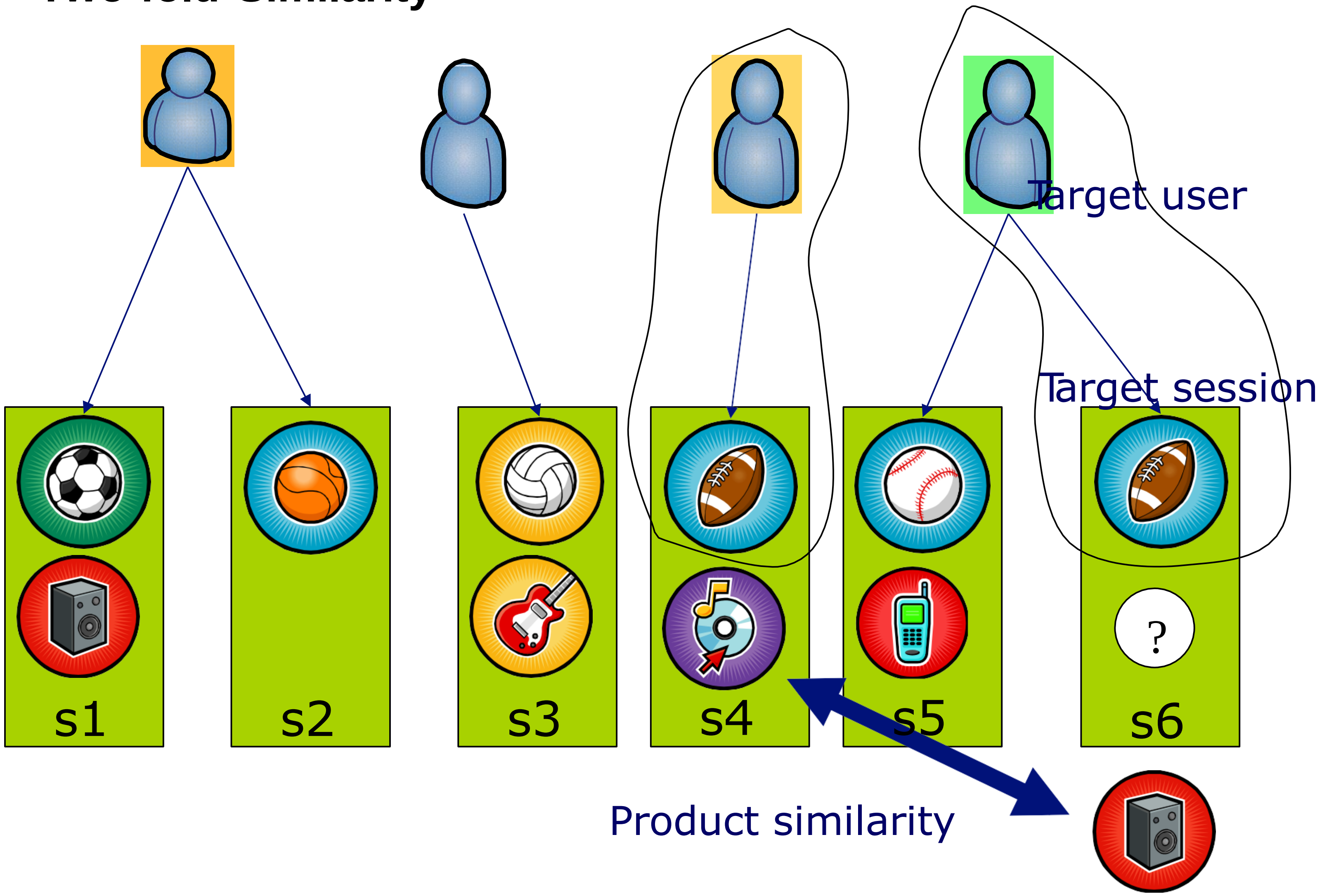
Example

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

- If the user query q is: **price=9** AND **cusine=B** AND **Atm=B**
- And the weights (importance) of the features is: 0.5 price, 0.3 Cusine, 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):
- Sim(q,Woodys) = 0.5 * (1 - 1/30) + 0.3 * 0 + 0.2 * 0 = **0.48**
- Sim(q, 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



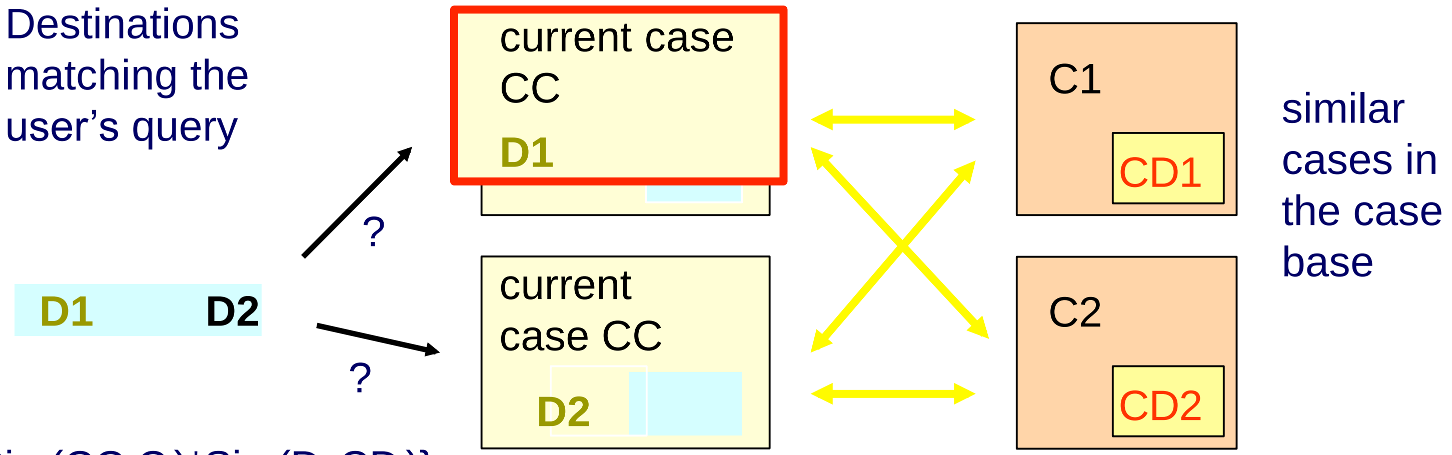
Rank using Two-Fold Similarity

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, …, 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, …, p_{10}) from cases (c_1, …, c_{10}) of the same type as those in R
3. For each product r in R compute the 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 Score(r).

Knowledge-Based Recommender Systems

Example: Scoring Two Destinations



$\text{Score}(D_i) = \text{Max}_j \{\text{Sim}(CC,C_j)*\text{Sim}(D_i,CD_j)\}$

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

Score(D1)=Max{0.2*0.4,0.6*0.7}=0.42

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

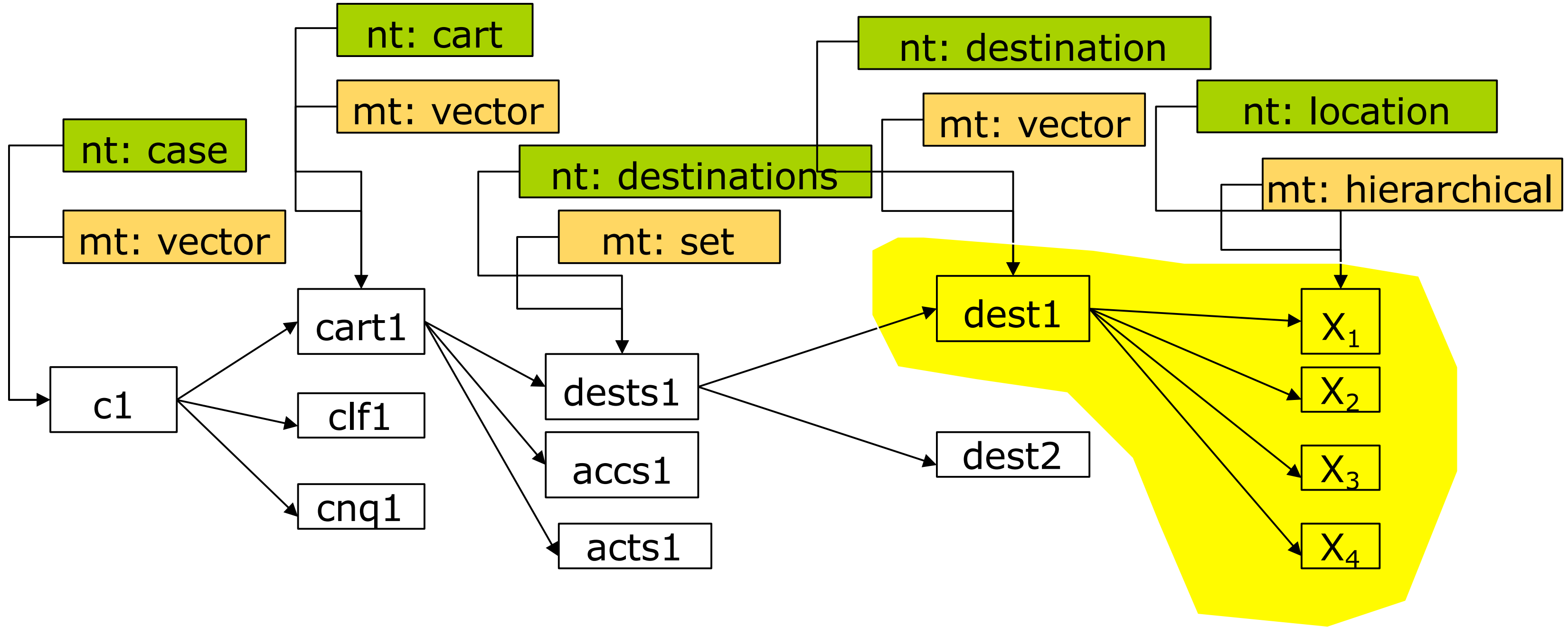
Knowledge-Based Recommender Systems

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

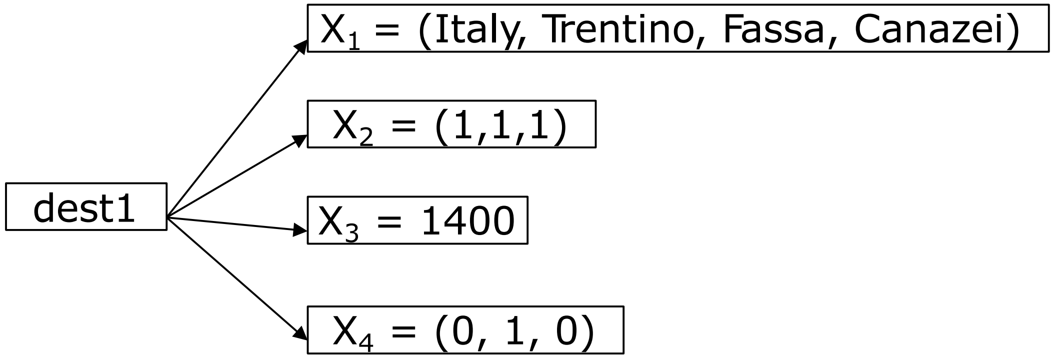
- **Metric type:** node content structure - how to measure the node similarity with another node in a second case



Knowledge-Based Recommender Systems

Item Representation

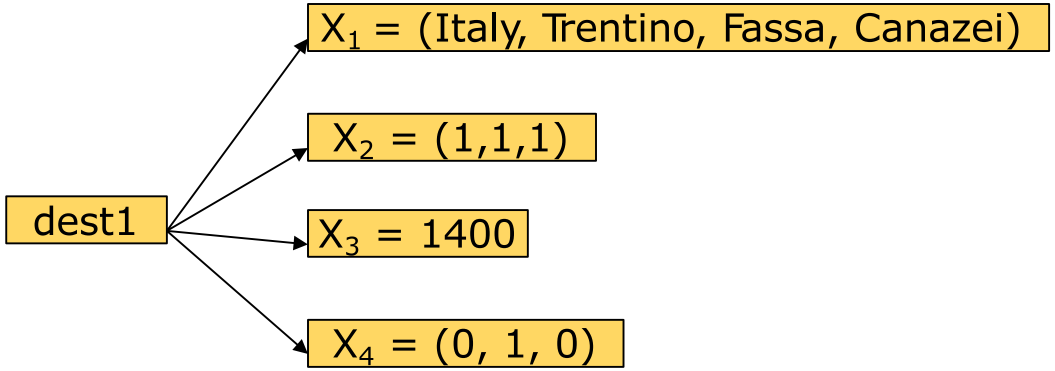
	Node Type	Metric Type	Example: Canazei
X_1	LOCATION	Set of hierarchical related symbols	Country=ITALY, Region=TRENTINO, TouristArea=FASSA, Village=CANAZEI
X_2	INTERESTS	Array of Booleans	Hiking=1, Trekking=1, Biking=1
X_3	ALTITUDE	Numeric	1400
X_4	LOCTYPE	Array of Booleans	Urban=0, Mountain=1, Rivereside=0



Knowledge-Based Recommender Systems

Item Query Language

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



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

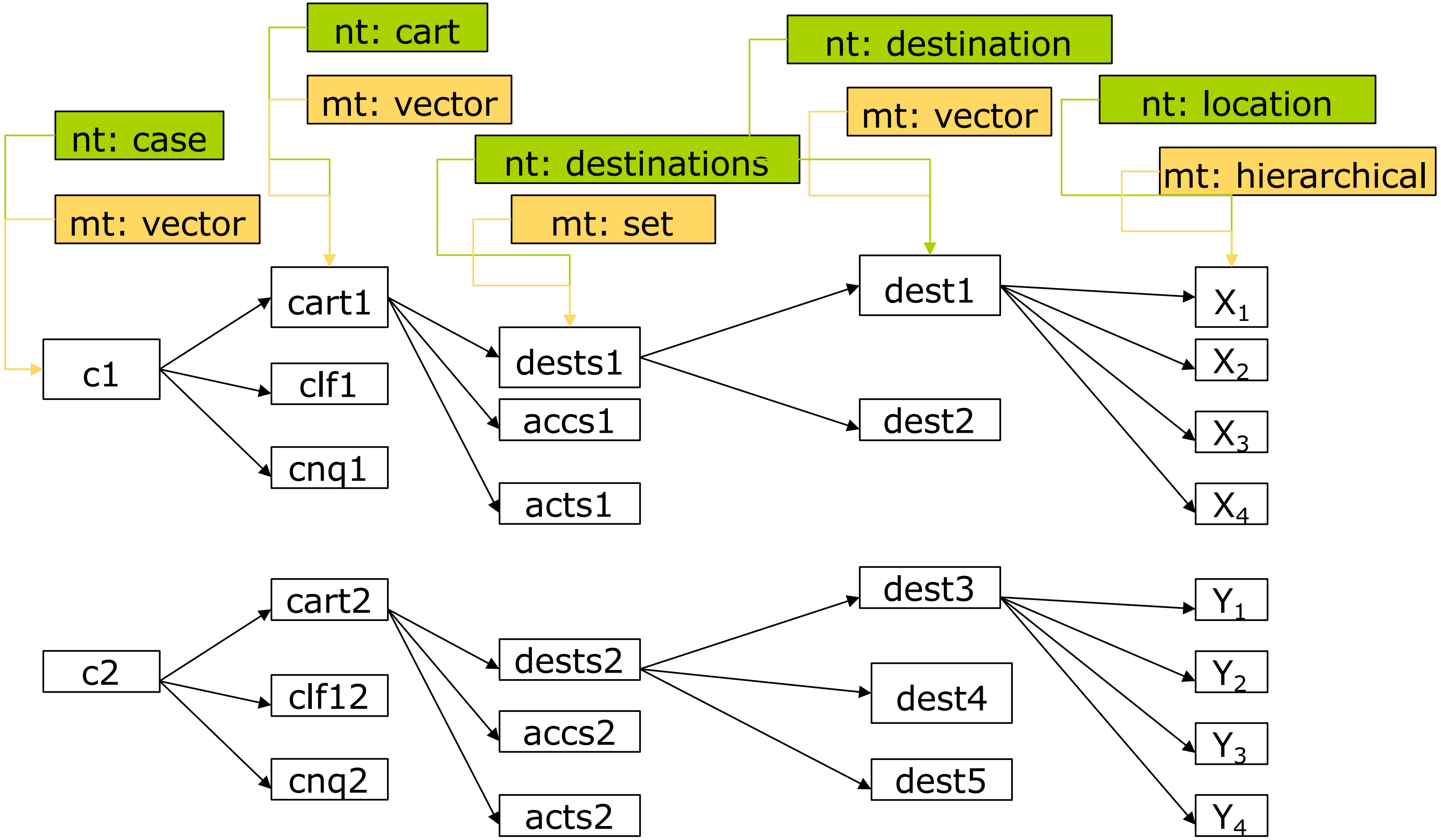
- A query is a conjunction of constraints over features:

$q = c_1 \wedge c_2 \wedge \ldots \wedge c_m$ where $m \leq n$ and

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

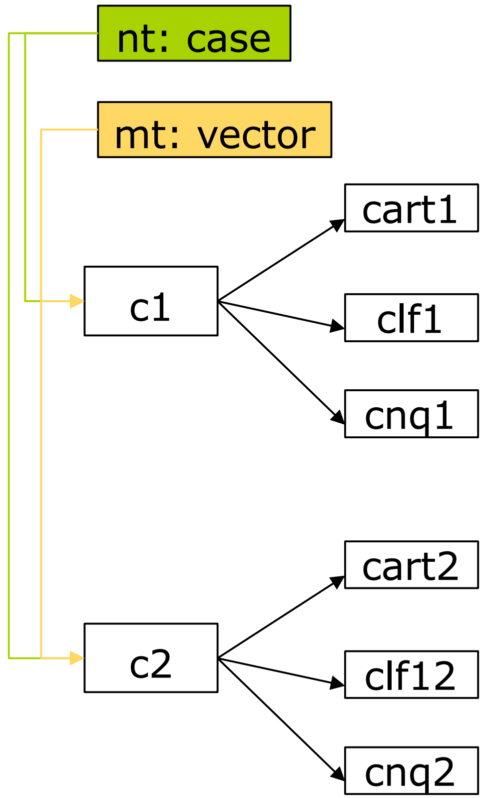
Knowledge-Based Recommender Systems

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 is 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.

Image Courtesy

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

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=h9gpufJFF-0

THANK YOU

Jyothi R.
Assistant Professor,
Department of Computer Science
jyothir@pes.edu

DATA ANALYTICS

Unit 4: Decision trees-CART

Jyothi R.
Assistant Professor
Department of Computer Science and Engineering

Decision Tree

- 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.

DATA ANALYTICS

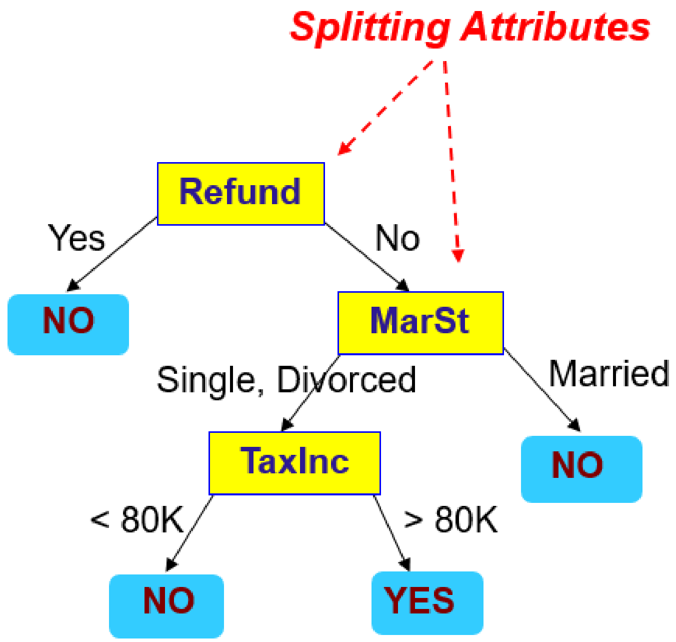
Decision Tree

Example1 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

Training Data



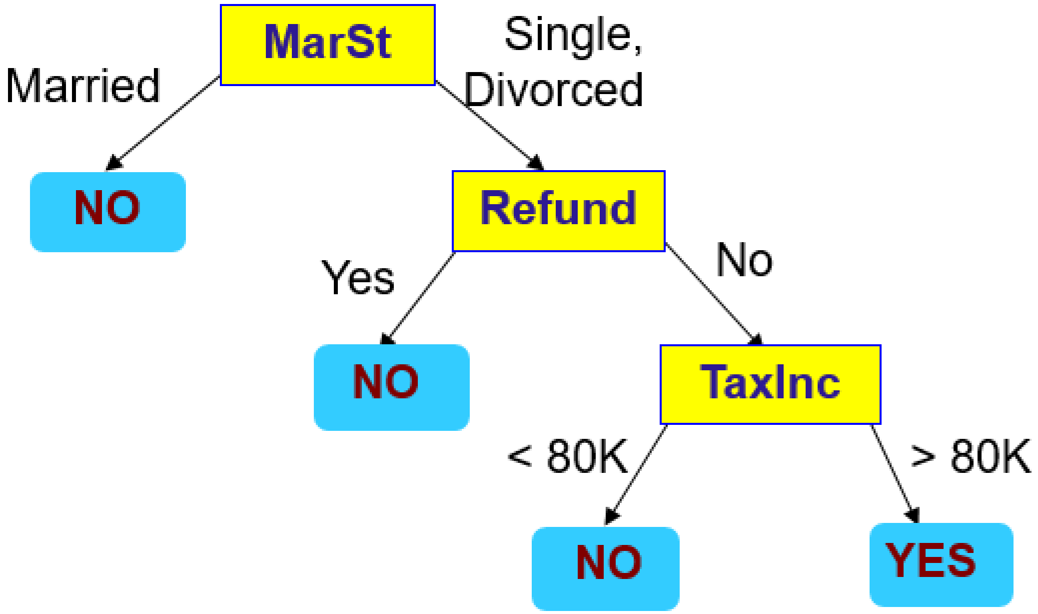
Model: Decision Tree

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

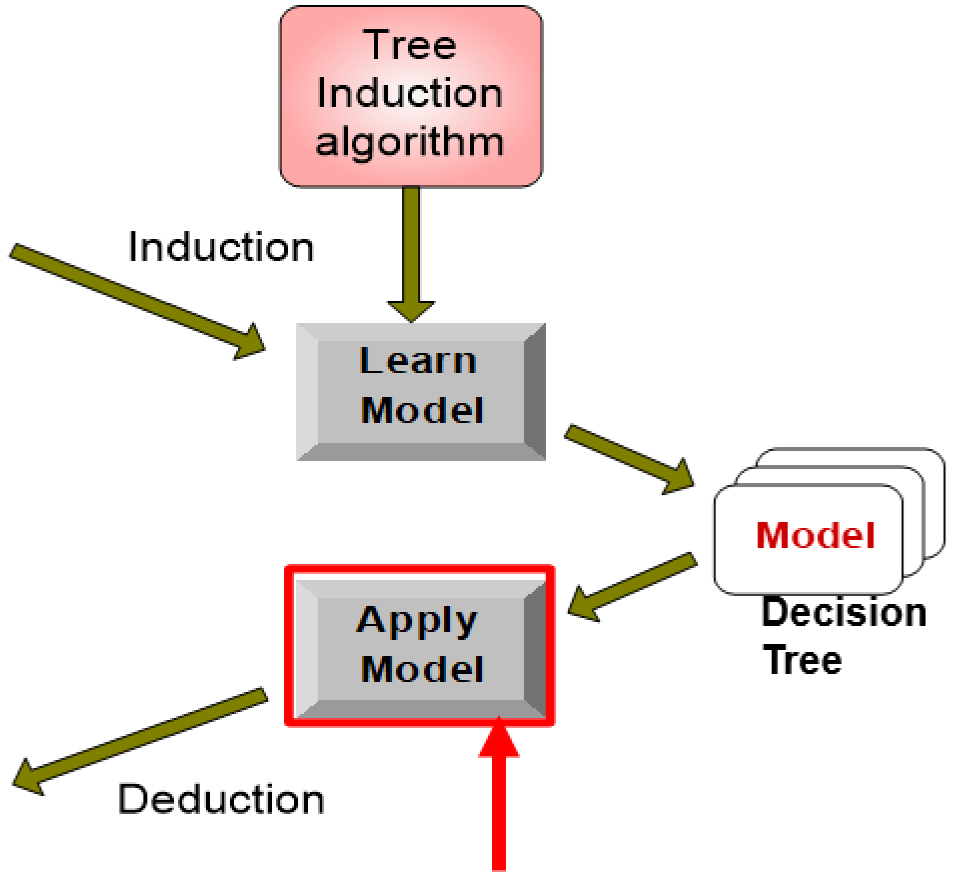


There could be more than one tree that fits the same data!

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

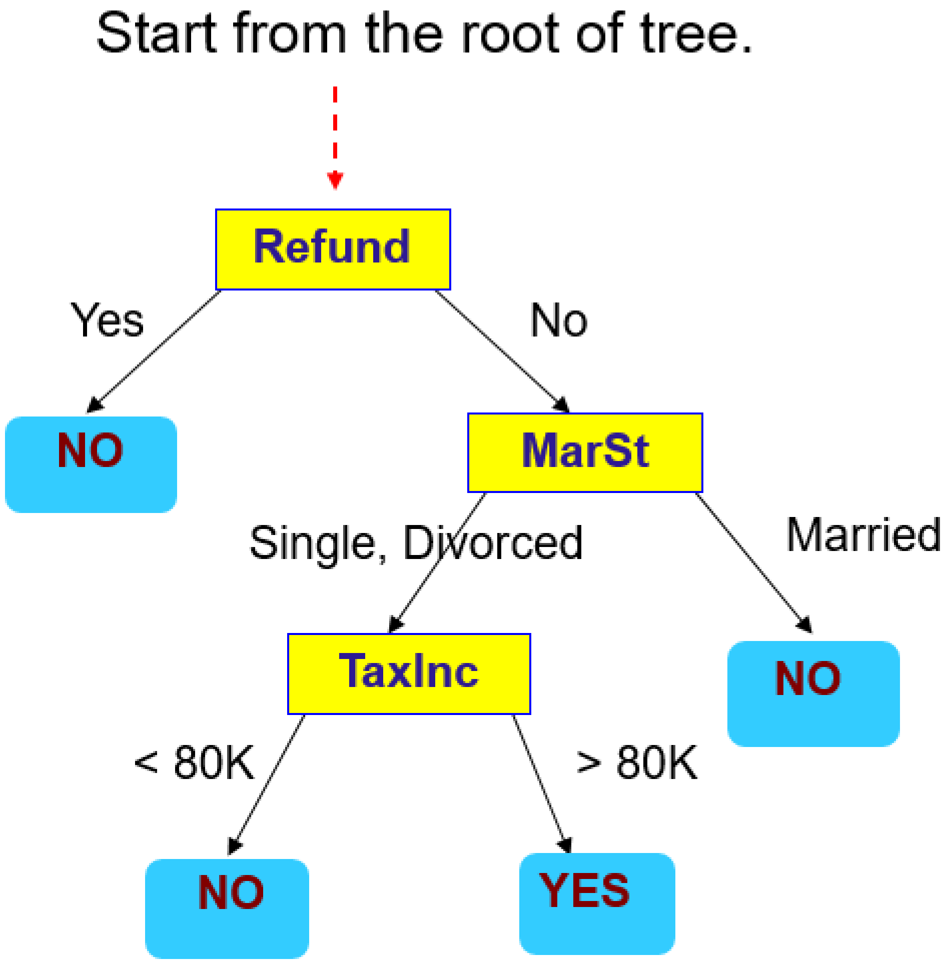
Training Set



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

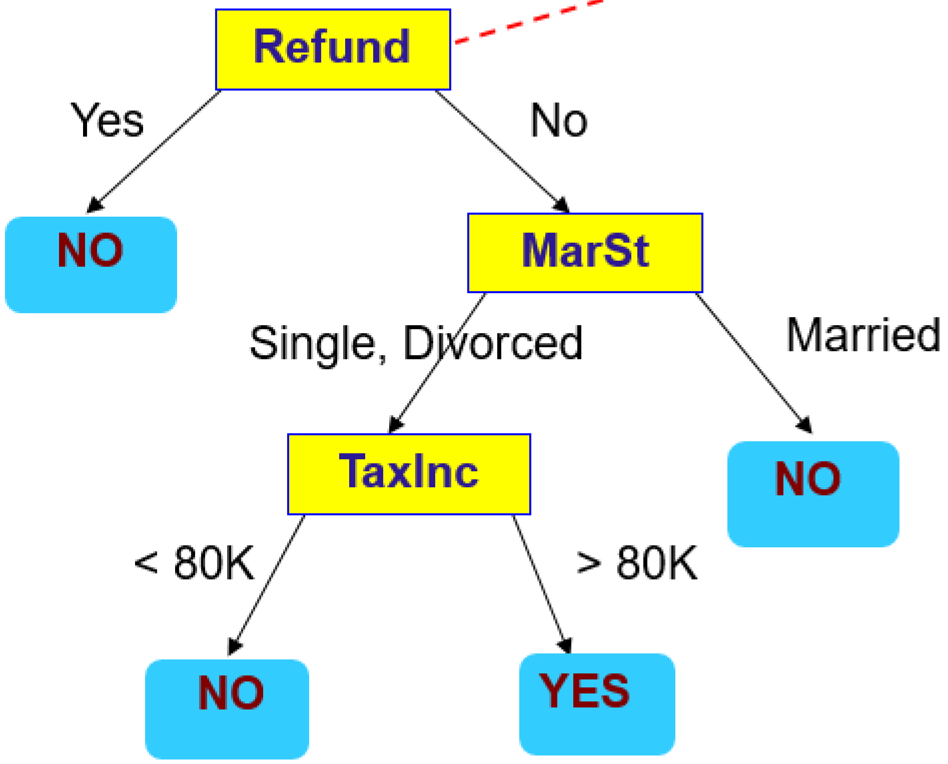
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data



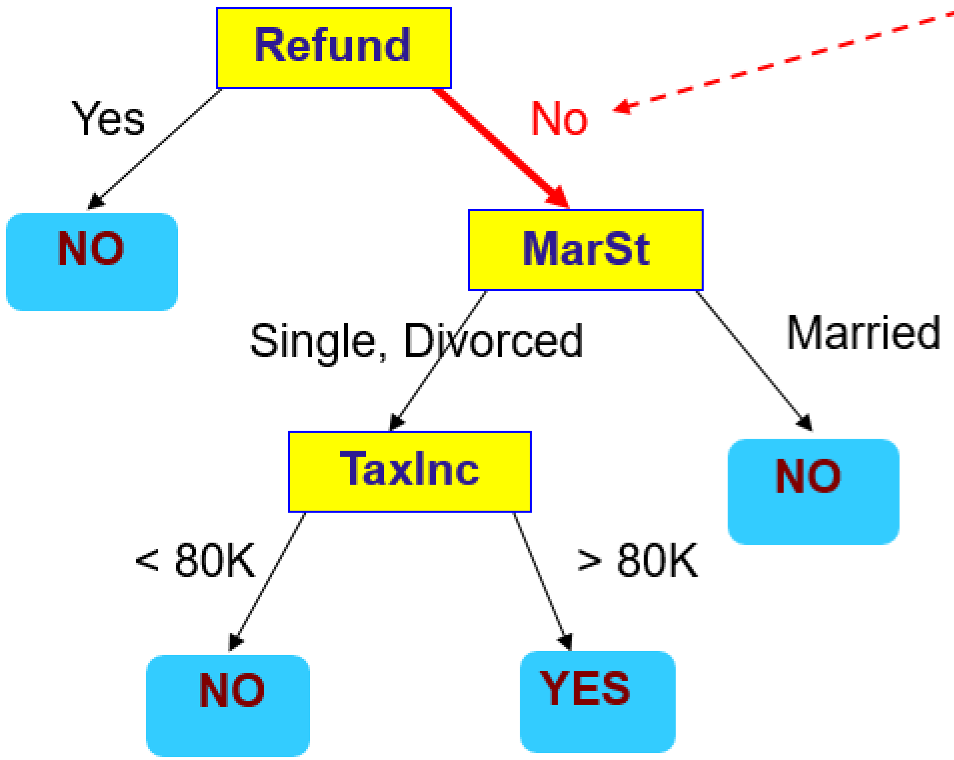
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Test Data

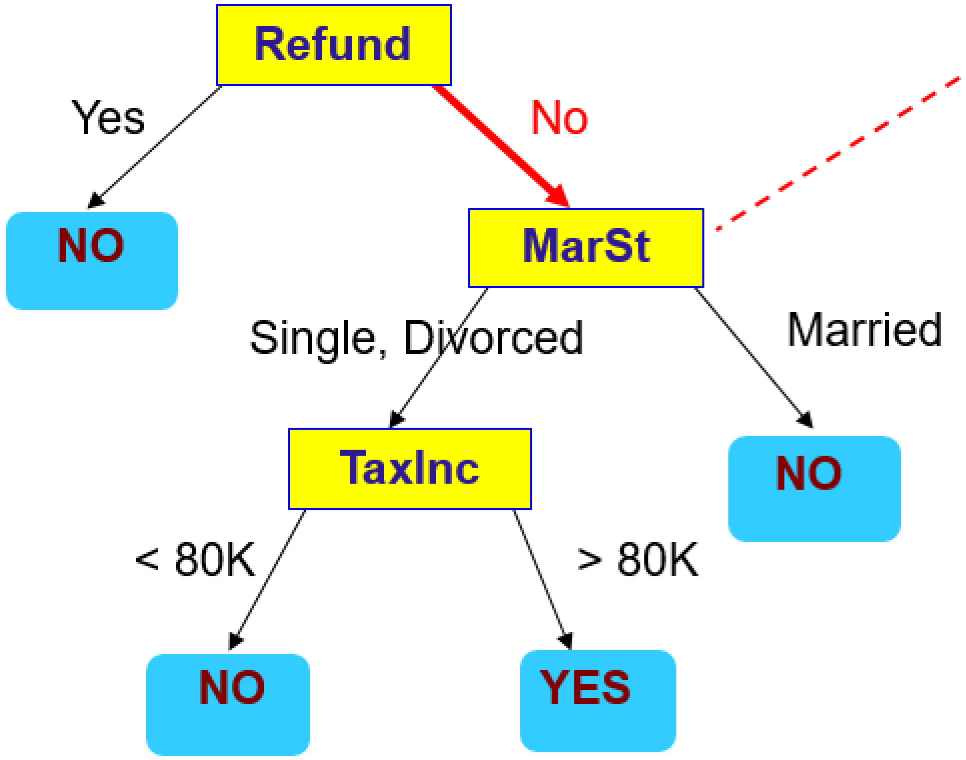
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

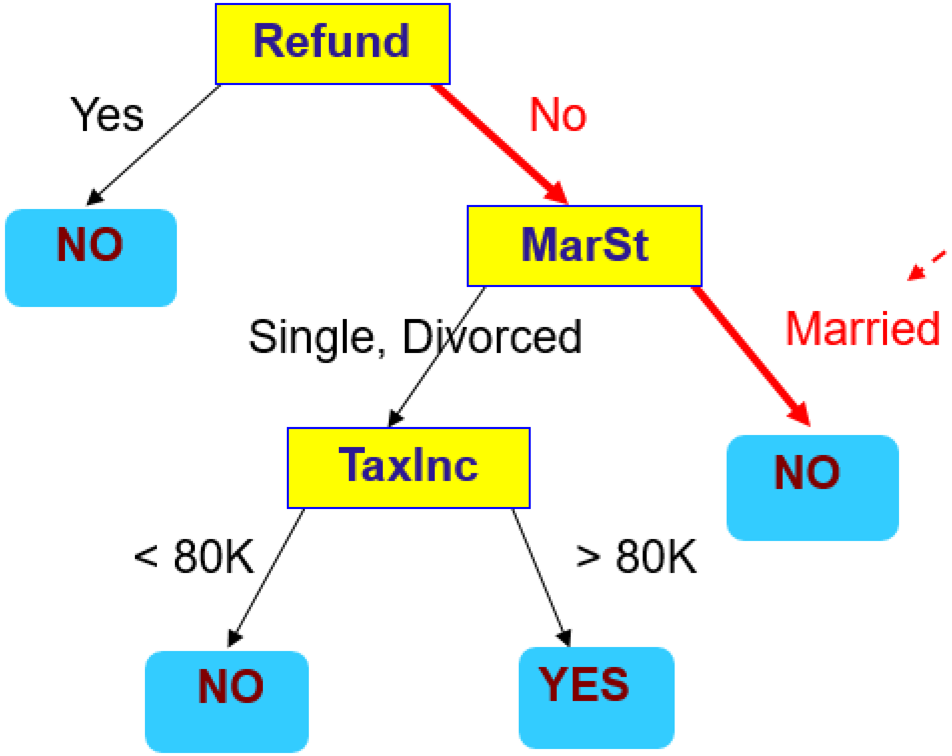
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

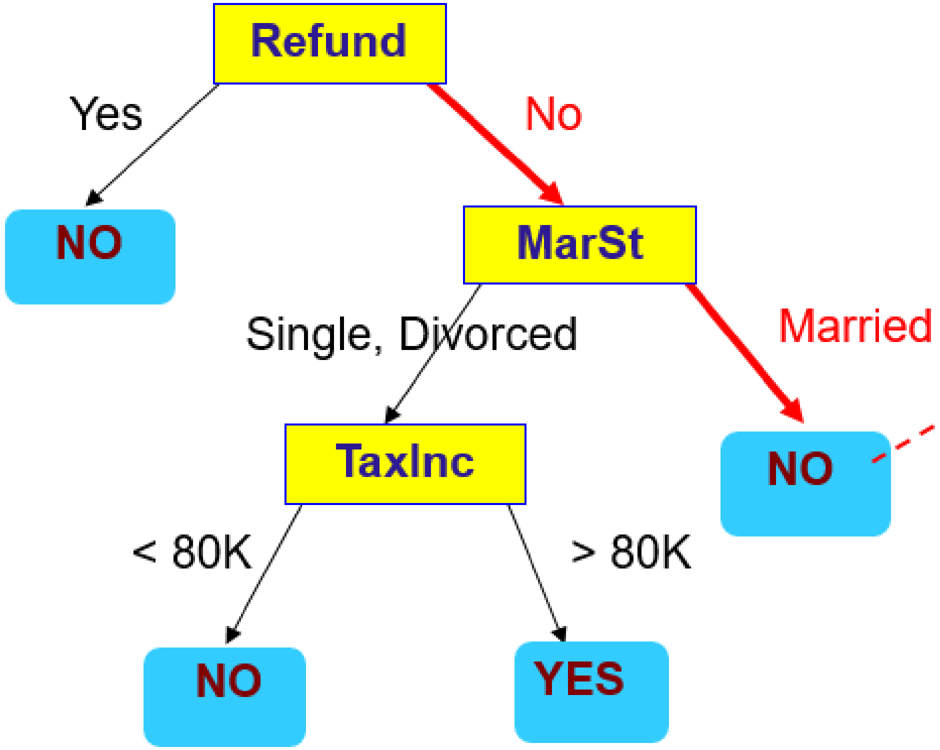
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



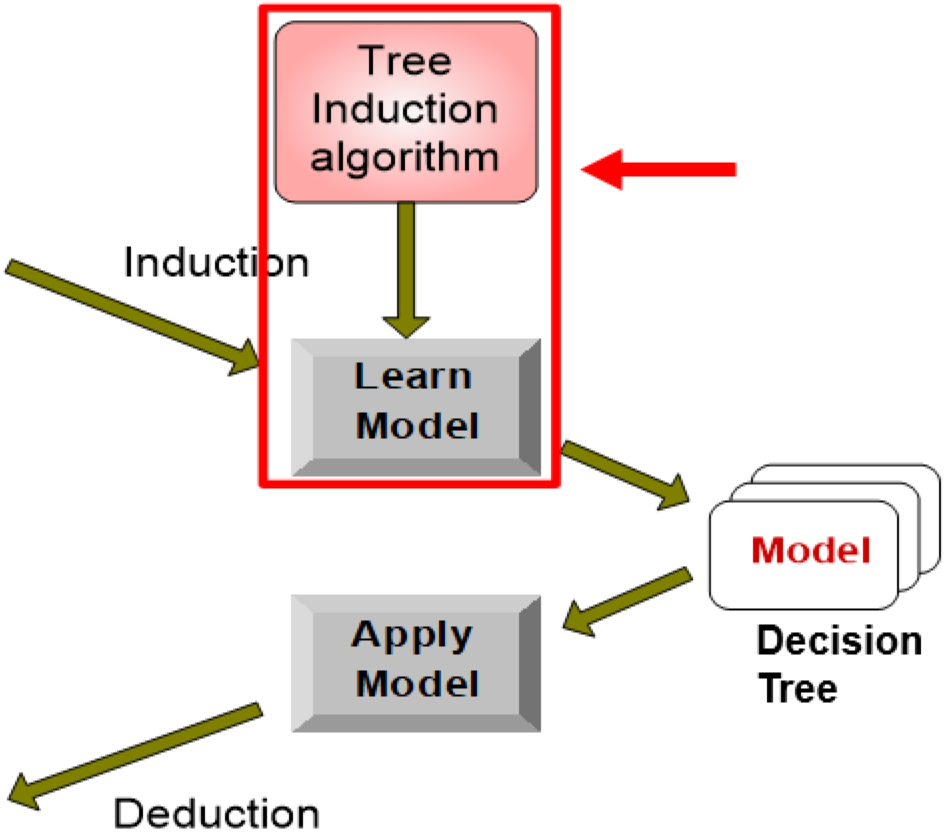
Assign Cheat to “No”

DATA ANALYTICS

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

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

THANK YOU

Jyothi R.
Assistant Professor,
Department of Computer Science
jyothir@pes.edu