

Data Mining

Topic Four- part II

Data mining techniques

Tibebe B. (PhD)

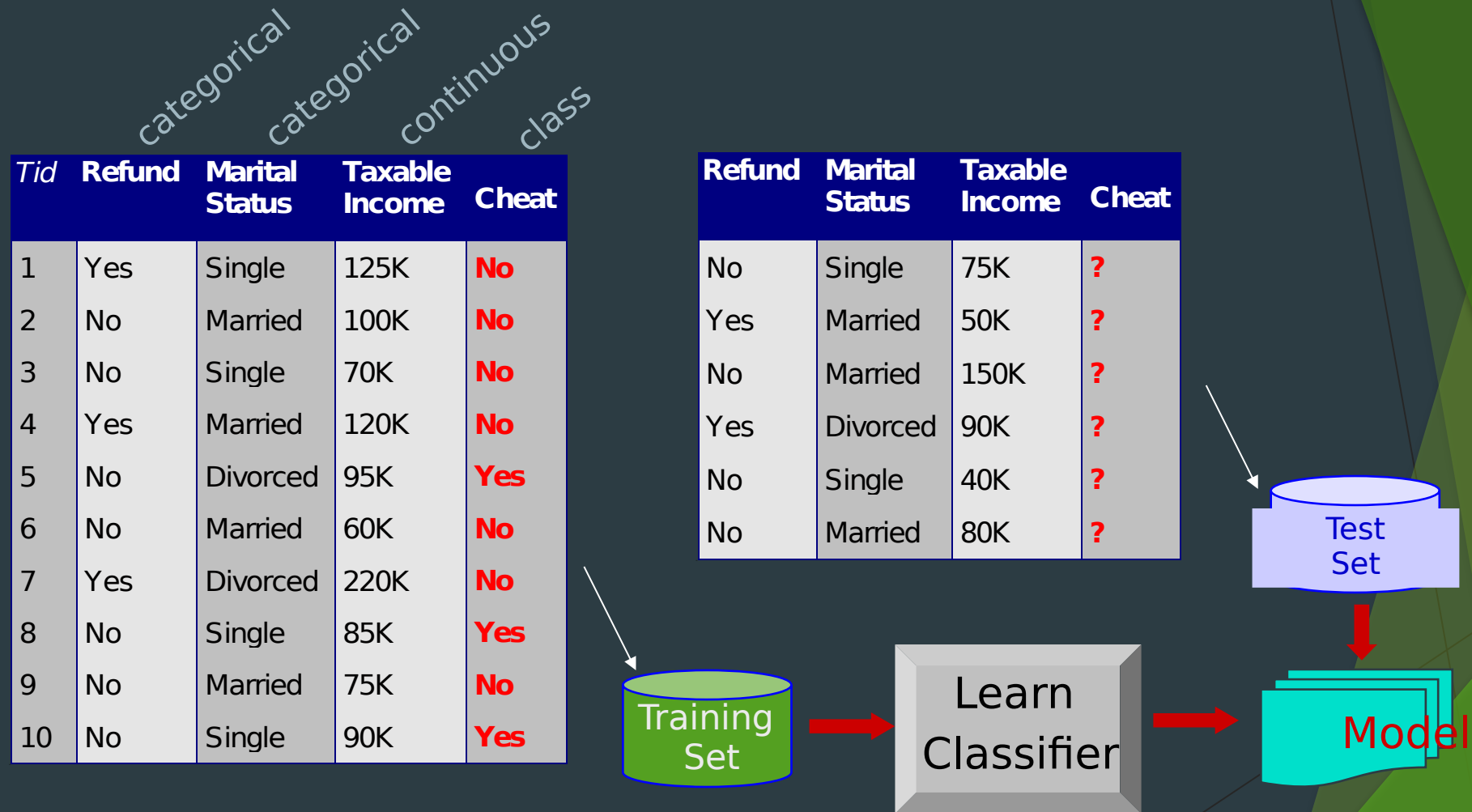
Topics

- ▶ Fundamental concepts and the need for business intelligence, data mining , big data analysis
- ▶ BI applications, DA models and frameworks
- ▶ Data and data warehousing
- ▶ Data mining techniques ;Association rule mining, Classification and Cluster analysis
 - ▶ Including Text/ web mining, sentiment mining , Big data and BI technologies, applications, and case studies
- ▶ Current trends in big data and BI

Classification

- ▶ Given a collection of records (*training set*)
 - ▶ Each record contains a set of *attributes*, one of the attributes is the *class*.
- ▶ Find a *model* for class attribute as a function of the values of other attributes.
- ▶ Goal: previously unseen records should be assigned a class as accurately as possible.
 - ▶ A *test set* is used to determine the accuracy of the model. Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it.

Classification Example



Cont...

- Classification is the process of finding a set of models (or functions) that describe and distinguish data classes or concepts for the purpose of being able to use the model to predict the class of an object whose class is unknown
- The derived class is based on training data set and can be represented in various forms such as classification IF—THEN rule, decision tree, mathematical formulae or neural networks

Classification: Application 1

- ▶ Direct Marketing
 - ▶ Goal: Reduce cost of mailing by *targeting* a set of consumers likely to buy a new cell-phone product.
 - ▶ Approach:
 - ▶ Use the data for a similar product introduced before.
 - ▶ We know which customers decided to buy and which decided otherwise. This $\{buy, don't\}$ decision forms the *class attribute*.
 - ▶ Collect various demographic, lifestyle, and company-interaction related information about all such customers.
 - ▶ Type of business, where they stay, how much they earn, etc.
 - ▶ Use this information as input attributes to learn a classifier model.

Classification: Application 2

- Fraud Detection
 - Goal: Predict fraudulent cases in credit card transactions.
 - Approach:
 - Use credit card transactions and the information on its account-holder as attributes.
 - When does a customer buy, what does he buy, how often he pays on time, etc
 - Label past transactions as fraud or fair transactions. This forms the class attribute.
 - Learn a model for the class of the transactions.
 - Use this model to detect fraud by observing credit card transactions on an account.

Classification: Application 3

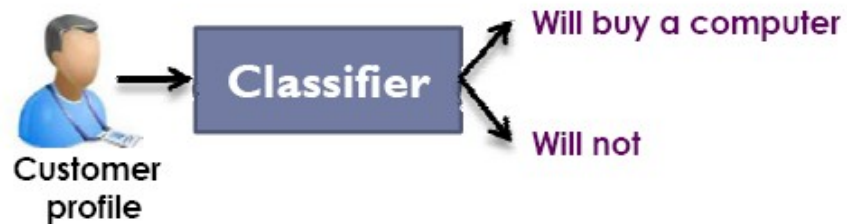
- ▶ Customer Attrition /Churn:
 - ▶ Goal: To predict whether a customer is likely to be lost to a competitor.
 - ▶ Approach:
 - ▶ Use detailed record of transactions with each of the past and present customers, to find attributes.
 - ▶ How often the customer calls, where he calls, what time-of-the day he calls most, his financial status, marital status, etc.
 - ▶ Label the customers as loyal or disloyal.
 - ▶ Find a model for loyalty.

Classification vs Prediction

► Classification

- Predicts categorical class labels (discrete or nominal)
- Use labels of the training data to classify new data

► Example



- A model or classifier is constructed to predict **categorical labels** such as “safe” or “risky” for a loan application data.

► Prediction

- Models continuous-valued functions, i.e., predicts unknown or missing values

► Example

- A marketing manager would like to predict how much a given customer will spend during a sale



- Unlike classification, it provides ordered values
- **Regression** analysis is used for prediction
- Prediction is a short name for **numeric prediction**

Classification—A Two-Step Process

► **Step1: Model Construction (learning step, or training step)**

→ Construct a classification model based on **training data**

→ **Training data**

- A set of tuples
- Each tuple is assumed to belong to a predefined class
- Labeled data (ground truth)

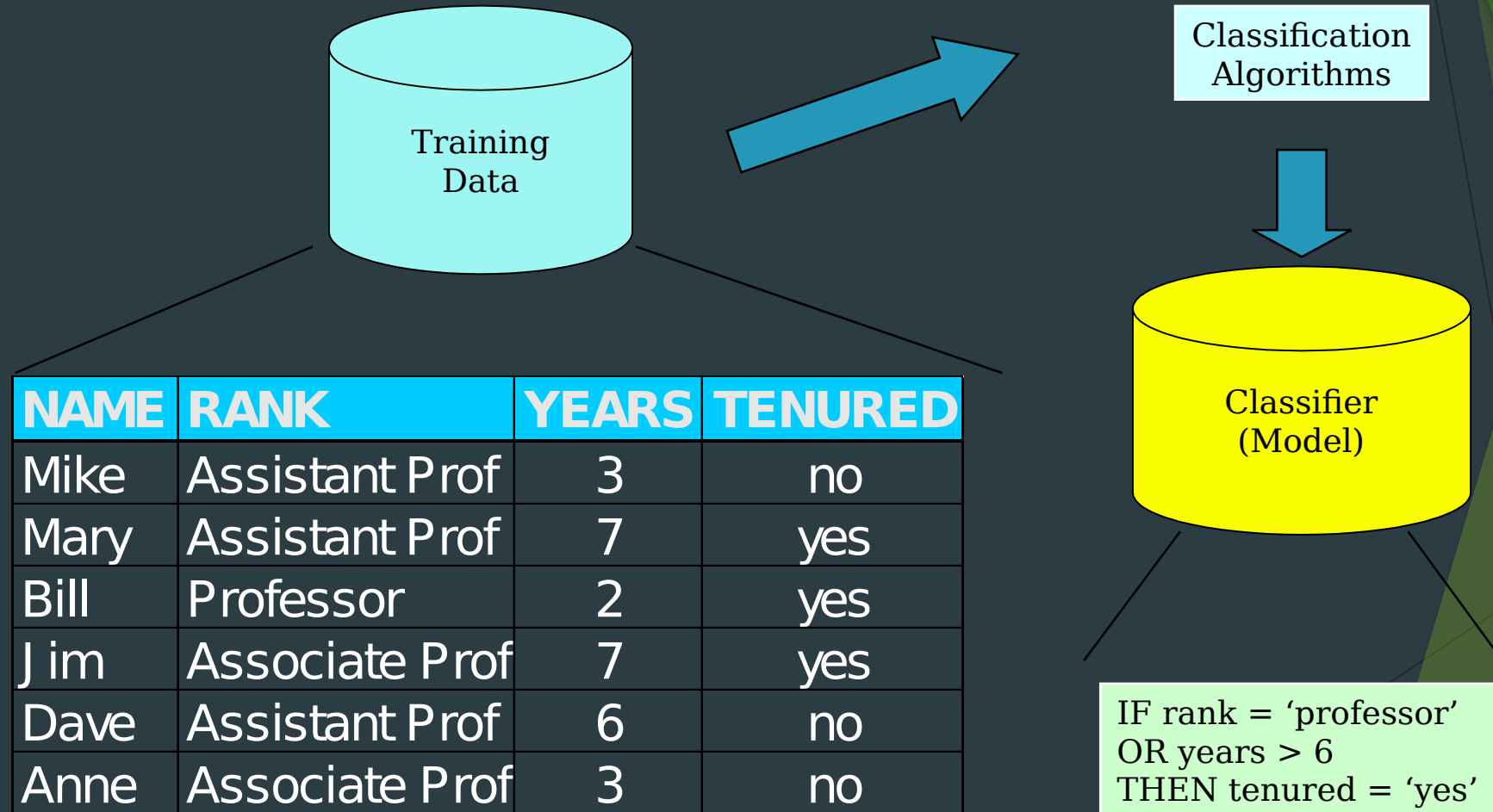
→ **How a classification model looks like?**

A classification model can be represented by one of the following forms:

- Classification rules
- Decision trees
- Mathematical formulae

Thus, model construction refers to describing a set of predetermined classes using training data set

Classification Process (1): Model Construction



Cont...

► Step2: Model Usage

Before using the model, we first need to test its accuracy

→ Measuring model accuracy

- To measure the accuracy of a model we need **test data**
- Test data is similar in its structure to training data (labeled data)

• How to test?

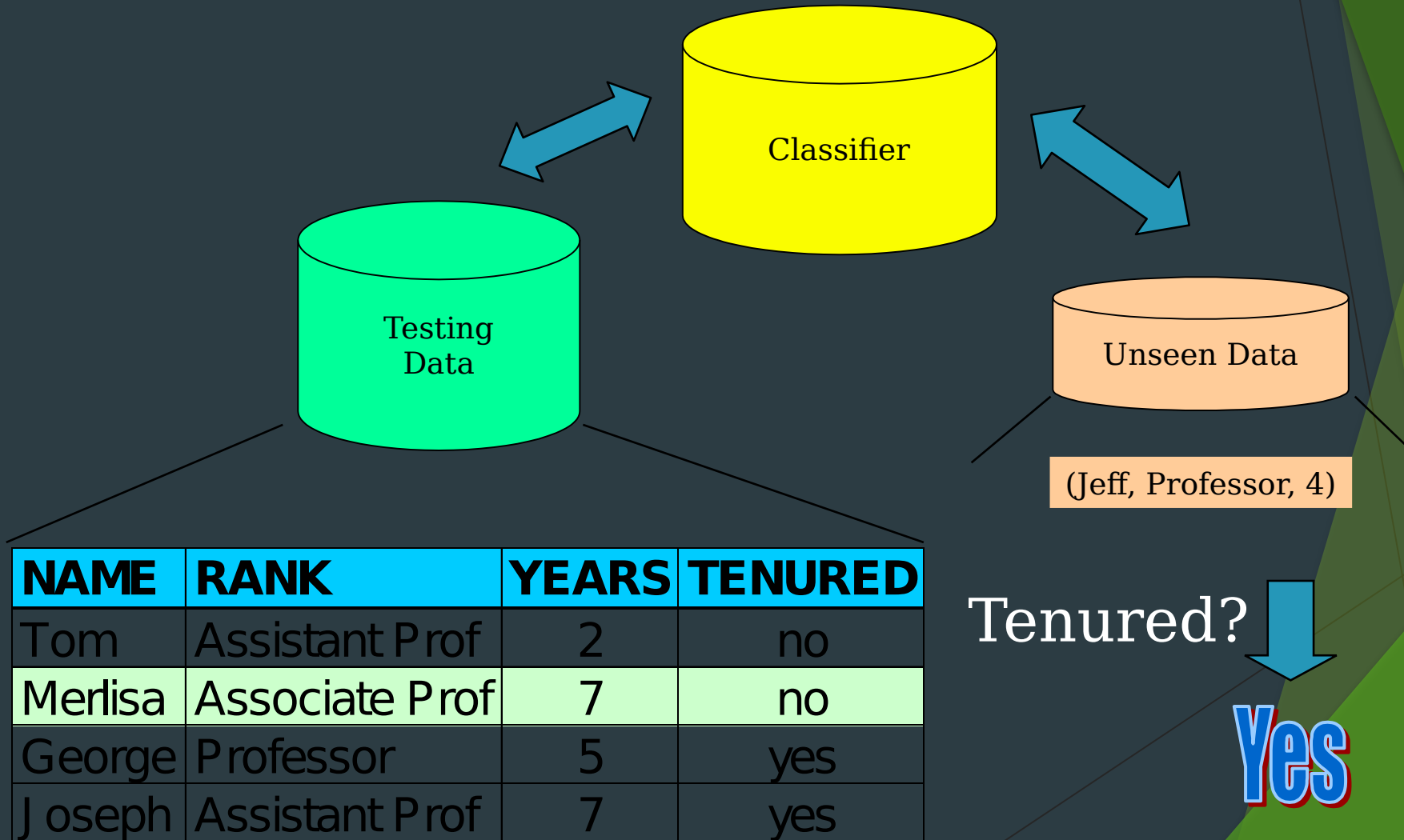
The known label of test sample is compared with the classified result from the model



- **Accuracy rate** is the percentage of test set samples that are correctly classified by the model
- **Important:** test data should be independent of training set, otherwise over-fitting will occur

→ **Using the model:** If the accuracy is acceptable, use the model to **classify data** tuples whose class labels are not known

Classification Process (2): Use the Model in Prediction



Issues regarding classification: Data Preparation and experimentation

- ▶ Data preprocessing
 - ▶ Preprocess data in order to reduce noise and handle missing values
 - ▶ Data transformation
 - ▶ Generalize and/or normalize data
- ▶ Relevance analysis (feature selection)
 - ▶ Remove the irrelevant or redundant attributes
- ▶ Conducting various experiments using different algorithms
 - ▶ Select the best model

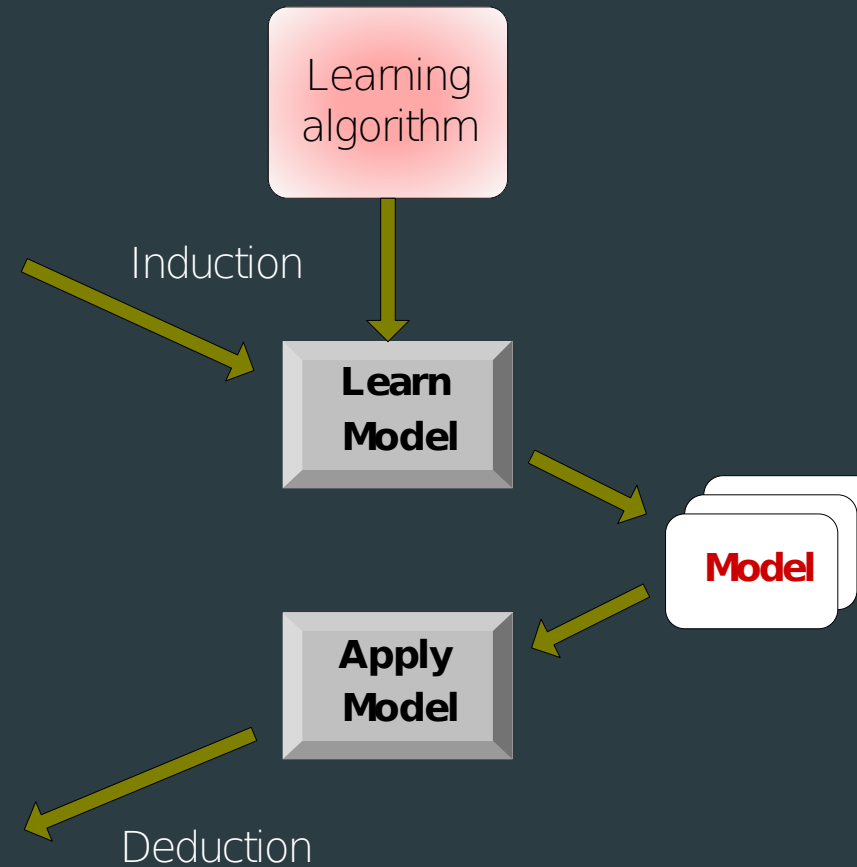
Illustrating Classification Task- induction and deduction

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



Thus Classification

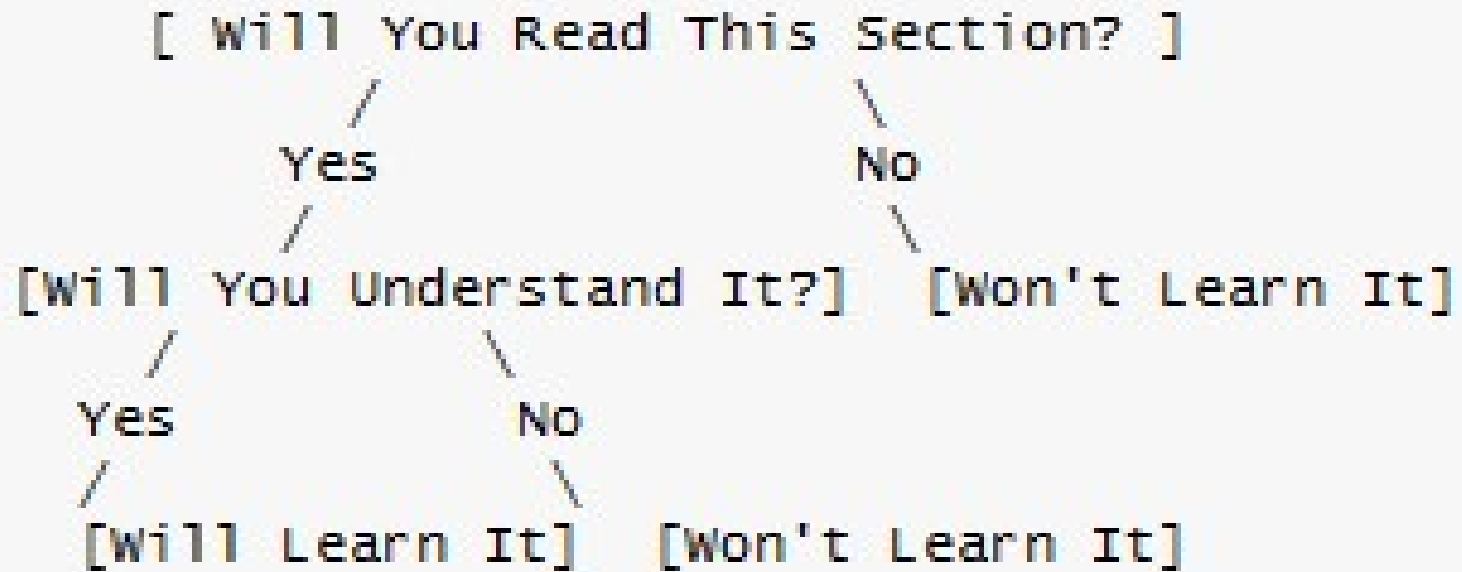
- ▶ Classification is a data mining (machine learning) technique used to predict **group membership** for data instances.
- ▶ Given a collection of records (training set), each record contains a set of attributes, one of the attributes is the class.
 - ▶ It is **finding a model** for class attribute **as a function of the values of other attributes**.
- ▶ Goal: previously unseen records should be assigned a class as accurately as possible. A test set is used to determine the accuracy of the model.
 - ▶ Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it.
- ▶ For example, one may use classification to predict whether the weather on a particular day will be “sunny”, “rainy” or “cloudy”.

Classification approaches/methods

- ▶ There are various classification methods. Popular classification techniques include the following.
 - ▶ Decision tree classifier: divide decision space into piecewise constant regions.
 - ▶ Rule based – Association based classifier
 - ▶ K-Nearest Neighbour: classify based on similarity measurement
 - ▶ Neural networks: partition by non-linear boundaries
 - ▶ Bayesian network: a probabilistic model
 - ▶ Support vector machine: solves non-linearly separable problems

Decision Trees

Simple classification using decision tree



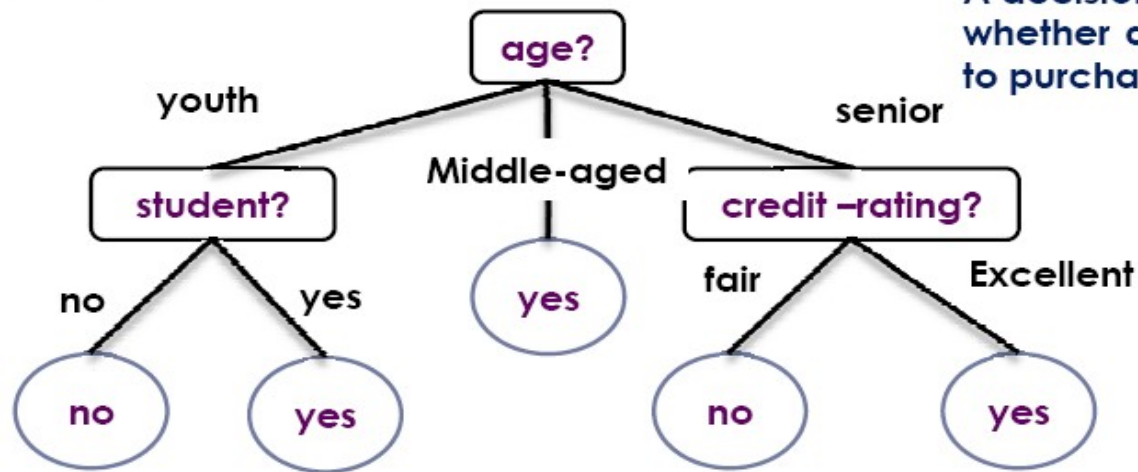
What are the two known classes?
How many attributes are used? (What are they?)

Decision tree classifier

- ▶ *Decision tree* performs classification by constructing a tree based on training instances with leaves having class labels.
 - ▶ The tree is traversed for each test instance to find a leaf, and the class of the leaf is the predicted class.
 - ▶ This is a directed knowledge discovery in the sense that there is a specific field whose value we want to predict.
- ▶ Widely used learning method. It has been applied to:
 - ▶ classify medical patients based on the disease,
 - ▶ equipment malfunction by cause,
 - ▶ loan applicant by likelihood of payment.
 - ▶ Accidents by severity

Cont...

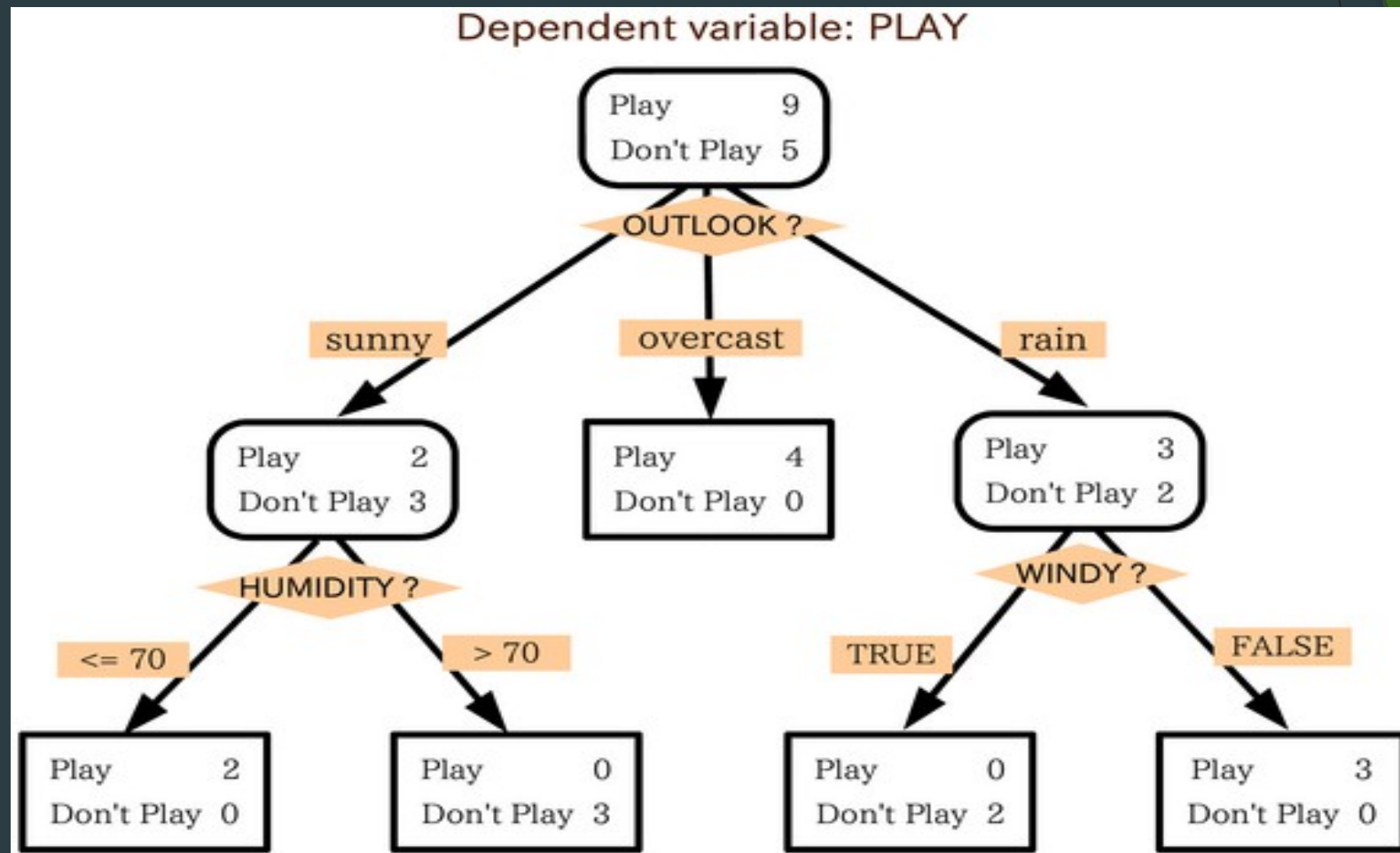
- ▶ **Decision tree induction** is the learning of decision trees from class-labeled training tuples
- ▶ A decision tree is a flowchart-like tree structure
 - **Internal nodes** (non leaf node) denotes a test on an attribute
 - **Branches** represent outcomes of tests
 - **Leaf nodes** (terminal nodes) hold class labels
 - **Root node** is the topmost node



A decision tree indicating whether a customer is likely to purchase a computer

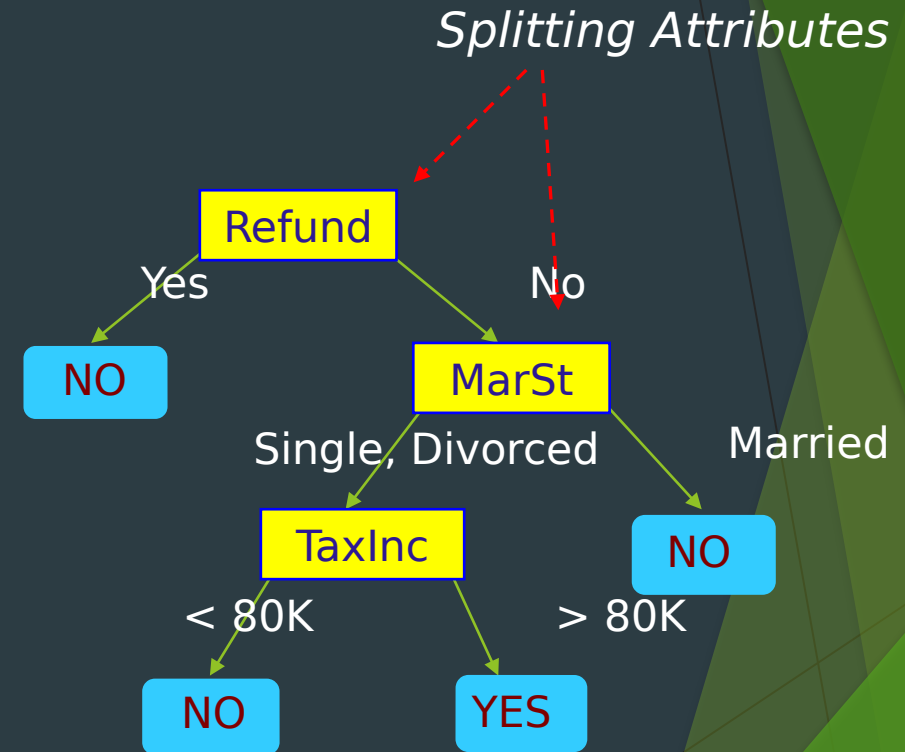
Class-label Yes: The customer is likely to buy a computer
Class-label no: The customer is unlikely to buy a computer

Decision Tree (detail)



Example of a Decision Tree

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

What is the preprocessing task done on the attribute "taxable income" ?

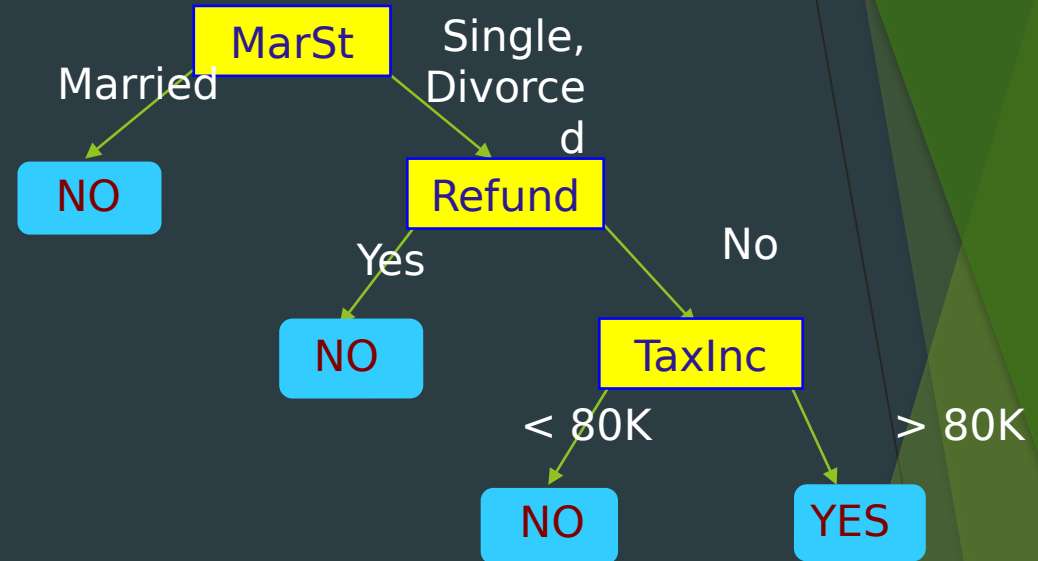
Explain the model usage stage with this example?

Model: Decision Tree

Another Example of Decision Tree

<i>Tid</i>	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

categorical
categorical
continuous
class

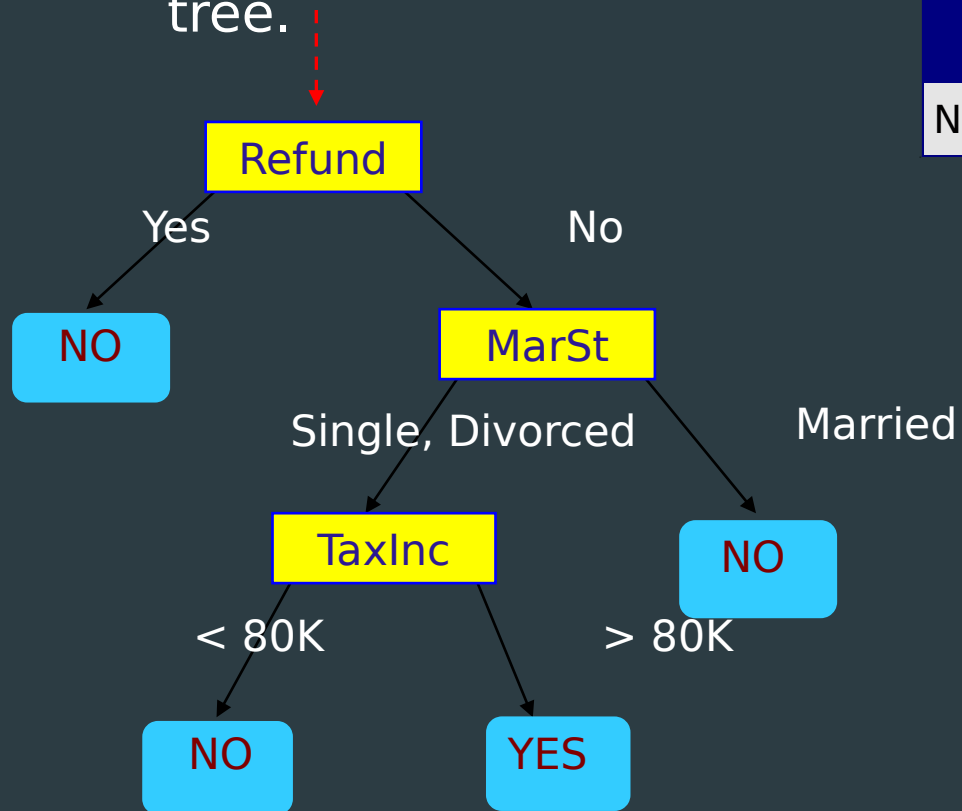


There could be more than one tree that fits the same data!

What is the difference with the previous tree?

Apply Model to Test Data

Start from the root of tree.



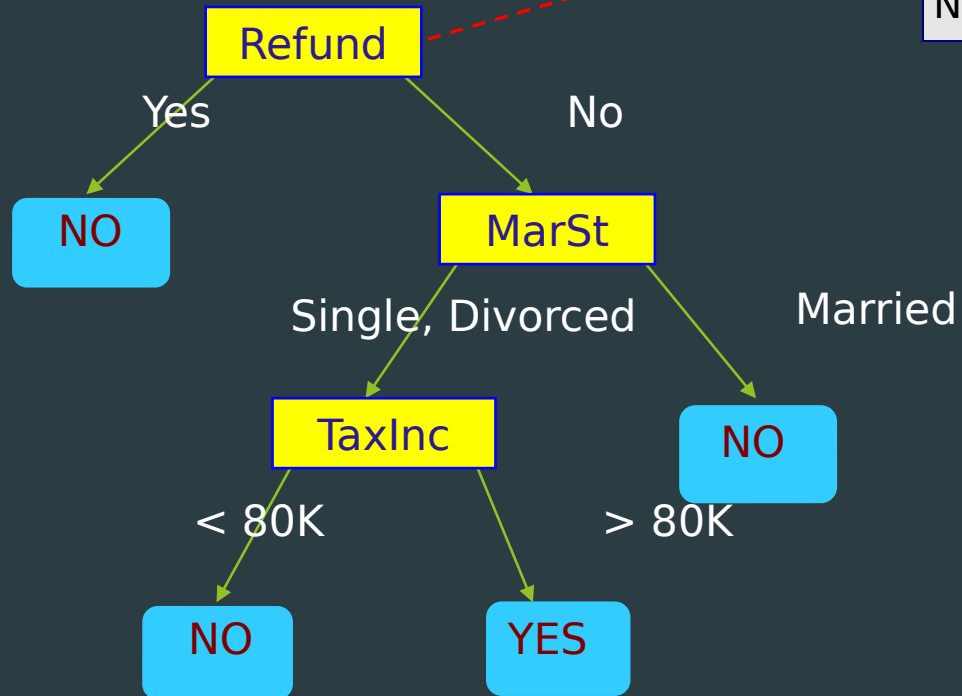
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Test Data

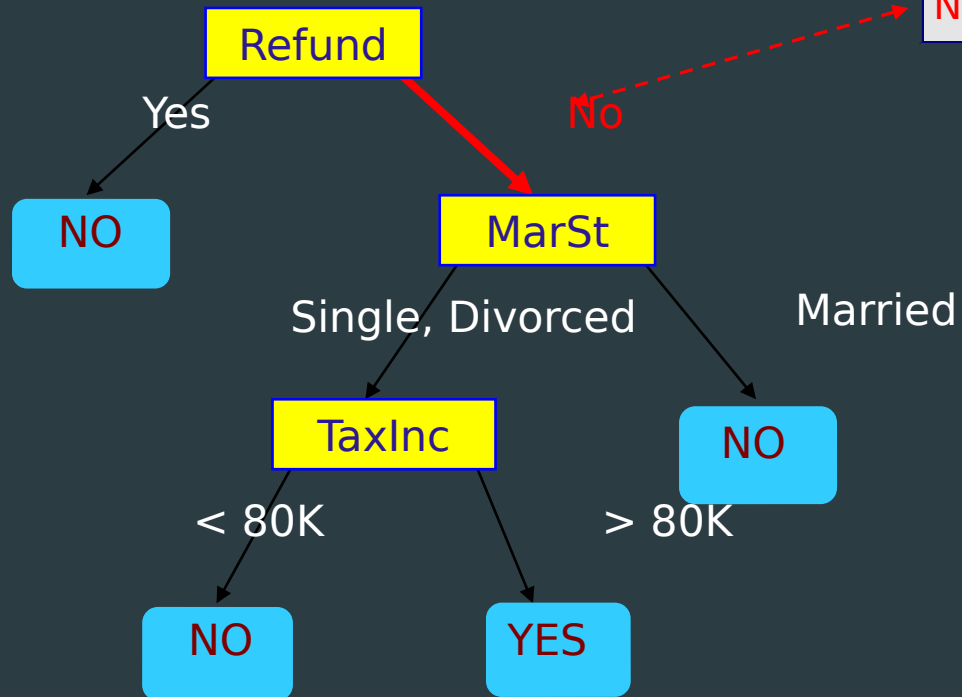
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

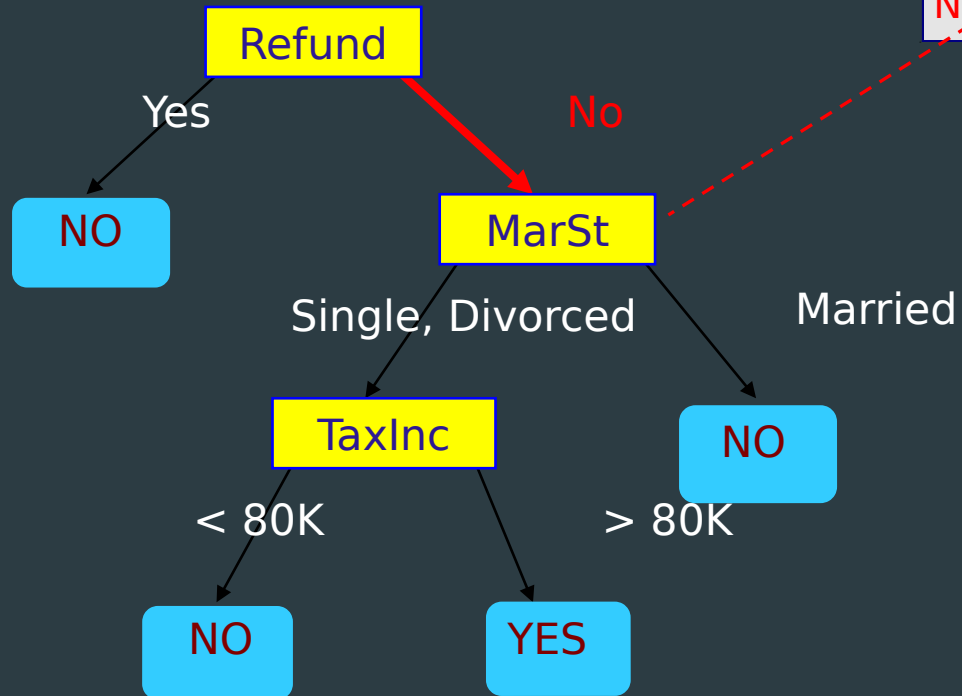
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

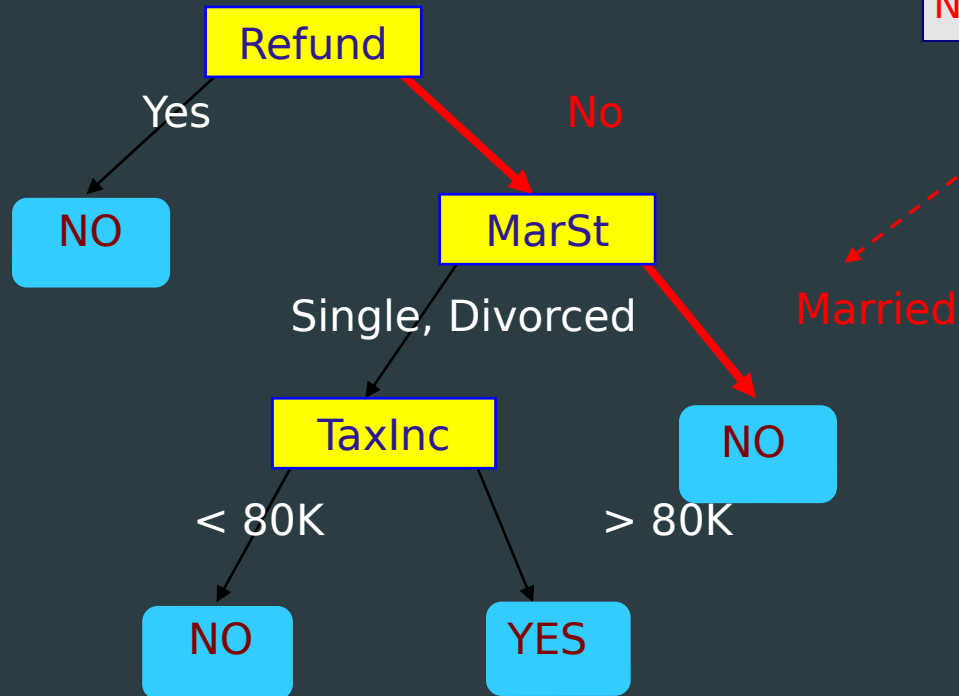
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

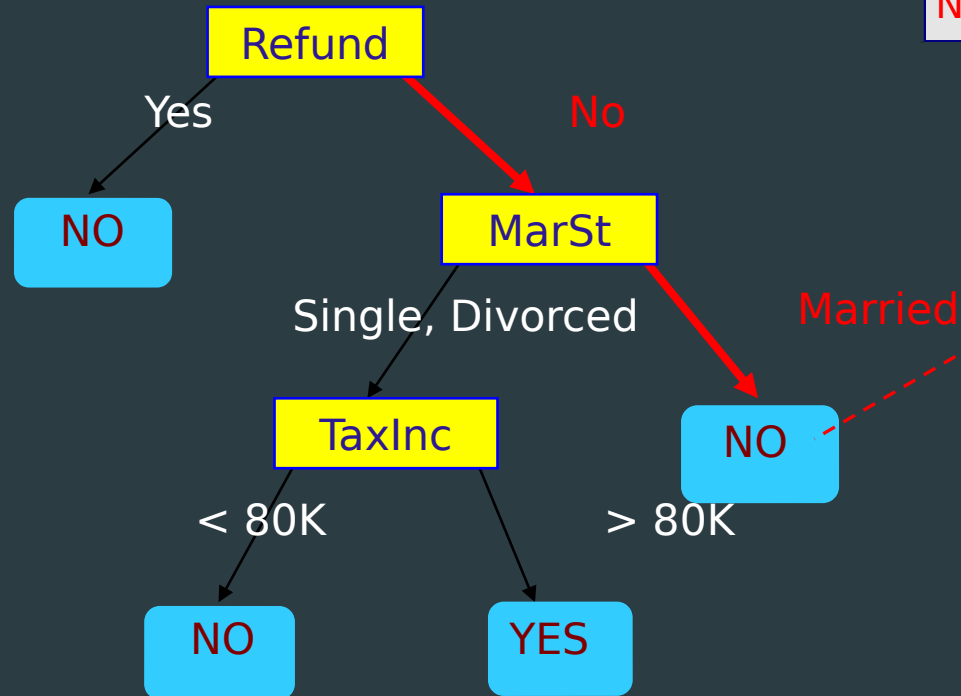
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Assign Cheat to
"No"

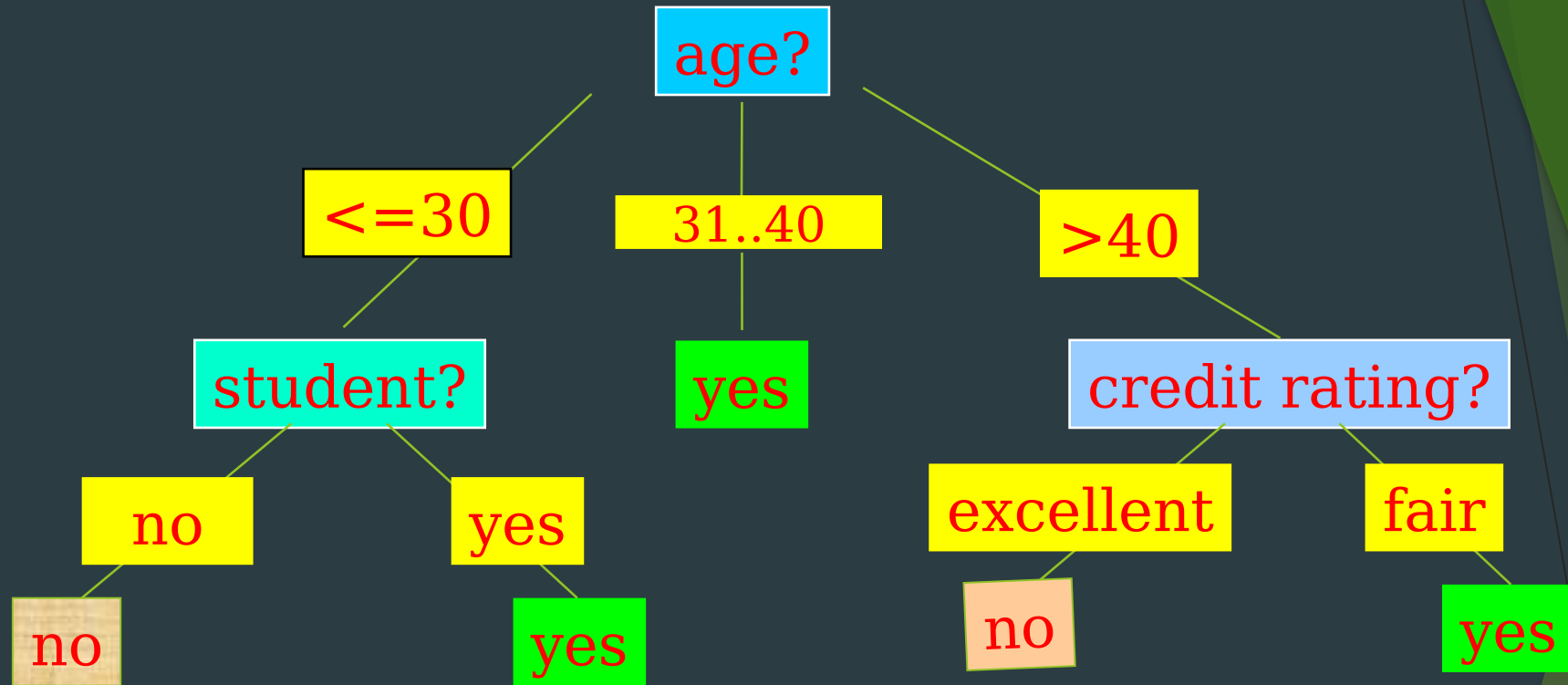
Thus...

- ▶ As it is a classification approach; Decision tree generation consists of two phases
 - ▶ Tree construction
 - ▶ At start, all the training examples are at the root
 - ▶ Partition examples recursively based on selected attributes
 - ▶ Tree pruning
 - ▶ Identify and remove branches that reflect noise or outliers
 - ▶ Use of decision tree:
 - ▶ Classifying an unknown sample
 - ▶ Test the attribute values of the sample against the decision tree

Training Dataset- example

age	income	student	credit_rating	buys_computer
≤30	high	no	fair	no
≤30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
≤30	medium	no	fair	no
≤30	low	yes	fair	yes
>40	medium	yes	fair	yes
≤30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no

Output: A Decision Tree for “*buys_computer*”



Extracting Classification Rules from Trees

- ▶ Represent the knowledge in the form of IF-THEN rules
- ▶ One rule is created for each path from the root to a leaf
- ▶ Each attribute-value pair along a path forms a conjunction
- ▶ The leaf node holds the class prediction
- ▶ Rules are easier for humans to understand
- ▶ Example

IF *age* = " ≤ 30 " AND *student* = "*no*" THEN *buys_computer* = "*no*"

IF *age* = " ≤ 30 " AND *student* = "*yes*" THEN *buys_computer* = "*yes*"

IF *age* = "31...40" THEN *buys_computer* = "*yes*"

IF *age* = " > 40 " AND *credit_rating* = "*excellent*" THEN *buys_computer* = "*yes*"

IF *age* = " > 40 " AND *credit_rating* = "*fair*" THEN *buys_computer* = "*no*"

What do you think is the key question now in DT learning?

Points to recall

- ▶ Classification
 - ▶ a two step process
 - ▶ PREDICT categorical class labels
- ▶ Classification approaches
 - ▶ Decision tree based techniques
 - ▶ Rule based techniques
 - ▶ Function based
- ▶ Decision tree
 - ▶ Attribute selection is the core idea

Decision Tree Induction

- ▶ Many Algorithms:
 - ▶ **Hunt's Algorithm (foundational):** Most of the existing induction-based algorithms use **Hunt's** method as the basic algorithm
 - ▶ ID3: Developed by J. Ross Quinlan in late 1970s,. All attributes are assumed to be categorical. Uses information gain to split node into branches
 - ▶ C4.5 (successor of ID3):It is an improvement of ID3. Can work for continuous-valued attributes
 - ▶ CART (Classification and regression Tree):It works in the same way as ID3- with some improvement in missing value treatment.
 - ▶ SLIQ,SPRINT....
- ▶ **ID3, C4.5, and CART adopt a greedy (i.e., nonbacktracking) approach in which decision trees are constructed in a top-down recursive divide-and-conquer** manner using Information gain attribute selection approach.

Constructing decision trees: the Hunt's algorithm

- ▶ X_t : the set of training records for node t
- ▶ $y = \{y_1, \dots, y_c\}$: class labels
- ▶ **Step 1:** If all records in X_t belong to the same class y_t , then t is a leaf node labeled as y_t
- ▶ **Step 2:** If X_t contains records that belong to more than one class,
 - ▶ select **attribute test condition** to partition the records into smaller subsets
 - ▶ Through information gain or error rate
 - ▶ Create a **child node** for each outcome of test condition
 - ▶ Apply algorithm **recursively** for each **child**

Algorithm for Decision Tree Induction- focusing on test attributes

- ▶ Basic algorithm (a greedy algorithm)
 - ▶ Tree is constructed in a top-down recursive divide-and-conquer manner
 - ▶ At start, all the training examples are at the root
 - ▶ Attributes are categorical (if continuous-valued, they are discretized in advance)
 - ▶ Examples are partitioned recursively based on selected attributes
 - ▶ Test attributes are selected on the basis of a heuristic or measures like - information gain
 - ▶ The process continues until the stopping criteria is satisfied

Tree Induction (more detail)

- ▶ Greedy strategy.
 - ▶ Split the records based on an attribute test that optimizes certain criterion.
- ▶ Issues
 - ▶ Determine **how to split** the records
 - ▶ How to specify the attribute test condition?
 - ▶ How to determine the best split?
 - ▶ Determine **when to stop** splitting

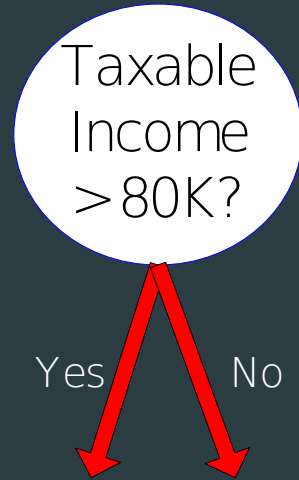
How to Specify Test Condition?

- ▶ Depends on attribute types and number of ways to split
 - ▶ Attribute type
 - ▶ Nominal
 - ▶ Ordinal
 - ▶ Continuous
 - ▶ ways to split
 - ▶ 2-way split
 - ▶ Multi-way split

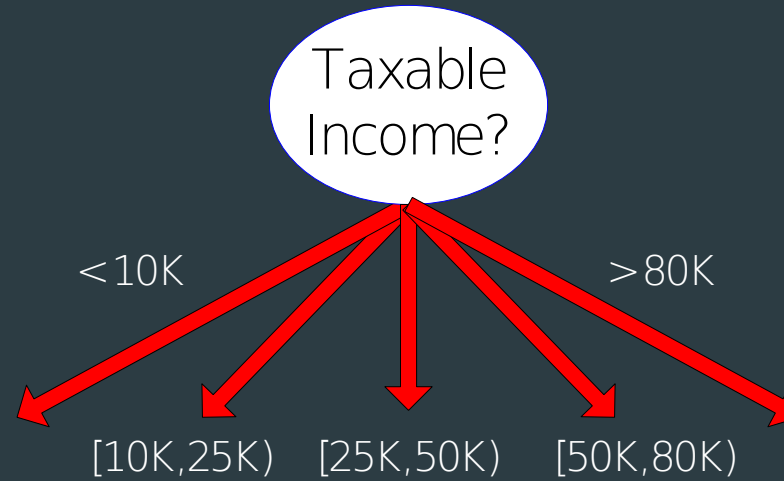
Splitting Based on Continuous Attributes

- ▶ 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 finds the best cut
 - ▶ can be more compute intensive

Splitting Based on Continuous Attributes



(i) Binary split



(ii) Multi-way split

How to determine the Best Split

Ultimately classification using decision tree can be reduced to the problem of selecting the best splitting attribute. Before Splitting: 10 records of class 0, 10 records of class 1



Which test condition is the best?

Cont...

- ▶ **$p(i|t)$** : fraction of records belonging to class **i**
- ▶ **Best split** is selected based on the degree of **impurity** of the child nodes
 - ▶ Class distribution **(0,1)** has **high purity**
 - ▶ Class distribution **(0.5,0.5)** has the **smallest purity (highest impurity)**
- ▶ **Intuition**: high purity $\square$ small value of impurity measures $\square$ better split

Cont...

- ▶ Nodes with homogeneous class distribution are preferred
- ▶ Need a measure of node impurity:

C0: 5
C1: 5

Non-homogeneous,
High degree of
impurity

C0: 9
C1: 1

Homogeneous,
Low degree of
impurity

Attribute Selection Measure

- Attribute selection measure also called splitting rule is the process of selecting an attribute that need to be used to define a condition for creating branches based on the tuples at a given node.
- Ideally, we need a condition that create branches in which each successor node holds tuples from only one class.
- As it is difficult to find such a single condition, conditioned on a single attribute, we need to find an attribute that best approximate the stated scenario
- The following are some of the techniques for attribute (test condition) selection
 - Information Gain
 - Information GainRatio
 - Classification error

Attribute Selection Measure (Measures of Node Impurity)

- ▶ **Information gain:** Select the attribute with the highest information gain
 - First, compute the disorder using Entropy; the expected information needed to classify objects into classes
 - Second, measure the Information Gain; to calculate by how much the disorder of a set would reduce by knowing the value of a particular attribute.
 - In information gain measure we want:-
 - Large Gain
 - Same as: small average disorder created

Use of InfoGain - three step procedure

- ▶ **Step1:** compute **Expected information** (entropy) of the current partition **Info(D)**
- ▶ **Step2:** compute **Info_A(D)**, the amount of information would we still need to arrive at an exact classification after partitioning using attribute A
- ▶ **Step3:** compute information gain:
- ▶ Information gain by branching on A is

$$Gain(A) = Info(D) - Info_A(D)$$

- ▶ **Information gain** is the expected reduction in the information requirements caused by knowing the value of A
 - The attribute **A** with the **highest information gain**, (Gain(A)), is **chosen** as the splitting attribute at node N

Information Gain (in ID3 and its successor)

- Select the attribute with the highest information gain
- This attribute minimizes the information needed to classify the tuples in the resulting partitions and reflects the least randomness or “impurity” in these partitions
- Such an approach minimizes the expected number of tests needed to classify a given tuple and guarantees that a simple (but not necessarily the simplest) tree is found.
- Assume there are n classes $C_1, C_2, C_3, \dots, C_n$
- Let S be the set of tuples at a given node P with a total of N tuples
- Let S contain c_i elements from class C_i
- **Now We can calculate $\text{Inf}(D)$ or $I(S)$**

Cont...

- o The amount of information needed to decide if an arbitrary example in S belongs to one of the class is defined as

$$I(S) = I(c_1, c_2, \dots, c_n) = - \sum_{i=1}^n \frac{c_i}{N} \log_2 \frac{c_i}{N}$$

- o Consider S to be the tuples in the next slide in which the tuples has classifying attribute **buy computer** with two class $C_1 = \text{yes}$, $C_2 = \text{no}$

- o **S has a total of 14 tuples, 9 of them are in class one and 5 of them are class two**

$$I(S) = I(c_1, c_2) = - \frac{9}{14} \log_2 \frac{9}{14} - \frac{5}{14} \log_2 \frac{5}{14} \\ = 0.41 + 0.53 = 0.94$$

- o Hence

Cont...

age	income	student	credit_rating	buys_computer
≤ 30	high	no	fair	no
≤ 30	high	no	excellent	no
31...40	high	no	fair	yes
> 40	medium	no	fair	yes
> 40	low	yes	fair	yes
> 40	low	yes	excellent	no
31...40	low	yes	excellent	yes
≤ 30	medium	no	fair	no
≤ 30	low	yes	fair	yes
> 40	medium	yes	fair	yes
≤ 30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
> 40	medium	no	excellent	no

Cont...

- ▶ Note the following
 - ▶ The amount of information need becomes maximum if all the classes has the same number of tuples (i.e. $I(S)=1$)
 - ▶ The amount of information need becomes minimal if all the tuples belongs to one class (i.e $I(S)= 0$)

$$I(S) = I(c_1, c_2, \dots, c_n) = - \sum_{i=1}^n \frac{c_i}{N} \log_2 \frac{c_i}{N}$$

Cont...

- Once we know the information needed to classify a given tuple into the classes C for a given S , we can see which attribute to be chosen for classification
- Assume that using attribute A will partition the set of tuples in S into sets $\{S_1, S_2, \dots, S_m\}$
 - One can calculate the required information to classify a tuple from each of the subsets (S_i) which is $I(S_i)$
- This can be used to define the Entropy when A is selected as an attribute for conditioning S as

$$Ent(A) = \sum_{i=1}^m \frac{|S_i|}{|S|} I(S_i)$$

Cont...

- Then the encoding information that would be gained by branching on A becomes

$$\text{Gain}(A) = I(S) - E(A)$$

- Choose the attribute with the maximum gain
 - Take attribute age for the above example, it will divide S into three and find **$E(\text{age})$**
 - Take attribute Income for the above example, it will divide S into three and find **$E(\text{Income})$**
 - Take attribute Student for the above example, it will divide S into two and find **$E(\text{Student})$**
 - Take attribute CreditRating for the above example, it will divide S into two and find **$E(\text{CreditRating})$**
 - And chose the one with the **maximum** gain

Attribute Selection by Information Gain

- Class P (c1): buys_computer = "yes"
- Class N (c2): buys_computer = "no"
- $E(P, N) = E(9, 5) = 0.940$
- Compute the entropy for *age*:

age	p_i	n_i	$E(p_i, n_i)$
≤ 30	2	3	0.971
30...40	4	0	0
> 40	3	2	0.971

$$E(\text{age}) = \frac{5}{14}E(2,3) + \frac{4}{14}E(4,0) + \frac{5}{14}E(3,2) = 0.69$$

Hence

$$Gain(\text{age}) = E(P, N) - E(\text{age})$$

Similarly

$$Gain(\text{age}) = 0.25$$

$$Gain(\text{income}) = 0.029$$

$$Gain(\text{student}) = 0.151$$

$$Gain(\text{credit_rating}) = 0.048$$

Use the information bellow and manually develop the decision tree

age	income	student	credit_rating	buys_computer
≤30	high	no	fair	no
≤30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
≤30	medium	no	fair	no
≤30	low	yes	fair	yes
>40	medium	yes	fair	yes
≤30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no

Cont...

The mutual information of the two classes $I(S_{\text{Yes}}, S_{\text{No}}) = I(9,5) = -9/14 \log_2(9/14) - 5/14 \log_2(5/14) = 0.94$

- For Age we have three values $\text{age}_{\leq 30}$ (2 yes and 3 no), $\text{age}_{31..40}$ (4 yes and 0 no) and $\text{age}_{>40}$ (3 yes 2 no)

$$\begin{aligned}\text{Entropy}(\text{age}) &= 5/14 (-2/5 \log(2/5) - 3/5 \log(3/5)) + 4/14 (0) + 5/14 (-3/5 \log(3/5) - 2/5 \log(2/5)) \\ &= 5/14(0.9709) + 0 + 5/14(0.9709) \\ &= 0.6935\end{aligned}$$

$$\text{Gain}(\text{age}) = 0.94 - 0.6935 = 0.2465$$

Information GainRatio (C4.5)

- Consider a case in which ID is one of the attribute in addition to some more attribute for the given list of tuples
 - Note that value of ID is unique to each tuple
 - Observe that also ID is the attribute that will have the maximum $\text{Gain}(\text{ID}) \geq \text{Gain}(A)$ for any other attribute A of the data set
 - Hence it will be chosen as a splitting attribute which don't have any relevance
- Hence C4.5 uses **GainRatio** rather than **Gain** of an attribute in attribute selection method

Information GainRatio (C4.5)

- GainRatio of an attribute on a data set S is simply defined as the ratio of the Gain of the attribute as stated in ID3 divided by the information required to classify a tuple from S into a class label

$$GainRatio(A) = \frac{Gain(A)}{- \sum_{i=1}^m \frac{|S_i|}{|S|} \log_h \frac{|S_i|}{|S|}} = \frac{I(S) - Entropy(A)}{- \sum_{i=1}^m \frac{|S_i|}{|S|} \log_h \frac{|S_i|}{|S|}}$$

- Where m is the number of classes
- Note: the definition below

$$I(S) = - \sum_{i=1}^m \frac{c_i}{N} \log_2 \frac{c_i}{N}$$

$$Ent(A) = \sum_{i=1}^m \frac{|S_i|}{|S|} I(S_i)$$

Splitting Criteria based on Classification Error

- ▶ Classification error at a node t :

$$Error(t) = 1 - \max_i P(i | t)$$

- ▶ Measures misclassification error made by a node.
 - ▶ Maximum ($1 - 1/n_c$) when records are equally distributed among all classes, implying least interesting information
 - ▶ Minimum (0.0) when all records belong to one class, implying most interesting information

Examples for Computing Error: three potential attributes checked for error rate at a given node while classifying records in to two classes.

$$\text{Error}(t) = 1 - \max_i P(i | t)$$

C1	0
C2	6

$$P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1$$

$$\text{Error} = 1 - \max(0, 1) = 1 - 1 = 0$$

C1	1
C2	5

$$P(C1) = 1/6 \quad P(C2) = 5/6$$

$$\text{Error} = 1 - \max(1/6, 5/6) = 1 - 5/6 = 1/6$$

C1	2
C2	4

$$P(C1) = 2/6 \quad P(C2) = 4/6$$

$$\text{Error} = 1 - \max(2/6, 4/6) = 1 - 4/6 = 1/3$$

Stopping Criteria for Tree Induction

- Stop expanding a node when all the records belong to the same class
 - Stop expanding a node when all the records have similar attribute values
- There are no remaining attributes for further partitioning – **majority voting** is employed for classifying the leaf
- Stop expanding a node when there are no samples (instances) left.

Some core points (from previous discussions)

- ▶ Classification
 - ▶ Predicting categorical class labels
 - ▶ Through model building(training) and usage (testing)
 - ▶ Decision tree is one technique (with different algorithms like Hunt's, ID3 C4.5, CART...)
 - ▶ Splitting a tree at a given node (using attribute)
 - ▶ How to specify splitting conditions (attribute values)?
 - ▶ How to determine the best splits?
(information gain, Error rate, Gini)
 - ▶ When to stop splitting?
 - ▶ Evaluation and other issues

Practical Issues of Classification

Underfitting and Overfitting

Underfitting

Underfitting means that the model performs poorly on new examples as it is too simplistic to distinguish between them (i.e. has not picked up the important patterns from the training examples)

Underfitting: when model is too simple, both training and test errors are large

Overfitting

Overfitting means that the model performs poorly on new examples (e.g. testing examples) as it is too highly trained to the specific (non-general) nuances of the training examples.

- Insufficient number of training records causes the decision tree to predict the test examples using other training records that are irrelevant to the classification task

Cont....

- ▶ Underfitting
 - ▶ Using too few components/attributes
 - ▶ The model is not large enough to capture important variability in data
 - ▶ Towards more model error
- ▶ Overfitting
 - ▶ Using too many components /attributes
 - ▶ Prediction will be data dependent
 - ▶ Towards more estimation error.

Cont...

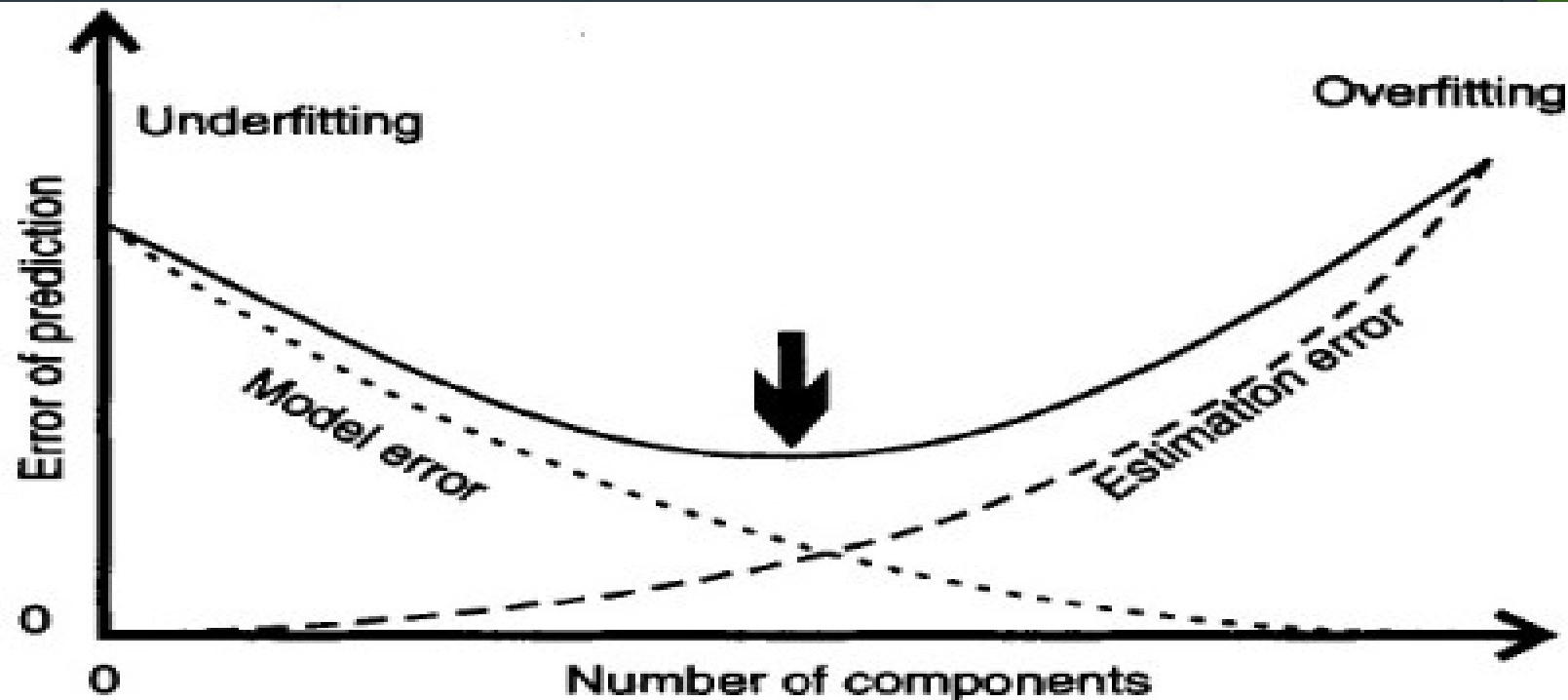


Figure 4.3. The estimation error (---) and model error (···) contribute to the prediction error (——). If too large a model is used, an overfitted solution is the result. The opposite is called underfitting. Usually a medium size model is to be preferred. Reproduced with permission from H. Martens and T. Næs, *Multivariate Calibration* (1989). © John Wiley & Sons Limited.

How to Address Overfitting

- In case of decision tree, the generated tree by the stated approach may overfit the training data
- Which means small samples may tend to generate a rule which creates problems when it comes to the real world
- This results in
 - Too many branches, some may reflect anomalies due to noise or outliers
 - Result is in poor accuracy for unseen samples
- There are two approaches to avoid over fitting, pre-pruning and post -pruning

Cont...

- ▶ Pre-Pruning (Early Stopping Rule)
 - ▶ Stop the algorithm before it becomes a fully-grown tree
 - ▶ Typical stopping conditions for a node:
 - ▶ Stop if all instances belong to the same class
 - ▶ Stop if all the attribute values are the same
 - ▶ Stop if number of instances is less than some user-specified threshold
 - ▶ Stop if expanding the current node does not improve impurity measures (e.g., Gini or information gain).

Cont...

- ▶ Post-pruning: Remove branches from a “fully grown” tree
 - ▶ A subtree at a given node is pruned by removing its branches and replacing it with a leaf.
 - ▶ The leaf is labeled with the most frequent class among the subtree being replaced.
 - ▶ Use a set of data different from the training and testing data called validation data to decide which is the “best pruned tree”
- ▶ What about a solution for Underfitting?

Metrics for Performance Evaluation

- ▶ **Confusion Matrix and Cost Matrix**
- ▶ Confusion Matrix (classification matrix)
 - ▶ Focus on the predictive capability of a model rather than how fast it takes to classify or build models, scalability, etc.

		PREDICTED CLASS	
		Class=Y es	Class=N o
ACTUAL CLASS	Class=Y es	a	b
	Class=N o	c	d

a: TP (true positive)

b: FN (false
negative)

c: FP (false positive)

d: TN (true negative)

Cont...

- ▶ A confusion matrix displays the number of correct and incorrect predictions made by the model compared with the actual classifications in the test data.
- ▶ The matrix is n -by- n , where n is the number of classes.
 - ▶ Allows the computation of
 - ▶ Accuracy
 - ▶ Error rate

Accuracy and error rate

Counts of test records that are correctly (or incorrectly) predicted by the classification model

$$\text{Accuracy} = \frac{\text{\#correct predictions}}{\text{total \# of predictions}}$$

$$\text{Error rate} = \frac{\text{\#wrong predictions}}{\text{total \# of predictions}}$$

	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

- Most widely-used metric:

$$\text{Accuracy} = \frac{a + d}{a + b + c + d} = \frac{TP + TN}{TP + TN + FP + FN}$$

[Figure 5-1](#) shows a confusion matrix for a binary classification model. The rows present the number of actual classifications in the test data. The columns present the number of predicted classifications made by the model.

Figure 5-1 Confusion Matrix for a Binary Classification Model

		PREDICTED CLASS	
		affinity_card = 1	affinity_card = 0
ACTUAL CLASS	affinity_card = 1	516	25
	affinity_card = 0	10	725

Cont...

Description of "Figure 5-1 Confusion Matrix for a Binary Classification Model"

In this example, the model correctly predicted the positive class for `affinity_card` 516 times and incorrectly predicted it 25 times. The model correctly predicted the negative class for `affinity_card` 725 times and incorrectly predicted it 10 times. The following can be computed from this confusion matrix:

- The model made 1241 **correct predictions** ($516 + 725$).
- The model made 35 **incorrect predictions** ($25 + 10$).
- There are 1276 **total scored cases** ($516 + 25 + 10 + 725$).
- The **error rate** is $35/1276 = 0.0274$.
- The **overall accuracy rate** is $1241/1276 = 0.9725$.

Cont...

- ▶ Estimate accuracy of the model
 - ▶ The known label of test sample is compared with the classified result from the model
 - ▶ Accuracy rate is the percentage of test set samples that are correctly classified by the model
 - ▶ Test set is independent of training set, otherwise over-fitting will occur

Limitation of Accuracy

- ▶ Consider a 2-class problem
 - ▶ Number of Class 0 examples = 9990
 - ▶ Number of Class 1 examples = 10
- ▶ If model predicts everything to be class 0, accuracy is $9990/10000 = 99.9\%$
 - ▶ Accuracy is misleading because model does not detect any class 1 example

Cont...

▶ **Cost Matrix**

- ▶ A cost matrix is mechanism for influencing the decision making a of a model.
- ▶ A cost matrix can cause the model to minimize costly misclassifications.
- ▶ It can also cause the model to maximize beneficial accurate classifications.

Cont...

- ▶ A cost matrix is used to specify the relative importance of accuracy for different predictions.
- A confusion matrix is used to measure accuracy, the ratio of correct predictions to the total number of predictions.
- ▶ In most business applications, it is important to consider costs in addition to accuracy when evaluating model quality.

Cost Matrix

	PREDICTED CLASS		
	$C(i j)$	Class=Yes	Class=No
	Class=Yes	$C(\text{Yes} \text{Yes})$	$C(\text{No} \text{Yes})$
	Class=No	$C(\text{Yes} \text{No})$	$C(\text{No} \text{No})$

$C(i|j)$: Cost of misclassifying class j example as class i

Assigning Costs and Benefits

- ▶ In a cost matrix, positive numbers (costs) can be used to influence negative outcomes.
- ▶ Since negative costs are interpreted as benefits, negative numbers (benefits) can be used to influence positive outcomes.
- ▶ Suppose you have calculated that it costs your business \$1500 when you do not give an affinity card to a customer who would increase spending.
 - ▶ Using the model with the confusion matrix, each false negative (misclassification of a responder) would cost \$1500
 - ▶ Misclassifying a non-responder is less expensive to your business. You figure that each false positive (misclassification of a non-responder) would only cost \$300.

Cont...

- ▶ You want to keep these costs in mind when you design a promotion campaign. You estimate that it will cost \$10 to include a customer in the promotion.
- ▶ For this reason, you associate a benefit of \$10 with each true negative prediction, because you can simply eliminate those customers from your promotion.
- ▶ Each customer that you eliminate represents a savings of \$10.
- ▶ In your cost matrix, you would specify this benefit as -10, a negative cost.

Cont...

The ff- Figure shows how you would represent these costs and benefits in a cost matrix

Figure 5-9 Cost Matrix

		PREDICTED	
		affinity_card = 1	affinity_card = 0
ACTUAL	affinity_card = 1	0	1500
	affinity_card = 0	300	-10

Figure 5-8 Positive and Negative Predictions

		PREDICTED CLASS	
		affinity_card = 1	affinity_card = 0
ACTUAL CLASS	affinity_card = 1	516 (true positive)	25 (false negative)
	affinity_card = 0	10 (false positive)	725 (true negative)

The true and false positive rates in this confusion matrix are:

- False positive rate — $10 / (10 + 725) = .01$
- True positive rate — $516 / (516 + 25) = .95$

Computing Cost of Classification & Accuracy

Count	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

$$\text{Accuracy} = (a+d)/(a+b+c+d)$$

$$\text{Cost} = p*a + q*b + r*c + s*d$$

Cost matrix	PREDICTED CLASS		
	Class=Yes	Class=No	
	Class=Yes	p	q
ACTUAL CLASS	Class=No	r	s

Computing Cost of Classification

Cost Matrix	PREDICTED CLASS		
	C(i j)	+	-
ACTUAL CLASS	+	-1	100
	-	1	0

Model M_1	PREDICTED CLASS		
		+	-
ACTUAL CLASS	+	150	40
	-	60	250

Accuracy = 80%
Cost = 3910

Model M_2	PREDICTED CLASS		
		+	-
ACTUAL CLASS	+	250	45
	-	5	200

Accuracy = 90%
Cost = 4255

Other Cost-Sensitive Measures

$$\text{Precision}(p) = \frac{a}{a+c}$$

$$\text{Recall}(r) = \frac{a}{a+b}$$

$$\text{F-measure}(F) = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$$

- Precision is biased towards $C(\text{Yes}|\text{Yes})$ & $C(\text{Yes}|\text{No})$
- Recall is biased towards $C(\text{Yes}|\text{Yes})$ & $C(\text{No}|\text{Yes})$
- F-measure is biased towards all except $C(\text{No}|\text{No})$

$$\text{Weighted Accuracy} = \frac{w_1 a + w_4 d}{w_1 a + w_2 b + w_3 c + w_4 d}$$

Exercise

1. Calculate error rates and use cost sensitive measures precision, recall, and F-measure to evaluate the performance of the following two classification models.
2. Based on the result describe the models along with their accuracy calculated in the previous slides.

Model M_1	PREDICTED CLASS		
ACTUAL CLASS		+	-
	+	150	40
	-	60	250

Model M_2	PREDICTED CLASS		
ACTUAL CLASS		+	-
	+	250	45
	-	5	200

Methods for Performance Evaluation

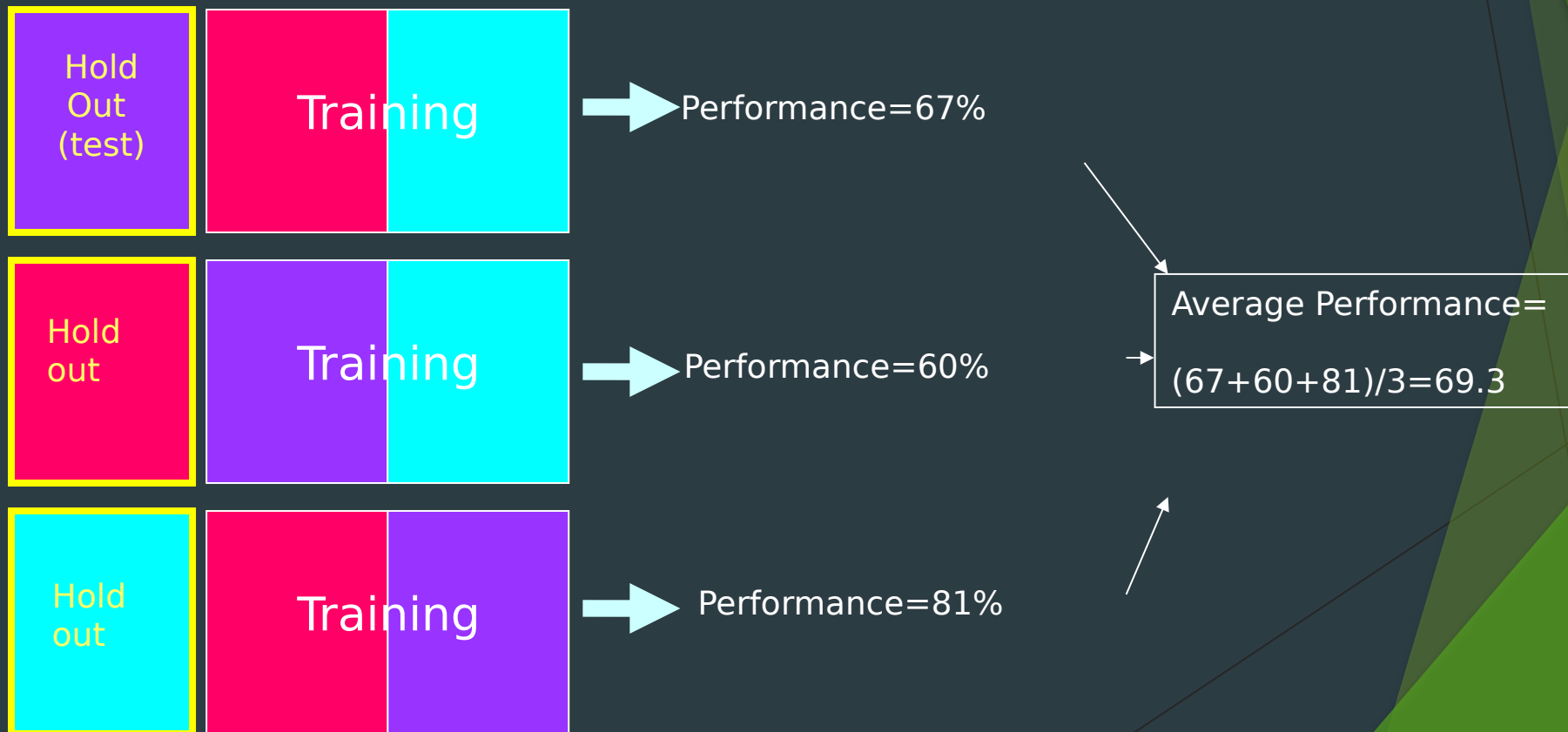
- ▶ How to obtain a reliable estimate of performance?
- ▶ Performance of a model may depend on other factors besides the learning algorithm:
 - ▶ Class distribution
 - ▶ Cost of misclassification
 - ▶ Size of training and test sets
 - ▶ Test options

Two most common method of estimation (test options)

- **Partition:** Training-and-testing (hold out method, random sub sampling) (used for data set with large number of samples)
 - ▶ use two independent data sets, e.g., training set (2/3), test set(1/3)
 - ▶ Either as a separate test file or in a percentage split
- **Cross-validation (N-fold)** (for data set with moderate and small size)
 - ▶ Do repeated experiment of model building and testing. Then average the performance measures resulting from these experiments.
 - ▶ divide the data set into k subsamples (Randomly partition data into N folds/partitions)
 - ▶ use $k-1$ subsamples as training data and one sub-sample as test data --- k -fold cross-validation (In turn, hold out one partition, train model on remaining data, and test its performance on the hold out set.)

Example: 3-Fold Cross-Validation

- Data is partitioned into 3 sets
- 3 experiments: Each with a different holdout set for testing



Test of Significance

- ▶ Given two models:
 - ▶ Model M1: accuracy = 85%, tested on 30 instances
 - ▶ Model M2: accuracy = 75%, tested on 5000 instances
- ▶ Can we say M1 is better than M2?
 - ▶ How much confidence can we place on accuracy of M1 and M2?
 - ▶ Can the difference in performance measure be explained as a result of random fluctuations in the test set?

Rule based classification

Rule based classification

- ▶ Rules are a good way of representing information or bits of knowledge.
- ▶ A rule-based classifier uses a set of IF-THEN rules for classification.
- ▶ An IF-THEN rule is an expression of the form
IF condition THEN conclusion
- ▶ An example is rule R1,
R1: IF age = youth AND student = yes THEN buys computer = yes.
- ▶ The “IF”-part (or left-hand side) of a rule is known as the ***rule antecedent or precondition***.
- ▶ The “THEN”-part (or right-hand side) is the ***rule consequent***.

Rule based classification

- ▶ Rules can be extracted from
 - ▶ **Decision tree** (Discussed with decision tree learning algorithm)
 - ▶ **Sequential covering**
 - ▶ Rule induction method in which the rules are learning sequential (one at a time)
 - ▶ There are several approach in sequential covering some of them are AQ, CN2, **PRISM** and RIPPER.
 - ▶ Each of them work with the same general principle and every approach tries to learn the best rule for a class first and continue to the more specific.

How does sequential covering algorithms work?

- ▶ In contrast to decision tree learning algorithms such as ID3 and C4.5 which proceed by repeatedly splitting the dataset based on the most promising attribute at each stage,
 - ▶ sequential covering algorithms proceed by repeatedly removing a portion of the dataset consisting of all instances covered by the most promising rule at each stage.
- ▶ we will see a **covering algorithm** known as **PRISM**.

Rule based classification using sequential covering

Algorithm: Sequential covering. Learn a set of IF-THEN rules for classification.

Input:

- D , a data set class-labeled tuples;
- Att_vals , the set of all attributes and their possible values.

Output: A set of IF-THEN rules.

Method:

```
(1)  $Rule\_set = \{\}$ ; // initial set of rules learned is empty
(2) for each class  $c$  do
(3)     repeat
(4)          $Rule = \text{Learn\_One\_Rule}(D, Att\_vals, c)$ ;
(5)         remove tuples covered by  $Rule$  from  $D$ ;
(6)     until terminating condition;
(7)      $Rule\_set = Rule\_set + Rule$ ; // add new rule to rule set
(8) endfor
(9) return  $Rule\_Set$ ;
```

Rule based classification

- ▶ A rule R can be assessed by its coverage and accuracy.
- ▶ Given a tuple, X , from a class labeled data set, D , let n_{covers} be the number of tuples covered by R ; n_{correct} be the number of tuples correctly classified by R ; and $|D|$ be the number of tuples in D .
- ▶ We can define the **coverage** and **accuracy** of R as

$$\text{coverage}(R) = \frac{n_{\text{covers}}}{|D|}$$

$$\text{accuracy}(R) = \frac{n_{\text{correct}}}{n_{\text{covers}}}$$

Review Questions-1

- ❑ Describe supervised learning, classification, prediction?
- ❑ Is that possible to generate more than one tree that fits the same data? Explain?
- ❑ Explain when to stop expanding a tree?
- ❑ What is the major critics on classification accuracy as a performance measure?
- ❑ Explain N-fold cross validation?
- ❑ Explain the concepts over fitting and under fitting

Review Questions -2

- ❑ Explain how you extract rules from decision trees?
- ❑ Describe greedy approach to determine the Best Split?
- ❑ Describe accuracy and classification error.
- ❑ Explain the two steps in classification learning ?
- ❑ What is sequential covering and how does rule based classification works?
- ❑ What is the difference between confusion matrix and cost matrix?
- ❑ Is that possible to generate more than one tree that fits the same data? Explain?

The background features a dark blue-grey field on the left, transitioning into a series of overlapping, semi-transparent green and yellow-green geometric shapes on the right, creating a layered, paper-like effect.

Thank you