Classification

Classification

- Used to classify objects to one of the classes.
  - Generally the classes are categorical or discrete values (values also known as labels of the classes)
- Assumption: finite classes and knows the characteristics of each class → this would require a model be built first and use the model to classify the objects.
  - Each new object is classified (to assigned) to one of the classes already defined in the model
- Supervised learning

Classification

- Two stage process:
  - model building
  - classifying the objects whose classes are not known
- Criteria:
  - fast classification (take least amount of resources) → compactness of model
  - Typically a tree based model is attempted because a tree can be implemented as a program by means of If... THEN... ELSE construct.
  - Usually based on a set of classes that are pre-defined and a set of training samples that are used for building class models

Classification Vs. Regression

- Classification and Regression are two major types of prediction techniques
- Classification
  - Predict nominal or discrete values
- Regression
  - Predict continuous or ordered values

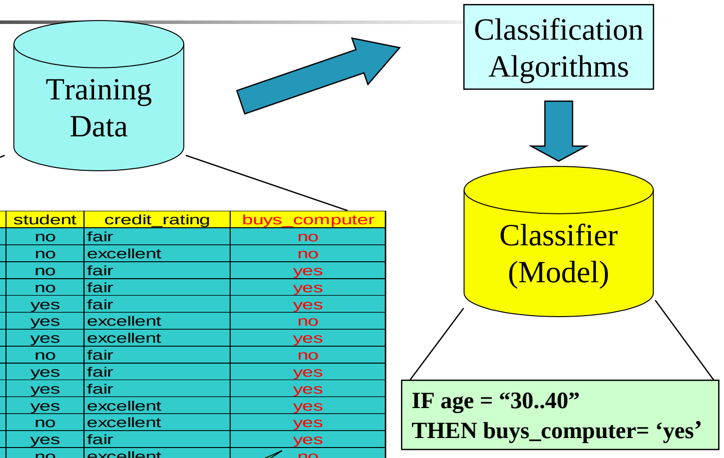
Classification

- Data is divided into two parts
  - Sample/training data -- for each data we know which class it belongs to.
    - normally the class will be an attribute of the data – called the decision/output attribute
    - E.g. Buys a computer has two values – yes or no (two classes)
  - Classification data – for which we do not know the value of the decision attribute
    - i.e. the class of the each is to be determined.

Classification

- Sample data is further divided into
  - data to build the model (typically 2/3 of the sample/training data)
  - test data (remaining sample data) to verify the validity of the model (the rest of the sample data).
- Typically the sample data will be around 10% of the total data that you want to classify.
- Also you need to know the value of the decision attribute of this sample data.

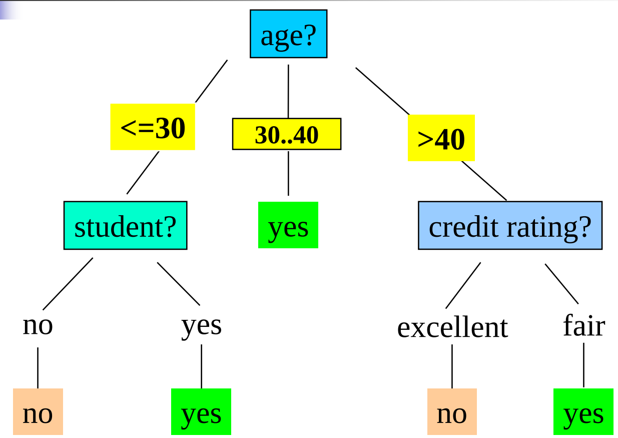
Classification Process (1): Model Construction



age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
30…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

Decision Attribute

Classification Process (1): Model Construction - **A Decision Tree for *"buys_computer"***



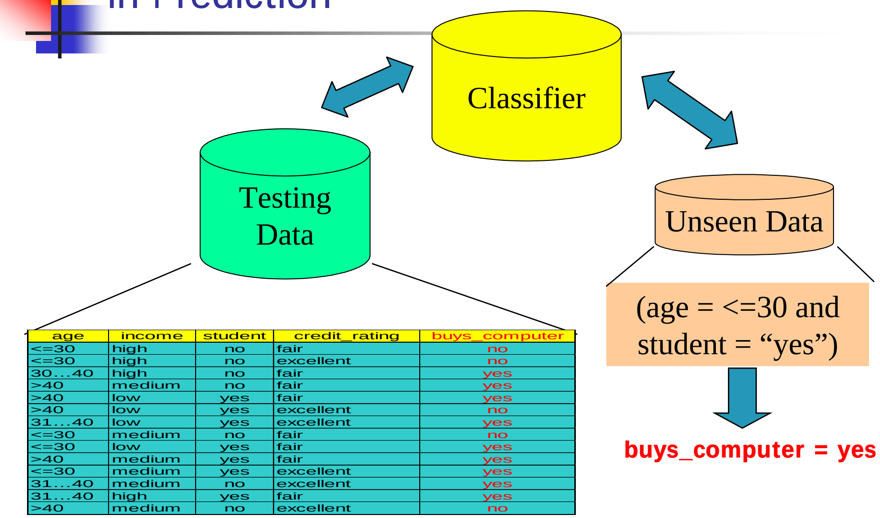
Classification process(1)– verification of the model

- Model Verification
  - Need to estimate the accuracy of the model
  - The known labels of the test data from the sample data is compared with the classified result by the model
  - Accuracy is the percentage of test set samples that are correctly classified by the model
  - Note that test data set is independent of the training data set

Classification - Stage 2

- Use the model to classify unknown objects or future data.
- Some example applications:
  - credit approval, insurance/mortgage risk, medical treatment effectiveness analysis; etc.

Classification Process (2): Use the Model in Prediction



age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
30…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

Evaluating Classification Methods

- Predictive accuracy
- Speed and scalability
  - time to construct the model
  - time to use the model
- Robustness
  - handling noise and missing values
- Scalability
  - efficiency in disk-resident databases
- Interpretability:
  - understanding and insight provided by the model
- Goodness of rules
  - decision tree size
  - compactness of classification rules

Decision Trees

- Assumption:
  - Data consists of records that have a number of input attributes and a output (decision) attribute
  - A flow-chart-like tree structure
    - Each internal node represents a test on an attribute
    - Each branch represents the outcome of a test
    - Each leaf node represents a class

Classification by Decision Tree Induction

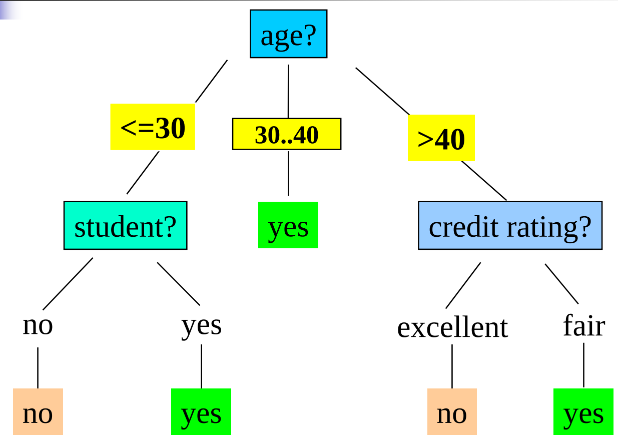
Decision tree generation consists of two phases

- Tree construction
  - At start, all the training data are at the root
  - Partition data recursively based on selected attributes until data at a node belongs to the same class.
- Tree pruning (optional)
  - Identify and remove branches that reflect noise or outliers
- Use of decision tree:
  - Verify the accuracy of the tree using the test data
  - Classifying unknown sample(s)

Training Data Set (Example from Quinlan's ID3)

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
30…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"*



Decision Trees

- Classify objects based on the values of their input attribute(s)
- Classification is based on a tree structure where each node of the tree is a test that involves a multi-way decision
- To classify an object, the attributes are compared with tests in each node starting from the root. The path found leads to a leaf node which is the class to which the objects belongs to.

Decision Trees

- Attractive due to ease of understanding
- Rules can be expressed in natural language
- Aim: Build a DT consisting of a *root* node, number of *internal nodes* and a set of (pre-defined = known classes) *leaf nodes.*
- Continuous splitting of root node until the process is complete.

Algorithm for Decision Tree Constrcution

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training data are at the root
  - Attributes are categorical (if continuous-valued, they are discretised in advance → opposite to that of clustering)
  - Samples are partitioned recursively based on selected attributes
  - Which attribute to choose for testing →Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)
- Conditions for stopping partitioning
  - All data in a given node (becomes a leaf) belong to the same class
  - There are no more remaining attributes for further partitioning – in which case majority voting is employed for classifying the leaf because all objects does not belong to the same class.
  - There are no more samples of data left to further classify (leaf node is empty).

Decision Trees

- The quality of the tree depends on the quality of the training data
  - Quality – able to correctly classify all the sample data
  - Height of the tree is minimum (why?)
  - ...
- 100% accurate for training data, but training data is only a sample...cannot be accurate for all data.

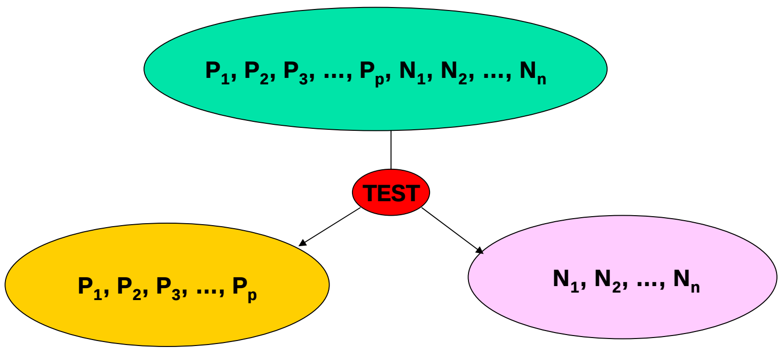
Finding the Split

- An attribute has to be chosen to split the data while forming the tree.
- This attribute has to be such that it does the best job of discriminating objects among its target classes.
- Finding which variable will affect the height of the tree is non-trivial.
- One approach – finding the data's diversity and choose that attribute that minimises the diversity amongst its children nodes.
- Several Techniques have been proposed – Information Gain/Entropy, Gini, Chi-Squared, Rough Sets etc.

How to select that attribute for splitting?

- Information gain (ID3/C4.5)
  - All attributes are assumed to be categorical
  - Can be modified for continuous-valued attributes
- Gini index (IBM IntelligentMiner)
  - All attributes are assumed continuous-valued
  - Assume there exist several possible split values for each attribute
  - May need other tools, such as clustering, to get the possible split values
  - Can be modified for categorical attributes

Finding that attribute for splitting



Finding the Split

- One approach involves finding the data's diversity (or uncertainty) and choosing a split attribute that minimises diversity amongst the children nodes or maximises the following:

diversity(before split) – (diversity(left child) + diversity(right child))

or (Finding the Split)

- Since our aim is to find nodes that belong to the same class (called *pure*), a term *impurity* is sometimes used to measure how far the node is from being pure.
- The aim of the split then is to reduce impurity:

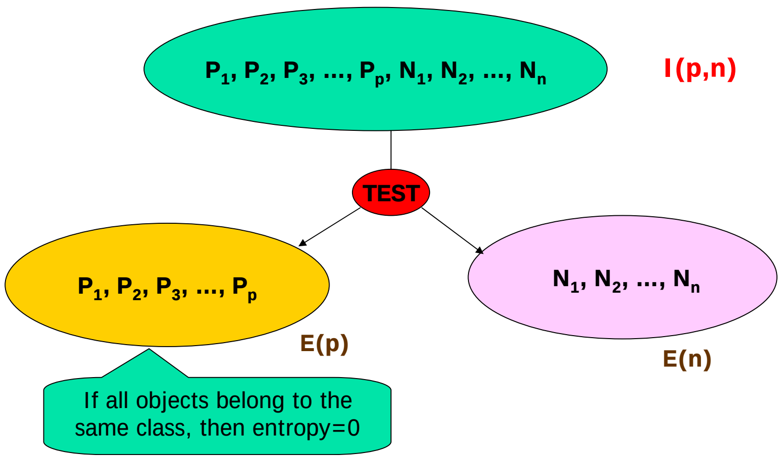
  impurity(before split) – (impurity(left child) + impurity(right child))

- Impurity is just a different term. Information theory or the Gini index may be used to find the split attribute that reduces impurity by the largest amount.

Information Theory – FIT**101** ☺

- Suppose there is a variable *s* that can take either value *a* or value *b.*
- If *s* is always going to be *a,* then there is no uncertainty and no information
  - if all the objects in a group belongs to the same class, then there is no information in the group for classification.
- The most common measure of the amount of information (also known as **entropy**) is:
  - $I = \Sigma -(p_i \log_2 (p_i))$
- Let p(a) = 0.5 and p(b) = 0.5 – tossing a coin –The value of I = 2 * (-0.5 $\log_2$ 0.5) = 1
- How many bits are required to represent the outcome of rolling the loaded dice?

Information Gain (ID3/C4.5)



Information Gain (ID3/C4.5)

- Select the attribute with the highest information gain
- Assume there are two classes, *P* and *N*
  - Let the set of examples *S* contain *p* elements of class *P* and *n* elements of class *N*
  - The amount of information, needed to decide if an arbitrary example in *S* belongs to *P* or *N* is defined as

$$I(p,n) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

Information Gain in Decision Tree Induction

- Assume that using attribute A a set *S* will be partitioned into sets $\{S_1, S_2, \ldots, S_v\}$
  - If S_i contains p_i examples of *P* and n_i examples of *N*, the entropy, or the expected information needed to classify objects in all subtrees S_i is

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

- The encoding information that would be gained by branching on *A*

$$Gain(A) = I(p,n) - E(A)$$

Attribute Selection by Information Gain Computation

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"
- I(p, n) = I(9, 5) =0.940
- Compute the entropy for *age*:

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
30…40	4	0	0
>40	3	2	0.971

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

Hence

$$Gain(age) = I(p,n) - E(age) = 0.94 - 0.69 = 0.25$$

Similarly

$$Gain(income) = 0.029$$
$$Gain(student) = 0.151$$
$$Gain(credit\_rating) = 0.048$$

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

Extracting Classification Rules from Trees

- 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*"

Revisit the Accuracy of a Classification

- A percentage of test set tuples that are correctly classified by the classifier.
- Example:
  - Consider the buying computer data set, accuracy of the classifier can be calculated based on the percentage of
    - tuples in the test data set with attribute "buys_computer = yes" are correctly classified to "buys_computer = yes"
    - PLUS
    - tuples in the test data set with attribute "buys_computer = no" are correctly classified to "buys_computer = no" .

Confusion Matrix

	Predicted				
Actual	Classes	Buy="yes"	Buy="no"	Total	Measures
	Buys="yes"	600	10	610	98.36%
	Buys="no"	40	350	390	89.74%
	Total	640	360	1000	95%

Confusion Matrix

		Predicted	
Actual		C1 (buy=y)	C2 (buy=n)
	C1 (buy=y)	True positive	False negative
	C2 (buy=n)	False positive	True negative

- Positive tuples refers to the main class of interest, eg buy="y".
- True positive refers to the positive tuples that are correctly classified.

Classification measures

	Predicted				
Actual	Classes	Buy="yes"	Buy="no"	Total	Measures
	Buy="yes"	600 (true_pos)	10 (false_neg)	610 (positive tuples)	600/601= 98.36% (sensitivity)
	Buy="no"	40 (false_pos)	350 (true_neg)	390 (negative tuples)	350/390= 89.74% (specificity)
	Total	640	360	1000	950/1000= 95% (accuracy)

$$Sensitivity = \frac{true\_pos}{total\ pos\_tuples}$$

$$Specificity = \frac{true\_neg}{total\ neg\_tuples}$$

$$Accuracy = \frac{true\_pos + true\_neg}{total\ tuples}$$

Accuracy is low...

- The training samples may not be good enough to train the model.
- Choose another samples set.
- It is possible that the data set is not good enough to use for classification.

Avoid Overfitting in Classification

- The generated tree may overfit the training data
  - Too many branches, some of the branches may be due to the noise or outliers
  - Result is in poor accuracy for unseen samples

Avoid Overfitting in Classification

- Two approaches to avoid overfitting
  - Pre-pruning: Halt tree construction early—do not split a node if this would result in the goodness measure falling below a threshold
    - Difficult to choose an appropriate threshold
  - Post-pruning: Remove branches from a "fully grown" tree—get a sequence of progressively pruned trees
    - Use a set of data different from the training data to decide which is the "best pruned tree"

Approaches to Determine the Final Tree Size

- Separate training (2/3) and testing (1/3) sets
- Use cross validation, e.g., 10-fold cross validation
- Use all the data for training
  - but apply a statistical test (e.g., chi-square) to estimate whether expanding or pruning a node may improve the entire distribution
- Use minimum description length (MDL) principle:
  - halting growth of the tree when the encoding is minimized

Decision Trees: Strengths and Weaknesses

- Strengths
  - Ability to generate understandable rules
  - Classify with minimal computational overhead
- Issues
  - Need to find the right split variable
  - Visualising Trees is tedious – particularly as data dimensionality increases
  - Handling both continuous and categorical data

Enhancements to basic decision tree induction

- Allow for continuous-valued attributes
  - Dynamically define new discrete-valued attributes that partition the continuous attribute value into a discrete set of intervals
- Handle missing attribute values
  - Assign the most common value of the attribute
  - Assign probability to each of the possible values
- Attribute construction
  - Create new attributes based on existing ones that are sparsely represented
  - This reduces fragmentation, repetition, and replication

Classification in Large Databases

- Classification—a classical problem extensively studied by statisticians and machine learning researchers
- Scalability: Classifying data sets with millions of examples and hundreds of attributes with reasonable speed
- Why decision tree induction in data mining?
  - relatively faster learning speed (than other classification methods)
  - convertible to simple and easy to understand classification rules
  - can use SQL queries for accessing databases
  - comparable classification accuracy with other methods

Other Classification Methods

- Bayesian approaches
- Neural Networks
- k-nearest neighbor classifier
- case-based reasoning
- Genetic algorithm
- Rough set approach
- Fuzzy set approaches
- ...

Summary

- Classification is a supervised learning.
- Model needs to be created using training data and tests using a test data.
- Measures of “goodness”: sensitivity, specificity and accuracy.