Rule based classification

We need labelled data to build a classifier

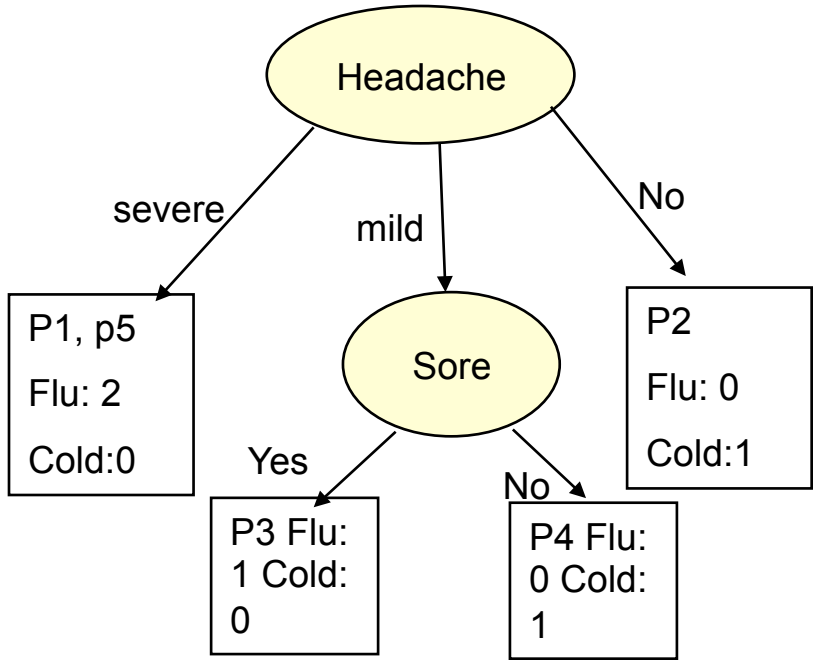
Headache	Cough	Temperature	Sore	Diagnosis
severe	mild	high	yes	Flu
no	severe	normal	yes	Cold
mild	mild	normal	yes	Flu
mild	no	normal	no	cold
severe	severe	normal	yes	Flu

Rule based classification

Patient#	Headache	Cough	Temperature	Sore	Diagnosis
p1	severe	mild	high	yes	Flu
P2	no	severe	normal	yes	Cold
P3	mild	mild	normal	yes	Flu
P4	mild	no	normal	no	cold
p5	severe	severe	normal	yes	Flu

One good approach is

- Construct a decision tree
- Extract one rule for each leaf node
- Example:
  - Rule1: if (headache = severe) then it is Flu
  - Rule2: if (headache = mild) and (Sore = yes) then it is Flu
  - Rule3: if (headache = mild) and (Sore = no) then it is Cold
  - Rule4: if (headache = no) then it is Cold



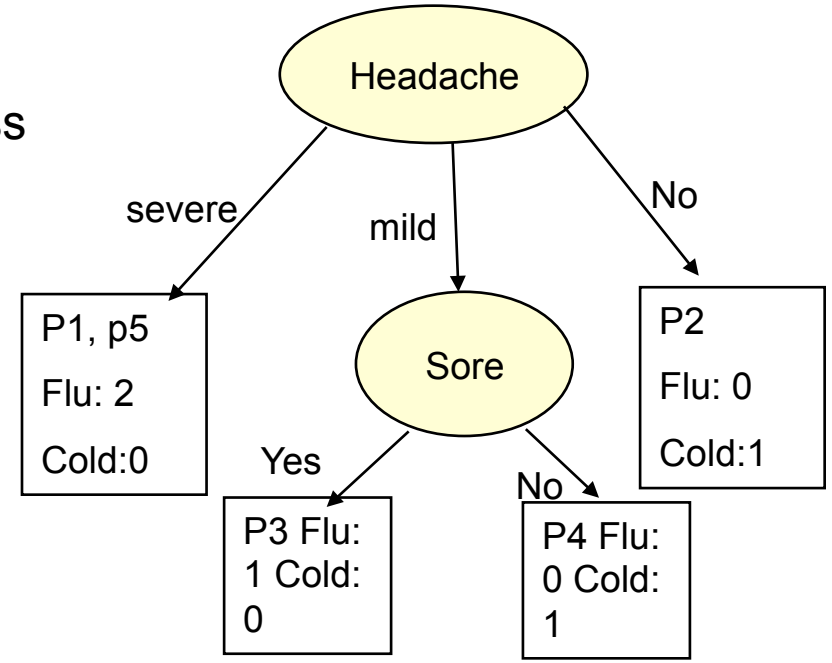
Rule based classification

Advantages of Rule synthesis based on Decision Tress are

- All rules are mutually exclusive this implies only one rule will be applicable at the time of decision making
- Easy to extract the rules from traversing the decision tree

- Disadvantages
  - Too restrictive in terms of number rules learned
  - The rules may be too specific and can be very complex

- There are many other schemes proposed in the literature and we cover them in the topic of frequent pattern mining.



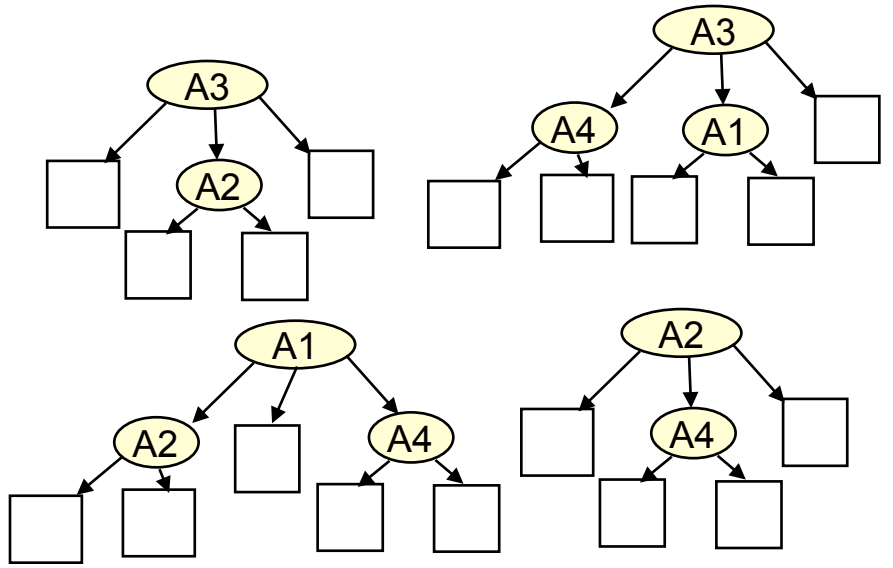
Variants of Decision Trees

Patient#	A1	A2	A3	A4	Diagnosis
p1	a11	a21	a31	a41	c1
P2	a12	a22	a32	a42	c1
P3	a13	a23	a33	a43	c2
P4	a14	a24	a34	a44	c1
p5	a15	a25	a35	a45	c2

Issue: Decision Trees cannot capture full information available in the data. This is especially true for data sets with very large number of dimensions.

Solutions:

• Bagging: Build several Decision Trees by sampling the data. For example, for a data set containing N records we sample N records randomly with replacement. On average each sample will have (2/3) of N original records. For each sample we build a decision Tree. We may build say 20-50 such trees. At the time of classification we collect all the decisions from the decision trees and apply majority rule to make the final decision. Generally this method works well.



Variants of Decision Trees

Random Forests : This method is similar to bagging in many respects but the construction of each tree is different to the standard decision tree method.

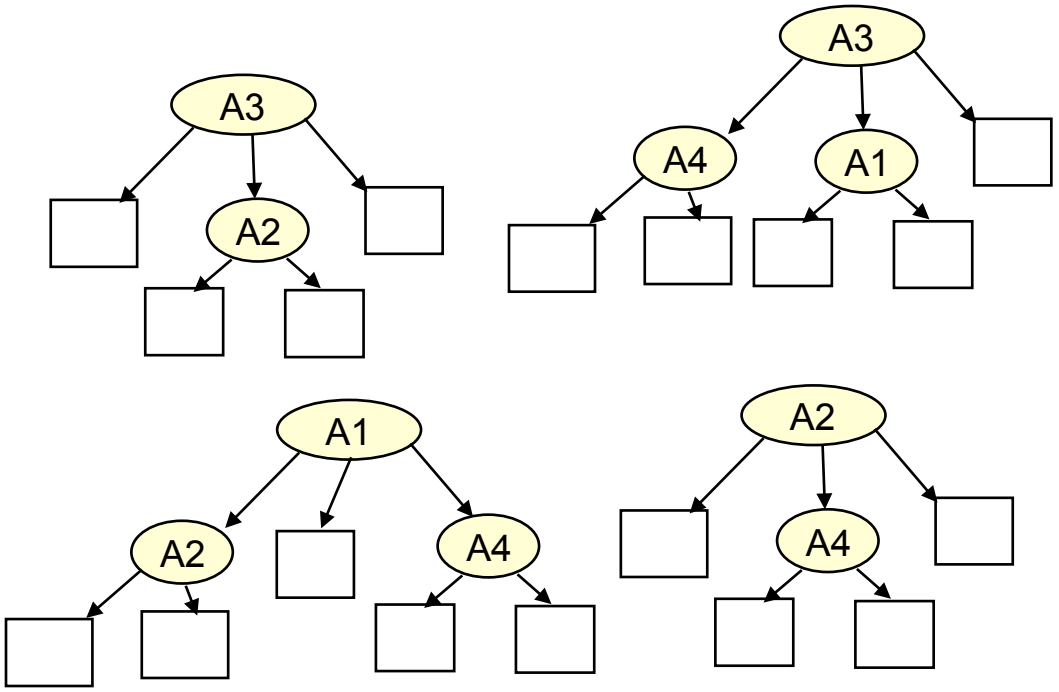
- At each node we first randomly select "m" features (attributes) out of all the features unlike the standard method which considers all the non-chosen features so far and chooses the best attribute to split the node among these attributes.
- Unlike standard Decision Tree method, there is no need to prune (to reduce the size of the tree) the constructed trees due to the randomness imposed both on the selection of data and the tree construction.
- Random Forests are shown to be one of the best classification methods experimentally.
- Random Forests can make use of large number attributes unlike standard decision trees.

Variants of Decision Trees

- The best performance is obtained for Random Forests when the trees are not correlated to each other (largely independent of each other). That is the overlap of splitting attributes between any two trees is small.
- We are able to make the trees random and less correlated:
  - by choosing randomly N records with replacement.
  - selecting the best attribute for splitting from randomly chosen "m" attributes for each splitting node.
- The quality of a decision tree depends on its accuracy which means large m value.
- The independence of each tree depends on the small values of m (smaller m means the chosen splitting variable at each node is more likely different to other trees.
- Overall quality of the Random Forest depends on both quality of the trees and their independence. This implies there must be some optimal values for m which can be determined for example experimentally.

Variants of Decision Trees

Patient#	A1	A2	A3	A4	Diagnosis
p1	a11	a21	a31	a41	c1
P2	a12	a22	a32	a42	c1
P3	a13	a23	a33	a43	c2
P4	a14	a24	a34	a44	c1
p5	a15	a25	a35	a45	c2



In the example we have 4 trees.

Each tree is largely different to each other.

This implies the decision errors are not correlated and the overall system should give us the best accuracy possible.

Practical Issues of Classification

Underfitting and Overfitting:

- For decision trees we need to make sure it does not over fit the data.
- One way to do this is to have a stopping criterion that makes sure that if the GINI index is smaller than some threshold value we stop splitting the node.
- Build the tree completely such that no further splitting is possible. That is either the node is pure (contains one class) or all the attribute values identical (data is inconsistent – records having identical feature values are labeled differently). Once the tree is built we start pruning the tree using some statistical significance test – that is improvement in accuracy is significant.

• Missing Values:

- Missing values can be predicated by using the distribution of the attribute values. Especially if we find two highly correlated attributes we can use this correlation to predict the missing value.

• Costs of Classification:

- We need to optimize the decision based on cost involved using confusion matrix

Practical Issues of Classification

Costs of Classification:

We need to optimize the decision based on cost involved using the confusion matrix

Assuming cost $Cost_{ij}$ >=0 the cost of classifying Class i as class j and f_{ij} is the frequency class I is classified as class j.

We need to build the decision system that minimizes the function:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} Cost_{ij} f_{ij}$$

This is quite difficult as probability theory cannot come in to rescue easily.

Confusion matrix		Actual class			
		C1	C2	…	Cn
Predicted class	C1	Cost11	Cost21	…	Costn1
	C2	Cost12	Cost22	…	Costn2
	…	…	…	…	
	Cn	Cost1n	Cost2n	…	Costnn

Prediction frequencies		Actual class			
		C1	C2	…	Cn
Predicted class	C1	f11	ft21	…	fn1
	C2	f12	f22	…	ftn2
	…	…	…	…	
	Cn	f1n	f2n	…	fnn