

Artificial Intelligence

Dr. Uzma Jamil

Department of Computer Science

Government College University, Faisalabad.

We have now seen one classification algorithm, and we are about to see more. How should we compare them?

- Predictive accuracy
- Speed and scalability
 - time to construct the model
 - time to use the model
 - efficiency in disk-resident databases
- Robustness
 - handling noise, missing values and irrelevant features, streaming data
- Interpretability:
 - understanding and insight provided by the model

Predictive Accuracy I

- How do we *estimate* the **accuracy** of our classifier?

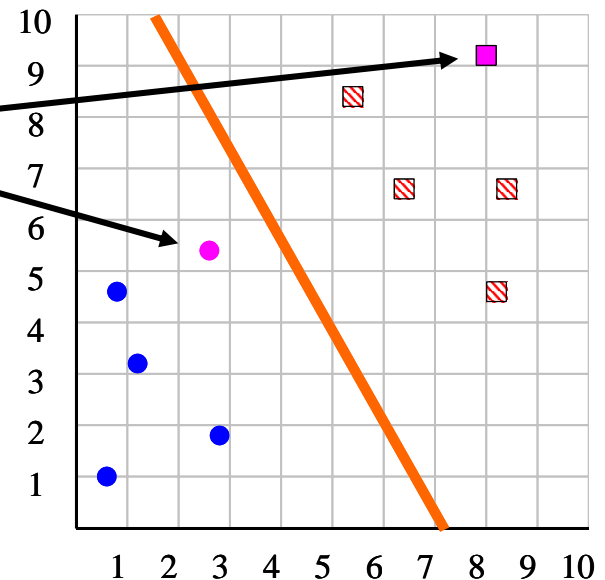
We can use ***K*-fold cross validation**

We divide the dataset into K equal sized sections. The algorithm is tested K times, each time leaving out one of the K section from building the classifier, but using it to *test* the classifier instead

$$\text{Accuracy} = \frac{\text{Number of correct classifications}}{\text{Number of instances in our database}}$$

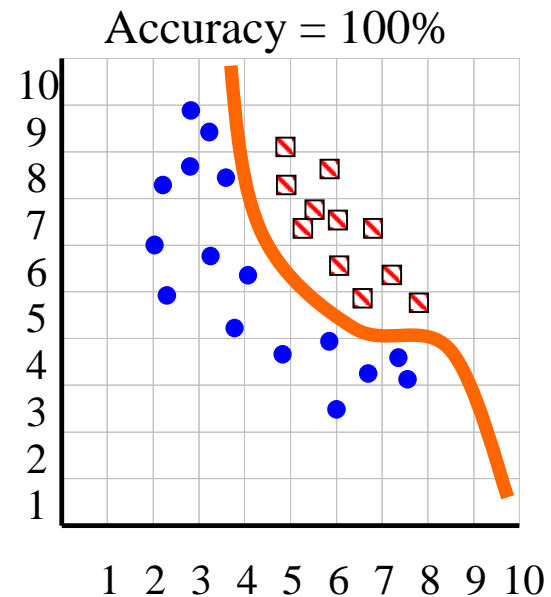
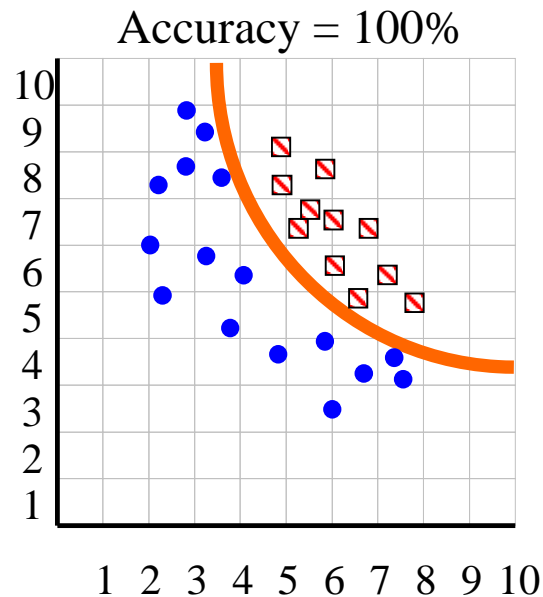
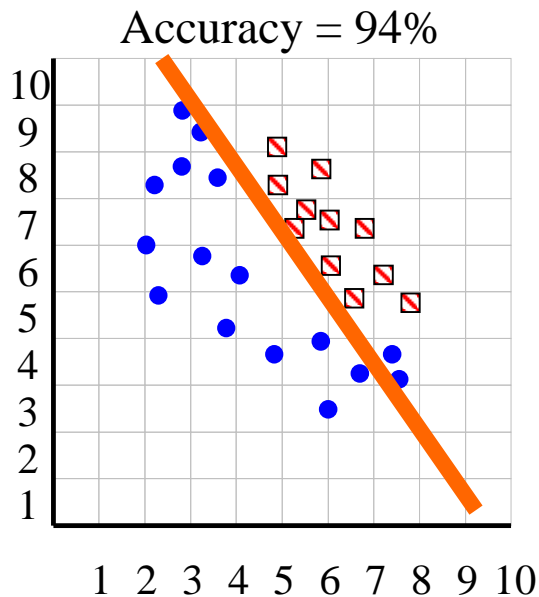
$K = 5$

	Insect ID	Abdomen Length	Antennae Length	Insect Class
	1	2.7	5.5	Grasshopper
	2	8.0	9.1	Katydid
	3	0.9	4.7	Grasshopper
	4	1.1	3.1	Grasshopper
	5	5.4	8.5	Katydid
	6	2.9	1.9	Grasshopper
	7	6.1	6.6	Katydid
	8	0.5	1.0	Grasshopper
	9	8.3	6.6	Katydid
	10	8.1	4.7	Katydid



Predictive Accuracy II

- Using K-fold cross validation is a good way to set any parameters we may need to adjust in (any) classifier.
- We can do K-fold cross validation for each possible setting, and choose the model with the highest accuracy. Where there is a tie, we choose the simpler model.
- Actually, we should probably penalize the more complex models, even if they are more accurate, since more complex models are more likely to overfit (discussed later).



Predictive Accuracy III

$$\text{Accuracy} = \frac{\text{Number of correct classifications}}{\text{Number of instances in our database}}$$

Accuracy is a single number, we may be better off looking at a **confusion matrix**. This gives us additional useful information...

True label is...

	Cat	Dog	Pig
Cat	100	0	0
Dog	9	90	1
Pig	45	45	10

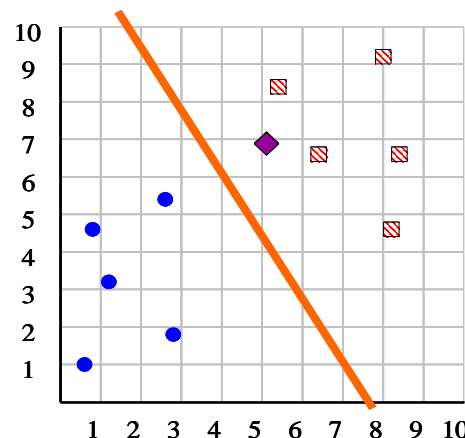
Classified as a...

Speed and Scalability I

We need to consider the time and space requirements for the two distinct phases of classification:

- Time to **construct** the classifier
 - In the case of the simpler linear classifier, the time taken to fit the line, this is linear in the number of instances.
- Time to **use** the model
 - In the case of the simpler linear classifier, the time taken to test which side of the line the unlabeled instance is. This can be done in constant time.

As we shall see, some classification algorithms are very efficient in one aspect, and very poor in the other.



Speed and Scalability II

For learning with small datasets, this is the whole picture →

However, for data mining with massive datasets, it is not so much the (main memory) time complexity that matters, rather it is how many times we have to scan the database.

This is because for most data mining operations, disk access times completely dominate the CPU times.

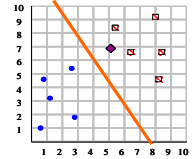
For data mining, researchers often report the number of times you must scan the database.

Speed and Scalability I

We need to consider the time and space requirements for the two distinct phases of classification:

- Time to **construct** the classifier
 - In the case of the simpler linear classifier, the time taken to fit the line, this is linear in the number of instances.
- Time to **use** the model
 - In the case of the simpler linear classifier, the time taken to test which side of the line the unlabeled instance is. This can be done in constant time.

As we shall see, some classification algorithms are very efficient in one aspect, and very poor in the other.



Robustness I

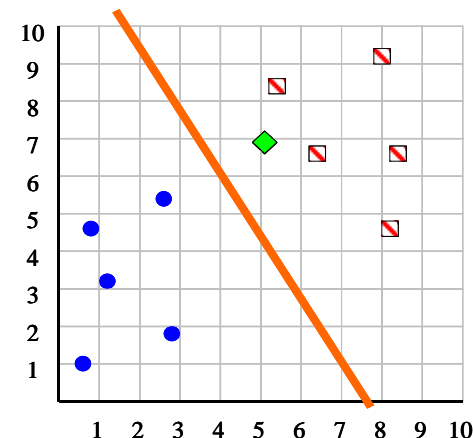
We need to consider what happens when we have:

- Noise

- For example, a persons age could have been mistyped as 650 instead of 65, how does this effect our classifier? (This is important only for building the classifier, if the instance to be classified is noisy we can do nothing).

- Missing values

- For example suppose we want to classify an insect, but we only know the abdomen length (X-axis), and not the antennae length (Y-axis), can we still classify the instance?



Robustness II

We need to consider what happens when we have:

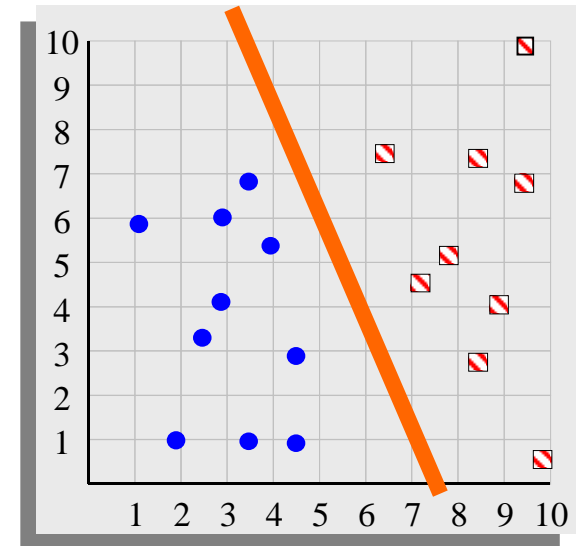
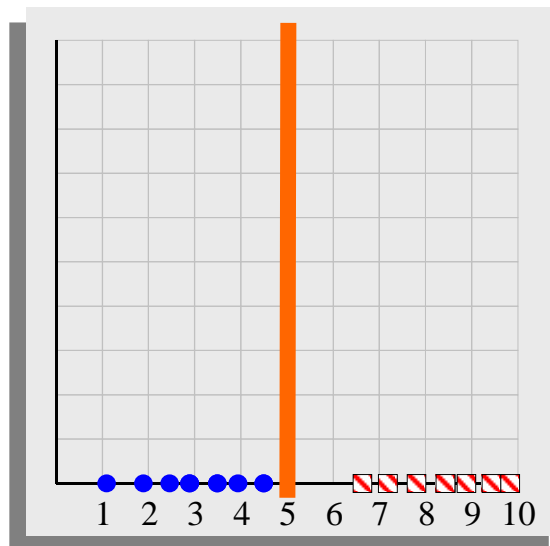
- Irrelevant features

For example, suppose we want to classify people as either

- **Suitable_Grad_Student**
- **Unsuitable_Grad_Student**

And it happens that scoring more than 5 on a particular test is a perfect indicator for this problem...

If we also use
“hair_length” as a
feature, how will this
effect our classifier?



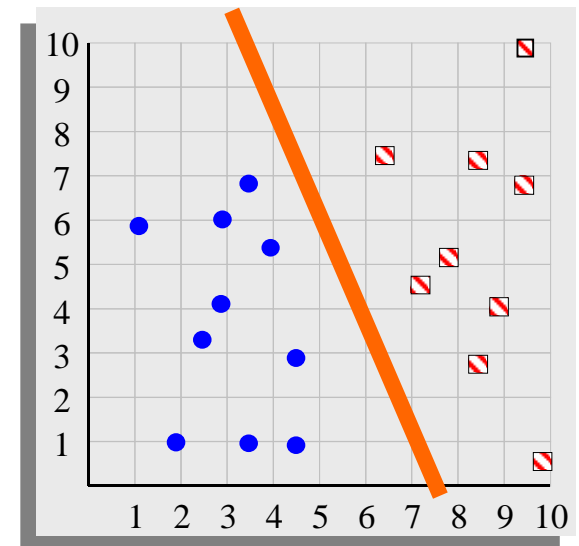
Robustness III

We need to consider what happens when we have:

- Streaming data

For many real world problems, we don't have a single fixed dataset. Instead, the data continuously arrives, potentially forever... (stock market, weather data, sensor data etc)

Can our classifier handle streaming data?

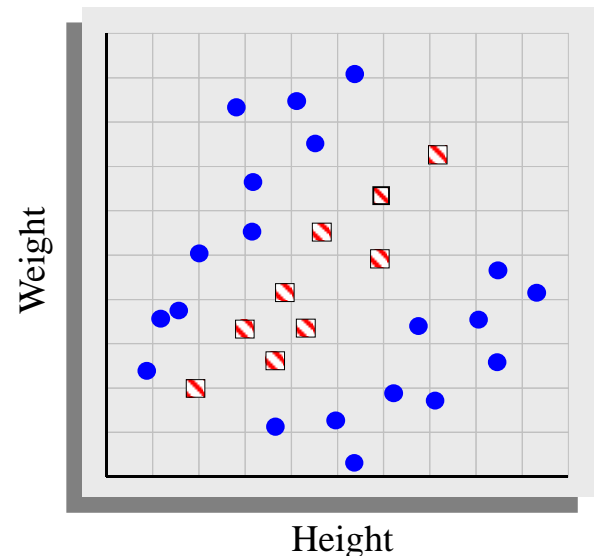


Interpretability

Some classifiers offer a *bonus* feature. The structure of the learned classifier tells use something about the domain.

As a trivial example, if we try to classify peoples health risks based on just their height and weight, we could gain the following insight (Based of the observation that a single linear classifier does not work well, but two linear classifiers do).

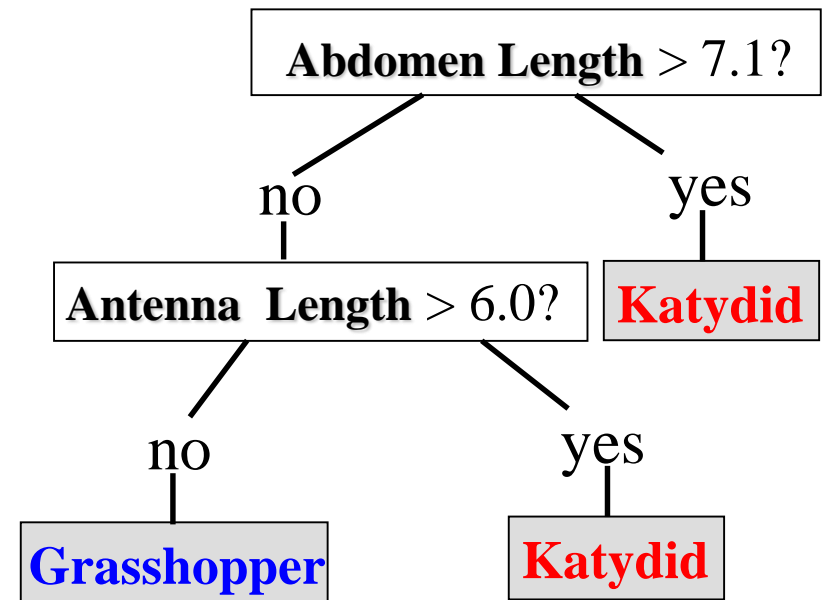
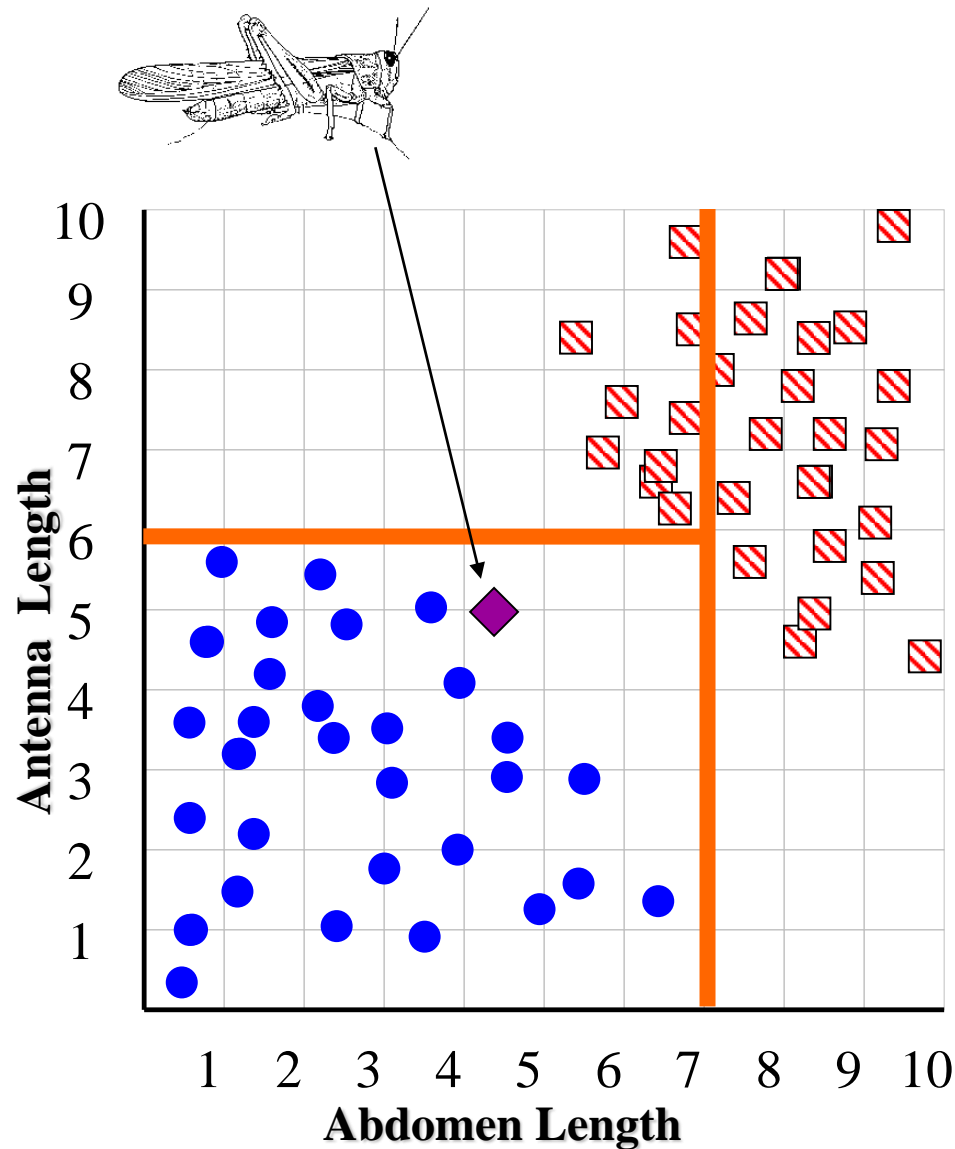
There are two ways to be unhealthy, being obese and being too skinny.



Decision Tree Classifier



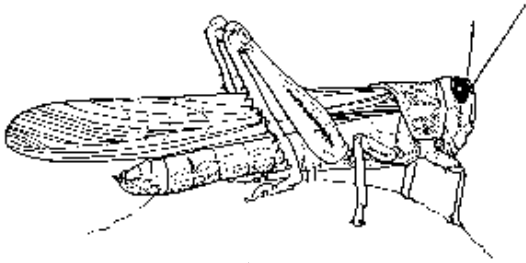
Ross Quinlan



Antennae shorter than body?

Yes

No



Grasshopper

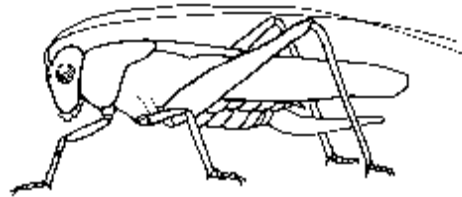
3 Tarsi?



Yes



No

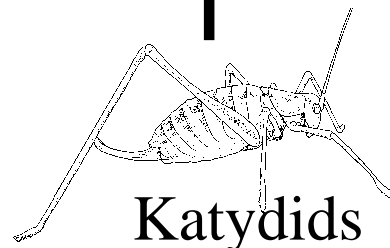


Cricket

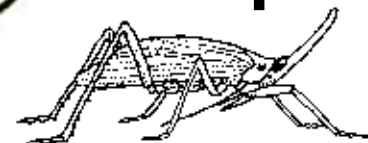
Foretibia has ears?

Yes

No



Katydids



Camel Cricket

Decision Tree Classification

- Decision tree
 - A flow-chart-like tree structure
 - Internal node denotes a test on an attribute
 - Branch represents an outcome of the test
 - Leaf nodes represent class labels or class distribution
- 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

How do we construct the decision tree?

- 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 can be discretized in advance)
 - Examples are partitioned recursively based on selected attributes.
 - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)
- Conditions for stopping partitioning
 - All samples for a given node belong to the same class
 - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
 - There are no samples left

Information Gain as A Splitting Criteria

- Select the attribute with the highest information gain (information gain is the expected reduction in entropy).
- Entropy is the quantitative measure of disorder in system (It is a measure of uncertainty associated with the random variable).
- 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

$$E(S) = -\frac{p}{p+n} \log_2 \left(\frac{p}{p+n} \right) - \frac{n}{p+n} \log_2 \left(\frac{n}{p+n} \right)$$










$0 \log(0)$ is defined as 0

Information Gain in Decision Tree Induction

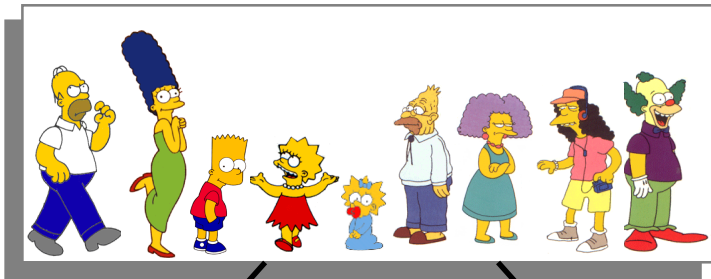
- Assume that using attribute A , a current set will be partitioned into some number of child sets
- The encoding information that would be gained by branching on A

$$Gain(A) = E(Current\ set) - \sum E(all\ child\ sets)$$

Note: entropy is at its minimum if the collection of objects is completely uniform

Person	Hair Length	Weight	Age	Class
 Homer	0"	250	36	M
 Marge	10"	150	34	F
 Bart	2"	90	10	M
 Lisa	6"	78	8	F
 Maggie	4"	20	1	F
 Abe	1"	170	70	M
 Selma	8"	160	41	F
 Otto	10"	180	38	M
 Krusty	6"	200	45	M

	Comic	8"	290	38	?
---	-------	----	-----	----	---



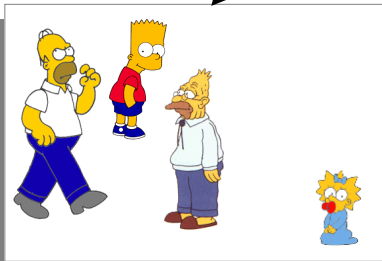
$$Entropy(S) = -\frac{p}{p+n} \log_2 \left(\frac{p}{p+n} \right) - \frac{n}{p+n} \log_2 \left(\frac{n}{p+n} \right)$$

$$Entropy(4\mathbf{F}, 5\mathbf{M}) = -(4/9) \log_2(4/9) - (5/9) \log_2(5/9) = \mathbf{0.9911}$$

yes

Hair Length <= 5?

no



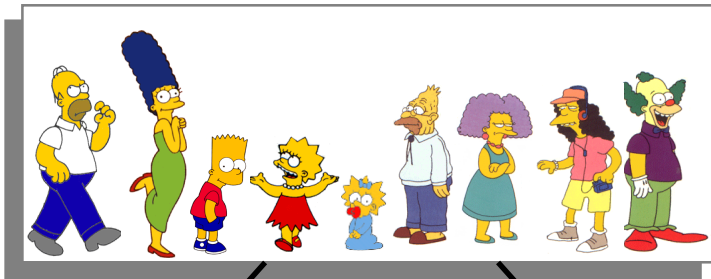
Let us try splitting
on *Hair length*

$$Entropy(1\mathbf{F}, 3\mathbf{M}) = -(1/4) \log_2(1/4) - (3/4) \log_2(3/4) = \mathbf{0.8113}$$

$$Entropy(3\mathbf{F}, 2\mathbf{M}) = -(3/5) \log_2(3/5) - (2/5) \log_2(2/5) = \mathbf{0.9710}$$

$$Gain(A) = E(\text{Current set}) - \sum E(\text{all child sets})$$

$$Gain(\text{Hair Length} \leq 5) = \mathbf{0.9911} - (4/9 * \mathbf{0.8113} + 5/9 * \mathbf{0.9710}) = \mathbf{0.0911}$$



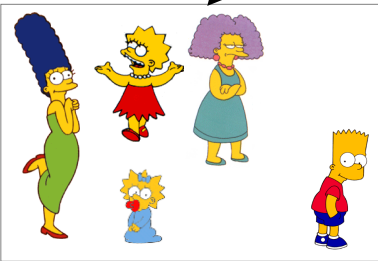
$$Entropy(S) = -\frac{p}{p+n} \log_2 \left(\frac{p}{p+n} \right) - \frac{n}{p+n} \log_2 \left(\frac{n}{p+n} \right)$$

$$Entropy(4\mathbf{F}, 5\mathbf{M}) = -(4/9) \log_2(4/9) - (5/9) \log_2(5/9) = \mathbf{0.9911}$$

yes

no

Weight ≤ 160?



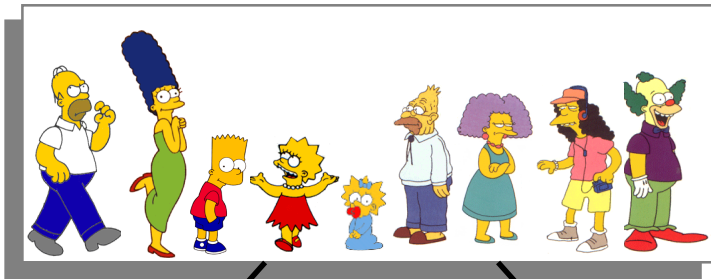
Let us try splitting
on *Weight*

$$Entropy(4\mathbf{F}, 1\mathbf{M}) = -(4/5) \log_2(4/5) - (1/5) \log_2(1/5) = \mathbf{0.7219}$$

$$Entropy(0\mathbf{F}, 4\mathbf{M}) = -(0/4) \log_2(0/4) - (4/4) \log_2(4/4) = \mathbf{0}$$

$$Gain(A) = E(\text{Current set}) - \sum E(\text{all child sets})$$

$$Gain(\text{Weight} \leq 160) = \mathbf{0.9911} - (5/9 * \mathbf{0.7219} + 4/9 * \mathbf{0}) = \mathbf{0.5900}$$



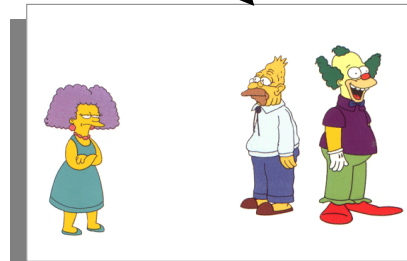
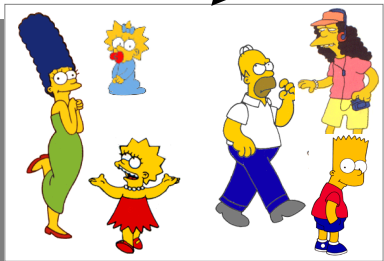
$$Entropy(S) = -\frac{p}{p+n} \log_2 \left(\frac{p}{p+n} \right) - \frac{n}{p+n} \log_2 \left(\frac{n}{p+n} \right)$$

$$Entropy(4\mathbf{F}, 5\mathbf{M}) = -(4/9) \log_2(4/9) - (5/9) \log_2(5/9) = \mathbf{0.9911}$$

yes

age <= 40?

no



Let us try splitting
on *Age*

$$Entropy(3\mathbf{F}, 3\mathbf{M}) = -(3/6) \log_2(3/6) - (3/6) \log_2(3/6) = \mathbf{1}$$

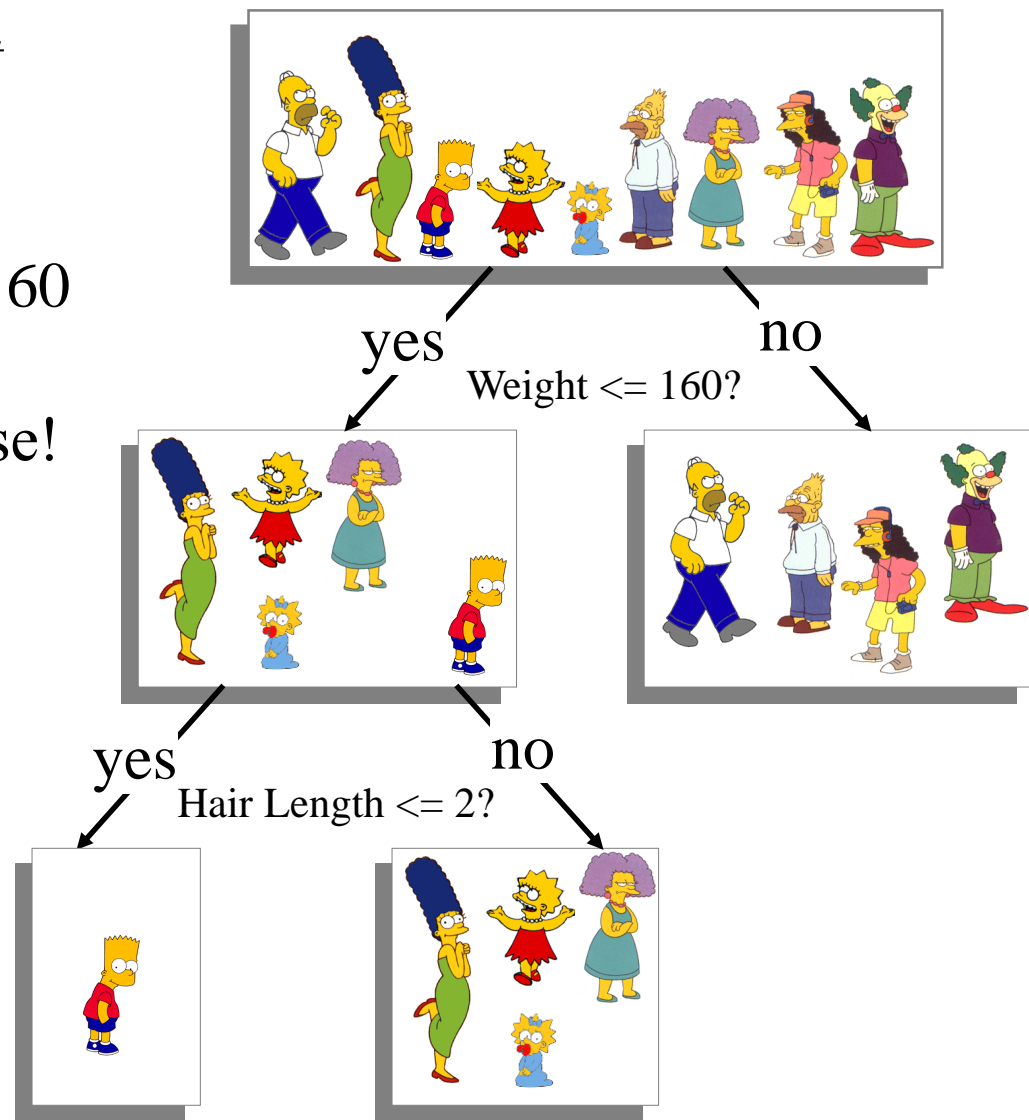
$$Entropy(1\mathbf{F}, 2\mathbf{M}) = -(1/3) \log_2(1/3) - (2/3) \log_2(2/3) = \mathbf{0.9183}$$

$$Gain(A) = E(\text{Current set}) - \sum E(\text{all child sets})$$

$$Gain(\text{Age} \leq 40) = \mathbf{0.9911} - (6/9 * \mathbf{1} + 3/9 * \mathbf{0.9183}) = \mathbf{0.0183}$$

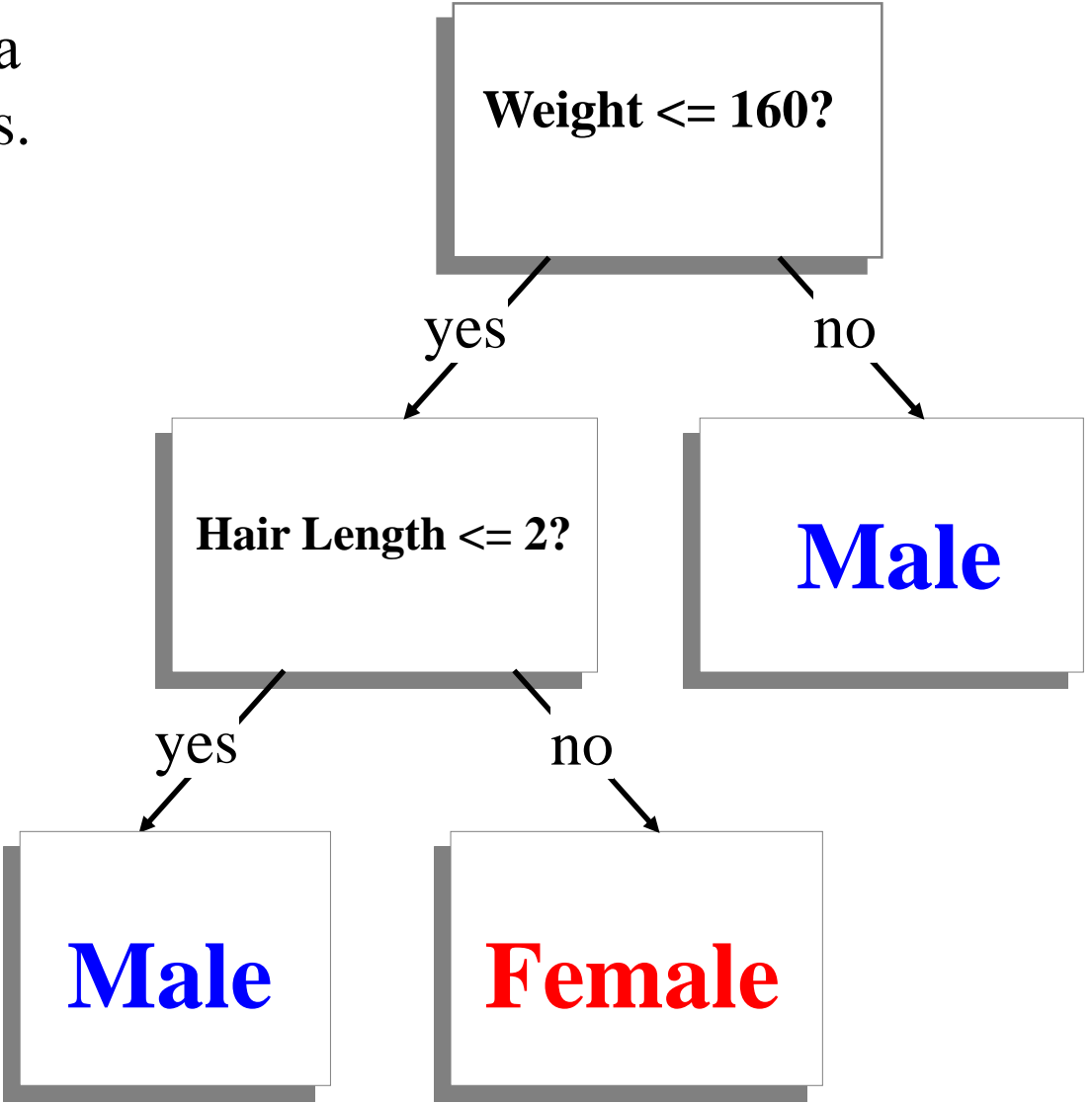
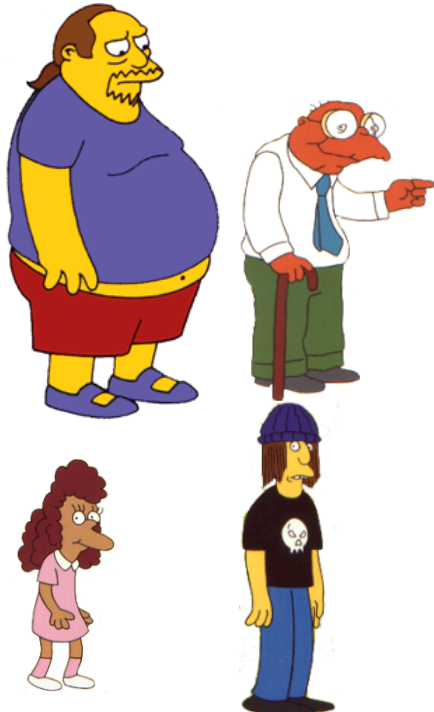
Of the 3 features we had, *Weight* was best. But while people who weigh over 160 are perfectly classified (as males), the under 160 people are not perfectly classified... So we simply recurse!

This time we find that we can split on *Hair length*, and we are done!

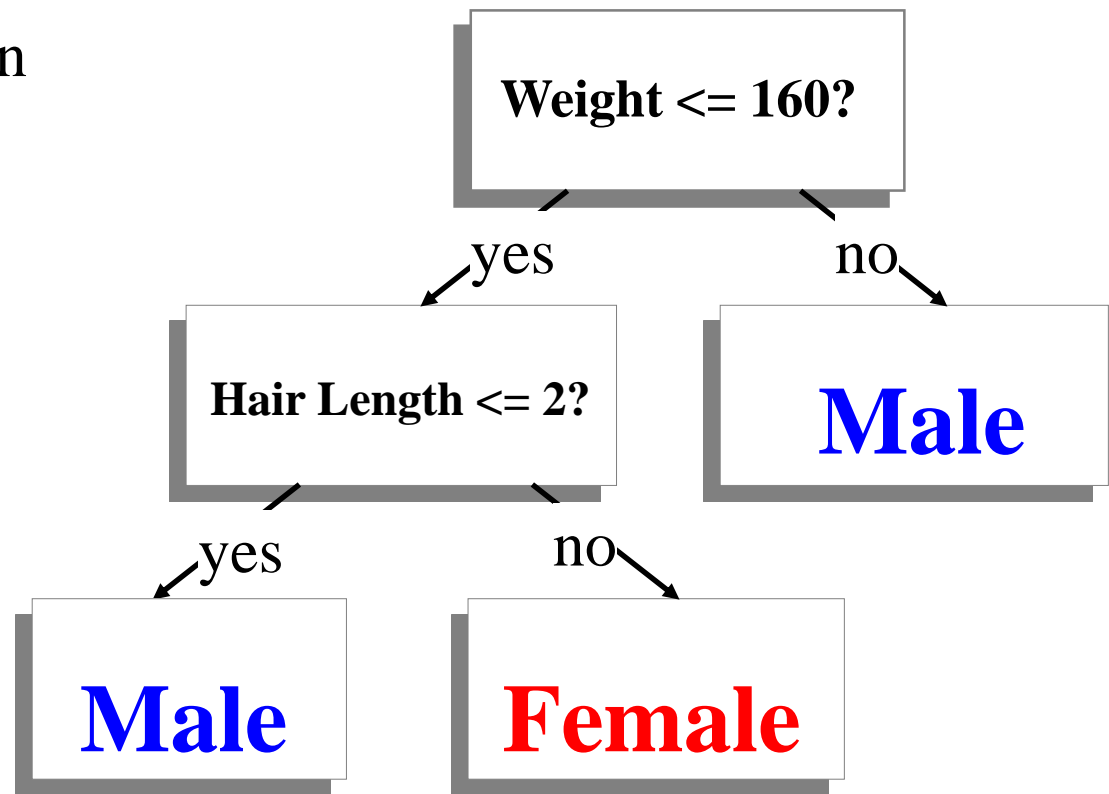


We don't need to keep the data around, just the test conditions.

How would these people be classified?



It is trivial to convert Decision Trees to rules...



Rules to Classify Males/Females

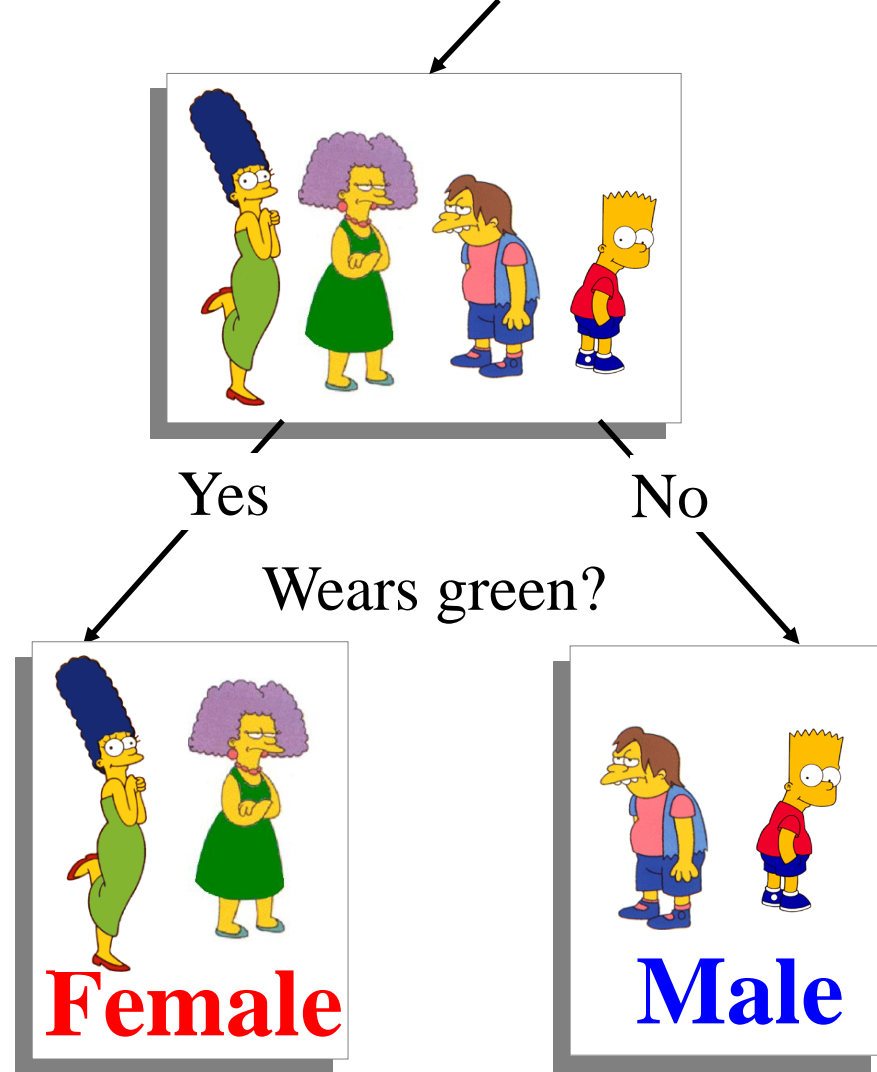
If *Weight* greater than 160, classify as **Male**

Elseif *Hair Length* less than or equal to 2, classify as **Male**

Else classify as **Female**

The worked examples we have seen were performed on small datasets. However with small datasets there is a great danger of overfitting the data...

When you have few datapoints, there are many possible splitting rules that perfectly classify the data, but will not generalize to future datasets.



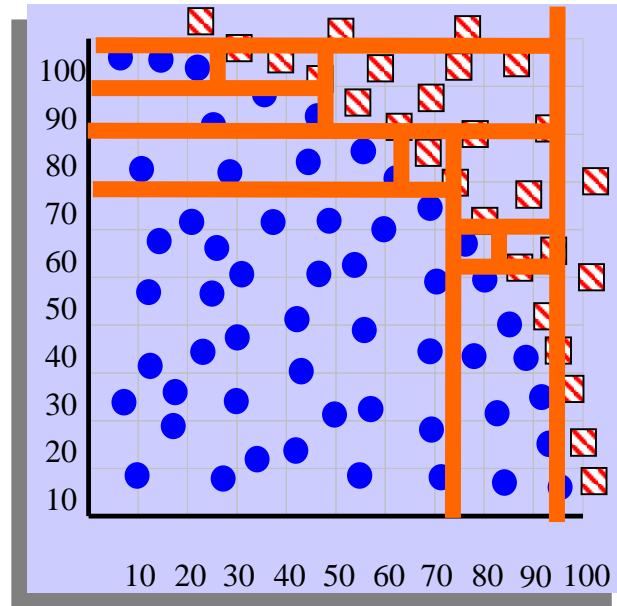
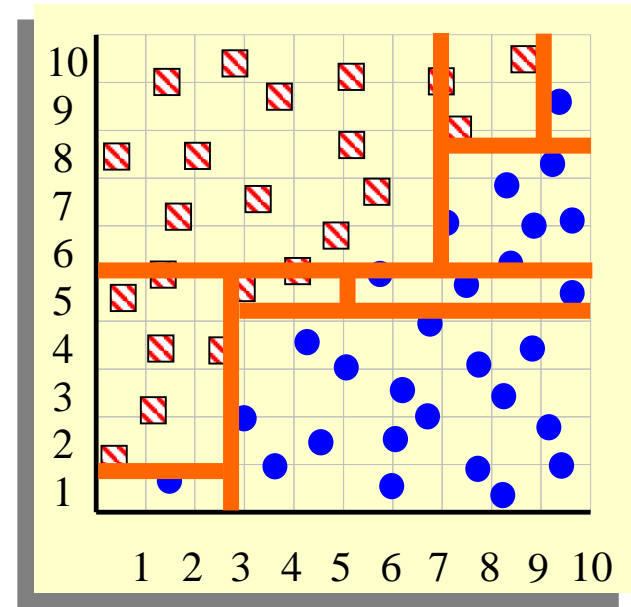
For example, the rule “Wears green?” perfectly classifies the data, so does “Mothers name is Jacqueline?”, so does “Has blue shoes”...

Avoid Overfitting in Classification

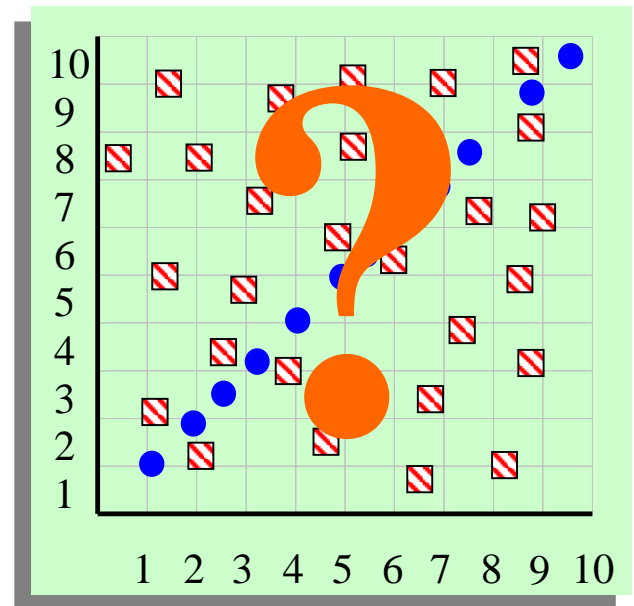
- The generated tree may overfit the training data
 - Too many branches, some may reflect anomalies due to noise or outliers
 - Result is in poor accuracy for unseen samples
- Two approaches to avoid overfitting
 - Prepruning: Halt tree construction early—do not split a node if this would result in the goodness measure falling below a threshold
 - Difficult to choose an appropriate threshold
 - Postpruning: 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”

Which of the “Pigeon Problems” can be solved by a Decision Tree?

- 1) Deep Bushy Tree
- 2) Useless
- 3) Deep Bushy Tree



The Decision Tree
has a hard time with
correlated attributes



Advantages/Disadvantages of Decision Trees

- Advantages:
 - Easy to understand (Doctors love them!)
 - Easy to generate rules
- Disadvantages:
 - May suffer from overfitting.
 - Classifies by rectangular partitioning (so does not handle correlated features very well).
 - Can be quite large – pruning is necessary.
 - Does not handle streaming data easily