Data Mining

Lecture 4: Decision Trees & k-Nearest Neighbour

Lecturer: G. Gray

Lecture Overview

Classification Algorithms

How a decision tree classifies data

- Information gain
- Pruning

How *k*-Nearest Neighbour classified data

- choosing the correct value for k

Classification Algorithms

- There are a wide range of algorithms which '**learn**' the patterns of a dataset in different ways.
- Each is suited to different type of attributes, or different types of labels. The more commonly used algorithms include:
  - Decision Tree based Methods (categorical attributes, categorical label)
  - Rule-based Methods (categorical attributes, categorical label)
  - Neural Networks (numeric attributes, categorical label)
  - Naïve Bayes and Bayesian Belief Networks (all attributes, categorical label)
  - Support Vector Machines (numeric attributes, binary class label)
  - *k*-Nearest Neighbour (all attributes, categorical label)

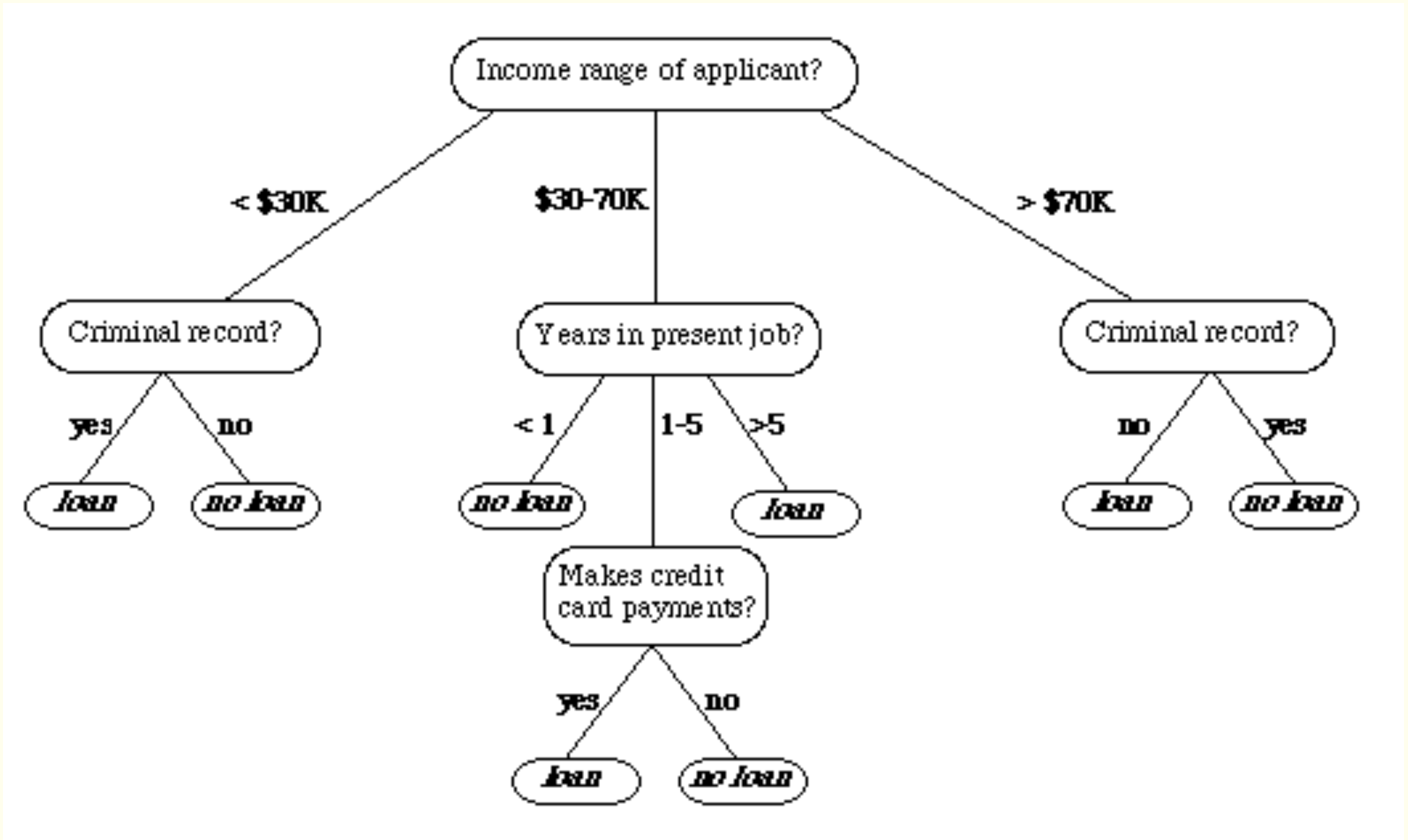
Criteria to Evaluate Classification Algorithms

1. What type of data can it handle?
2. Predictive accuracy
3. Speed and scalability
   - time to construct the model
   - time to use the model
4. Robustness
   - handling noise and missing values
5. Interpretability:
   - understanding and insight provided by the model

Data Mining Algorithms

First we will look at: **Decision Trees**

- Widely used practical method
  - Collection of decision nodes
  - Connected by branches
  - Extending downwards from root node to terminating leaf node

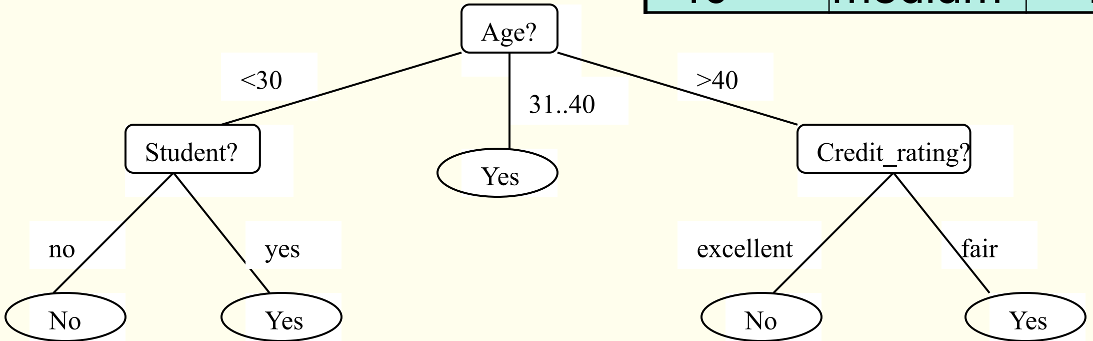


Decision trees

- A tree structure where each internal node denotes a test on an attribute.
- Each branch represents an outcome of the test,
- and each leaf node represents a class, or class distribution.

Training dataset

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

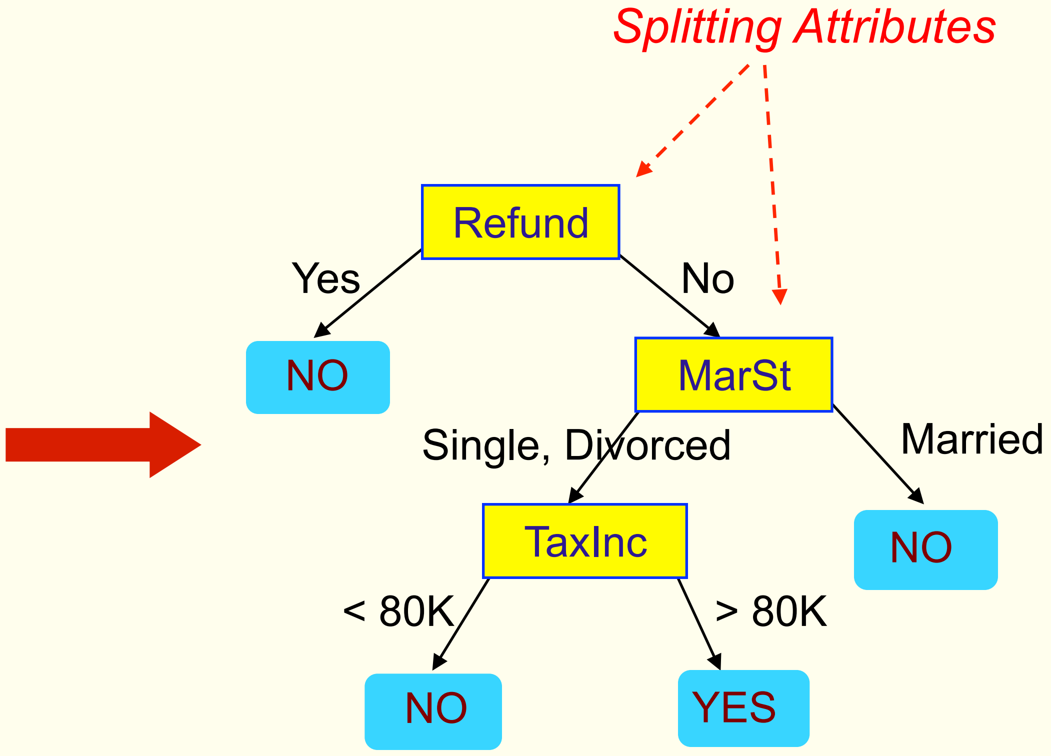


A decision tree indicating if a customer is likely to buy a computer (from Han, Kamber)

Another Example of a Decision Tree

Tid	Refund categorical	Marital Status categorical	Taxable Income continuous	Cheat Class label
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**

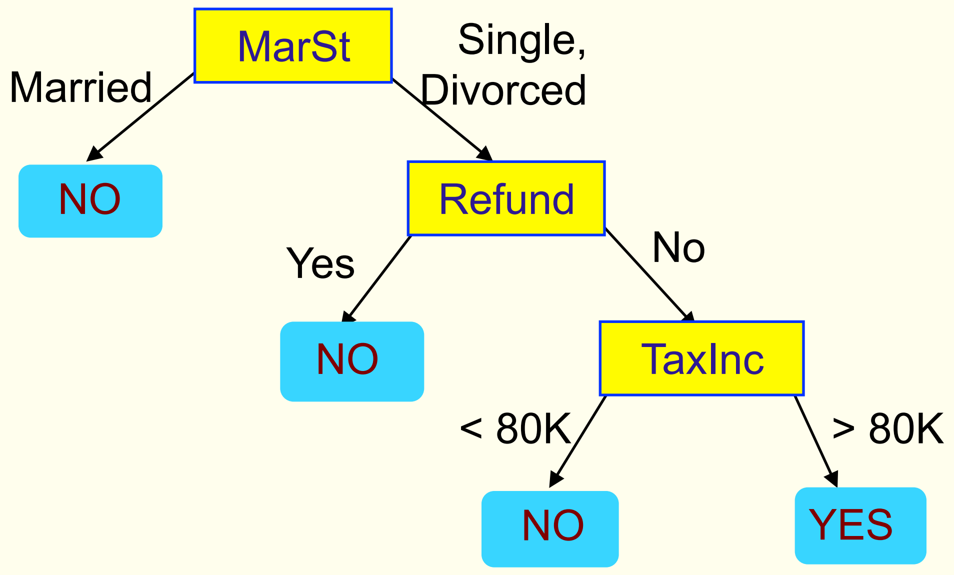
Training Data



Model: Decision Tree

Another Example of Decision Tree

Tid	**Refund** categorical	**Marital Status** categorical	**Taxable Income** continuous	**Cheat** Class label
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!

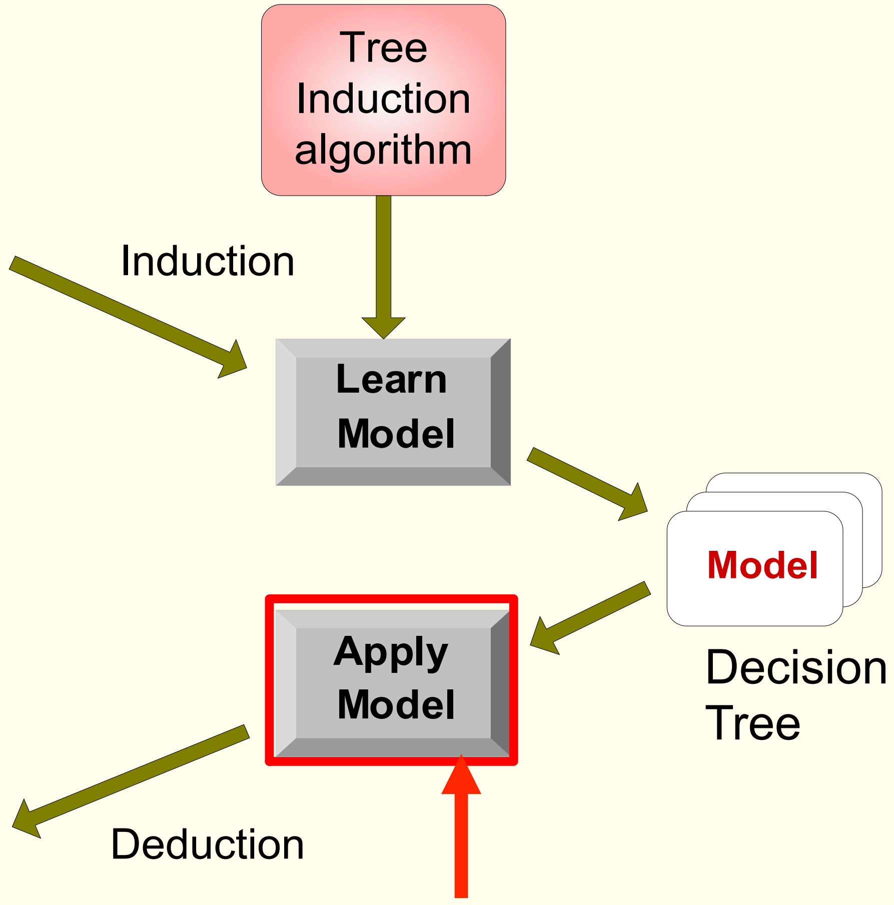
Using the tree . . .

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

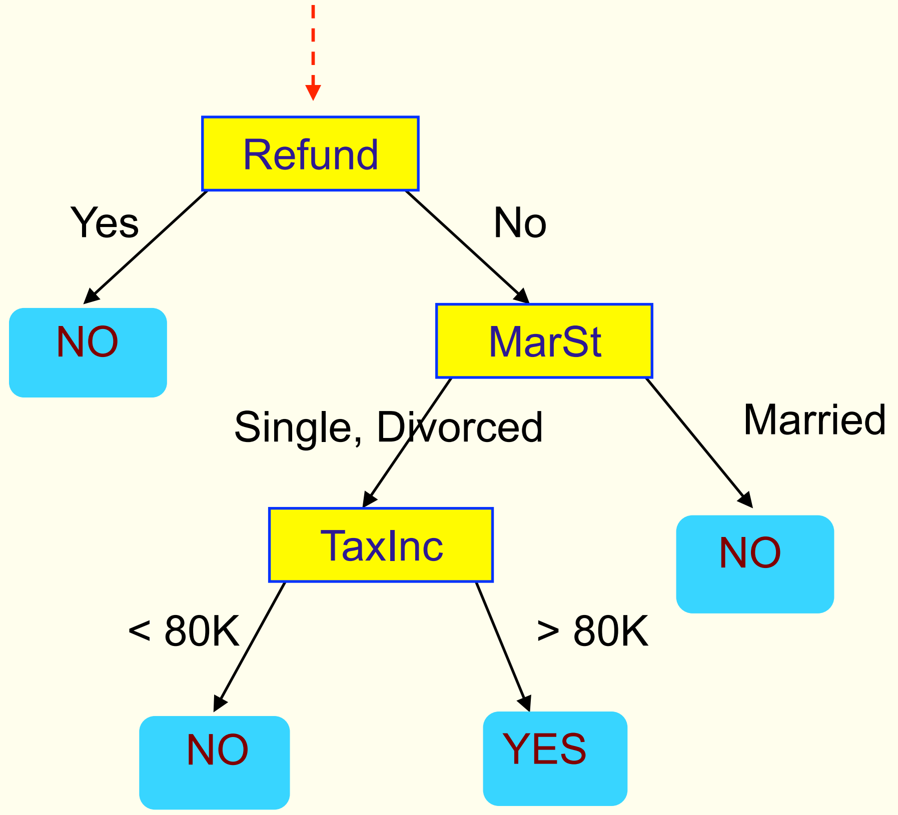
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



Apply Model to Test Data

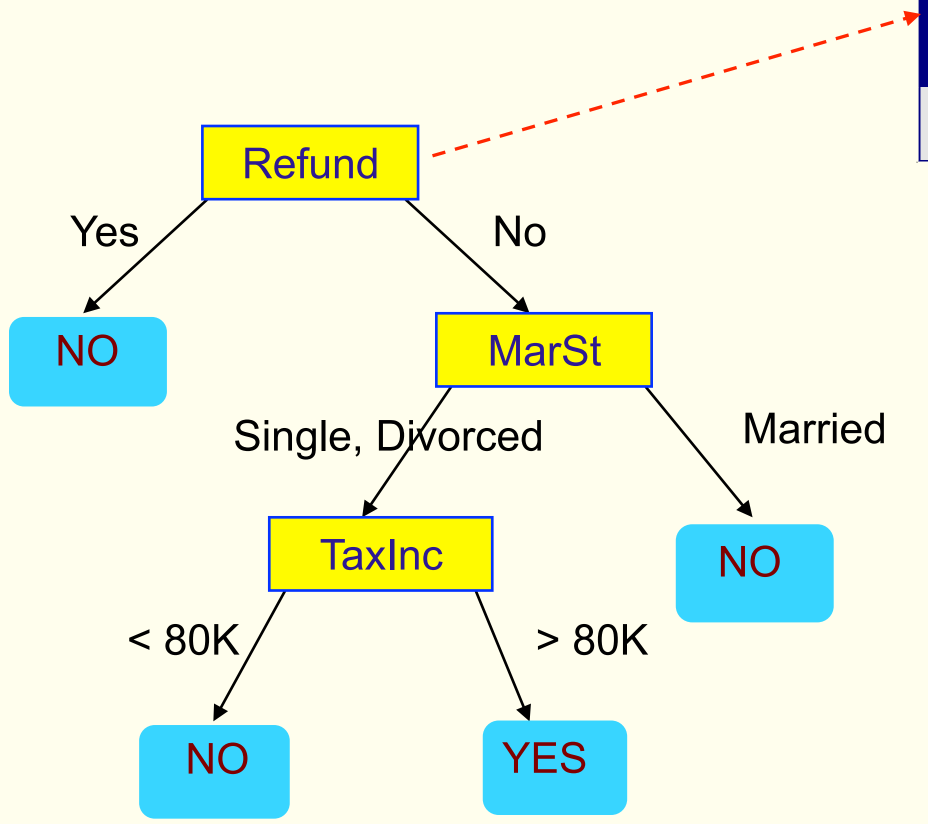
Start from the root of tree.



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

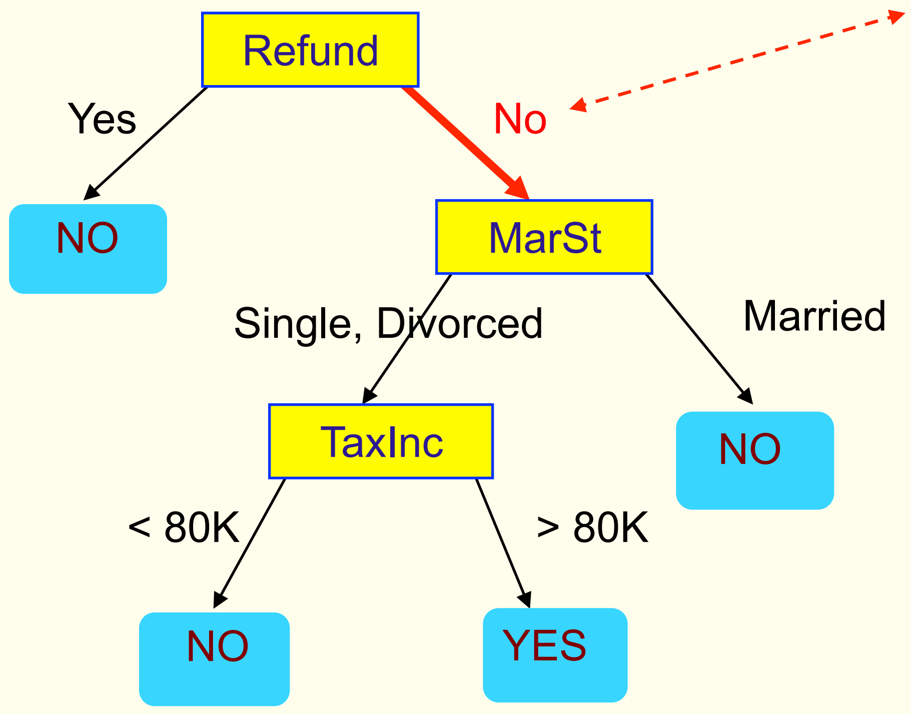
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data



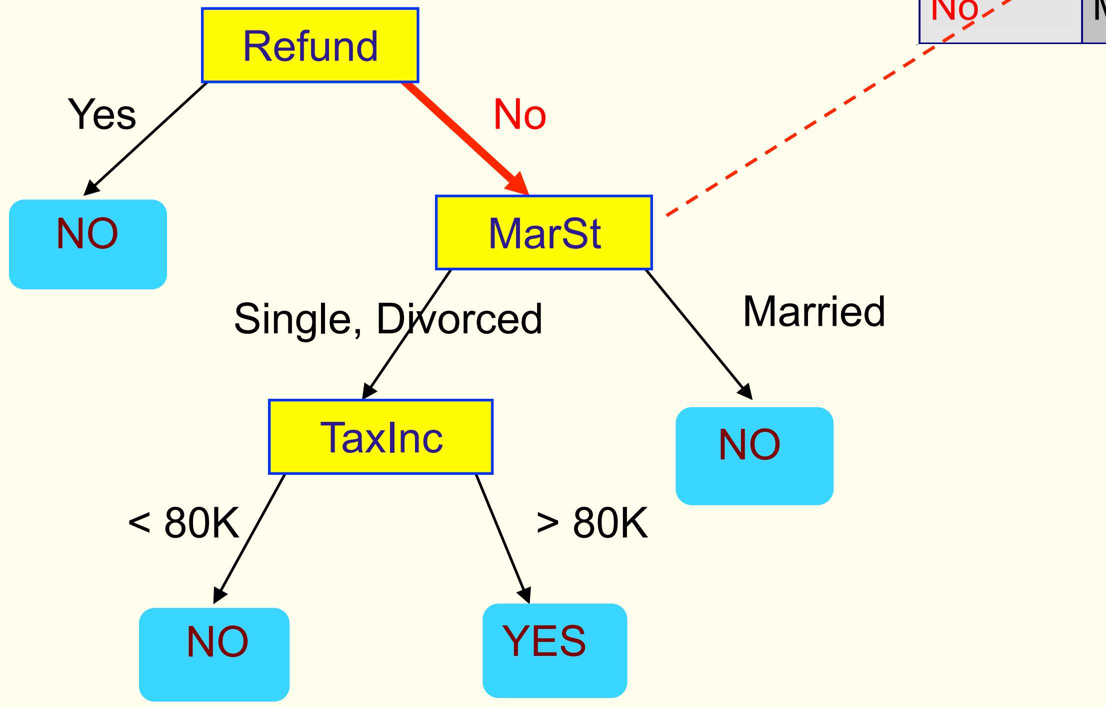
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Test Data

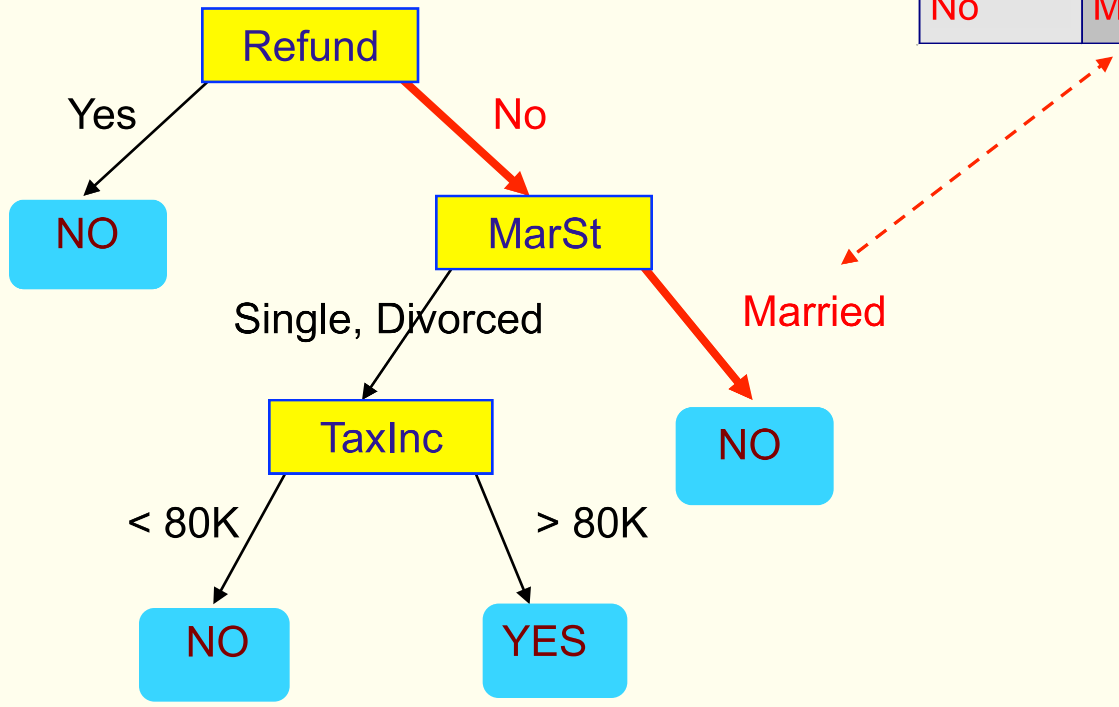
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



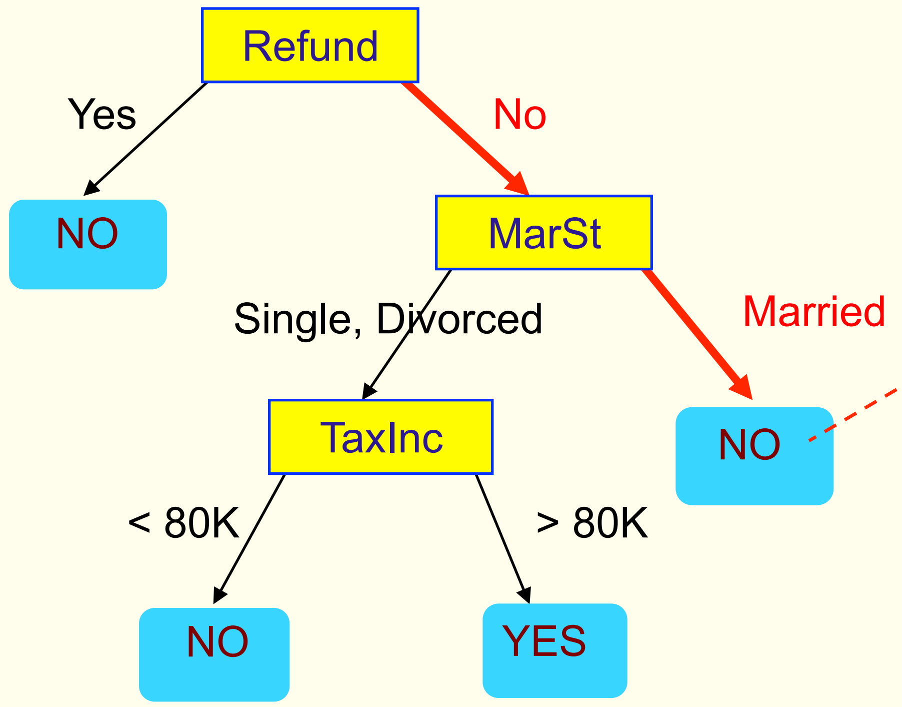
Apply Model to Test Data

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"

Decision trees - review

- To classify an unknown sample
  - the attributes values are tested against the decision tree.
  - A path is traced from root to a leaf node that holds the class prediction for that sample.
  - Decision trees can easily be converted to classification rules.
- Strength
  - Understand decision trees intuitively
- Weakness
  - Harder to manage as complexity of data increases – increasing the number of branches in tree
  - Can not model complex patterns

Works best with highly summarised, categorical data

Decision trees

- Generating the tree
  - Many decision tree algorithms (CART, CLS, ID3, C4.5, AID, CHAID, TREEDISC…)
  - All algorithms undergo a similar type of process, although some employ different mathematical algorithms to determine how to calculate and rank the importance of different variables
- The discussion below is based on the C5.0 algorithm used widely in off-the-shelf decision tree tools
- It works with categorical variables ONLY.
- It is a recursive process, that has to decide, at each node, what attribute best splits the data at this node.

Decision trees

Outline of the algorithm:

1. The tree starts as a single node representing the full training sample.
2. If the sample is all of the same class (e.g churn=false), then the node becomes a leaf node, and is labelled with that class
3. Otherwise the algorithm decides on the best attribute to use at this node, to partition the sample into groups that belong to the same class.
4. A branch is created for each value of this attribute, and the data is partitioned accordingly.

The algorithm uses the same process (steps 2 to 4) recursively to form a decision tree for the samples at each partition.

Once an attribute has been used at a node, it need not be considered in any of the nodes descendants.

Decision trees

The recursive processing stops when any one of the following conditions are true:

- All samples for a given node belong to the same class
- There are no remaining attributes on which the split the sample. In this case it is either labelled with the class in majority on that node, or the distribution of classes for that node is stored
- There is no data applicable to the node.

Decision trees

A **key decision** in the algorithm above is which attribute to use on each non-leaf node.

There are two criteria on which to evaluate an attribute:

1. How well does it split the data?
2. How much information will be required to classify the data samples in the child nodes generated by this attribute.

- Attribute selection
  - Information gain is commonly used to select the test attribute at each node in the tree
  - The attribute which results in the highest information gain is chosen

Information Gain

- Information Gain is a measure of how much easier it is to classify branches of a tree compared to the parent node.
- Measuring 'how easy' it is to classify rows of data is dependent on how homogenous that group is:
  - Do they all belong to the one class, or are they split across several classes?
- Entropy is a formula which measures how easy it is to classify a row of data, giving an answer between 0 and 1.
  - If all rows belong to the same class, the entropy is 0. If a sample is evenly split across classes, the entropy is 1.

- Information gain is the difference between the Entropy of the Parent node, and the combined entropy of the child nodes on a decision tree

Attribute Selection by information gain

The formula for Entropy is as follows:

$$E(t) = -\sum_{i=0}^{c-1} p(i \mid t) \log_2 p(i \mid t)$$

Where:

t is the current node on the Tree

c is the number of Classes in the dataset

i represents each individual class, e.g. churn=yes or churn=no.

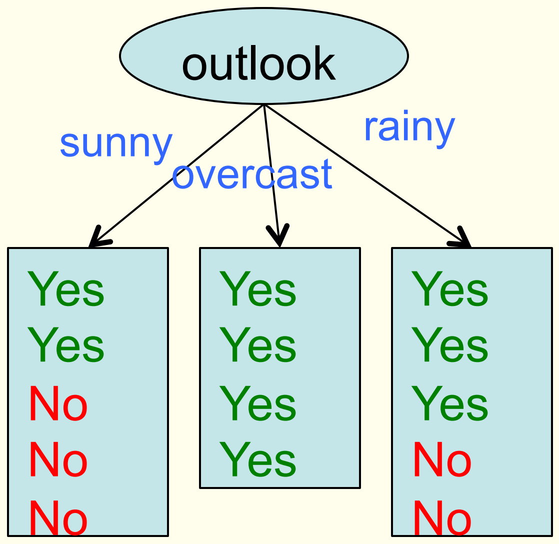
p(i|t) is the proportion or rows at node *t* that are in class *I*

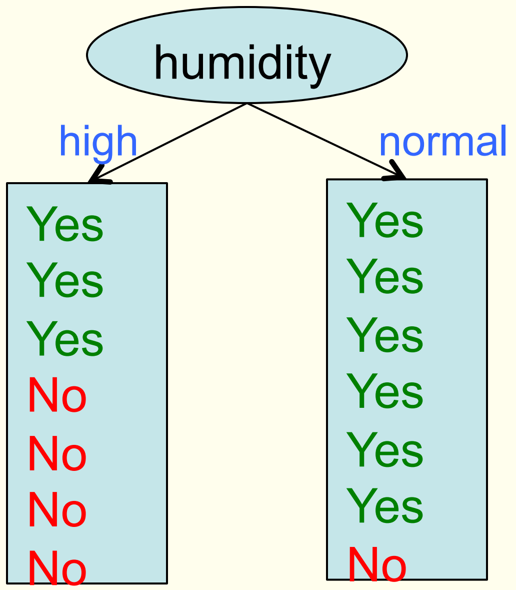
Lets look at an example . . .

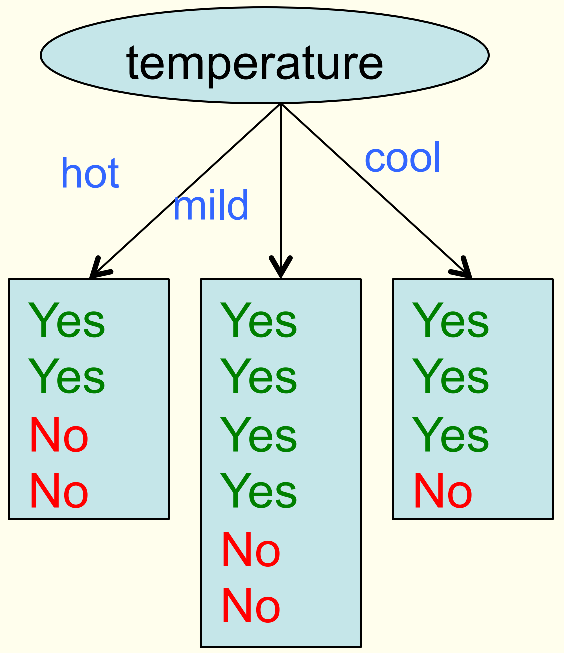
The calculations on the next few slides are based on this Golf dataset

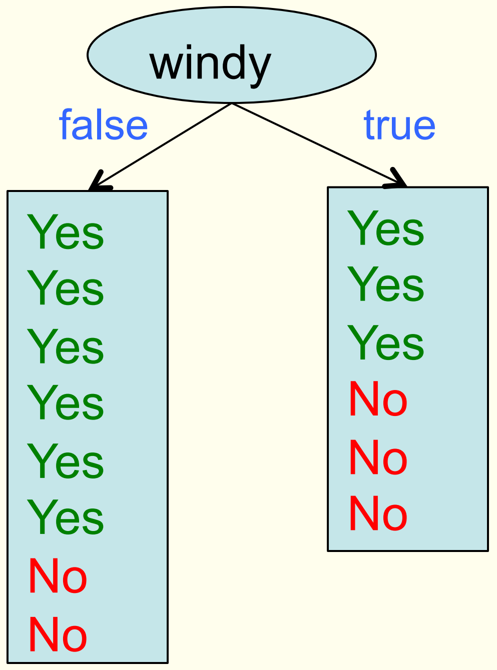
Outlook	Temperature	Humidity	Windy	Play
sunny	hot	high	FALSE	no
sunny	hot	high	TRUE	no
overcast	hot	normal	FALSE	yes
rain	mild	high	FALSE	yes
rain	cool	high	FALSE	yes
rain	cool	normal	TRUE	no
overcast	cool	normal	TRUE	yes
sunny	mild	high	FALSE	no
sunny	cool	normal	FALSE	yes
rain	mild	normal	FALSE	yes
sunny	mild	normal	TRUE	yes
overcast	mild	high	TRUE	yes
overcast	hot	normal	FALSE	yes
rain	mild	high	TRUE	no

Which of the following attributes gives the best split of the data?









$$E(t) = -\sum_{i=0}^{c-1} p(i|t)\log_2 p(i|t)$$

Taking the first branch for outlook, i.e. **outlook=sunny**, which had 2 yes's and 3 no's (5 rows of data). The entropy is:

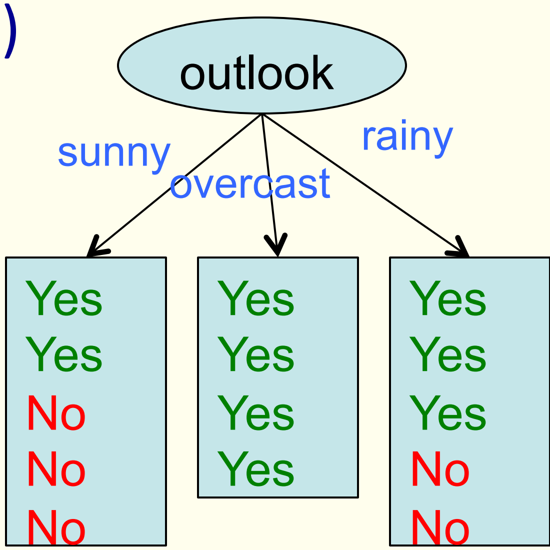
Entropy([2,3]) = $-(2/5)\log_2(2/5)-(3/5)\log_2(3/5)$

$= -(0.4)\log_2(0.4)-(0.6)\log_2(0.6)$

$= -(0.4)(-1.32) -(0.4))(-0.73)$

$= 0.528 + 0.442$

$= 0.97$



Note: If all samples belong to the same class, the entropy is 0. If a sample is evenly split across classes, the entropy is 1.

$$E(t) = -\sum_{i=0}^{c-1} p(i|t)\log_2 p(i|t)$$

Taking the second branch for outlook, i.e. **outlook=overcast**, which had 4 yes's and 0 no's (4 rows of data). The entropy is:

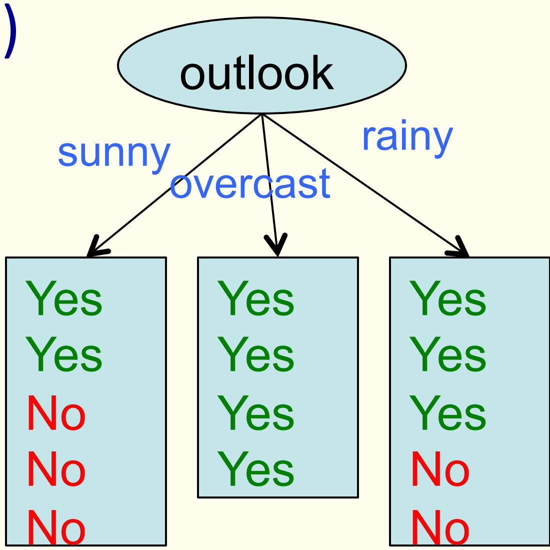
Entropy([4,0]) = $-(4/4)\log_2(4/4)-(0/4)\log_2(0/4)$

$= -(1)\log_2(1)-(0)\log_2(0)$

$= -(1)(0) -(0))(-0.73)$

$= 0 + 0$

$= 0$



Recap: If all samples belong to the same class, the entropy is 0. If a sample is evenly split across classes, the entropy is 1.

$$E(t) = -\sum_{i=0}^{c-1} p(i|t)\log_2 p(i|t)$$

Taking the second branch for outlook, i.e. **outlook=rainy**, which had 3 yes's and 2 no's (5 rows of data). The entropy is:

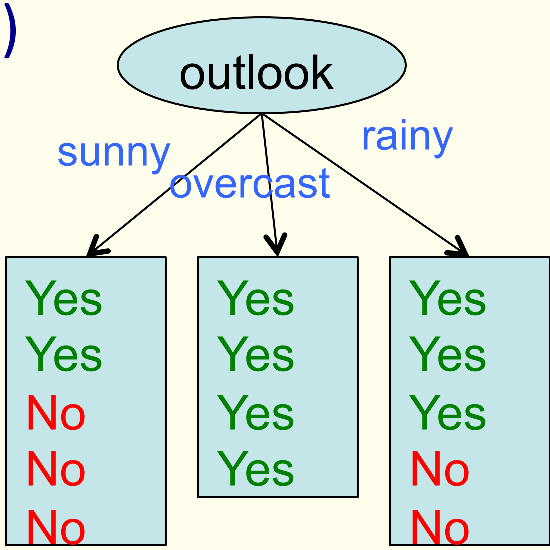
Entropy([3,2]) = $-(3/5)\log_2(3/5)-(2/5)\log_2(2/5)$

$= -(0.6)\log_2(0.6) -(0.4)\log_2(0.4)$

$= -(0.4))(-0.73) -(0.4)(-1.32)$

$= 0.442 + 0.528$

$= 0.97$



Recap: If all samples belong to the same class, the entropy is 0. If a sample is evenly split across classes, the entropy is 1.

Calculating the <u>average entropy</u> for outlook takes into account the number of rows in each branch:

Sunny:	5 rows @ 0.97:	0.97 + 0.97 + 0.97 + 0.97+ 0.97
Overcast:	4 rows @ 0:	+ 0 + 0 + 0 + 0
Rainy:	5 rows @ 0.97:	+ 0.97 + 0.97 + 0.97 + 0.97+ 0.97
Total:		= 9.7
Average:		9.7 / 14 = **0.693**

Note: There are 14 rows altogether across the three braches

The parent node of this branch contained nine yes's and five no's.

Its entropy is: Entropy([9,5]) = $-(9/14)\log_2(9/14)-(5/14)\log_2(4/14)$

$= \mathbf{0.940}$

The **information gain** of this new node is the difference between the average entropy of the child nodes, and the entropy of the parent, which is:

0.940 – 0.693 = **0.247.**

Doing the same calculation for the remaining three attributes gives the following Information Gains:

gain(outlook) = 0.247

gain(temperature) = 0.029

gain(humidity) = 0.152

gain(windy) = 0.048

Which is the best attribute to use?

Based on this calculation, outlook is the best attribute to use. This is because splitting by outlook gives the highest information gain.

Looking back at the data, outlook is favoured because it produces a leaf branch. The middle sample for outlook (overcast), has all **yes's.**

The second ranking attribute is humidity, because it produced a branch with all but 1 yes.

Exercise

Take a look at the simpsons example on moodle

Complete the exercises in
ExcersiseLect4.doc

Overfitting

- Sometimes, if the training dataset contains examples that are not very typical, the model created (the decision tree in this case) may be a very good representation of the training data, but not a good general model for the domain, and so may perform poorly when tested against a different dataset. This is called OVERFITTING.
- In general, if a model is too accurate on a training dataset (close to 100%) it is overfitted, and needs to be adjusted.
- Overfitting: When model performs well on the training dataset but poorly on the test dataset

Avoid Over fitting in Decision Trees

- Over fitting occurs in decision trees if it contains too many branches, with some branches matching only one or two rows of data, which could be noisy or erroneous data

- Two approaches to avoid over fitting in a decision tree:
  - Prepruning - Prune the tree while it is being built. This is done using the TRAINING DATASET
  - Postpruning – Build a full tree, and prune the tree afterwards. This can be done using a different dataset called a PRUNING DATASET, and so can be more accurate, but is more time consuming.

Pruning the tree

- **How to you decide what to prune?**
  - **There are a number of ways to decide what branches to prune, such as:**
    - Only allow branches if the information gain is above some threshold value. If no attribute provides an information gain above this threshold value, the node becomes a leaf.
    - Only allow branches that have more than a certain number of rows matching them
    - Calculate the error rate (number of rows incorrectly classified) with and without the branch. If accuracy is improved by some set value (e.g. more than 1%), then include the branch.
      - This works for post pruning only, as it needs the full tree to calculate the error rate.

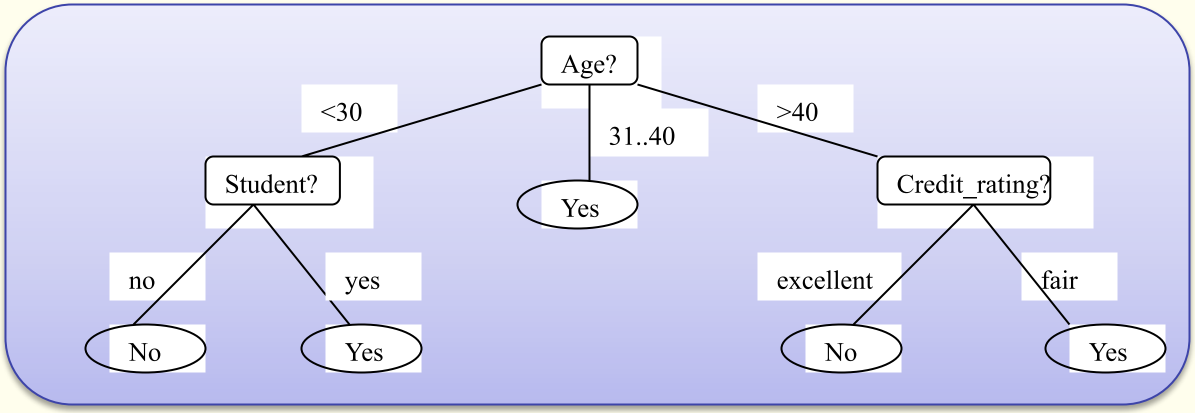
See post pruning.doc for an example of error rate calculations

Extracting classification rules from a decision tree

The knowledge represented in a decision tree can be converted into IF-THEN-ELSE rules.

– One rule is created for each path from root to leaf.

The classification rules from the full tree below are:



IF age < 30 and student = No, THEN smoker = no

IF age < 30 and student = Yes, THEN smoker = yes

IF age between 31 and 40, then THEN smoker = yes

IF age < 40 and credit rating = excellent, THEN smoker = no

IF age < 40 and credit rating = fair, THEN smoker = yes

K Nearest Neighbour

"If it walks like a duck, quacks like a duck and looks like a duck then it is probably a duck"

K Nearest Neighbour

- Most classifiers, including decision trees, are **eager learners**:
  - given a training sample they will generate a model in advance, which can subsequently be used to classify other data.
- Nearest neighbour classifiers are **lazy learners**:
  - the training set is not modeled until it is needed to classify the test examples.
  - To classify a test example, it is compared to objects in the training data set. If it is similar to one of those objects, then it is classified in the same way.

K Nearest Neighbour

A nearest neighbour classifier works by comparing the test example with a number of objects which are '**close**' to it.

- The *k*-nearest neighbours of a given row, *t*, are the *k* rows closest to it.
- The row, *t*, is allocated to the same class as the pre-dominate class amongst those *k* nearest neighbours.
- If there is a tie between two or more classes, one of the classes is be chosen randomly.

Calculating distances

- For nominal and categorical variables (which have no inherent ordering) distance is calculated as:

  = 1 if the attribute values are different, and

  = 0 if the values are the same.

	Attribute – eye colour
Row 1	Brown
Row 2	Blue
Row 3	Brown

d(row1, row2) = 1
d(row2, row3) = 1
d(Row1, row3) = 0

Exercise

- What is the distance between the following rows?

	Marital Status	Gender	Refund given
1	Single	M	Yes
2	Divorced	M	Yes
3	Married	F	No

Distance between:

Row 1 and Row 2? _________

Row 1 and Row 3? _________

Row 2 and Row 3? __________

Calculating distance

- For numeric attributes, distance can be calculated as:

  d = |x-y| / range

- Total range below is 53 (max value) minus 20 (min value) = 33

	Age	
Row 1	20	d(row1, row2) = |20-36|/33 = 0.48 d(row2, row3) = |36-53|/33 = 0.51 d(Row1, row3) = |20-53|/33 = 1.0
Row 2	36	
Row 3	53	

Exercise

- What is the distance between the following rows?

	Marital Status	Income	Age
1	Single	22000	20
2	Divorced	50000	60
3	Married	20000	45

Distance between:

Row 1 and Row 2? __________

Row 1 and Row 3? __________

Row 2 and Row 3? ___________

Which attribute is the most influential in the distance between rows?

How would you make all rows have equal influence?

Worked example

- Taking the dataset we looked at before:

Refund	**MaritalStatus**	**TaxableIncome**	**Cheat**
Yes	Single	125000	No
No	Married	100000	No
No	Single	70000	No
Yes	Married	120000	No
No	Divorced	95000	Yes
No	Married	60000	No
Yes	Divorced	220000	No
No	Single	85000	Yes
No	Married	75000	No
No	Single	90000	Yes
No	Married	85000	No
Yes	Married	85000	No
Yes	Single	125000	No

Worked example – calculating distance between a row from the test dataset, and row 1 in the training dataset

How would k-Nearest Neighbour classify the following example from the test dataset

Row 1: TEST dataset

Refund	MaritalStatu	TaxableIncor
Yes	Single	125000

Distance from row 1 in the training dataset:

Row 1: training dataset

Training data			
Refund	**MaritalStatu**	**TaxableIncome**	**Cheat**
Yes	Single	125000	No

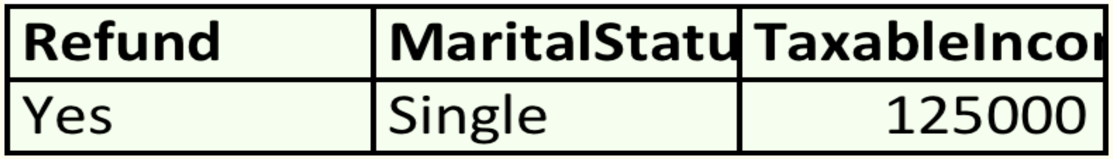
Distances			
Refund	**MaritalStatu**	**TaxableIncor**	**Average Dist**
0.00	0.00	0.00	0.00

All values were the same as row 1 in the training dataset, hence a distance of 0.

Worked example – calculating distance between a row from the test dataset, and row 2 in the training dataset

- Test data row again . . .

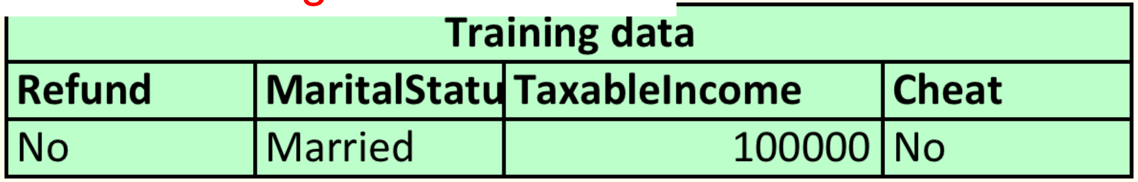
Row 1: TEST dataset



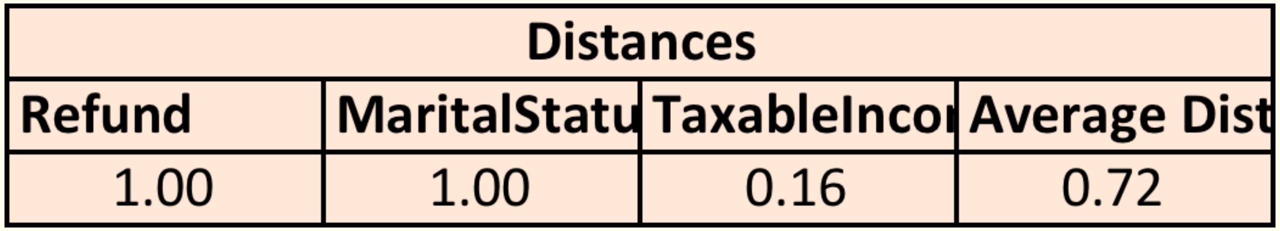
Refund	MaritalStatu	TaxableInco
Yes	Single	125000

- Distance to 2nd row in the training dataset:

Row 2: training dataset



Training data			
Refund	MaritalStatu	TaxableIncome	Cheat
No	Married	100000	No



Distances			
Refund	MaritalStatu	TaxableInco	Average Dist
1.00	1.00	0.16	0.72

Income has been scaled to the range [0,1]

Worked example

- The table below shows the distance between each row in the training example and the single test example. For calculations, see TaxReturns-knn.xlsx

Training dataset

Distance for:

Refund	Marital Status	Taxable Income	Cheat
Yes	Single	125000	No

Training data				Distances				
Refund	**Marital Status**	**Taxable Income**	**Cheat**	**Refund**	**Marital Status**	**Taxable Income**	**Average Distance**	**Rank (Nearest)**
Yes	Single	125000	No	0.00	0.00	0.00	0.00	1
No	Married	100000	No	1.00	1.00	0.16	0.72	8
No	Single	70000	No	1.00	0.00	0.34	0.45	6
Yes	Married	120000	No	0.00	1.00	0.03	0.34	2
No	Divorced	95000	Yes	1.00	1.00	0.19	0.73	9
No	Married	60000	No	1.00	1.00	0.41	0.80	12
Yes	Divorced	220000	No	0.00	1.00	0.59	0.53	7
No	Single	85000	Yes	1.00	0.00	0.25	0.42	4
No	Married	75000	No	1.00	1.00	0.31	0.77	11
No	Single	90000	Yes	1.00	0.00	0.22	0.41	3
No	Married	85000	No	1.00	1.00	0.25	0.75	10
Yes	Married	85000	No	0.00	1.00	0.25	0.42	4

The actual label for the test data row is 'No'

Will k-NN get this correct for k=3?

What about k=4?

Worked examples

Table from last slide explained:

Columns 1 to 4: the training dataset - the three attributes and class label for the 12 rows in the training dataset.

Column 5: The distance between ‘refund’ in the training dataset, and refund=“yes” in the test row.

Column 6: The distance between ‘marital status’ in the training dataset, and marital status=“single” in the test row.

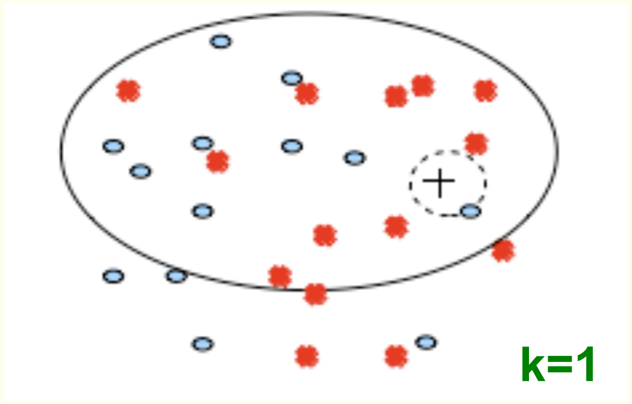
Column 7: The distance between ‘taxable income’ in the training dataset, and taxable income=“125000” in the test row.

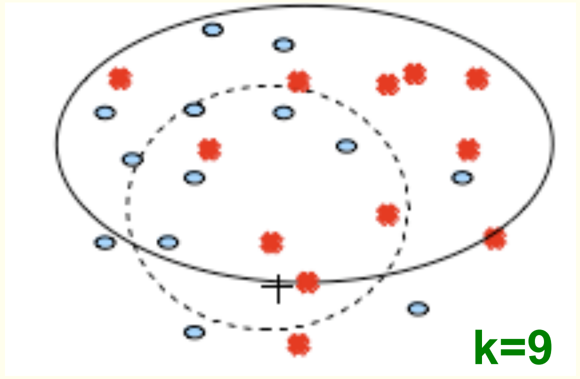
Column 8: The average of the three distances from columns 5 – 7.

Rank: The average distances in order, i.e. Rank=1 is the row most similar to the test row; Ramk=12 is the row least similar to the test row.

Choosing the right number for K

- If *k* is too small, results could be influenced by noise
- If *k* is too large, objects not close to the example will influence the result.





The dataset is made up of objects from two classes, ‘red’ points and ‘blue’ points. **+** is the point to be classified. Assume it should be classified as a ‘red’ point. The dotted circle indicates the k closest objects, depending on the value of k.

Choosing the right number for K

- If the dataset has 50 rows, 20 belonging to one class, and 30 rows belonging to another class, which of the following values would you choose for K?
  - K=5?
  - K=15?
  - K=25?
  - K=50?

A small value of *k* over-fits the data. Just like a fully grown tree, decision are made based on comparisons with a small number of rows

Strengths and Weaknesses for K nearest neighbour

- Classifying a test sample can be quite expensive because a model hasn't been built in advance.
- The fast training time makes them suitable for environments where models are changing rapidly.
  - For example a model classifying an e-mail is spam needs to be updated regularly to identify new formations in spam mail.
- With Nearest neighbour, decisions are made locally, which makes them more sensitive to details in the data. This is good for data which has a lot of variability by nature and so difficult to generalise about, e.g. classifying people. However, it also makes them more sensitive to noise in the data.

Summary

Classification algorithms:

1. What type of data can it handle?
2. Predictive accuracy
3. Speed and scalability
   time to construct the model
   time to use the model
4. Robustness
   handling noise and missing values
5. Interpretability:
   understanding and insight provided by the model

Decision trees

- C5.0 algorithm
- Attribute selection – Information gain
- Tree pruning

K-Nearest neighbour

- Lazy classifier.
- No model is build in advance.
- New data is compared with all rows in the training sample, and classified the same way as those rows that it is most simillar to.

Solution: Which of the following predictions would you trust?

1. A model says the customer you are talking to <u>will churn</u>
   – The model has high precision for churn=yes

Yes

2. A model says the customer you are talking to will churn
   – The model has high recall for churn=yes

Depends on precision

3. A model says the customer you are talking to will churn
   – The model has high recall for churn=yes, but low precision for churn=yes.

No

4. A model returns a list of customers that will churn.
   – The model has high precision but low recall for churn=yes

No