FIT3152 Data analytics – Lecture 7

Introduction to decision trees

- Assignment Q & A
- Regression Review Questions
- Overview: machine learning
- Introduction to classification and decision trees
- A specific decision tree algorithm: ID3
- Entropy and information gain
- Model accuracy; training and testing
- Decision trees in R

Consultations on Zoom

Clayton consultations have commenced:

- Any student can attend any consultation.
- Schedule on Moodle, https://lms.monash.edu/
- Current days/times:
- Monday 9:30-10:30AM, 2:00-3:00PM, 6:00-7:00PM,
- Tuesday 9:00-10:00AM, 12:00PM-1:00PM,
- Wednesday 10:00AM-11:00, 11:00-12:00PM,
- Thursday 1:00PM-02:00PM, 6:00PM-7:00PM.
- Please check the schedule for any changes.
- Consultation will run over the mid-semester break. Note Friday 15th, Mon 18th, Tues 19th, and Mon 25th of April are holidays so no consultations will run on those days.

Week-by-week

Week Starting	Lecture	Topic	Tutorial	A1	A2
28/2/22	1	Intro to Data Science, review of basic statistics using R	...		
7/3/22	2	Exploring data using graphics in R	T1		
14/3/22	3	Data manipulation in R	T2	Released	
21/3/22	4	Data Science methodologies, dirty/clean/tidy data, data manipulation	T3		
28/3/22	5	Network analysis	T4		
4/4/22	6	Regression modelling	T5		
11/4/22	7	Classification using decision trees	T6		
		Mid-semester Break		Submitted	
25/4/22	8	Naïve Bayes, evaluating classifiers	T7		Released
2/5/22	9	Ensemble methods, artificial neural networks	T8		
9/5/22	10	Clustering	T9		
16/5/22	11	Text analysis	T10		Submitted
23/5/22	12	Review of course, Exam preparation	T11		

Assignment 1

Assignment 1: Summary

FIT3152 Data analytics – 2022: Assignment 1

Your task	• Analyse the activity, language use and social interactions of an on-line community using metadata and linguistic summary from a real on-line forum and submit a report of your findings. • This is an individual assignment.
Value	• This assignment is worth **20%** of your total marks for the unit. • It has 30 marks in total.
Suggested Length	• 6 – 8 A4 pages (for your report) + extra pages as appendix (for your code) • Font size 11 or 12pt, single spacing
Due Date	**11.55pm Friday 22nd April 2022**
Submission	• PDF file only. Naming convention: *FirstnameSecondnameID.pdf* • Via Moodle Assignment Submission. • Turnitin will be used for similarity checking of all submissions.
Late Penalties	• 10% (3 mark) deduction per calendar day for up to one week. • Submissions more than 7 calendar days after the due date will receive a mark of zero (0) and no assessment feedback will be provided.

Assignment 1: Instructions

Instructions

Submit the results of your analysis, answering the research questions and report anything else you discover of relevance. If you choose to analyse only a subset of your data, you should explain why.

You are expected to include at least one multivariate graphic summarising key results. You may also include simpler graphs and tables. Report any assumptions you've made in modelling, and include your R code as an appendix.

There are two options for compiling your report:
(1) You can submit a single pdf with R code pasted in as machine-readable text as an appendix, or
(2) As an R Markup document that contains the R code with the discussion/text interleaved. Render this as an HTML file and print off as a pdf and submit.

Regardless of which method you choose, you will submit a single pdf, and your R code will be machine readable text. We need to conform to this format as the university now requires all student submission to be processed by plagiarism detection software.

Submit your report as a single PDF with the file name ***FirstnameSecondnameID.pdf*** on Moodle.

Assignment 1: Questions a & b

Questions

Activity, language use and social interactions in an on-line community. Analyse the metadata and linguistic summary from a real on-line forum and submit a report of your findings. Do the following:

(a) Analyse activity and language on the forum over time:

1. How active are participants over the longer term (that is, over months and/or years)? Are there periods where activity increases or decreases? Is there a trend over time? **(3 Marks)**
2. Looking at the linguistic variables, do the levels of these change over the duration of the forum? Is there a relationship between linguistic variables over the longer term? **(3 Marks)**

(b) Analyse the language used by threads:

We can think of threads as groups of participants posting on the same topic.

1. Using the relevant linguistic variables, is it possible to see whether or not particular threads are happier or more optimistic than other threads, or the forum in general, at different periods in time. **(3 Marks)**

Assignment 1: Question c

(c) Analyse social networks online:

We can think of authors posting to the same thread at similar times (for example during the same month) as having a connection to each other, forming a social network. This is called a two-mode network. When an author posts to more than one network during the same time period their social network extends to include authors from both networks, and so on. We will cover social network analysis in Lecture 5.

1. Create a non-trivial social network of all authors who are posting over a particular time period. For example, over one month. To create this, your social network should include at least 30 authors, some of whom will have posted to multiple (2 or more) threads during this period. Your social network should be connected, although some authors may be disconnected from the main group. Present your result as a network graph. **(3 Marks)**
2. Identify the most important author in the social network you created. Looking at the language they use, can you observe any difference between them and other members of their social network? **(3 Marks)**

Assignment 1: Overall considerations

(d) Overall considerations:

- The quality and clarity of your reasoning and assumptions. **(3 Marks)**
- The strength of support for your findings. **(3 Marks)**
- The quality of your writing in general and communication of results. **(3 Marks)**
- The quality of your graphics throughout, including at least one high-quality multivariate graphic. **(3 Marks)**
- The quality of your R coding. **(3 Marks)**

Assignment 1: Draft rubric

Question	Part	Mark 1	Mark 2	Mark 3
a	1	Activity over time (posts and/or threads) calculated.	Suitable graphical presentation is created.	Visual inspection and descriptive analysis performed.
	2	Suitable time series summary calculations of LIWC variables performed.	Suitable graphical presentation is created.	Descriptive analysis of relationship between variables over time.
b	1	Relevant LIWC variables identified with justification.	Summary of relevant LIWC variables presented.	Descriptive or graphical comparison of threads performed.
c	1	Suitable time interval for calculating social network identified.	Social network with Author IDs as nodes created.	Suitable network plot created.
	2	Vertex importance measures created.	Most important author identified with reasons given.	Comparison of most important authors with others.
d	1	Data pre-processing (or not) is justified and performed.	Suitable time division identified and time coded accordingly.	Data analysis and conclusions are logical overall.
	2	Higher level of justification for Part a (hyp tests other statistical models etc.)	Higher level of justification for Part b (hyp tests other statistical models etc.).	Higher level of justification for Part c (hyp tests other statistical models etc.).
	3	Report has good structure and flow.	Quality of writing is good throughout.	Quality of writing is excellent throughout.
	4	One high quality multivarate graphic included.	Graph choice is appropriate throughout.	Graphs are high quality throughout.
	5	R coding looks sensible and has good readability.	Coding is used to automate analysis across multiple fields - Parts a and b.	Coding used to automate network construction - Part c.

Response to student questions

- I'm a bit confused about whether WC is a linguistic variable or not, could you confirm that for me, please?
  - > Think about what LIWC means... Word Count is a linguistic variable because it tells us how many words are in the post... A measure like this might tell us about the reliability of the other summary stats. For example, would a post of 1000 words give a more reliable estimate of sentiment than a post of 10 words?

Response to student questions

- For our coding part, are we allowed to use the method that has not been taught in class? If so, do we need a reference for that method?
  - > You can definitely use methods not taught in class. You can put a reference in your notes as a commented-out URL etc. if you think it's relevant. Avoid copying slabs of other people's code verbatim, but please adapt any useful ideas you find in your research.

Response to student questions

- Do we need to have a cover sheet and table of content for this. Is that possible to post a report template or a marking rubric?
  - > No cover sheet or table of contents needed. Rubric was posted in last week's lecture. If you need a report template, what about:
  - > Brief introduction (any data pre-processing, and general observations that you think are relevant)
  - > Question 1.
  - > Question 2.
  - > ... other questions.
  - > Any brief closing comments.
  - > Appendix if required.
  - > R Script.

Response to student questions

- I'm using RMarkdown for Assignment and would like to double check on the structure of the submission.
- Proposed structure: Cover Page (optional and not counted as a page), Compiled RMarkdown PDF file (does not include code snippets to save space), (Appendix) RMarkdown in plain text attached to the compiled PDF file
  - > What you propose sounds fine. You don't need a cover page. You could interleave your code snippets in the answers if you wanted. That would not be counted in total pages. Note that page count is a guide - your report can be longer than this if necessary - but not too long!

Response to student questions

- I’m still a bit confused about the steps for pre-processing data. I’m wondering besides the lecture slides, are there useful resources such as books, journals and websites.
  - > Your data is clean, and tidy. However, when you analyse the data, you may need to consider what data you include, and what data you do not use. This a judgement you need to make, based on your understanding of the problem.

Response to student questions

- If I resolved the question with multiple simple codes instead of complex code, would I receive a mark down? For example, I want to get the coefficient for each year, so instead of continuing working on the table, I looked at the table and create a data set for each year ... and calculate the coefficient one by one.
  - > Marks will be given to coding that is more automated over hand coding (where that's relevant).

Quick review from regression lecture:

Please answer in the Zoom chat...

mtcars

The (inbuilt) data set Motor Trend Car Road Tests gives summary statistics including fuel usage (mpg), engine size (disp), number of cylinders (cyl), power (hp), body weight (wt) and the number of gears for a variety of cars.

How well do these variables predict fuel economy?

Summary data

```
> head(mtcars)
                   mpg cyl disp  hp drat   wt qsec vs am gear
Mazda RX4         21.0   6  160 110 3.90 2.62 16.5  0  1    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.88 17.0  0  1    4
Datsun 710        22.8   4  108  93 3.85 2.32 18.6  1  1    4
Hornet 4 Drive    21.4   6  258 110 3.08 3.21 19.4  1  0    3
Hornet Sportabout 18.7   8  360 175 3.15 3.44 17.0  0  0    3
Valiant           18.1   6  225 105 2.76 3.46 20.2  1  0    3
```

Model

```
> attach(mtcars)
> fitted = lm(mpg ~ cyl + disp + hp + wt + gear)
> fitted

Call:
lm(formula = mpg ~ cyl + disp + hp + wt + gear)

Coefficients:
(Intercept)          cyl         disp           hp           wt
    37.3626      -1.1186       0.0138      -0.0279      -3.7143
       gear
     0.6788
```

Summary (a)

```
Call:
lm(formula = mpg ~ cyl + disp + hp + wt + gear)

Residuals:
   Min     1Q Median     3Q    Max
-3.223 -1.686 -0.383  1.293  5.943
```

Summary (b)

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  37.3626     5.9725    6.26  1.3e-06 ***
cyl          -1.1186     0.7144   -1.57   0.1295
disp          0.0138     0.0123    1.12   0.2727
hp           -0.0279     0.0166   -1.68   0.1052
wt           -3.7143     1.0482   -3.54   0.0015 **
gear          0.6788     1.0345    0.66   0.5175
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘
’ 1

Residual standard error: 2.54 on 26 degrees of freedom
Multiple R-squared:  0.851,   Adjusted R-squared:  0.822
F-statistic: 29.7 on 5 and 26 DF,  p-value: 5.72e-10
```

Question 1

Ignoring the constant term, the most reliable predictor of fuel economy (mpg) is:

		Estimate	Std. Error	t value	Pr(>|t|)	
	(Intercept)	37.3626	5.9725	6.26	1.3e-06	***
(a)	cyl	-1.1186	0.7144	-1.57	0.1295	
(b)	disp	0.0138	0.0123	1.12	0.2727	
(c)	hp	-0.0279	0.0166	-1.68	0.1052	
(d)	wt	-3.7143	1.0482	-3.54	0.0015	**
(e)	gear	0.6788	1.0345	0.66	0.5175	

Question 2

The least reliable predictor of fuel economy is:

		Estimate	Std. Error	t value	Pr(>|t|)	
	(Intercept)	37.3626	5.9725	6.26	1.3e-06	***
(a)	cyl	-1.1186	0.7144	-1.57	0.1295	
(b)	disp	0.0138	0.0123	1.12	0.2727	
(c)	hp	-0.0279	0.0166	-1.68	0.1052	
(d)	wt	-3.7143	1.0482	-3.54	0.0015	**
(e)	gear	0.6788	1.0345	0.66	0.5175	

Question 3

Cars with more gears are more economical:

A. True
B. False
C. Can't tell

		Estimate	Std. Error	t value	Pr(>|t|)	
	(Intercept)	37.3626	5.9725	6.26	1.3e-06	***
(a)	cyl	-1.1186	0.7144	-1.57	0.1295	
(b)	disp	0.0138	0.0123	1.12	0.2727	
(c)	hp	-0.0279	0.0166	-1.68	0.1052	
(d)	wt	-3.7143	1.0482	-3.54	0.0015	**
(e)	gear	0.6788	1.0345	0.66	0.5175	

Question 4

Heaver cars are less economical:

A. True
B. False
C. Can't tell

		Estimate	Std. Error	t value	Pr(>|t|)	
	(Intercept)	37.3626	5.9725	6.26	1.3e-06	***
(a)	cyl	-1.1186	0.7144	-1.57	0.1295	
(b)	disp	0.0138	0.0123	1.12	0.2727	
(c)	hp	-0.0279	0.0166	-1.68	0.1052	
(d)	wt	-3.7143	1.0482	-3.54	0.0015	**
(e)	gear	0.6788	1.0345	0.66	0.5175	

Question 5

Overall, the predictive power of the model is high:

A. True (better than 70%)
B. False (worse than 30%)
C. Can't tell

```
Residual standard error: 2.54 on 26 degrees of freedom
Multiple R-squared:  0.851,   Adjusted R-squared:  0.822
F-statistic: 29.7 on 5 and 26 DF,  p-value: 5.72e-10
```

Machine learning

Machine Learning

We can think of Machine Learning (ML) as the automated (statistical) learning of a concept from labelled sample data.

- For example: Spam filtering with an algorithm that takes some examples of spam and makes a rule to predict whether an email should go to the spam folder.

How can a model learn a concept?

- Descriptive: captures the training data;
- Predictive: generalizes to unseen data;
- Explanatory: describes the concept to be learned.

A common application of ML is classification.

ML classification example – tax return

- Given the following data about people who submitted a tax return, we want to automatically create a model that will classify people into cheats or non-cheats.
- We then want to be able to Use the model to classify 'new' people into cheats or non-cheats.

	categorical	categorical	continuous	class
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**

Classification

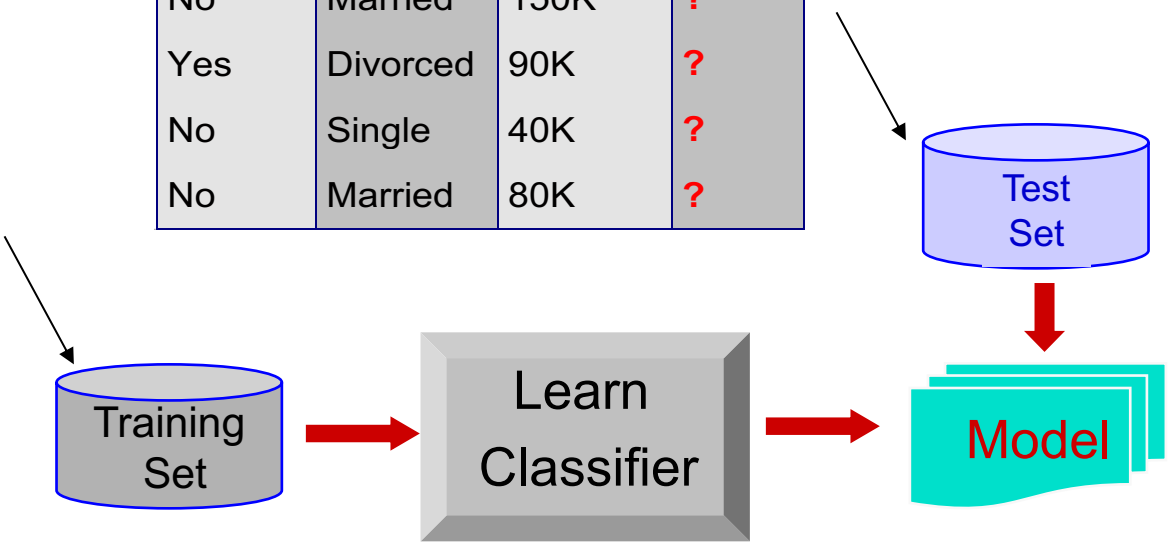
Using a collection of records containing a set of *attributes* where one of the attributes is the *class*, find a model to predict class as a function of the other attributes.

- Goal: previously unseen records should be assigned a class as accurately as possible.
- Data is usually divided into a *training set* to build the model, and a *test set* used to validate (test the accuracy) of the model.

ML classification example – tax return

Tid	Refund (categorical)	Marital Status (categorical)	Taxable Income (continuous)	Cheat (class)
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

Refund	Marital Status	Taxable Income	Cheat
No	Single	75K	?
Yes	Married	50K	?
No	Married	150K	?
Yes	Divorced	90K	?
No	Single	40K	?
No	Married	80K	?



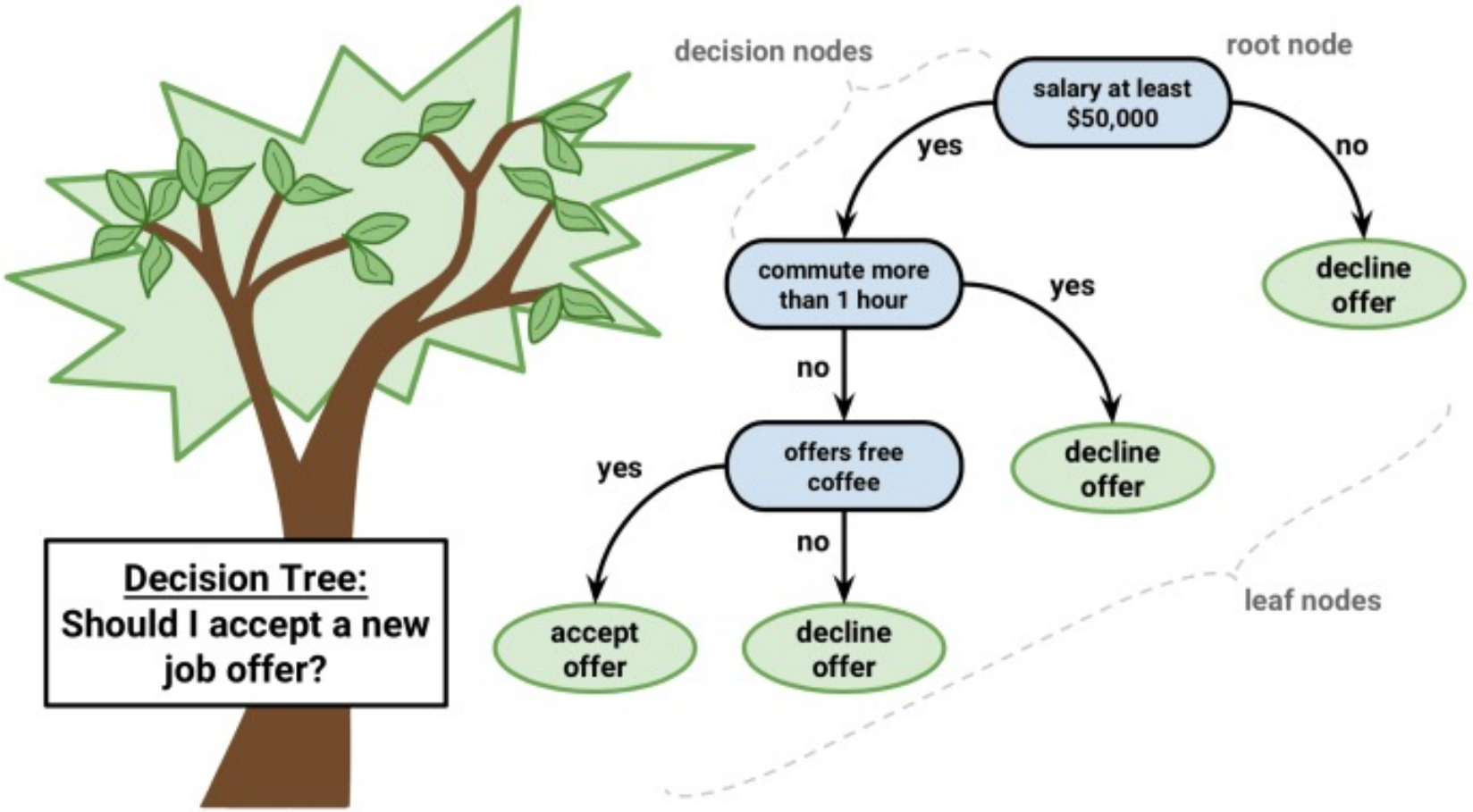
Classification

Machine Learning classification techniques include:

- Decision Tree based methods
- Naïve Bayes and Bayesian Belief Networks
- Ensemble methods
- Artificial neural networks
- Rule-based methods
- Memory based reasoning
- Support Vector Machines

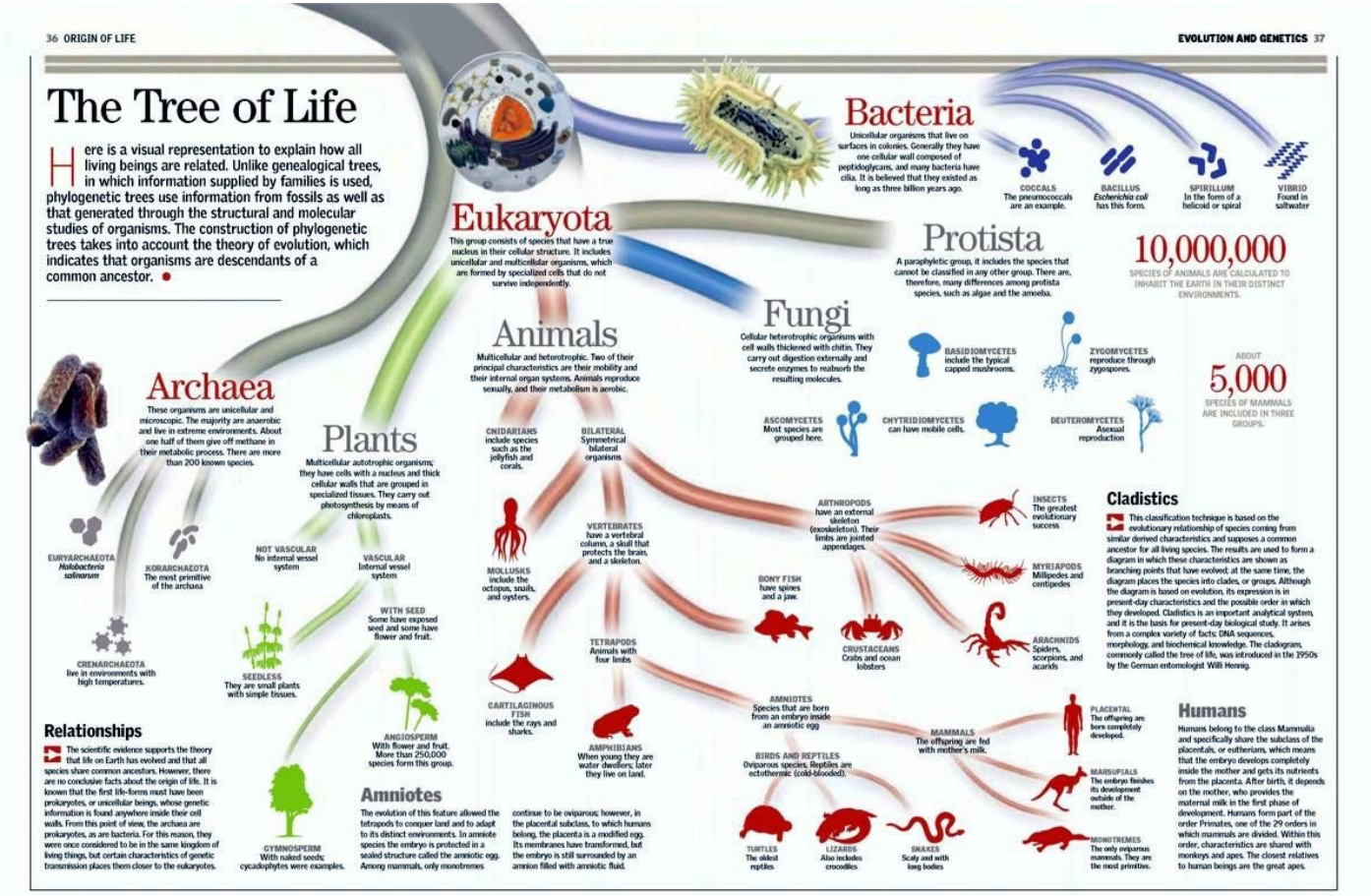
In each of these cases the model "learns" the classification rules from the data.

Decision tree: should I accept job?



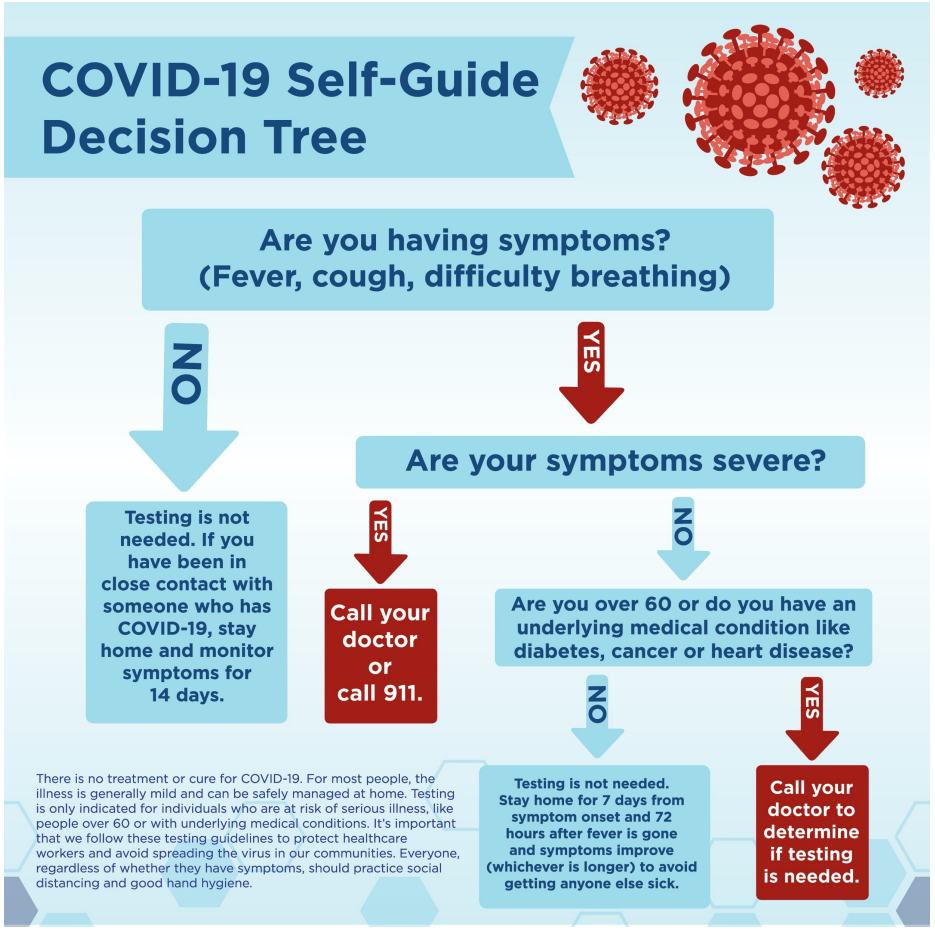
https://towardsdatascience.com/decision-tree-hugging-b8851f853486

Classification tree: life forms



https://medium.com/cracking-the-data-science-interview/decision-trees-how-to-optimize-my-decision-making-process-e1f327999c7a

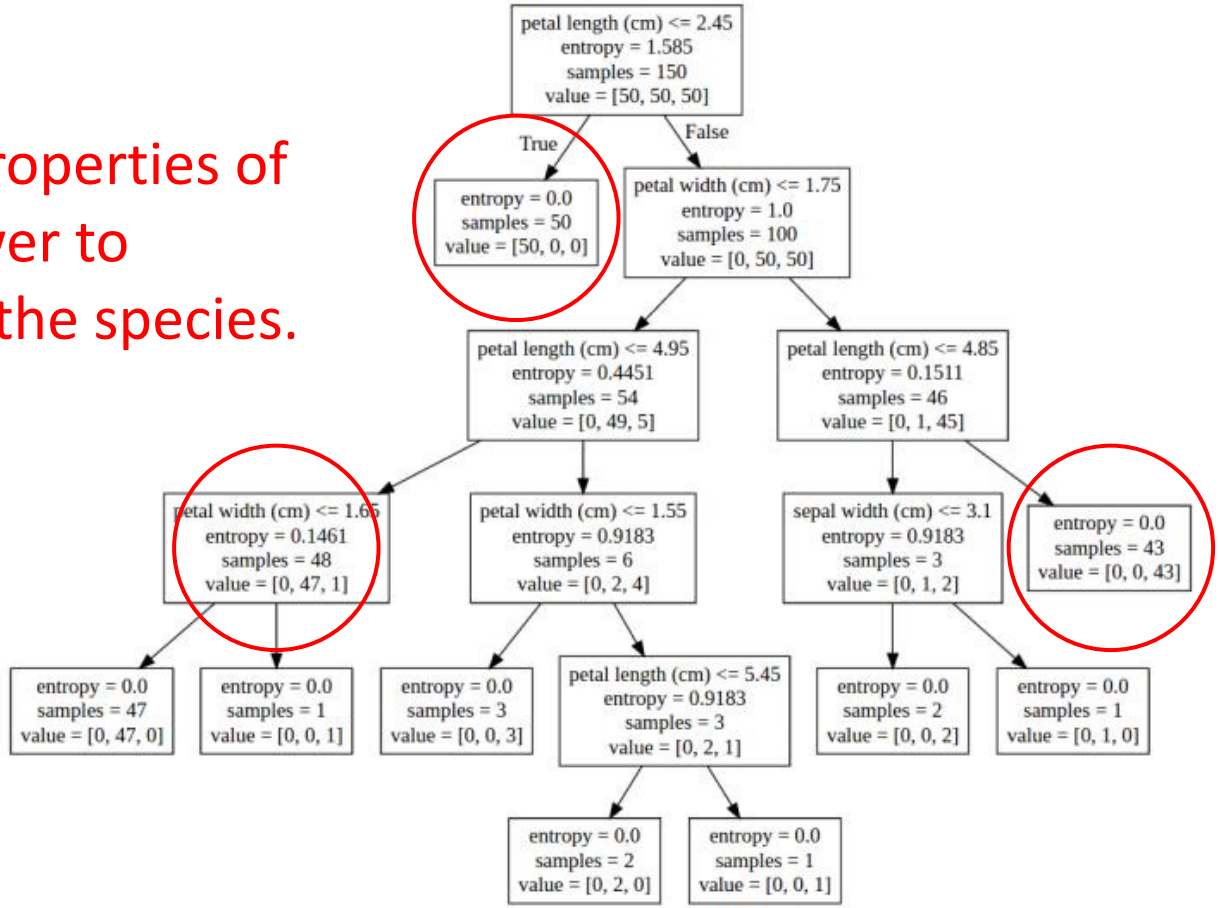
COVID-19 Self-diagnosis



https://www.holzer.org/coronavirus-covid-19-updates/

Iris species classification

Using properties of the flower to classify the species.



https://www.kdnuggets.com/2017/05/simplifying-decision-tree-interpretation-decision-rules-python.html

Decision Trees

Decision trees are one of the most widely used and practical methods in machine learning:

- Model uses existing data attributes and values
- Can be used to classify new instances
- Can be used to profile existing data
- Robust to noise and missing values
- Each tree can be viewed as a sequence of "if – then – else" statements, this readability is highly desirable.
- *We can construct simple decision trees by hand!*

Decision Trees

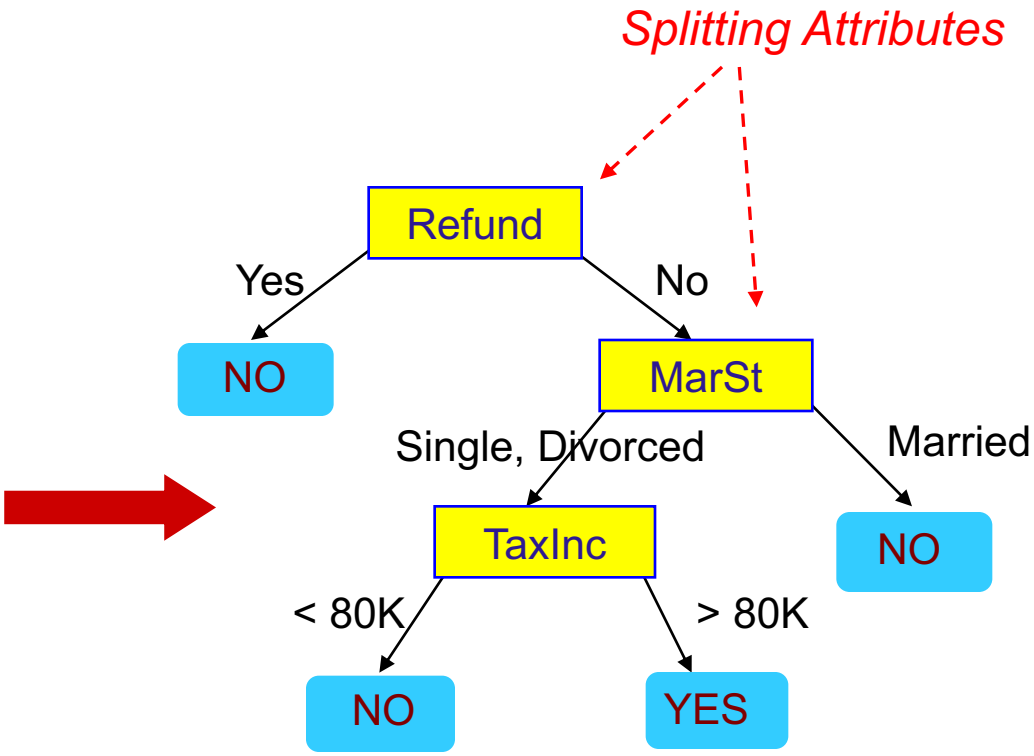
Tree constitutes:

- Leaf nodes (of each class) and non-leaf nodes, corresponding to the decision attributes,
- Branches – corresponding to the values of the decision attributes having either binary or multi-way splits.
- To classify an object, each decision node (starting from the root) compares an attribute of the object with a specific attribute value (or range) and takes the corresponding branch.
- A path from the root to a leaf node gives the class of the object.

Classification example – tax return

	categorical	categorical	continuous	class
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

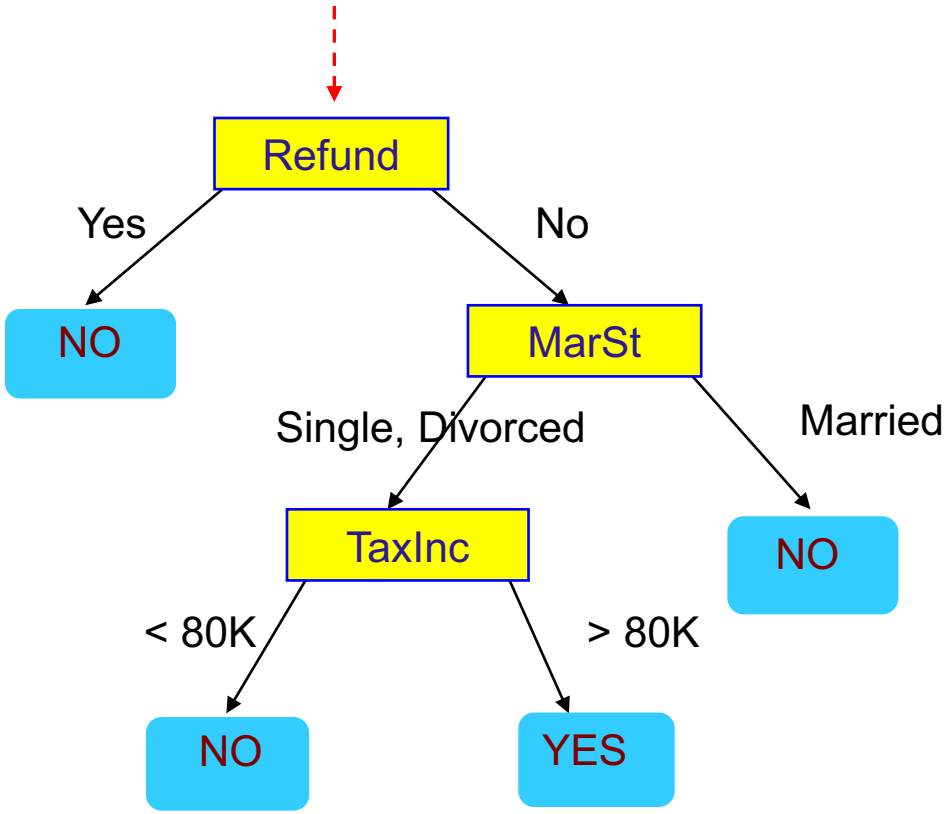


Model: Decision Tree

(One of many possible trees)

Apply Model to Test Data

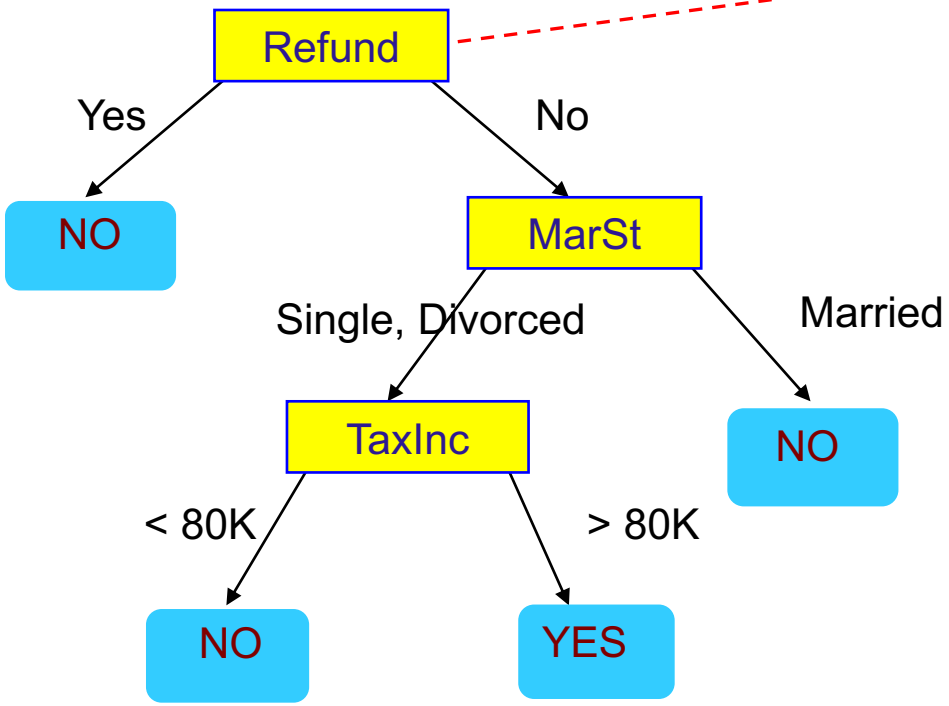
Start from the root of tree.



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

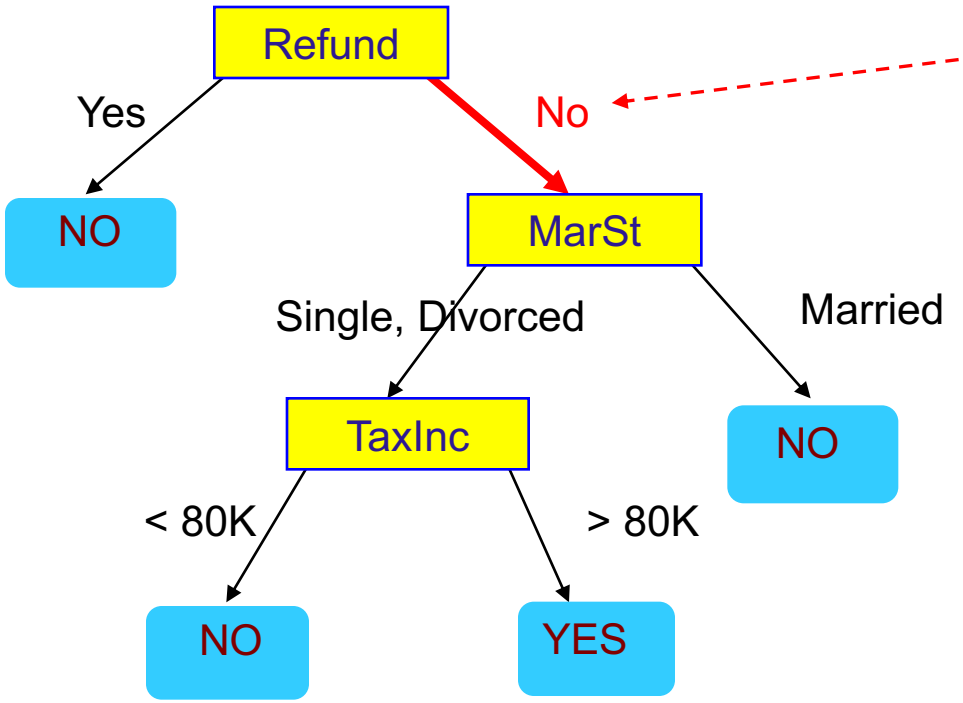
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

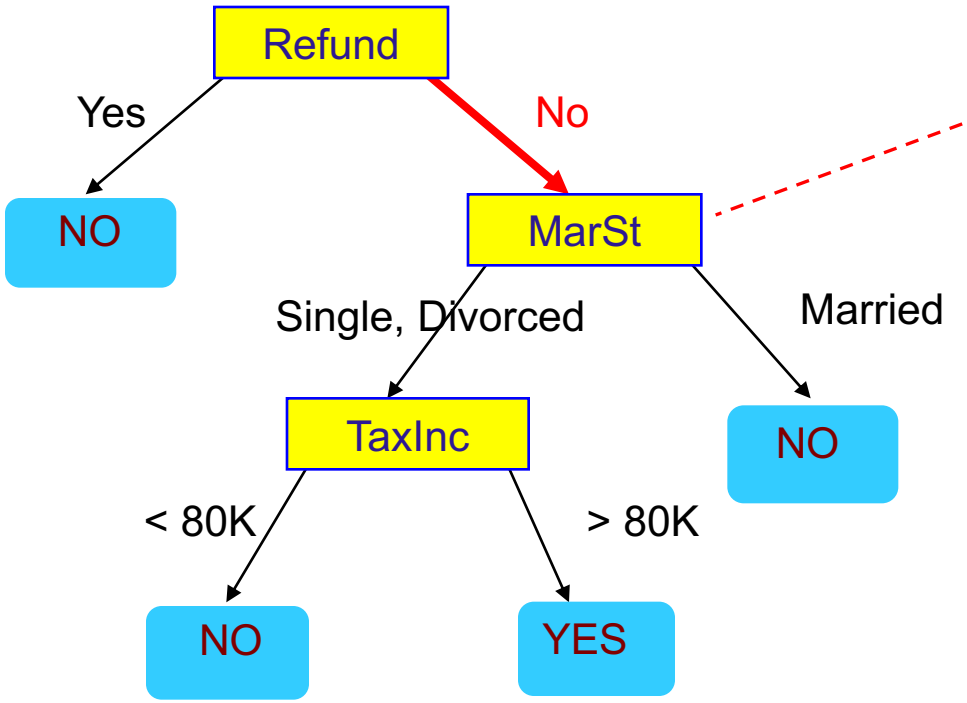
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data



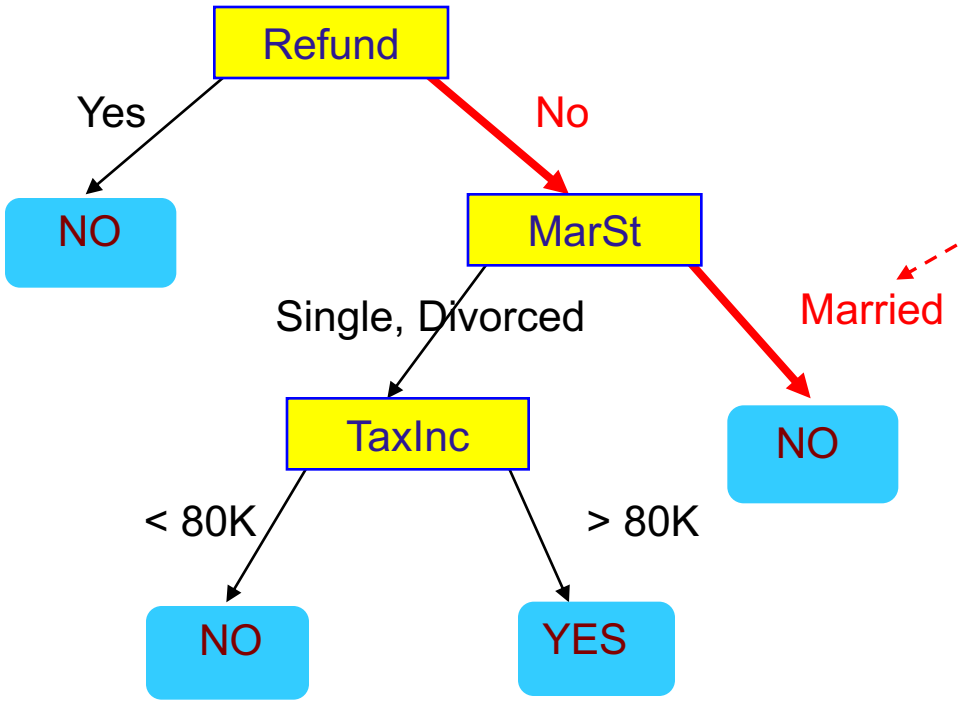
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

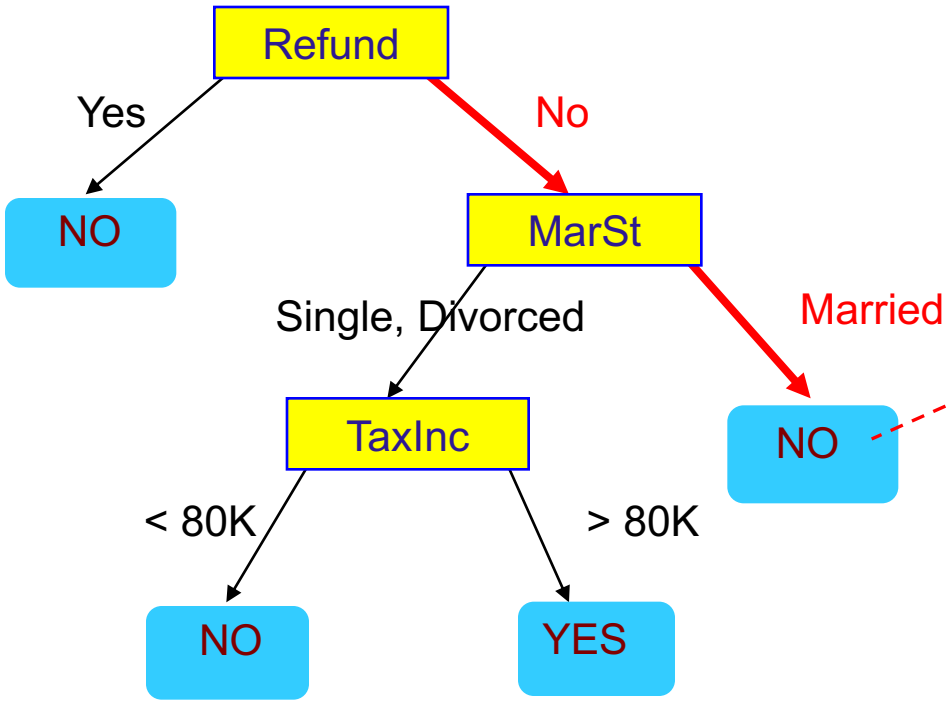
Apply Model to Test Data

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”

Building a decision tree

Building a decision tree requires answering:

- Which attribute should be tested at a node?
- When should a node be declared a leaf?
- What if tree becomes too large?
- How to handle missing values?
- Should the properties be restricted to binary-valued or allowed to be multi-valued?
- Answering these questions leads to different variants of decision trees, ID3, C4.5, C5, CART, etc.

Top-down induction: ID3

The algorithm (*Iterative Dichotomiser 3*):

- At each step, determine the "best" decision attribute, A, for next node.
- Assign A as decision attribute for node.
- For each value of A create new descendant.
- Sort training examples to that node according to the attribute value of the branch.
- If all training examples are homogenous (that is, perfectly classified, having same value of target attribute) stop, else iterate over new leaf nodes.

For this algorithm assume class attribute is categorical.

https://en.wikipedia.org/wiki/ID3_algorithm

That is, a category, not a number.

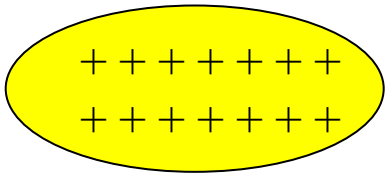
Which attribute to split on?

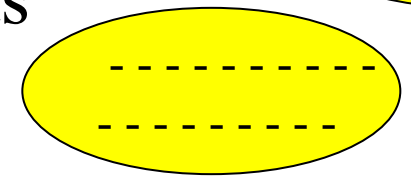
- At each stage of the process, we try to find the 'best' attribute and split to partition the data.
- That decision may not be the best overall – but once it is made, we stay with it for the rest of the tree.
- This is generally called a *greedy* approach and may not result in the best overall decision tree.
- At each split the goal is to increase the *homogeneity* of the resulting datasets with respect to the *class* or *target* variable (which we are trying to classify).

Homogeneity

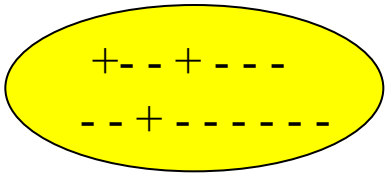
Suppose we have a binary target attribute with values ‘+’ and ‘–’.

- These two sets are homogeneous

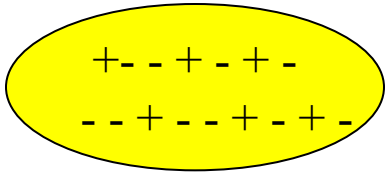




- This one is not



- This one even less so



Information gain

Which attribute to choose for splitting?

- A statistical property called *information gain* measures how well a given attribute separates the training examples into homogeneous groups according to target classification.
- ID3 uses information gain as the splitting criteria for building a tree and chooses the attribute which provides the greatest information gain.
- Information gain is determined using a measure from Information Theory called *Entropy*.

Entropy

In thermodynamics:

- It gives a measure of the amount of chaos present in a system (or a measure of the disorder in a system).

In Information Theory:

- Entropy measures the uncertainty in a random variable or message, or indicates how much information (or impurity) there is in an event.
- In general, the more uncertain or random the event is, the more information it will contain.

Entropy cont…

See Wikipedia:

https://en.wikipedia.org/wiki/Entropy_(information_theory)

- In information theory, systems are modeled by a transmitter, channel, and receiver… The receiver attempts to infer which message was sent. In this context, entropy (more specifically, Shannon entropy) is the expected value (average) of the information contained in each message. ‘Messages’ can be modeled by any flow of information…

Calculating entropy

For a two-class problem: c_1 and c_2:

- P indicates the probability of belonging to each class, the number in each class is $N_{c1} + N_{c2} = N$.

$$\begin{aligned} \text{Entropy(S)} &= -P_{c1}\log_2(P_{c1}) - P_{c2}\log_2(P_{c2}) \\ &= -\frac{N_{c1}}{N}\log_2(\frac{N_{c1}}{N}) - \frac{N_{c2}}{N}\log_2(\frac{N_{c2}}{N}) \end{aligned}$$

For a multi-class problem

$$\begin{aligned} \text{Entropy(S)} &= -\sum_{i=1}^{C} P_i \log_2(P_i) \\ &= -\sum_{i=1}^{C} \frac{N_i}{N}\log_2(\frac{N_i}{N}) \end{aligned}$$

Calculating entropy

Suppose S is a collection of 14 examples, 9 positive and 5 negative → [9+ ,5-]

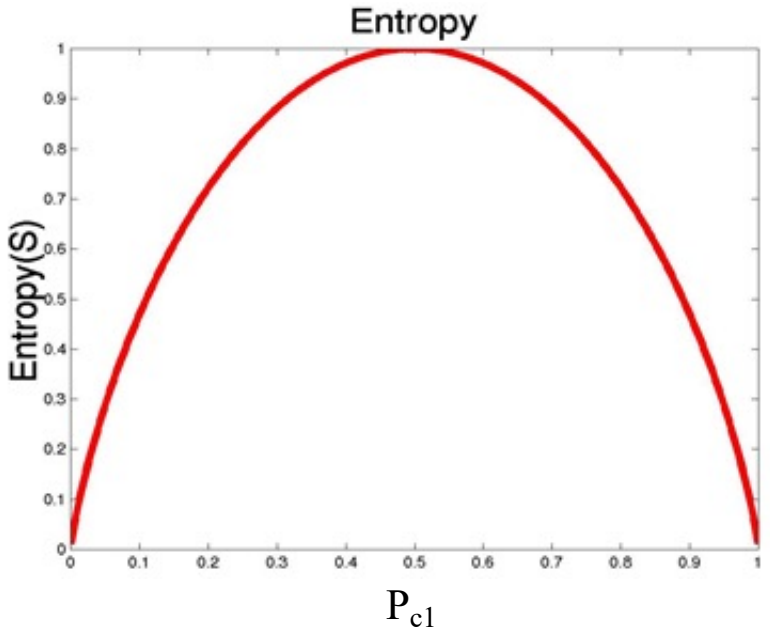
(+, -, +, -, +, -, +, +, +, -, -, +, +, +)

$$\begin{aligned} \text{Entropy}(S) &= -P_{c1}\log_2(P_{c1}) - P_{c2}\log_2(P_{c2}) \\ &= -\frac{9}{14}\log_2(\frac{9}{14}) - \frac{5}{14}\log_2(\frac{5}{14}) \\ &= 0.940 \end{aligned}$$

Suppose S has all positive or all negative Examples, then Entropy(S) = 0

(+, +, +, +, +, +, +, +, +), or

(-, -, -, -, -, -, -, -, -, -, -, -, -, -)



Entropy is 0 (minimum) if all members belong to the same class. Entropy is 1 (maximum) if the collection consists of equal number of positive and negative examples. Assume: $0\text{Log}_2 0 = 0$.

Calculating entropy

The previous example as a spreadsheet:

- If your calculator can't work out logs to base 2 then use the following:

$$\log_2(x) = \frac{\log_{10}(x)}{\log_{10}(2)} \approx \frac{\log_{10}(x)}{0.3010}$$

Yes	No	P(Yes)	Log2(Yes)	P(No)	Log2(No)	Entropy
9	5	0.6429	-0.6374	0.3571	-1.4854	0.9403

- Note: $\log_2$ yields entropy in units called "shannons"

Information gain

Information gain is the expected reduction in entropy caused by partitioning the examples according to an attribute A.

- Gain(S,A) of an attribute A, relative to a collection of examples S (with v groups having $|S_v|$ elements) is:

$$\text{Gain}(S,A) = \text{Entropy}(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \times \text{Entropy}(S_v)$$

Entropy before split

Expected entropy after split

How ID3 uses information gain

The algorithm ‘splits’ on the attribute that provides the most information gain – that is, gives the purest class breakdown at each step in the decision tree.

Recall: purer class = entropy reduction!

How ID3 uses information gain

The algorithm:

- At each step, determine the “best” decision attribute, A, for next node.
- Assign A as decision attribute for node.
- For each value of A create a new descendant.
- Sort training examples to that node according to the attribute value of the branch.
- If all training examples are perfectly classified (same value of target attribute) stop, else iterate over new leaf nodes.

Example: playing tennis

Build a decision tree for playing tennis based on weather conditions.

Training set (S):

Day	Outlook	Temperature	Humidity	Wind	Play
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

First split on: Outlook, Temperature, Humidity or Wind?

Terminology

- *Instance*: single row in a data set. (each observation).
- *Attribute*: an aspect of an instance. Also called feature, variable. (usually each column variable).
- *Value*: category that an attribute can take.
- *Class*: the thing to be learned. (This is what we are trying to classify from the attributes).
- It is usual to have several decision attributes and one target attribute.

Playing tennis: initial entropy

Training set (*S*): Initial entropy before splitting based on 9 Yes/5 No:

Day	Outlook	Temperature	Humidity	Wind	Play
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Yes	No	P(Yes)	Log2(Yes)	P(No)	Log2(No)	Entropy
9	5	0.6429	-0.6374	0.3571	-1.4854	0.9403

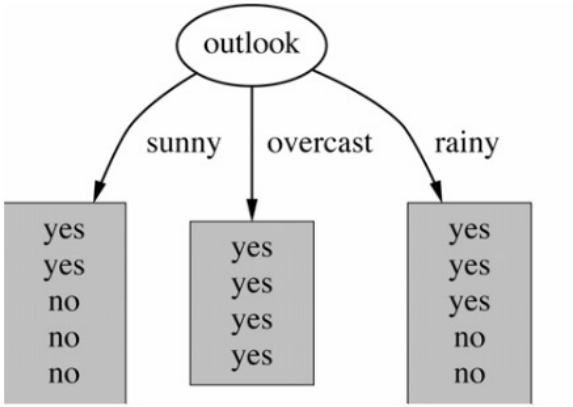
Initial entropy

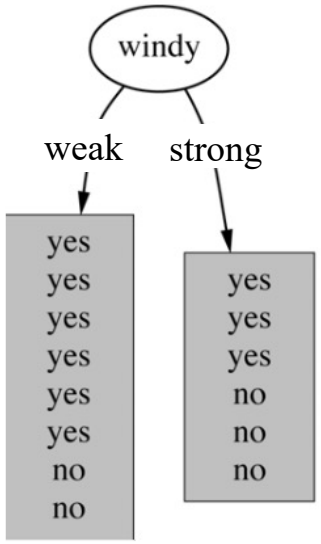
- Without any knowledge of the weather there are 9 Yes and 5 No cases. Initial entropy is:
- $E(S) = -\frac{9}{14} \cdot log_2\left(\frac{9}{14}\right) - \frac{5}{14} \cdot log_2\left(\frac{5}{14}\right)$
- $E(S) = 0.9403$

Which attribute to select?

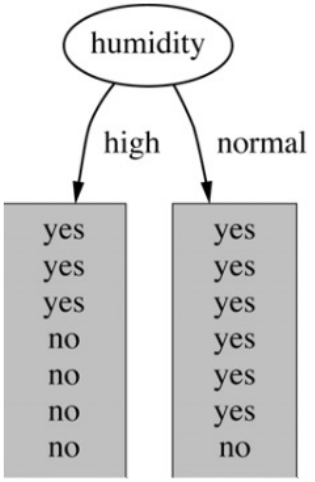
Remember - ID3 chooses the attribute which gives the greatest information gain (reduction in Entropy), or the 'purest' result.

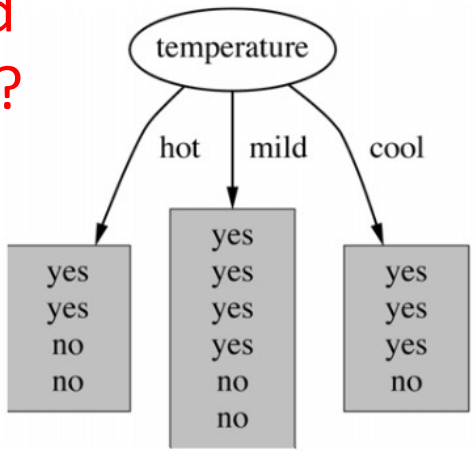
We next calculate the information gain for each attribute in turn.





What would you choose?

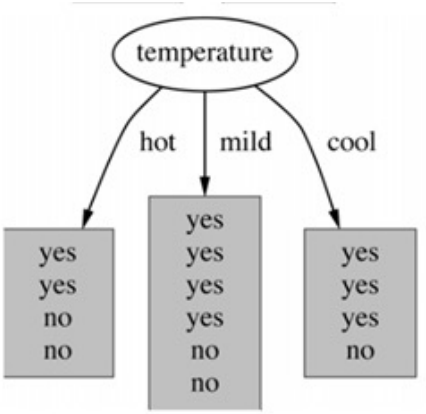




Information gain: Temperature

Calculate entropy for each branch first:

- $E(S_{hot}) = -\frac{2}{4} \cdot log_2\left(\frac{2}{4}\right) - \frac{2}{4} \cdot log_2\left(\frac{2}{4}\right) = 1$
- $E(S_{mild}) = -\frac{4}{6} \cdot log_2\left(\frac{4}{6}\right) - \frac{2}{6} \cdot log_2\left(\frac{2}{6}\right) = 0.918$
- $E(S_{cool}) = -\frac{3}{4} \cdot log_2\left(\frac{3}{4}\right) - \frac{1}{4} \cdot log_2\left(\frac{1}{4}\right) = 0.811$



Now calculate expected entropy and information gain

- $Gain(S, Temp) = E(S) - E(S, Temp)$
- $Gain(S, Temp) = E(S) - \left(\frac{4}{14}1 + \frac{6}{14}0.918 + \frac{4}{14}0.811\right)$
- $= 0.9403 - 0.910$
- $= 0.0292$

Expected entropy is the sum of entropy * probability for each branch

Information gain: Temperature

As a spreadsheet showing initial entropy and subsequent information gain:

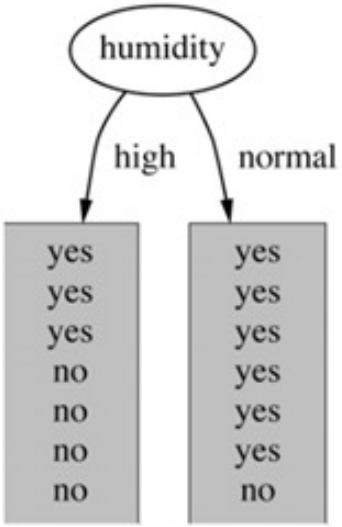
Initial State	Yes	No	P(Yes)	Log2(Yes)	P(No)	Log2(No)	Entropy
Entropy(S)	9	5	0.6429	-0.6374	0.3571	-1.4854	0.9403

Temperature	Yes	No	P(Yes)	Log2(Yes)	P(No)	Log2(No)	Entropy
Hot	2	2	0.5000	-1.0000	0.5000	-1.0000	1.0000
Mild	4	2	0.6667	-0.5850	0.3333	-1.5850	0.9183
Cool	3	1	0.7500	-0.4150	0.2500	-2.0000	0.8113
EEntropy(Temp)							0.9111
Gain(S, Temp)							**0.0292**

Information gain: Humidity

Calculate entropy for each branch first:

- $E(S_{high}) = -\frac{3}{7} \cdot log_2\left(\frac{3}{7}\right) - \frac{4}{7} \cdot log_2\left(\frac{4}{7}\right) = 0.9852$
- $E(S_{normal}) = -\frac{6}{7} \cdot log_2\left(\frac{6}{7}\right) - \frac{1}{7} \cdot log_2\left(\frac{1}{7}\right) = 0.5917$



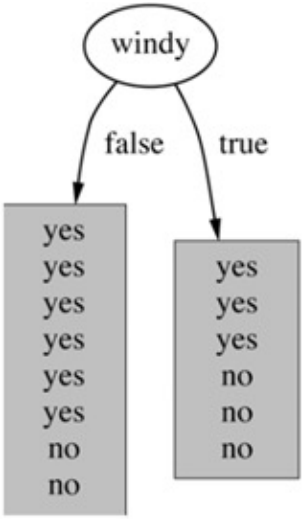
Now calculate expected entropy and information gain

- $Gain(S, Humidity) = E(S) - E(S, Humidity)$
- $Gain(S, Humidity) = E(S) - \left(\frac{7}{14}0.9852 + \frac{7}{14}0.5917\right)$
- $= 0.9403 - 0.7885$
- $= 0.1518$

Information gain: Windy (for you to do)

Calculate entropy for each branch first:

- $E(S_{false}) = -\frac{6}{8} \cdot log_2\left(\frac{6}{8}\right) - \frac{2}{8} \cdot log_2\left(\frac{2}{8}\right) =$ ☐
- $E(S_{true}) = -\frac{3}{6} \cdot log_2\left(\frac{3}{6}\right) - \frac{3}{6} \cdot log_2\left(\frac{3}{6}\right) =$ ☐



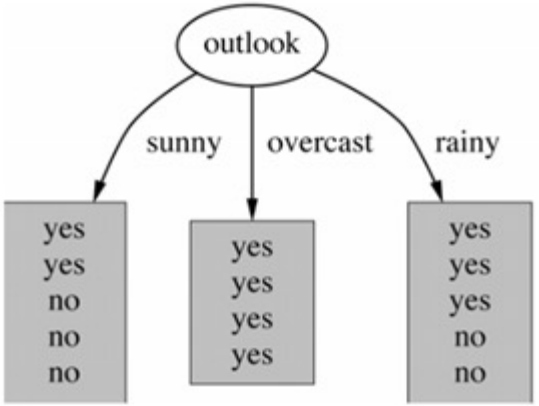
Now calculate expected entropy and information gain

- $Gain(S, Windy) = E(S) - E(S, Windy)$
- $Gain(S, Windy) = E(S) - \left(\frac{8}{14}$ ☐ $+ \frac{6}{14}$ ☐ $\right)$
- $= 0.9403 -$ ☐ $=$ ☐

Information gain: Outlook (for you to do)

Calculate entropy for each branch first:

- $E(S_{sunny}) = -\frac{2}{5} \cdot log_2\left(\frac{2}{5}\right) - \frac{3}{5} \cdot log_2\left(\frac{3}{5}\right) =$ ☐
- $E(S_{overcast}) = 0$
- $E(S_{rainy}) = -\frac{3}{5} \cdot log_2\left(\frac{3}{5}\right) - \frac{2}{5} \cdot log_2\left(\frac{2}{5}\right) =$ ☐



Now calculate expected entropy and information gain

- $Gain(S, Outlook) = E(S) - E(S, Outlook)$
- $Gain(S, Outlook) = E(S) - \left(\frac{5}{14}$ ☐ $+ \frac{4}{14}$ ☐ $+ \frac{5}{14}$ ☐ $\right)$
- $= 0.9403 -$ ☐ $=$ ☐

Calcs: Humidity, Windy, Outlook

Humidity	**Yes**	**No**	**P(Yes)**	**Log2(Yes)**	**P(No)**	**Log2(No)**	**Entropy**
High	3	4	0.4286	-1.2224	0.5714	-0.8074	0.9852
Normal	6	1	0.8571	-0.2224	0.1429	-2.8074	0.5917
EEntropy(Humidity)							0.7885
Gain(S, Humidity)							**0.1518**

Wind	**Yes**	**No**	**P(Yes)**	**Log2(Yes)**	**P(No)**	**Log2(No)**	**Entropy**
Weak	6	2	0.7500	-0.4150	0.2500	-2.0000	0.8113
Strong	3	3	0.5000	-1.0000	0.5000	-1.0000	1.0000
EEntropyWind)							0.8922
Gain(S, Wind)							**0.0481**

Outlook	**Yes**	**No**	**P(Yes)**	**Log2(Yes)**	**P(No)**	**Log2(No)**	**Entropy**
Sunny	2	3	0.4000	-1.3219	0.6000	-0.7370	0.9710
Overcast	4	0	1.0000	0.0000	0.0000	0.0000	0.0000
Rain	3	2	0.6000	-0.7370	0.4000	-1.3219	0.9710
EEntropy(Outlook)							0.6935
Gain(S, Outlook)							**0.2467**

Attribute giving greatest information gain

Which attribute to choose? Outlook, Temperature, Humidity or Wind?

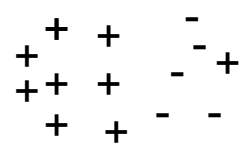
Gain(S, Temperature) = 0.029

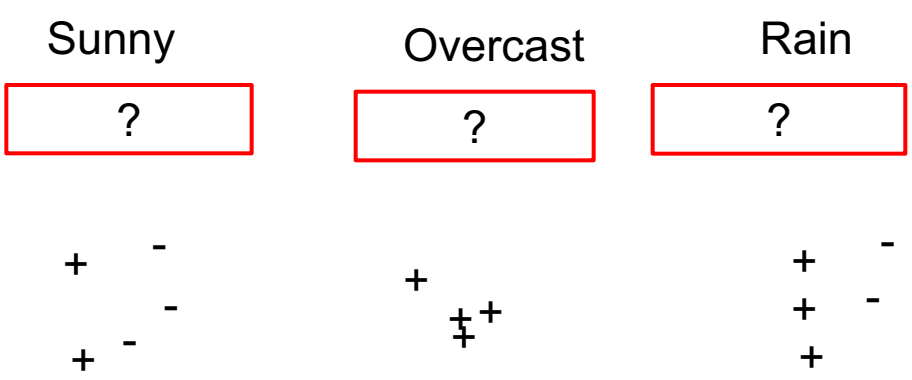
Gain(S, Humidity) = 0.151

Gain(S, Wind) = 0.048

Gain(S, Outlook) = 0.247

Choose this one!





Should we split again? If so Which attribute should we split on next?

Impure, non-leaf node. Apply splitting procedure again.

Pure, make it a leaf node.

Impure, non-leaf node. Apply splitting procedure again.

Entropy after "Outlook"

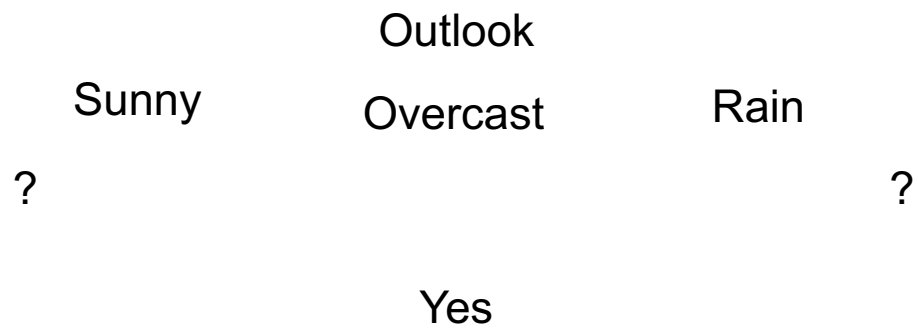
The entropy of each branch of the decision tree after split on Outlook is shown below.

- Information gain in descendent trees is now measured as change in the entropy of each branch.
- For example, Entropy(Sunny) = 0.971

Outlook	Yes	No	P(Yes)	og2(Yes)	P(No)	.og2(No)	Entropy
Sunny	2	3	0.400	-1.322	0.600	-0.737	0.971
Overcast	4	0	1.000	0.000	0.000	0.000	0.000
Rain	3	2	0.600	-0.737	0.400	-1.322	0.971
EEntropy(Outlook)							0.694

Which attribute to split on next?

Now, starting with Sunny, which attribute should be split on next? Temperature, Humidity or Wind?

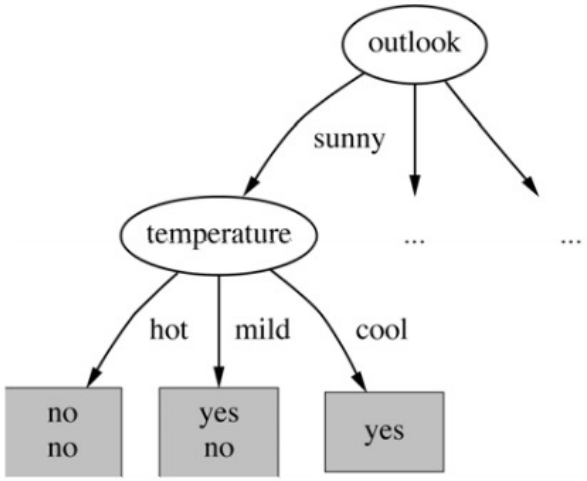


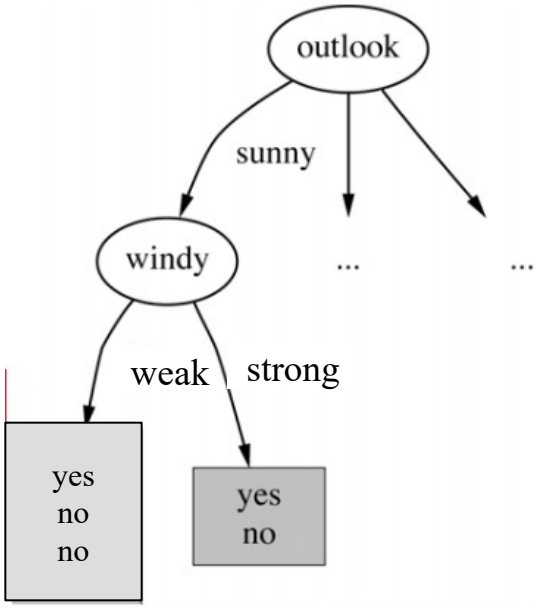
ID3 Step 2: gain(S_sunny, ???)

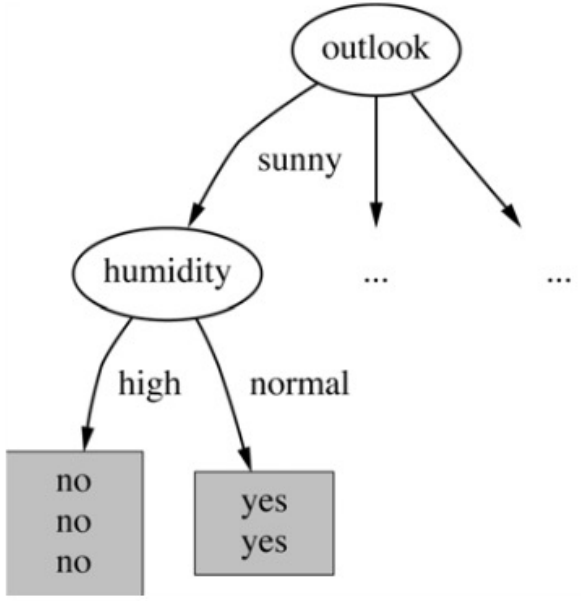
Now consider subset corresponding to “Sunny”:

Day	Outlook	Temperature	Humidity	Wind	Play
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Which attribute to split on next?







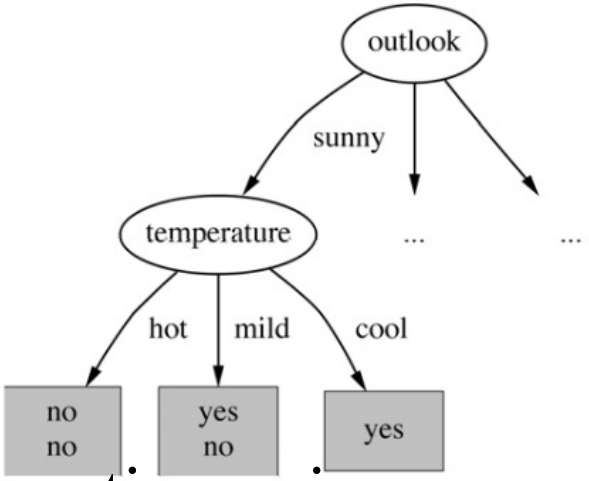
What attribute would you choose?

Do you need to do any calculations?

ID3 Step 2: Gain Sunny, Temperature

Calculate entropy for each branch first:

- $E(S_{sunny,\,hot}) = -\frac{0}{2} \cdot log_2\left(\frac{0}{2}\right) - \frac{2}{2} \cdot log_2\left(\frac{2}{2}\right) = 0$
- $E(S_{sunny,mild}) = -\frac{1}{2} \cdot log_2\left(\frac{1}{2}\right) - \frac{1}{2} \cdot log_2\left(\frac{1}{2}\right) = 1$
- $E(S_{sunny,cool}) = -\frac{1}{1} \cdot log_2\left(\frac{1}{1}\right) - \frac{0}{1} \cdot log_2\left(\frac{0}{1}\right) = 0$



Now calculate expected entropy and information gain

- $Gain(S, Sunny, Temp) = E(S, Sunny) - E(S, Sunny, Temp)$
- $Gain(S, Sunny, Temp) = E(S, Sunny) - \left(\frac{2}{5}0 + \frac{2}{5}1 + \frac{1}{5}0\right)$
- $= 0.971 - 0.4 = 0.571$

Calculations for all attributes shown on the next slide…

ID3 Step 2: Gain for Sunny Outlook

Sunny, Temp	Yes	No	P(Yes)	Log2(Yes)	P(No)	Log2(No)	Entropy
Hot	0	2	0.0000	0.0000	1.0000	0.0000	0.0000
Mild	1	1	0.5000	-1.0000	0.5000	-1.0000	1.0000
Cool	1	0	1.0000	0.0000	0.0000	0.0000	0.0000
EEntropy(Temp)							0.4000
Gain(Sunny, Temp)							**0.5710**

Sunny, Humid	Yes	No	P(Yes)	Log2(Yes)	P(No)	Log2(No)	Entropy
High	0	3	0.0000	0.0000	1.0000	0.0000	0.0000
Normal	2	0	1.0000	0.0000	0.0000	0.0000	0.0000
EEntropy(Temp)							0.0000
Gain(Sunny, Humid)							**0.9710**

Sunny, Wind	Yes	No	P(Yes)	Log2(Yes)	P(No)	Log2(No)	Entropy
Weak	1	2	0.3333	-1.5850	0.6667	-0.5850	0.9183
Strong	1	1	0.5000	-1.0000	0.5000	-1.0000	1.0000
EEntropyWind)							0.9510
Gain(Sunny, Wind)							**0.0200**

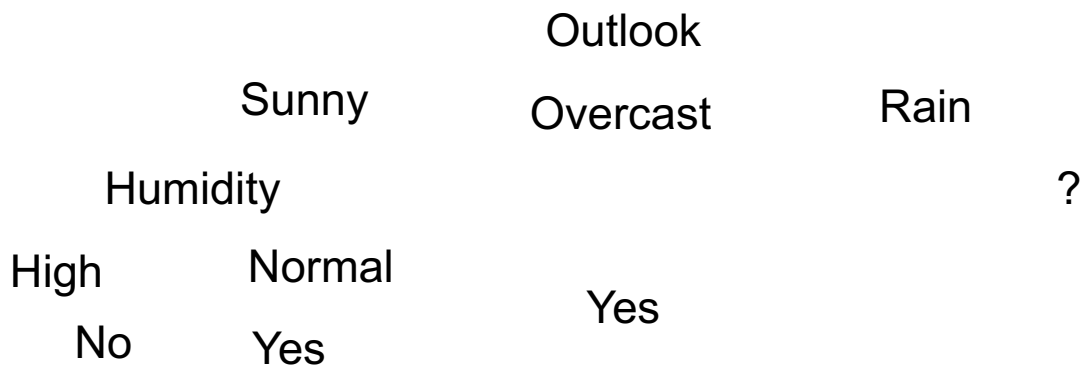
ID3 Step 2: Gain for Sunny Outlook

Which attribute to choose? Temperature, Humidity or Wind?

- $Gain(S_{sunny}, Wind) = 0.020$
- $Gain(S_{sunny}, Humidity) = 0.971$
- $Gain(S_{sunny}, Temperature) = 0.570$

Choose this one!

Repeat the process to find which attribute is the best to split on at this step



Not all leaves need to be 'pure'.
Splitting stops when the data can't be split any further.

Class activity

Now consider subset after “Rain Outlook”:

Day	Outlook	Temperature	Humidity	Wind	Play
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Class activity

Class counts and expected entropy after rain outlook.

Day	Outlook	Temperature	Humidity	Wind	Play
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Rain, Temp	Yes	No
Hot		
Mild		
Cool		

Rain, Humid	Yes	No
High		
Normal		

Rain, Wind	Yes	No
Weak	3	0
Strong	0	2

What would you choose?

ID3 Step 3: Gain for Rain Outlook

Rain, Temp	Yes	No	P(Yes)	Log2(Yes)	P(No)	Log2(No)	Entropy
Hot							
Mild							
Cool							
EEntropy(Temp)							
Gain(Rain, Temp)							

$$\text{Entropy}(S) = -P_{C_1}\log_2\left(P_{C_1}\right) - P_{C_2}\log_2\left(P_{C_2}\right) = -\frac{N_{C_1}}{N}\log_2\left(\frac{N_{C_1}}{N}\right) - \frac{N_{C_2}}{N}\log_2\left(\frac{N_{C_2}}{N}\right)$$

Rain, Humid	Yes	No	P(Yes)	Log2(Yes)	P(No)	Log2(No)	Entropy
High							
Normal							
EEntropy(Temp)							
Gain(Rain, Humid)							

$$\log_2(x) = \frac{\log_{10}(x)}{\log_{10}(2)} \approx \frac{\log_{10}(x)}{0.3010}$$

Rain, Wind	Yes	No	P(Yes)	Log2(Yes)	P(No)	Log2(No)	Entropy
Weak							
Strong							
EEntropyWind)							
Gain(Rain, Wind)							

ID3 Step 3: Gain for Rain Outlook

Rain, Temp	Yes	No	P(Yes)	Log2(Yes)	P(No)	Log2(No)	Entropy
Hot	0	0	0.0000	0.0000	0.0000	0.0000	0.0000
Mild	2	1	0.6667	-0.5850	0.3333	-1.5850	0.9183
Cool	1	1	0.5000	-1.0000	0.5000	-1.0000	1.0000
EEntropy(Temp)							0.9510
Gain(Rain, Temp)							**0.0200**

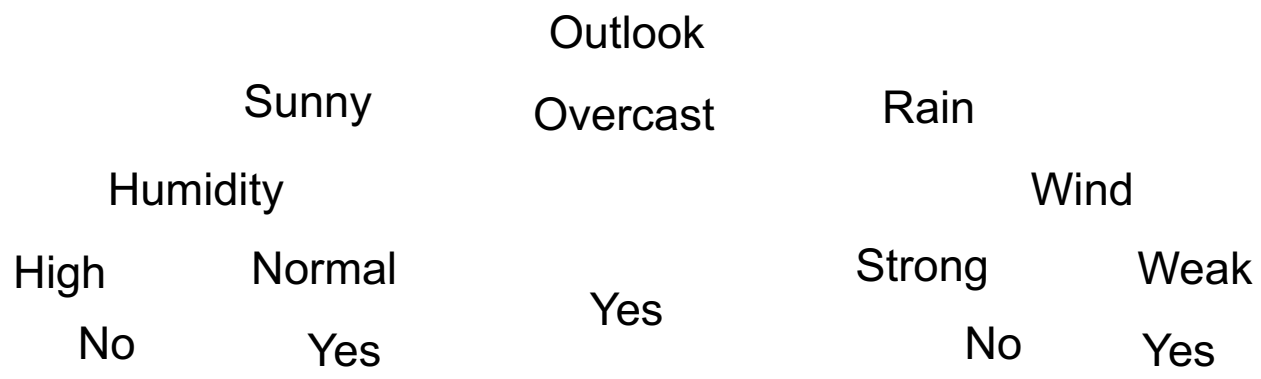
Rain, Humid	Yes	No	P(Yes)	Log2(Yes)	P(No)	Log2(No)	Entropy
High	1	1	0.5000	-1.0000	0.5000	-1.0000	1.0000
Normal	2	1	0.6667	-0.5850	0.3333	-1.5850	0.9183
EEntropy(Temp)							0.9510
Gain(Rain, Humid)							**0.0200**

Rain, Wind	Yes	No	P(Yes)	Log2(Yes)	P(No)	Log2(No)	Entropy
Weak	3	0	1.0000	0.0000	0.0000	0.0000	0.0000
Strong	0	2	0.0000	0.0000	1.0000	0.0000	0.0000
EEntropyWind)							0.0000
Gain(Rain, Wind)							**0.9710**

The Final tree

The process of selecting a new attribute and partitioning the training examples is repeated for each non-leaf node, this time only using the examples associated with that node.
Attributes that have been incorporated higher in the tree are excluded, so that any given attribute can appear at most once along any path through the tree.
The process continues for each leaf node until:

- every attribute has been included along that path through the tree or
- the training examples associated with this leaf node all have the same class.



The Decision Tree Rules

In addition to generating a tree structure, explicit rules for classifying 'play/don't play' are also generated:

If outlook = Overcast Then Play= Yes {No=0, Yes=4}

If outlook = Rain And wind = Strong Then Play= No {No=2, Yes=0}

If outlook = Rain And wind = Weak Then Play = Yes {No=0, Yes=3}

If outlook = Sunny And humidity = High Then Play = No {No=3, Yes=0}

If outlook = Sunny And humidity = Normal Then Play = Yes {No=0, Yes=2}

Further considerations

Types of decision trees?

- Classification Trees (categorical – nominal attributes)
- Regression Trees (numerical – continuous attributes)

How do we specify the splitting conditions?

How do we evaluate the decision tree model?

Further considerations

Classification Trees *vs* Regression Trees

- Target variable types

Splitting criteria:

- Information gain
- Gain ratio (reduces bias for highly branched attributes)
- Gini index

Decision tree algorithms

- ID3 (discrete), C4.5, C5 (continuous) target attributes
- CART, Chaid etc.
- See: https://en.wikipedia.org/wiki/Decision_tree_learning

Metrics for Performance Evaluation

How to evaluate the performance of a model?

- Training and testing
- Confusion matrix
- Cross validation

Training and testing

- You can measure a classifier's performance in terms of the error rate (proportion of errors made over a whole set of instances).
- Due to desirability of generalization, low error on the training data is not a good measure.
- To predict the performance of a classifier on a new data, we need to assess its error on data that was not used to build the model.
- In general, the data set is divided into two subsets: training and testing. Training for learning the model and testing for determining how well it will do on unseen data.

Training Data Testing Data

Performance evaluation of the Model

Focus on the predictive capability of a model

- Rather than how fast it takes to classify or build models, scalability, etc.

How to determine accuracy of decision tree in classifying/predicting?

- Usual to have two data sets:
  - A *Training Set* and a *Test Set*
  - This can be created by dividing the data set into two sets – e.g. 70%/30%
- We create the decision tree model using the training set.
- Then run the test set through the model to find out what the predicted class is.
- Then compare the predicted class with the actual class to see how accurate the model is.

Metrics for Performance Evaluation

One way of assessing performance is to calculate accuracy based on a *confusion matrix* (for the test data classification).

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

a: TP (true positive) b: FN (false negative)

c: FP (false positive) d: TN (true negative)

Metrics for Performance Evaluation

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

Most widely-used metric:

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

Also:

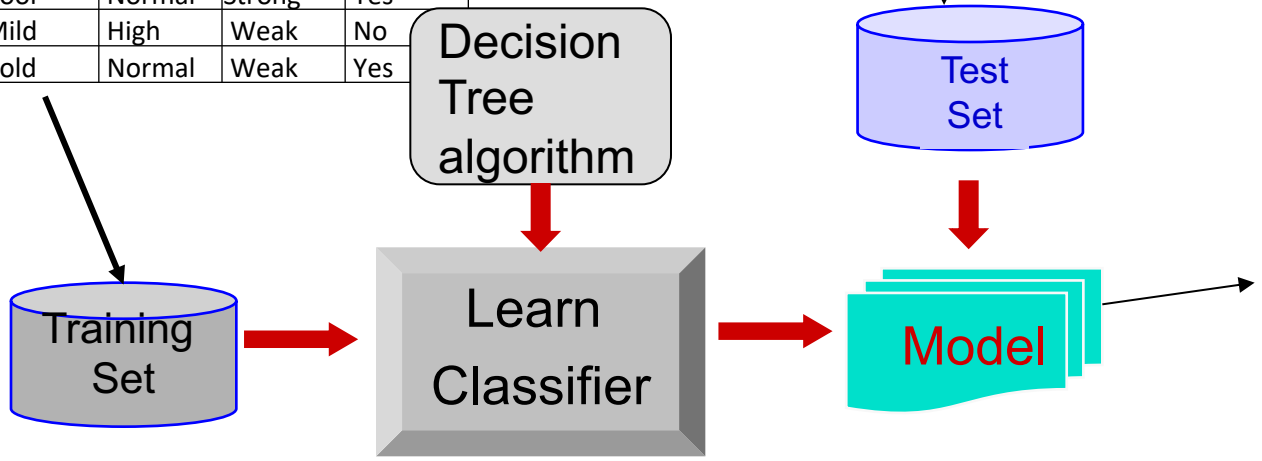
Precision = TP/(TP + FP)

Sensitivity = TP /(TP + FN)

The Play Tennis example

ID	outlook	temp	humidity	wind	play
D1	Sunny	Hot	High	Weak	No
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes

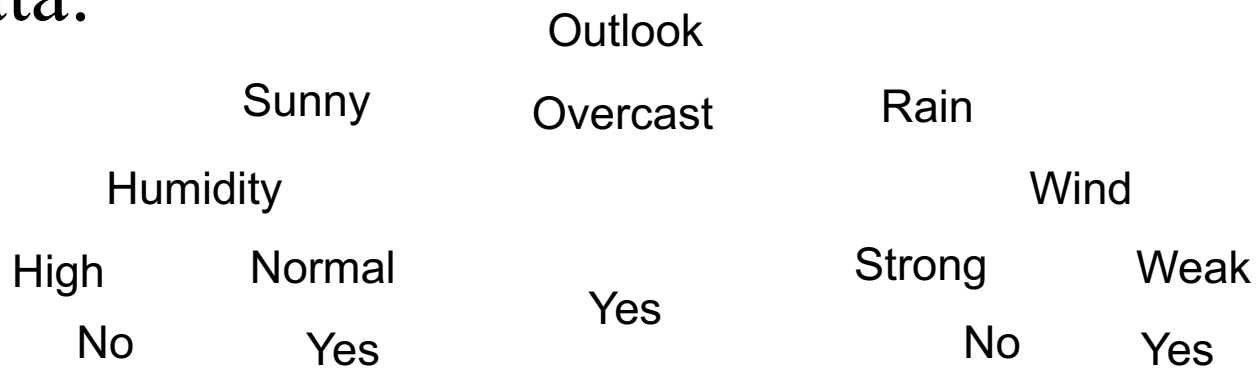
ID	outlook	temp	humidity	wind	play
D15	Sunny	Mild	Normal	Strong	No
D16	Sunny	Hot	High	Weak	Yes
D17	Rain	Hot	High	Weak	No
D18	Overcast	Cool	High	Strong	No
D19	Overcast	Mild	Normal	Weak	Yes
D20	Rain	Mild	Normal	Weak	Yes



actual class	Predicted class		
		Class=Yes	Class=No
	Class=Yes	a	b
	Class=No	c	d

The play tennis example

Let's see what our model would predict using the test data:



Day	Outlook	Temperature	Humidity	Wind	Play	Predict
D15	Sunny	Mild	Normal	Strong	No	yes
D16	Sunny	Hot	High	Weak	Yes	no
D17	Rain	Hot	High	Weak	No	yes
D18	Overcast	Cool	High	Strong	No	yes
D19	Overcast	Mild	Normal	Weak	Yes	yes
D20	Rain	Mild	Normal	Weak	Yes	yes

Metrics for Performance Evaluation…

		PREDICTED CLASS	
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	2 (TP)	1 (FN)
	Class=No	3 (FP)	0 (TN)

The most widely-used metric:

$$Accuracy = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN} = \frac{2+0}{6} = 33.3\%$$

More ways to measure classification performance next lecture...

Decision trees in R

Decision trees in R

There are a number of packages to create decision trees in R. We will start with the "tree" package.

```
> install.packages("tree")
> library(tree)
```

Note: "tree" aims to minimise 'impurity' by binary splitting. Similar, but not identical, to ID3 in effect.

https://cran.r-project.org/web/packages/tree/index.html

tree: details

- A tree is grown by binary recursive partitioning using the response in the specified formula and choosing splits from the terms of the right-hand-side. Numeric variables are divided into X < a and X > a;
- The levels of an unordered factor are divided into two non-empty groups.
- The split which maximizes the reduction in impurity is chosen, the data set split, and the process repeated.
- Splitting continues until the terminal nodes are too small or too few to be split.

https://cran.r-project.org/web/packages/tree/index.html

Classification tree: data

Build and test a model using the playtennis data.

Day	Outlook	Temperature	Humidity	Wind	Play
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Classification tree: data

Build and test a model using the playtennis data.

- Ensure inputs are factors – not character vars.

  > ptt <- read.csv("playtennistrain.csv", stringsAsFactors = T)

- As the training set has too few instances for tree package to fit a model, a larger synthetic data set is created by resampling with replacement.

  > set.seed(9999) #make random selection repeatable

  > # resampling with replacement

  > pttrain = ptt[sample(nrow(ptt), 100, replace = TRUE),]

  Training data has been resampled to make 100 rows.

Classification tree: building the tree

Build the tree using "Play" as the response and all input variables except "Day" as predictors.

Syntax is very similar to linear model function.

Output is a list.

```
> ptfit = tree(Play ~. -Day, data = pttrain)
```

Fit model to all attributes except "Day".

? tree

- Description

  A tree is grown by binary recursive partitioning using the response in the specified formula and choosing splits from the terms of the right-hand-side.

- Usage

```
tree(formula, data, weights, subset,
na.action = na.pass, control =
tree.control(nobs, ...),
method = "recursive.partition",
split = c("deviance", "gini"),
model = FALSE, x = FALSE, y = TRUE, wts = TRUE,
...)
```

Classification tree: summary

Use “summary” to get a basic idea of model performance: terminal nodes, error measures.

```
> summary(ptfit)
Classification tree:
tree(formula = Play ~ . - Day, data = pttrain)
Number of terminal nodes:  7
Residual mean deviance:  0 = 0 / 93
Misclassification error rate: 0 = 0 / 100
```

Classification tree: details

Details of each split, root to leaf, left to right.

- > ptfit

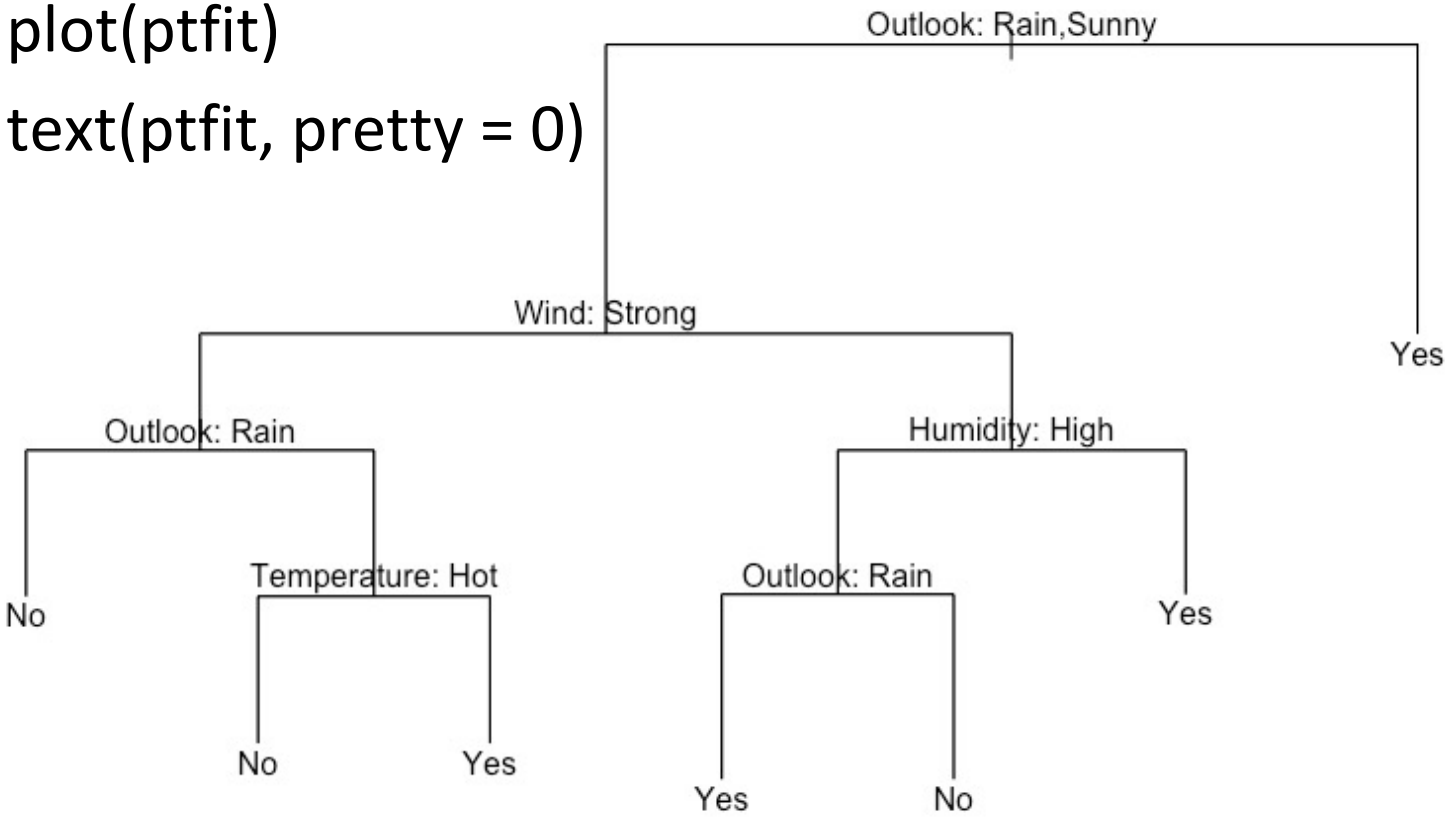
```
> ptfit
node), split, n, deviance, yval, (yprob)
      * denotes terminal node

 1) root 100 100 Yes ( 0.4 0.6 )
   2) Outlook: Rain,Sunny 70 100 No ( 0.6 0.4 )
     4) Wind: Strong 36  40 No ( 0.8 0.2 )
       8) Outlook: Rain 22   0 No ( 1.0 0.0 ) *
       9) Outlook: Sunny 14  20 Yes ( 0.4 0.6 )
        18) Temperature: Hot 6   0 No ( 1.0 0.0 ) *
        19) Temperature: Mild 8   0 Yes ( 0.0 1.0 ) *
     5) Wind: Weak 34  40 Yes ( 0.3 0.7 )
      10) Humidity: High 18  20 No ( 0.6 0.4 )
        20) Outlook: Rain 7   0 Yes ( 0.0 1.0 ) *
        21) Outlook: Sunny 11   0 No ( 1.0 0.0 ) *
      11) Humidity: Normal 16   0 Yes ( 0.0 1.0 ) *
   3) Outlook: Overcast 30   0 Yes ( 0.0 1.0 ) *
```

Classification tree: plot

Headers give rule for left branching.

```
> plot(ptfit)
> text(ptfit, pretty = 0)
```



Classification tree: testing the model

To test the model, make a prediction for each input from the test data set and cross tabulate with the actual classification in the test data:

```
> pttest <- read.csv("playtennistest.csv",
  stringsAsFactors = T)
> tpredict = predict(ptfit, pttest, type = "class")
> Tpredict
[1] Yes No  Yes Yes Yes Yes
Levels: No Yes
```

Classification tree: testing the model

Comparing predicted with actual values as a Confusion Matrix:

```
> tpredict
[1] Yes No  Yes Yes Yes Yes
> pttest$Play
[1] No  Yes No  No  Yes Yes
> table(observed = pttest$Play, predicted = tpredict)
          predicted
observed No Yes
     No   0   3
     Yes  1   2
```

Edgar Anderson's Iris data

50 samples from 3 species:

- Iris setosa, – virginica, – versicolor

Four features measured:

- Sepal width and length
- Petal width and length

Is it possible to distinguish species using physical measurements?

- Data is packaged with R: "iris"

http://en.wikipedia.org/wiki/Iris_flower_data_set

Regression tree: data

Build and test a model using the iris data.

Subset the data into training and test data sets.

```
> # to select 70% of rows
> set.seed(9999) # make random selection repeatable
> train.row = sample(1:nrow(iris), 0.7*nrow(iris))
> iris.train = iris[train.row,]  Row indices for training
> iris.test = iris[-train.row,]  Row indices for testing
```

Regression tree: building and testing

Adapting the same commands used for the tennis example:

```
> itree = tree(Species ~., data = iris.train)
> itree
> summary(itree)
> plot(itree)
> text(itree, pretty = 0)
> ipredict = predict(itree, iris.test, type = "class")
> table(observed = iris.test$Species, predicted = ipredict)
```

Regression tree: summary

Summary of terminal nodes, variables actually used, error measures.

```
> summary(itree)
Classification tree:
tree(formula = Species ~ ., data = iris.train)
Variables actually used in tree construction:
[1] "Petal.Length" "Petal.Width"
Number of terminal nodes:  5
Residual mean deviance:  0.1332 = 13.32 / 100
Misclassification error rate: 0.0381 = 4 / 105
```

Regression tree: details

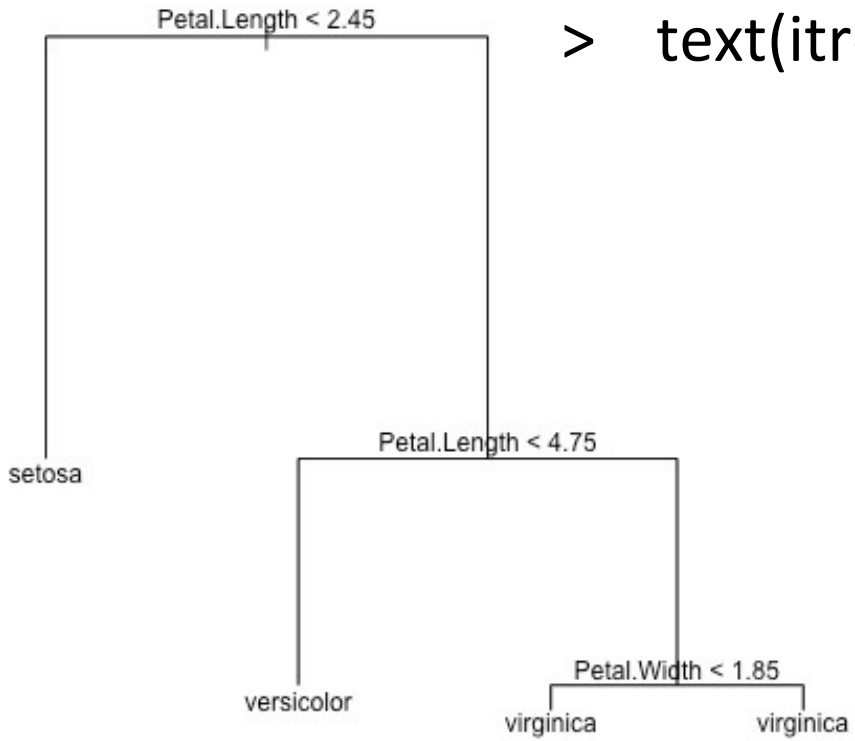
> itree

```
node), split, n, deviance, yval, (yprob)
      * denotes terminal node

 1) root 105 200 setosa ( 0.36 0.30 0.34 )
   2) Petal.Length < 2.45 38   0 setosa ( 1.00 0.00 0.00 ) *
   3) Petal.Length > 2.45 67  90 virginica ( 0.00 0.46 0.54 )
     6) Petal.Length < 4.75 27   0 versicolor ( 0.00 1.00 0.00 ) *
     7) Petal.Length > 4.75 40  30 virginica ( 0.00 0.10 0.90 )
      14) Petal.Width < 1.7 6   8 virginica ( 0.00 0.50 0.50 ) *
      15) Petal.Width > 1.7 34   9 virginica ( 0.00 0.03 0.97 )
        30) Petal.Length < 4.95 5   5 virginica ( 0.00 0.20 0.80 ) *
        31) Petal.Length > 4.95 29   0 virginica ( 0.00 0.00 1.00 ) *
```

Regression tree: plot

```
> plot(itree)
> text(itree, pretty = 0)
```



Classification tree: testing the model

To test the model, make a prediction for each and draw the confusion matrix.

```
> table(observed = iris.test$Species, predicted = ipredict)
                predicted
observed     setosa versicolor virginica
 setosa          13          0         0
 versicolor       0         14         3
 virginica        0          1        14
```

Discussion

How good is each model?

Could the models be improved?

Are they too specific, based on the training set?

Note that previous examples are sensitive to the value of the random seed. If this changed the decision tree model and/or accuracy may change.

More on decision devlopment and testing as well as other classifcation methods next lecture.

Answers to the quiz questions

1. D
2. E
3. C
4. A
5. A

Reading/Notes on the presentation

Further Reading:

- An Introduction to Statistical Learning with applications in R, 2nd Ed, 2021. (Springer Texts in Statistics), James, Witten, Hastie and Tibshirani, Chapter 8 (available on-line from Monash Library)

Notes:

- This presentation contains some slides created to accompany: *Introduction to Data Mining*, Tan, Steinbach, Kumar. Pearson Education Inc., 2006.
- Presentation originally created by Dr. Sue Bedingfield.