ID3 algorithm for learning classification

Simon Dixon
Queen Mary University of London
based on material by Boris Mailhé and Alan Neymark

Outline

- The classification problem
  - geometrical interpretation
- Attribute selection
  - entropy, mutual information
- Some classifiers
  - Bayesian, SVM, decision trees
- The ID3 algorithm
- Avoiding overfitting

Classification

Classification

- Goal
    - given a set of examples described by several attributes (or *features*) and a set of classes, label each example with a class

Classification

Queen Mary University of London

- Goal
  - given a set of examples described by several attributes (or *features*) and a set of classes, label each example with a class
- Applications:
  - biometry: identify people from their voice, face, gait, ...
  - category recognition: e.g. objects in images; speech; musical instrument, style, mood
  - data mining: predict user behaviour depending on their past history
  - medical diagnosis: knowing the symptoms and eventual test results, determine whether a patient has a particular disease

Classification: why is it hard?

Classification: why is it hard?

Queen Mary University of London

- The classes themselves are not observable
  - the classes represent high-level information (identity, behaviour,...) but only low-level attributes (pixel values, numbers of clicks on a web page,...) are observed
  - the human brain is good at making sense of data, but we still don't know how

Classification: why is it hard?

- The classes themselves are not observable
  - the classes represent high-level information (identity, behaviour,...) but only low-level attributes (pixel values, numbers of clicks on a web page,...) are observed
  - the human brain is good at making sense of data, but we still don't know how
- There are many attributes
  - each attribute individually has little correlation with the classes
    - so we need a lot of attributes
  - the correlation between the attributes and the classes is not known beforehand
    - it has to be learnt

Classification: why is it hard?

Queen Mary University of London

- The classes themselves are not observable
  - the classes represent high-level information (identity, behaviour,...) but only low-level attributes (pixel values, numbers of clicks on a web page,...) are observed
  - the human brain is good at making sense of data, but we still don't know how
- There are many attributes
  - each attribute individually has little correlation with the classes
    - so we need a lot of attributes
  - the correlation between the attributes and the classes is not known beforehand
    - it has to be learnt
- There are many examples
  - if not, it could be done by hand
  - learning algorithms may have high computational cost

Machine learning for classification

Queen Mary
University of London

Machine learning for classification

Queen Mary University of London

- Classifier: a function that takes features describing an object and returns the class of that object
  - a classifier can label any possible example

Machine learning for classification

Queen Mary University of London

- Classifier: a function that takes features describing an object and returns the class of that object
    - a classifier can label any possible example
- Machine learning approach
    - learn the classifier from a *training set*, then apply it to the problem examples
        - supervised learning: the classes of the training examples are known
        - the training set is much smaller than the problem set

Machine learning for classification

Queen Mary University of London

- Classifier: a function that takes features describing an object and returns the class of that object
  - a classifier can label any possible example
- Machine learning approach
  - learn the classifier from a *training set*, then apply it to the problem examples
    - supervised learning: the classes of the training examples are known
    - the training set is much smaller than the problem set
- The learning step is costly but performed only once

Machine learning for classification

- Classifier: a function that takes features describing an object and returns the class of that object
  - a classifier can label any possible example
- Machine learning approach
  - learn the classifier from a *training set*, then apply it to the problem examples
    - supervised learning: the classes of the training examples are known
    - the training set is much smaller than the problem set
- The learning step is costly but performed only once
- How can we ensure that the classifier will work on the problem data?
  - we cannot be certain
  - Occam's razor: the simplest solution is often the right one

Classification: geometrical view

Queen Mary
University of London

Classification: geometrical view

- The examples lie in the attribute space
  - each attribute is a dimension
  - each example is a point
  - in these slides, classes are represented by colours

Classification: geometrical view

- The examples lie in the attribute space
  - each attribute is a dimension
  - each example is a point
  - in these slides, classes are represented by colours
- A classifier embodies a partition of the attribute space
  - the whole space is divided into non-overlapping pieces (subspaces)

Classification: geometrical view

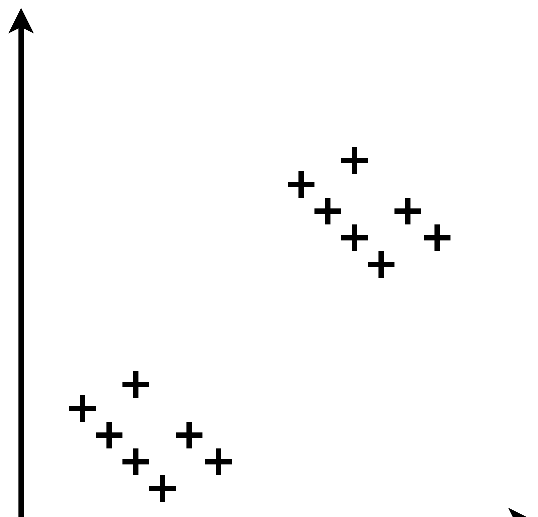
- The examples lie in the attribute space
  - each attribute is a dimension
  - each example is a point
  - in these slides, classes are represented by colours
- A classifier embodies a partition of the attribute space
  - the whole space is divided into non-overlapping pieces (subspaces)
- Learning a classifier is partitioning the training set into homogeneous subsets
  - each subspace contains training examples of exactly one class

Classification: geometrical view

- The examples lie in the attribute space
  - each attribute is a dimension
  - each example is a point
  - in these slides, classes are represented by colours
- A classifier embodies a partition of the attribute space
  - the whole space is divided into non-overlapping pieces (subspaces)
- Learning a classifier is partitioning the training set into homogeneous subsets
  - each subspace contains training examples of exactly one class
- Applying a classifier is therefore just checking in which subspace a point lies

Classification: geometrical view

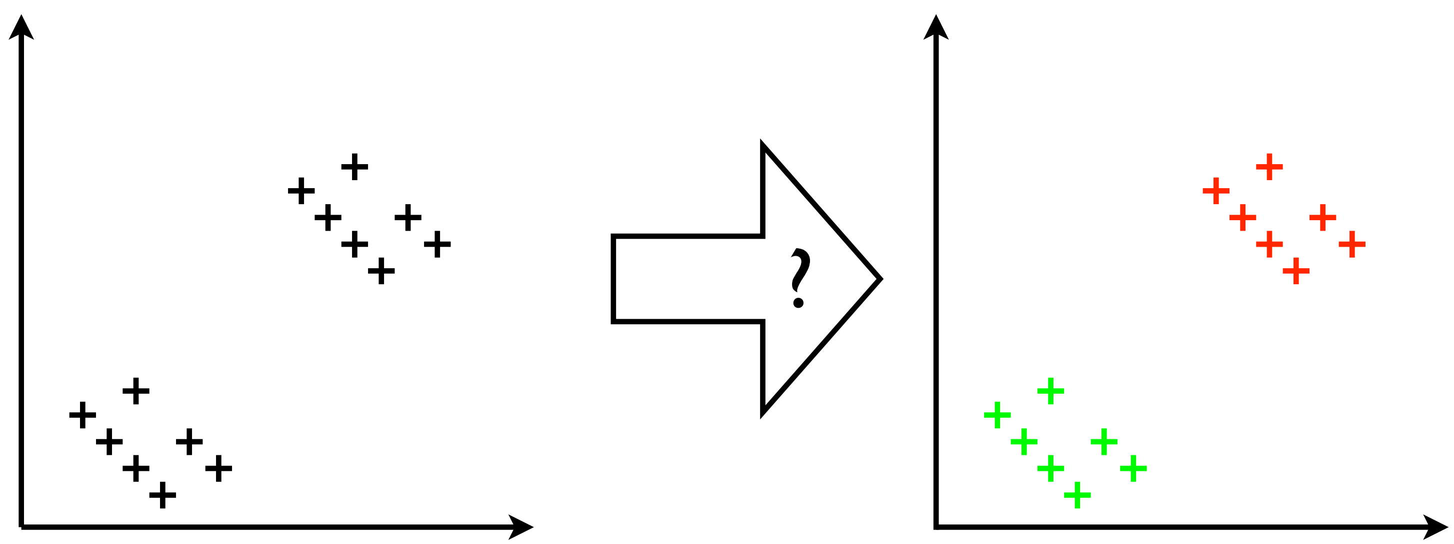
Classification: geometrical view



Problem

- Attributes are dimensions (axes)
- Examples are points

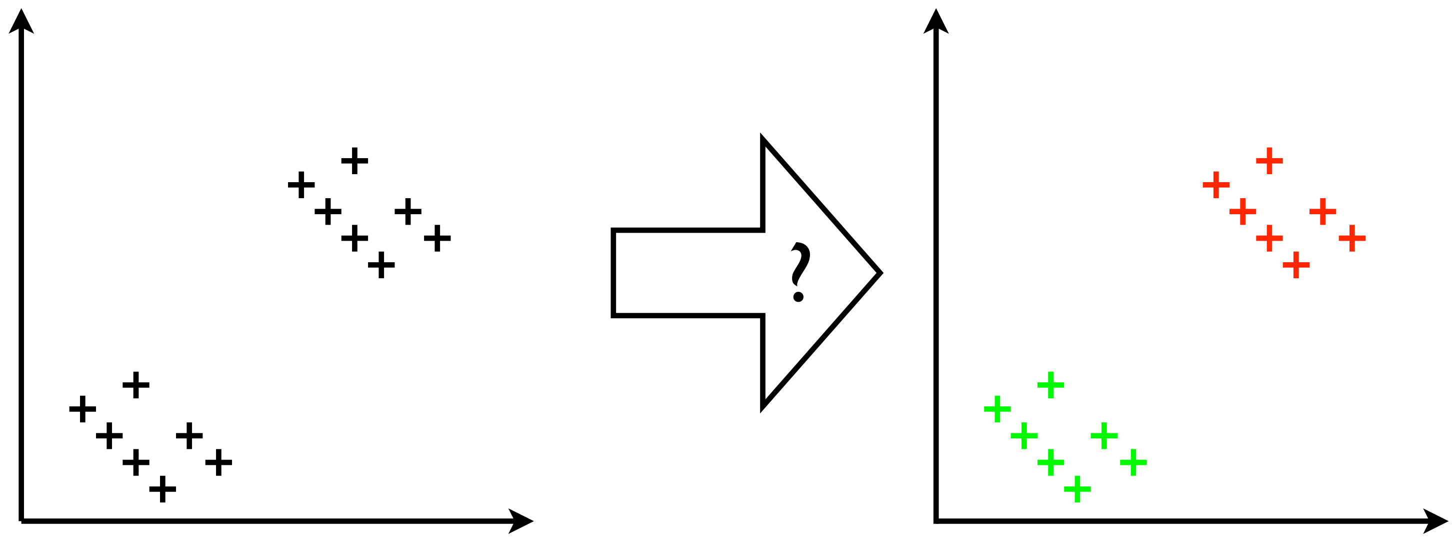
Classification: geometrical view

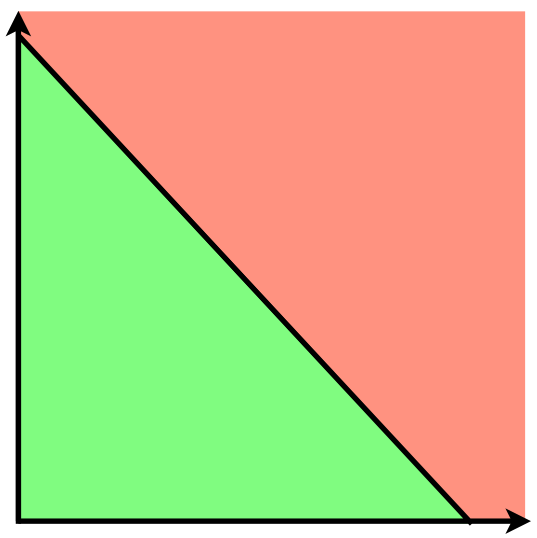


Problem

- Attributes are dimensions (axes)
- Examples are points

Classification: geometrical view





Problem

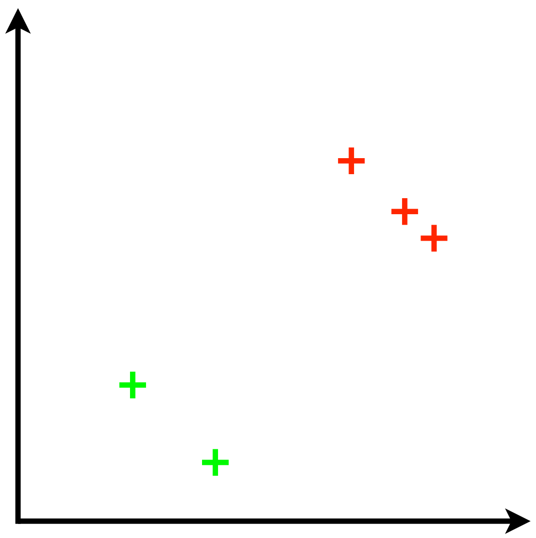
Classifier

- Attributes are dimensions (axes)
- Examples are points

- Colour the whole space

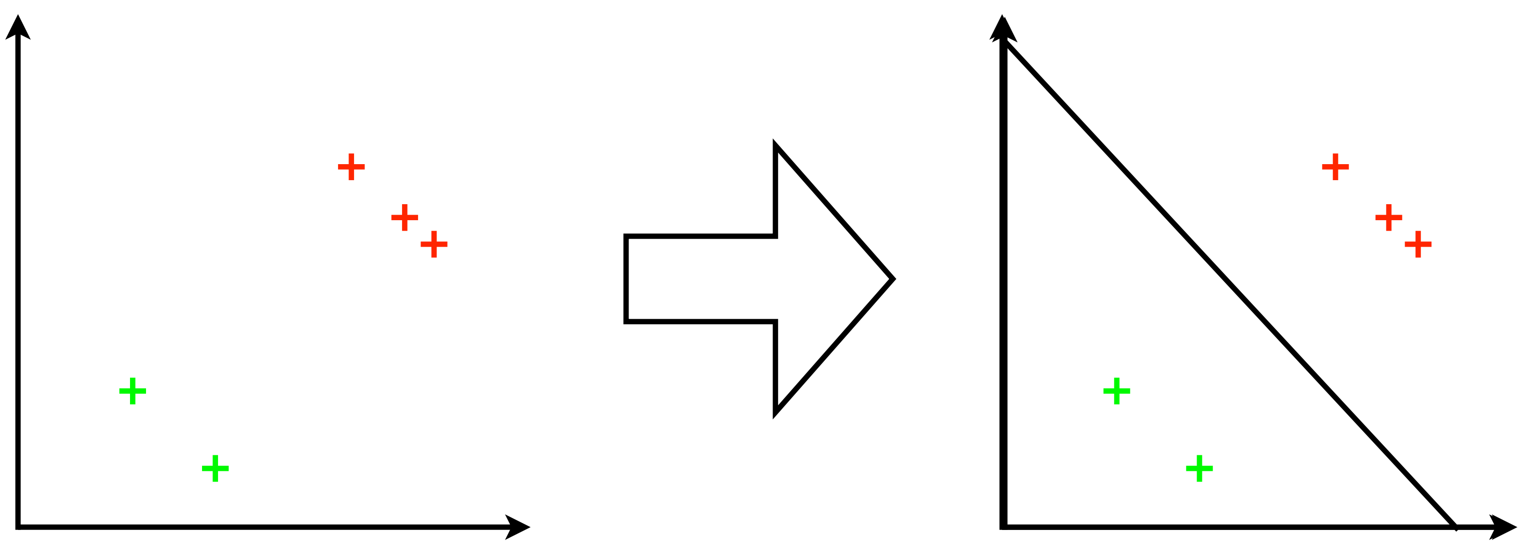
Learning a classifier

Learning a classifier



Select training examples

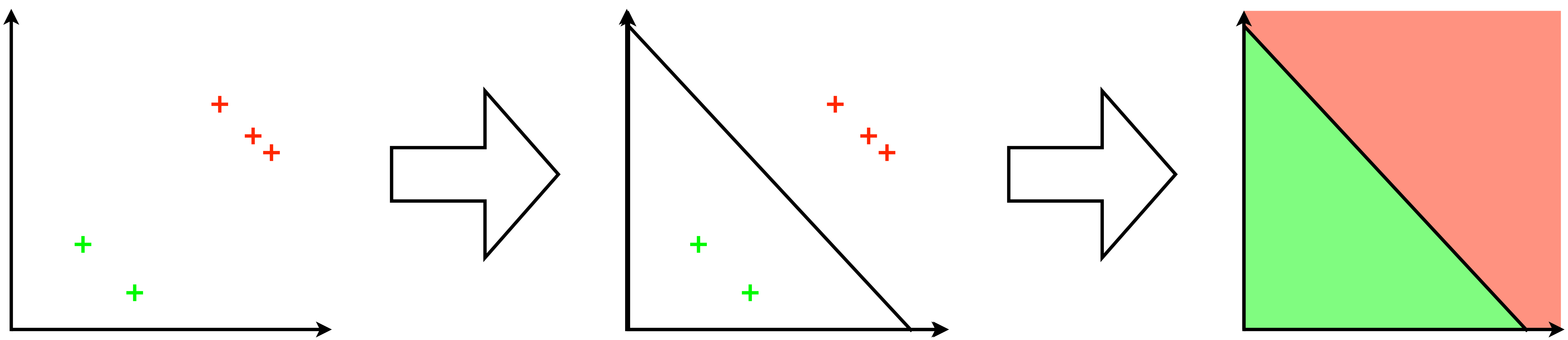
Learning a classifier



Select training examples

Draw the borders

Learning a classifier



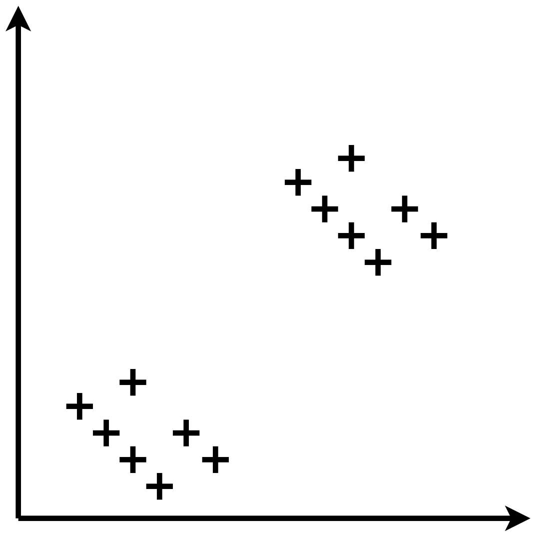
Select training examples

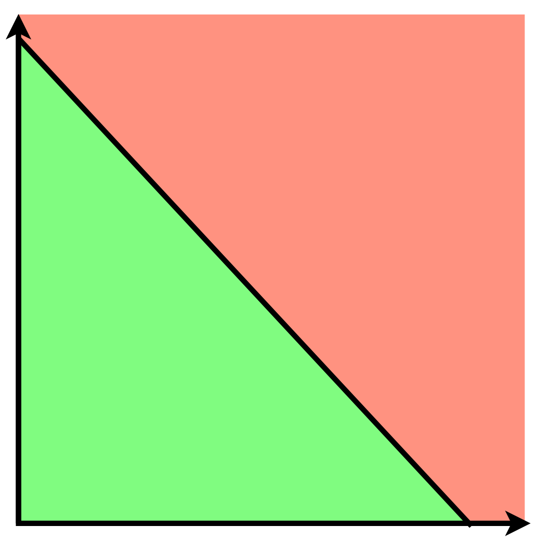
Draw the borders

Colour the subspaces

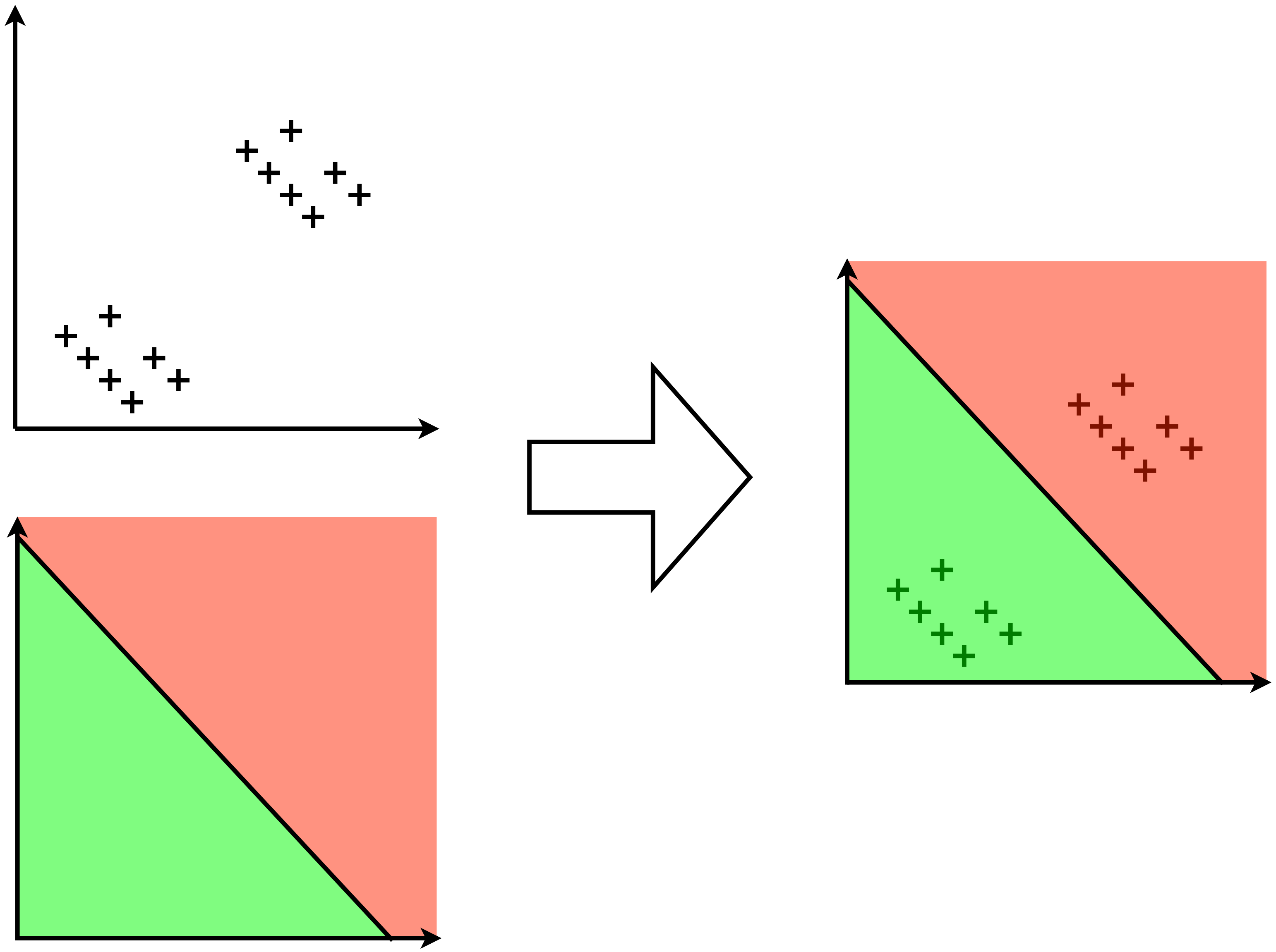
Applying the classifier

Applying the classifier



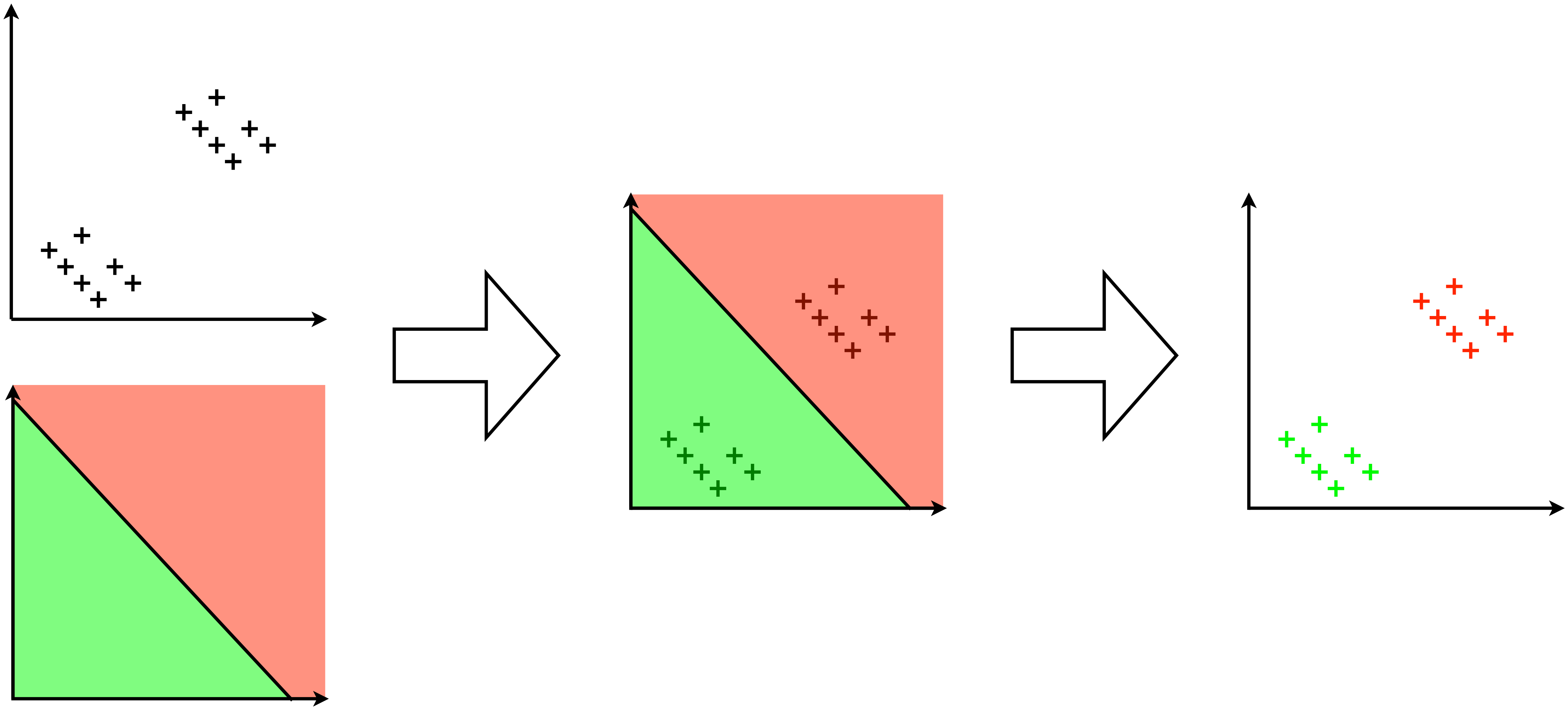


Applying the classifier



Locate the points

Applying the classifier

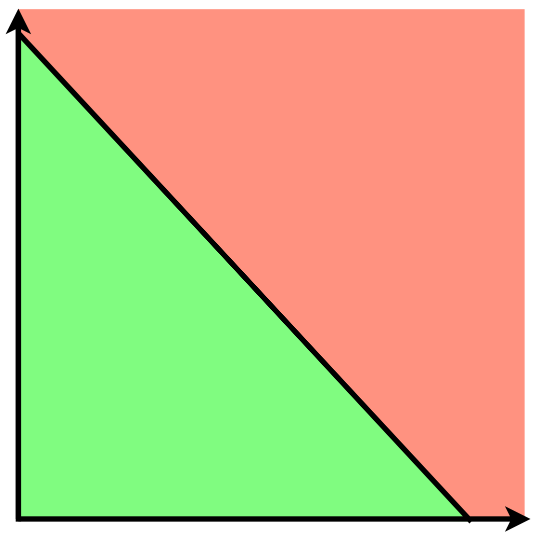


Locate the points

Colour the points

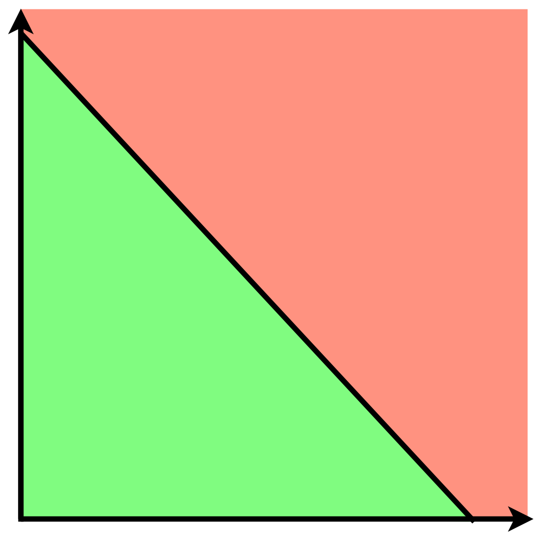
Structure of the classifier

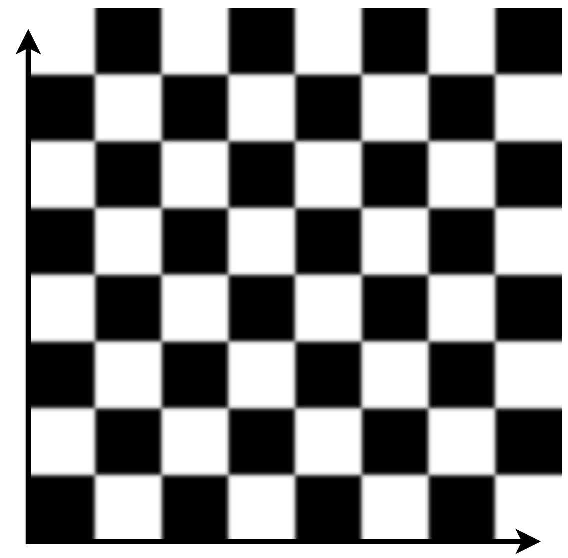
Structure of the classifier



- The subspaces of the classifier do not have to be connected

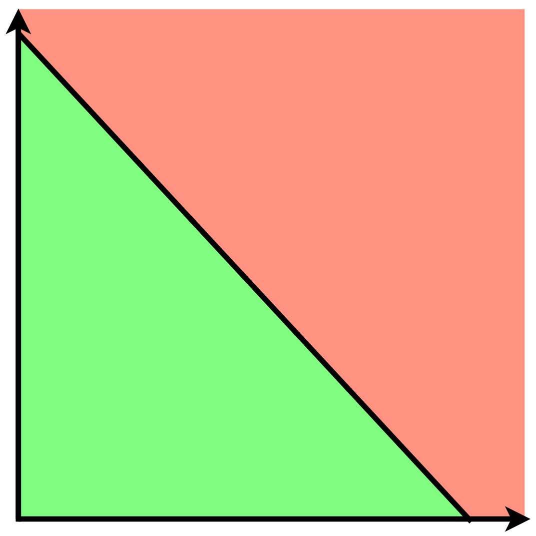
Structure of the classifier

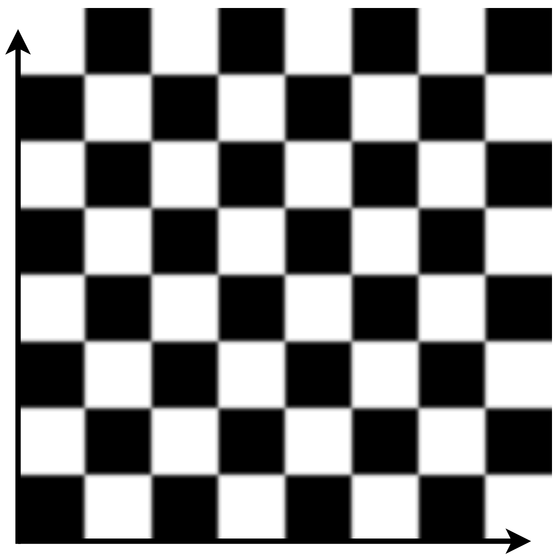




- The subspaces of the classifier do not have to be connected
- The border can be much more complex than a single line

Structure of the classifier





- The subspaces of the classifier do not have to be connected
- The border can be much more complex than a single line
- Generally, the types of borders allowed are set by the algorithm
    - need expert knowledge of the problem
    - there is no single algorithm that solves every problem

Attribute Types

Queen Mary
University of London

Attribute Types

- Symbolic attributes
  - finite, discrete valued: e.g. gender, marital status,...
  - no notion of order or distance between values, only Boolean comparison
  - can be enumerated
  - use *logic* to combine different attributes *after* testing them

Attribute Types

- Symbolic attributes
  - finite, discrete valued: e.g. gender, marital status,...
  - no notion of order or distance between values, only Boolean comparison
  - can be enumerated
  - use *logic* to combine different attributes *after* testing them
- Numeric attributes
  - real or integer valued: e.g. grades, temperature, pixels, audio samples, ...
  - can be measured
  - can use *algebra* to combine different attributes *before* testing them
    - can create new attributes

Attribute selection/engineering

Attribute selection/engineering

Queen Mary
University of London

- Learning is hard. Learning algorithms in general are both costly and suboptimal.

Attribute selection/engineering

- Learning is hard. Learning algorithms in general are both costly and suboptimal.
- Learning is easier if one starts with “good” attributes
  - fewer attributes decrease the cost of learning
  - a good initial guess helps finding the best classifier

Attribute selection/engineering

- Learning is hard. Learning algorithms in general are both costly and suboptimal.
- Learning is easier if one starts with "good" attributes
  - fewer attributes decrease the cost of learning
  - a good initial guess helps finding the best classifier
- What is a "good" attribute?
  - easiest case (trivial): the classes themselves as an attribute
  - hardest case (insoluble): no attributes at all
  - an attribute that carries a lot of "information"
  - an attribute whose "information" is related to the class

Information theory (Shannon 1948)

Queen Mary
University of London

Information theory (Shannon 1948)

- Communication problem: how to compress a signal, i.e. a stream of independent random variables X?
  - e.g.: signal = succession of coin tosses

Information theory (Shannon 1948)

- Communication problem: how to compress a signal, i.e. a stream of independent random variables X?
  - e.g.: signal = succession of coin tosses
- Quantity of information in the signal = minimum bit rate needed to transmit it, i.e. the average number of bits per symbol.

Information theory (Shannon 1948)

Queen Mary University of London

- Communication problem: how to compress a signal, i.e. a stream of independent random variables X?
  - e.g.: signal = succession of coin tosses
- Quantity of information in the signal = minimum bit rate needed to transmit it, i.e. the average number of bits per symbol.
- It is a function of the probability distribution of the signal
  - the distribution represents what is known before the actual transmission, i.e. what does not depend on the samples
  - the bit rate should be chosen before transmission

Information theory (Shannon 1948)

- Communication problem: how to compress a signal, i.e. a stream of independent random variables *X*?
  - e.g.: signal = succession of coin tosses
- Quantity of information in the signal = minimum bit rate needed to transmit it, i.e. the average number of bits per symbol.
- It is a function of the probability distribution of the signal
  - the distribution represents what is known before the actual transmission, i.e. what does not depend on the samples
  - the bit rate should be chosen before transmission
- The more uncertain the distribution, the more information in the content
  - if the coin is double-headed, then there is nothing to transmit

Information theory (Shannon 1948)

Queen Mary
University of London

$$H(X) = -\sum_{x \in X} p_x \log_2 p_x$$

Information theory (Shannon 1948)

Queen Mary
University of London

- Probabilities are multiplicative, information is additive
  - if ones tosses a coin n times, there are 2^n possible outcomes, but they can all be written using n bits only
  - information content depends on the logarithm of the probability distribution

$$H(X) = -\sum_{x \in X} p_x \log_2 p_x$$

Information theory (Shannon 1948)

- Probabilities are multiplicative, information is additive
  - if ones tosses a coin n times, there are 2^n possible outcomes, but they can all be written using n bits only
  - information content depends on the logarithm of the probability distribution
- Impossible events should not matter
  - adding "the coin can rest on its side with probability 0" should not change the amount of information
  - weight the logarithms by the probability p_x of each event x from the set X of all possible events

$$H(X) = -\sum_{x \in X} p_x \log_2 p_x$$

Information theory (Shannon 1948)

Queen Mary University of London

- Probabilities are multiplicative, information is additive
  - if ones tosses a coin n times, there are 2^n possible outcomes, but they can all be written using n bits only
  - information content depends on the logarithm of the probability distribution
- Impossible events should not matter
  - adding "the coin can rest on its side with probability 0" should not change the amount of information
  - weight the logarithms by the probability p_x of each event x from the set X of all possible events
- Entropy:

$$H(X) = -\sum_{x \in X} p_x \log_2 p_x$$

Example: coin toss

Example: coin toss

Queen Mary University of London

- Two values (e.g. Heads and Tails)
  - respective frequencies $p_{Heads} = p$ and $p_{Tails} = 1 - p$

Example: coin toss

Queen Mary University of London

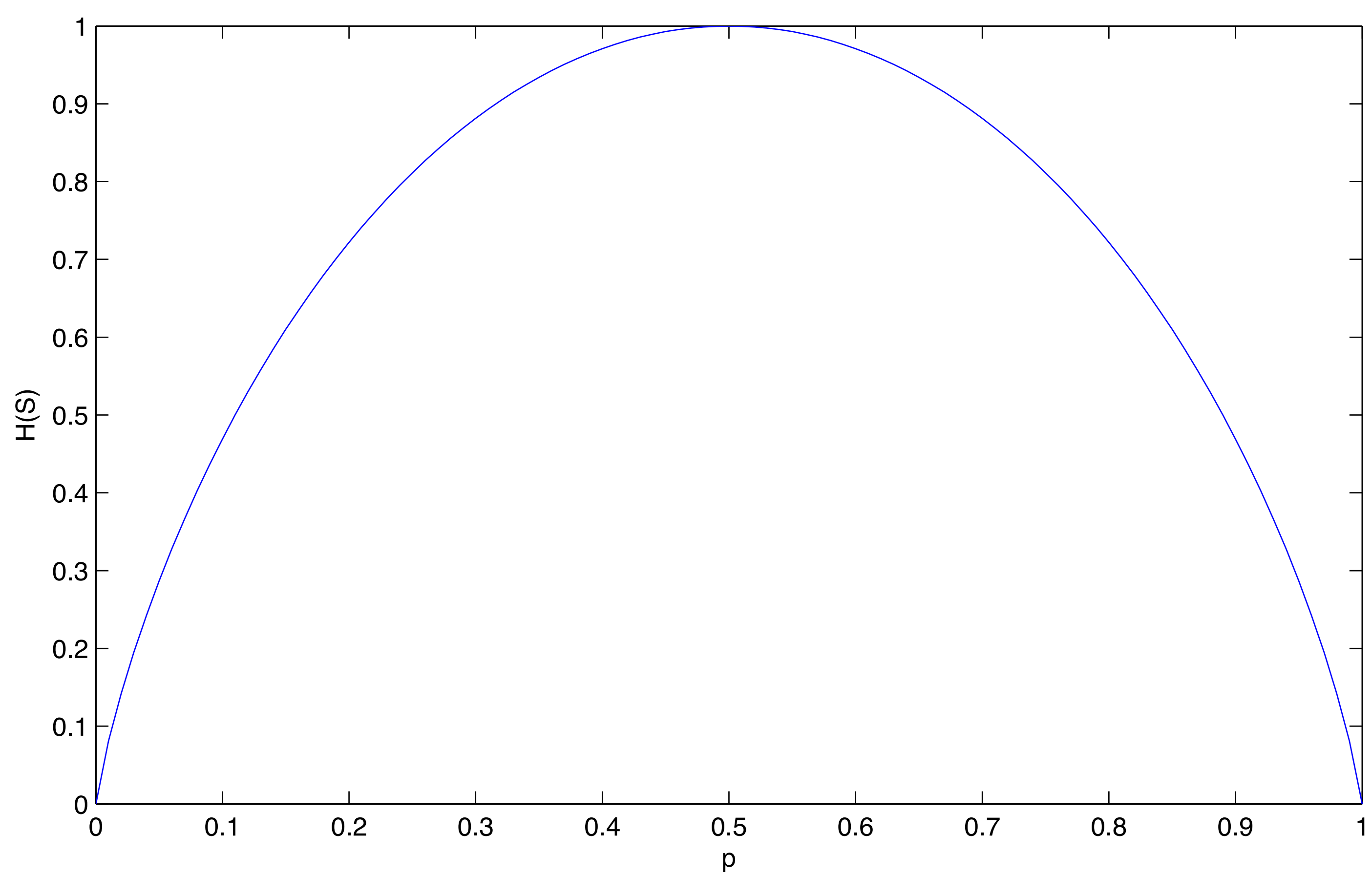
- Two values (e.g. Heads and Tails)
  - respective frequencies $p_{Heads} = p$ and $p_{Tails} = 1 - p$

$$H(X) = -p \log_2 p - (1-p) \log_2 (1-p)$$

Example: coin toss

- Two values (e.g. Heads and Tails)
  - respective frequencies $p_{Heads} = p$ and $p_{Tails} = 1 - p$

$$H(X) = -p \log_2 p - (1-p) \log_2 (1-p)$$



What does it mean?

Queen Mary
University of London

What does it mean?

- How can one coin-toss hold less than 1 bit of information when there are still 2 possible outcomes?
  - The result is *probabilistic*: for any bit rate $b > H(X)$ and any arbitrarily low error rate $e > 0$, there is an encoder with bit rate b and error rate e.

What does it mean?

- How can one coin-toss hold less than 1 bit of information when there are still 2 possible outcomes?
    - The result is *probabilistic*: for any bit rate $b > H(X)$ and any arbitrarily low error rate $e > 0$, there is an encoder with bit rate b and error rate e.
- What does an non-integer quantity of information mean?
    - It is a rate for a stream of independent samples
    - The stream is encoded by grouping samples together into words
    - $H(X) = 0.4$ means that words of 10 samples can be encoded using 4 bits only.

What does it mean?

- How can one coin-toss hold less than 1 bit of information when there are still 2 possible outcomes?
  - The result is *probabilistic*: for any bit rate $b > H(X)$ and any arbitrarily low error rate $e > 0$, there is an encoder with bit rate b and error rate e.
- What does an non-integer quantity of information mean?
  - It is a rate for a stream of independent samples
  - The stream is encoded by grouping samples together into words
  - $H(X) = 0.4$ means that words of 10 samples can be encoded using 4 bits only.
- How does it work?
  - If $H(X)$ is less than maximal, then there are rare events
  - Conjunctions of rare events become rarer and rarer as word length increases
  - At some point they become negligible and do not need to be encoded anymore

Mutual information

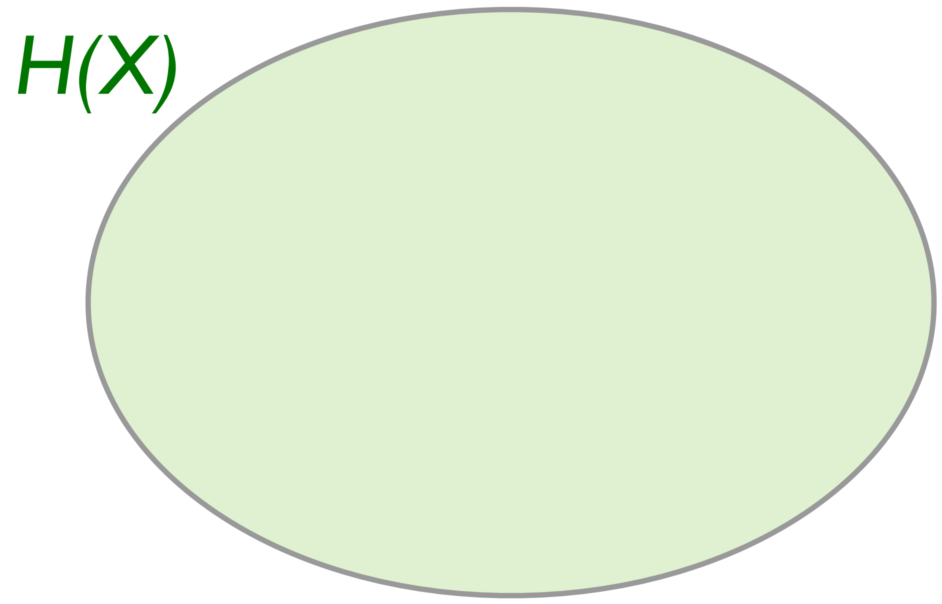
Mutual information

Queen Mary
University of London

- Objective: measure how much of the information in two random variables is shared
  - 0 if the variables are independent
  - maximal if one variable is a function of the other.
  - Mutual information: $I(X;Y) = H(X) + H(Y) - H(X,Y)$

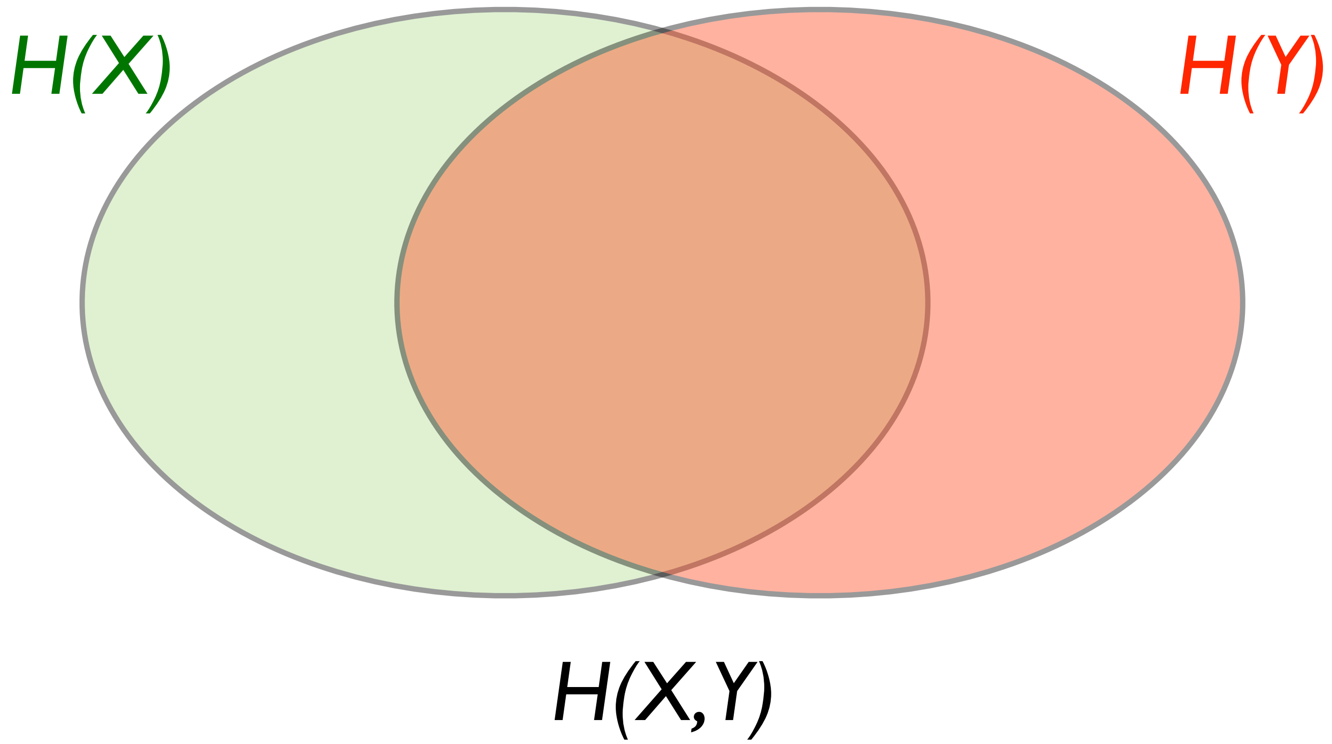
Mutual information

- Objective: measure how much of the information in two random variables is shared
  - 0 if the variables are independent
  - maximal if one variable is a function of the other.
  - Mutual information: $I(X;Y) = H(X) + H(Y) - H(X,Y)$



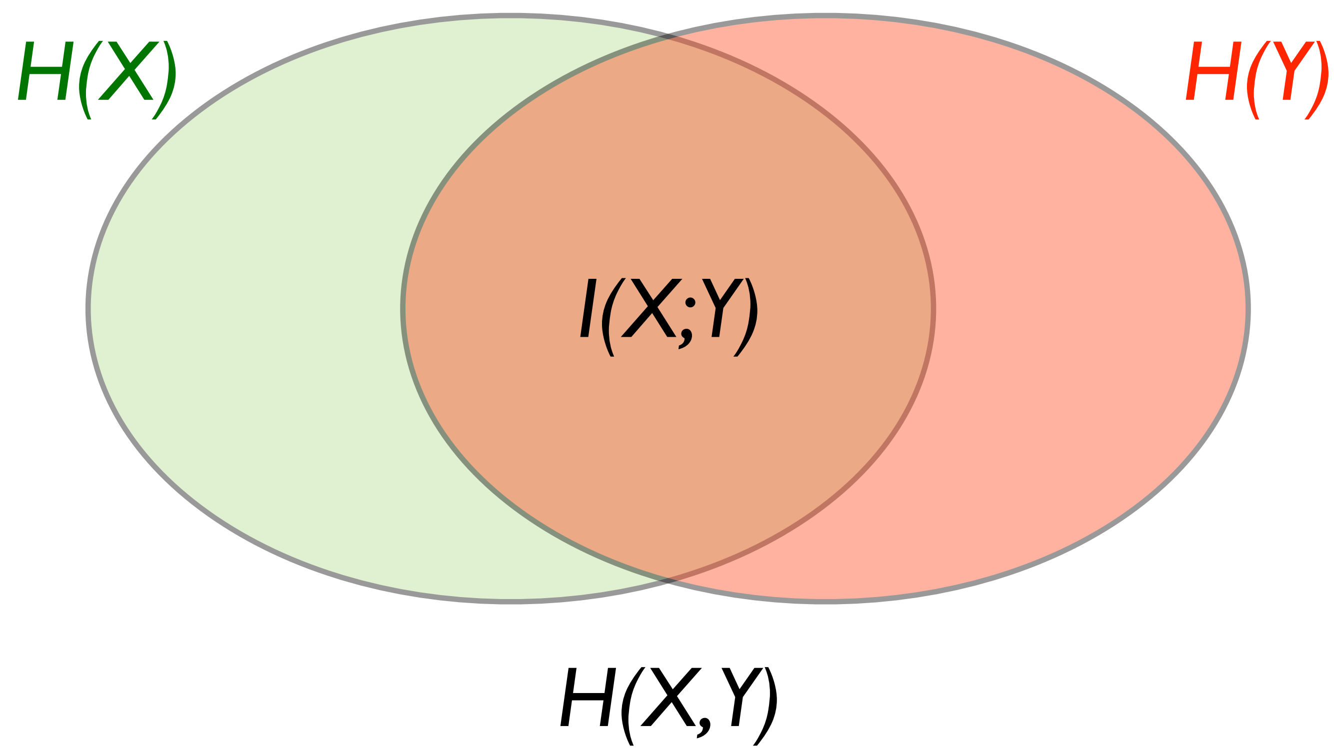
Mutual information

- Objective: measure how much of the information in two random variables is shared
  - 0 if the variables are independent
  - maximal if one variable is a function of the other.
  - Mutual information: $I(X;Y) = H(X) + H(Y) - H(X,Y)$



Mutual information

- Objective: measure how much of the information in two random variables is shared
  - 0 if the variables are independent
  - maximal if one variable is a function of the other.
  - Mutual information: $I(X;Y) = H(X) + H(Y) - H(X,Y)$



Computing the entropy numerically

Computing the entropy numerically

- Note that log(0) is undefined and will give an error if you compute it
    - $\lim_{x \to 0} \log(x) = -\infty$, for $x > 0$ but $\lim_{x \to 0} x \log(x) = 0$
    - test if the probability is 0, do not compute explicitly in that case

Computing the entropy numerically

Queen Mary University of London

- Note that log(0) is undefined and will give an error if you compute it
  - $\lim_{x\to 0} \log(x) = -\infty$, for $x > 0$ but $\lim_{x\to 0} x \log(x) = 0$
  - test if the probability is 0, do not compute explicitly in that case
- How to compute $\log_2$:
  - many languages do have a $\log_2$ function (MATLAB, C, C++,...)
  - ... but some do not (Java,...)
  - for any bases *a* and *b*, $\log_b x = \dfrac{\log_a x}{\log_a b}$
  - in our case *b*=2
  - *a* is usually e (2.71828…) or 10 (does it matter?)

Back to attribute selection

Back to attribute selection

Queen Mary University of London

- Finding attributes with high entropy:
  - start with a compressing transform (e.g. time/frequency for sounds, wavelet or DCT for images, movement vectors for video,...)
  - for numeric attributes, variance can often be used as a good surrogate for entropy, and is easier to compute

Back to attribute selection

Queen Mary University of London

- Finding attributes with high entropy:
  - start with a compressing transform (e.g. time/frequency for sounds, wavelet or DCT for images, movement vectors for video,...)
  - for numeric attributes, variance can often be used as a good surrogate for entropy, and is easier to compute
- Finding features with high mutual information with the classes:
  - expert knowledge may be available

Back to attribute selection

- Finding attributes with high entropy:
  - start with a compressing transform (e.g. time/frequency for sounds, wavelet or DCT for images, movement vectors for video,...)
  - for numeric attributes, variance can often be used as a good surrogate for entropy, and is easier to compute
- Finding features with high mutual information with the classes:
  - expert knowledge may be available
- Common sense works!
  - e.g., don't classify text documents based on frequency of "a", "the", "and", ...

Some common classifiers

Queen Mary
University of London

Some common classifiers

Queen Mary University of London

- Bayesian classifiers
  - Maximum Likelihood learning: for each class C, learn a probability distribution $p(.|C)$ that maximizes $\prod p(m|C)$ for all the m in the training data belonging to C
  - classification: for a point m, find the class C that maximises $p(C|m)$
    - using Bayes' rule: $p(C|m) = \frac{p(C)p(m|C)}{p(m)}$
    - scales linearly with the number of classes

Some common classifiers

- Bayesian classifiers
  - Maximum Likelihood learning: for each class *C*, learn a probability distribution *p(.|C)* that maximizes $\prod p(m|C)$ for all the *m* in the training data belonging to *C*
  - classification: for a point *m*, find the class *C* that maximises *p(C|m)*
    - using Bayes' rule: $p(C|m) = \frac{p(C)p(m|C)}{p(m)}$
    - scales linearly with the number of classes
- Support Vector Machines (SVM)
  - linear classifiers: boundaries are hyperplanes
  - Maximal Margin learning: learn the hyperplane that is the furthest away from making a mistake
    - maximize the distance to the closest (=worst) points of each class (the support vectors)
  - mainly for 2 classes (*binary* classification)

Decision trees

Queen Mary
University of London

Decision trees

Queen Mary
University of London

- A tool to represent decision making

Decision trees

- A tool to represent decision making
- Decompose a complex problem into a sequence of simple questions

Decision trees

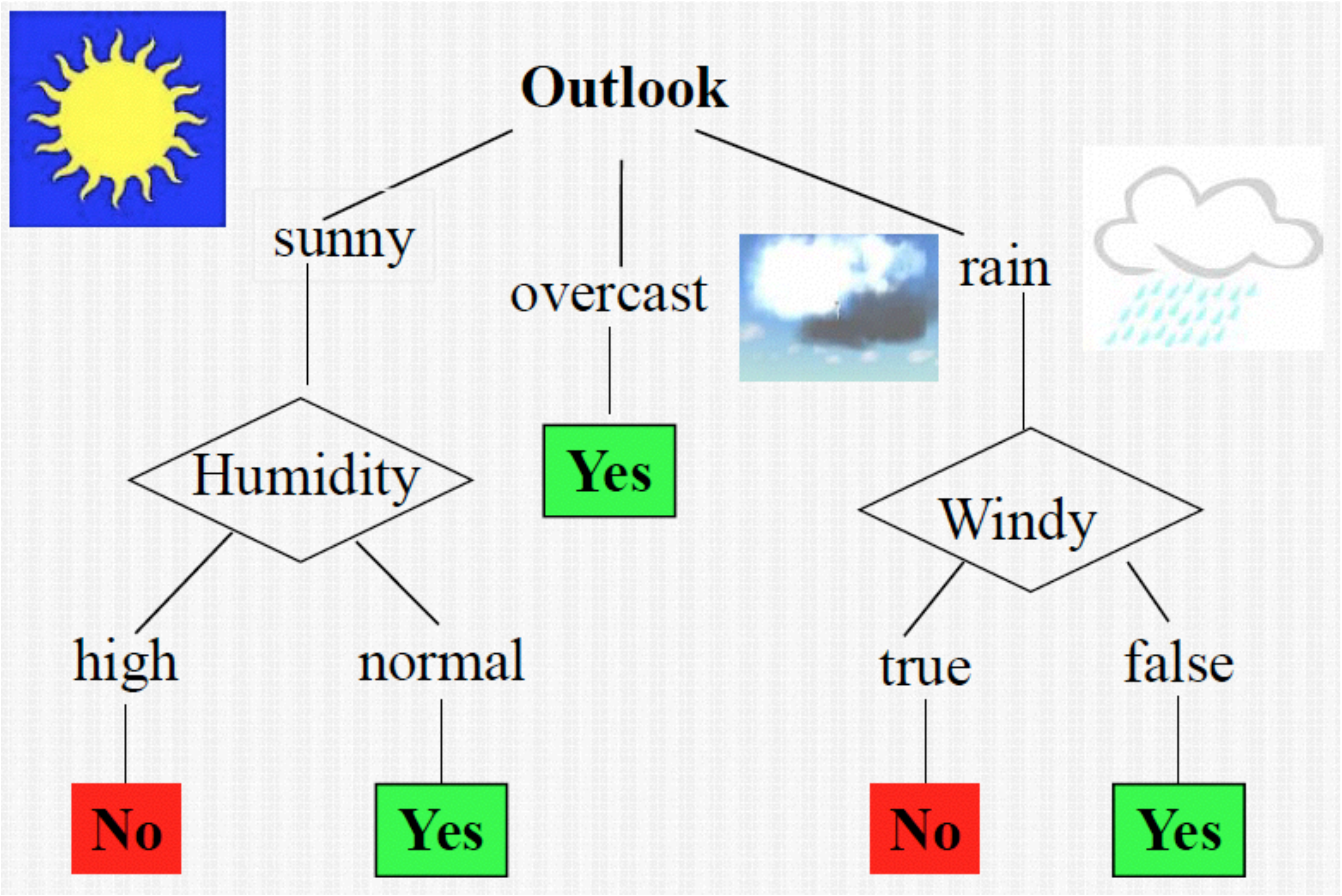
- A tool to represent decision making
- Decompose a complex problem into a sequence of simple questions
- Which question to ask next depends on the previous answers
  - can be represented as a tree

Decision trees

- A tool to represent decision making
- Decompose a complex problem into a sequence of simple questions
- Which question to ask next depends on the previous answers
  - can be represented as a tree
- Tree structure:
  - leaves: decisions
  - nodes: questions
  - branches: possible answers to the parent question

The morning problem

Should I attend the class or not?



Decision trees and classification

Queen Mary
University of London

Decision trees and classification

Queen Mary University of London

- Classification can be seen as a decision problem
  - the decision to take: which label apply to a data point?

Decision trees and classification

Queen Mary
University of London

- Classification can be seen as a decision problem
  - the decision to take: which label apply to a data point?
- Classifiers can be represented as a decision tree
  - a decision tree also partitions the attribute space
  - question: on which side of a given border is the point we're interested in?
  - several trees can represent the same classifier

Learning decision trees

Learning decision trees

Queen Mary
University of London

- Learning problem
  - which questions to ask?

Learning decision trees

- Learning problem
  - which questions to ask?
- If one asks enough questions, one will eventually classify the training set
  - 1 question = 1 new split of the space
  - with enough splits, one can eventually separate each training point from all others
  - but would that classifier generalise well to data it hasn't seen before?

Learning decision trees

- Learning problem
  - which questions to ask?
- If one asks enough questions, one will eventually classify the training set
  - 1 question = 1 new split of the space
  - with enough splits, one can eventually separate each training point from all others
  - but would that classifier generalise well to data it hasn't seen before?
- Goals of the learning:
  - the classifier should be general
  - it should be fast to apply
    - build a classifier that needs to ask few questions before taking a decision (a shallow, bushy tree)

ID3 algorithm (Quinlan 1986)

ID3 algorithm (Quinlan 1986)

- ID3 takes a training set and builds a decision tree such that each leaf is homogeneous
  - builds the tree from the root to the leaves
  - greedy algorithm: no backtracking

ID3 algorithm (Quinlan 1986)

- ID3 takes a training set and builds a decision tree such that each leaf is homogeneous
  - builds the tree from the root to the leaves
  - greedy algorithm: no backtracking

```
ID3(Training_Set) =
  if all the points in Training_Set have the same class C
    return Leaf(C);
  elseif no questions remain for Training_Set
    return MajorityClass(Training_Set);
  else
    Question = Find_Best_Question(Training_Set);
    [Set_1,...,Set_n] = Split(Training_set, Question);
    return Tree(Question, ID3(Set_1),...,ID3(Set_n));
```

ID3 algorithm (Quinlan 1986)

- ID3 takes a training set and builds a decision tree such that each leaf is homogeneous
  - builds the tree from the root to the leaves
  - greedy algorithm: no backtracking

```
ID3(Training_Set) =
  if all the points in Training_Set have the same class C
    return Leaf(C);
  elseif no questions remain for Training_Set
    return MajorityClass(Training_Set);
  else
    Question = Find_Best_Question(Training_Set);
    [Set_1,...,Set_n] = Split(Training_set, Question);
    return Tree(Question, ID3(Set_1),...,ID3(Set_n));
```

- How does `Find_Best_Question` work?
  - which questions are allowed?
  - how to select the best one?

Finding the next question to ask

$$H(S) = -\sum_{c \in C} p_c \log_2 p_c, \qquad \text{where } p_c = P(x \in c \mid x \in S)$$

Finding the next question to ask

- ID3 restricts itself to questions that test only one attribute
  - less questions to choose from
  - geometrically, all the borders are orthogonal to one axis
  - works well for symbolic attributes
    - symbolic attributes cannot be combined
    - symbolic attributes can be enumerated

$$H(S) = -\sum_{c \in C} p_c \log_2 p_c, \qquad \text{where } p_c = P(x \in c \mid x \in S)$$

Finding the next question to ask

- ID3 restricts itself to questions that test only one attribute
  - less questions to choose from
  - geometrically, all the borders are orthogonal to one axis
  - works well for symbolic attributes
    - symbolic attributes cannot be combined
    - symbolic attributes can be enumerated
- Each question is chosen so that the resulting subsets are as classified as possible
  - the entropy of the class variable c within a set S measures how classified it is

$$H(S) = -\sum_{c \in C} p_c \log_2 p_c, \qquad \text{where } p_c = P(x \in c \mid x \in S)$$

Finding the best question

Finding the best question

- The entropy of a partition is the sum of the entropies of the subsets weighted by their size

Finding the best question

- The entropy of a partition is the sum of the entropies of the subsets weighted by their size
- Entropy also provides a stopping criterion: a leaf is a node with entropy 0

Finding the best question

Queen Mary University of London

- The entropy of a partition is the sum of the entropies of the subsets weighted by their size
- Entropy also provides a stopping criterion: a leaf is a node with entropy 0
- The information gain is defined as the difference between the total entropies before and after partitioning the set S into subsets S_i
  - It is also the mutual information between the attribute and the class, conditioned to the previous splits

Finding the best question

Queen Mary University of London

- The entropy of a partition is the sum of the entropies of the subsets weighted by their size
- Entropy also provides a stopping criterion: a leaf is a node with entropy 0
- The information gain is defined as the difference between the total entropies before and after partitioning the set S into subsets S_i
  - It is also the mutual information between the attribute and the class, conditioned to the previous splits

$$G(S,Q) = H(S) - \sum_{S_i \in \mathrm{Split}(S,Q)} \frac{|S_i|}{|S|} H(S_i)$$

Finding the best question

Queen Mary
University of London

- The entropy of a partition is the sum of the entropies of the subsets weighted by their size
- Entropy also provides a stopping criterion: a leaf is a node with entropy 0
- The information gain is defined as the difference between the total entropies before and after partitioning the set *S* into subsets S_i
  - It is also the mutual information between the attribute and the class, conditioned to the previous splits

$$G(S,Q) = H(S) - \sum_{S_i \in \mathrm{Split}(S,Q)} \frac{|S_i|}{|S|} H(S_i)$$

- Compute *G* for all possible questions *Q* and select the *Q* with largest *G(S,Q)*

Which questions to ask?

Queen Mary
University of London

Which questions to ask?

- Symbolic attributes (gender, hair colour,...):
  - only a finite number of possible values for all data
  - one branch for each different value

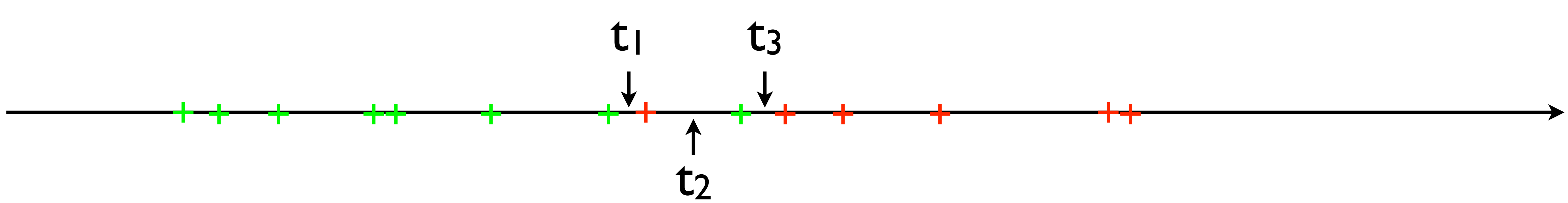
Which questions to ask?

Queen Mary University of London

- Symbolic attributes (gender, hair colour,...):
  - only a finite number of possible values for all data
  - one branch for each different value
- Numerical attributes (temperature, pixel value,...):
  - infinite range even if only a finite number of values are present in the training set
  - choose a threshold t and ask the question: value < t?

Which questions to ask?

- Symbolic attributes (gender, hair colour,...):
  - only a finite number of possible values for all data
  - one branch for each different value
- Numerical attributes (temperature, pixel value,...):
  - infinite range even if only a finite number of values are present in the training set
  - choose a threshold t and ask the question: value < t?
- Threshold choice:
  - the information gain criterion still holds
  - the best threshold must be between two training examples of different classes
  - try all possible thresholds



Example: attend the lecture or not?

Outlook	Temperature	Windy	Attend?
sunny	29	No	Yes
sunny	27	Yes	No
overcast	28	No	Yes
rain	21	Yes	No
rain	20	No	Yes
rain	18	Yes	No
overcast	17	Yes	Yes
sunny	22	No	No

$$
\begin{aligned}
H(S) &= -\frac{4}{8}\log_2\frac{4}{8} - \frac{4}{8}\log_2\frac{4}{8} \\
&= 1
\end{aligned}
$$

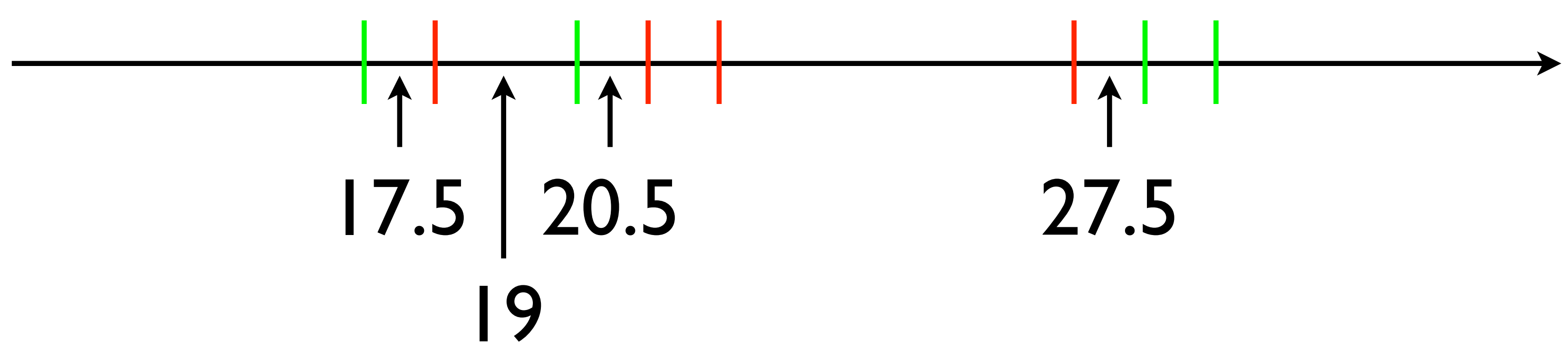
Example: split along Outlook

Outlook	Temperature	Windy	Attend?
sunny	29	No	Yes
sunny	27	Yes	No
overcast	28	No	Yes
rain	21	Yes	No
rain	20	No	Yes
rain	18	Yes	No
overcast	17	Yes	Yes
sunny	22	No	No

$$
\begin{aligned}
H(sunny) &= -\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3} \approx 0.9183 \\
H(overcast) &= -\frac{2}{2}\log_2\frac{2}{2} - \frac{0}{2}\log_2\frac{0}{2} \approx 0 \\
H(rain) &= -\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3} \approx 0.9183 \\
G(Outlook) &= H(S) - \frac{3}{8}H(sunny) - \frac{2}{8}H(overcast) - \frac{3}{8}H(rain) \\
&\approx 0.3113
\end{aligned}
$$

Example: split along Temperature

Outlook	Temperature	Windy	Attend?
sunny	29	No	Yes
sunny	27	Yes	No
overcast	28	No	Yes
rain	21	Yes	No
rain	20	No	Yes
rain	18	Yes	No
overcast	17	Yes	Yes
sunny	22	No	No



Example: split at t=17.5°C

Outlook	Temperature	Windy	Attend?
sunny	29	No	Yes
sunny	27	Yes	No
overcast	28	No	Yes
rain	21	Yes	No
rain	20	No	Yes
rain	18	Yes	No
overcast	17	Yes	Yes
sunny	22	No	No

$$H(t < 17.5) = -\frac{1}{1}\log_2\frac{1}{1} - \frac{0}{1}\log_2\frac{0}{1} \approx 0$$

$$H(t > 17.5) = -\frac{3}{7}\log_2\frac{3}{7} - \frac{4}{7}\log_2\frac{4}{7} \approx 0.9852$$

$$\begin{aligned} G(t = 17.5) &= H(S) - \frac{1}{8}H(t < 17.5) - \frac{7}{8}H(t > 17.5) \\ &\approx 0.1379 \end{aligned}$$

Example: split at t=19°C

Outlook	Temperature	Windy	Attend?
sunny	29	No	Yes
sunny	27	Yes	No
overcast	28	No	Yes
rain	21	Yes	No
rain	20	No	Yes
rain	18	Yes	No
overcast	17	Yes	Yes
sunny	22	No	No

$$
\begin{aligned}
H(t<19) &= -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} \approx 1 \\
H(t>19) &= -\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6} \approx 1 \\
G(t=19) &= H(S) - \frac{2}{8}H(t<19) - \frac{6}{8}H(t>19) \\
&= 0
\end{aligned}
$$

Example: split at t=20.5°C

Outlook	Temperature	Windy	Attend?
sunny	29	No	Yes
sunny	27	Yes	No
overcast	28	No	Yes
rain	21	Yes	No
rain	20	No	Yes
rain	18	Yes	No
overcast	17	Yes	Yes
sunny	22	No	No

$$H(t < 20.5) = -\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3} \approx 0.9183$$

$$H(t > 20.5) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} \approx 0.9710$$

$$\begin{aligned} G(t = 20.5) &= H(S) - \frac{3}{8}H(t < 20.5) - \frac{5}{8}H(t > 20.5) \\ &\approx 0.0488 \end{aligned}$$

Example: split at t=27.5°C

Outlook	Temperature	Windy	Attend?
sunny	29	No	Yes
sunny	27	Yes	No
overcast	28	No	Yes
rain	21	Yes	No
rain	20	No	Yes
rain	18	Yes	No
overcast	17	Yes	Yes
sunny	22	No	No

$$H(t < 27.5) = -\frac{2}{6}\log_2\frac{2}{6} - \frac{4}{6}\log_2\frac{4}{6} \approx 0.9183$$

$$H(t > 27.5) = -\frac{2}{2}\log_2\frac{2}{2} - \frac{0}{2}\log_2\frac{0}{2} = 0$$

$$G(t = 27.5) = H(S) - \frac{6}{8}H(t < 27.5) - \frac{2}{8}H(t > 27.5)$$

$$\approx 0.3113$$

Example: split along windy

Queen Mary University of London

Outlook	Temperature	Windy	Attend?
sunny	29	No	Yes
sunny	27	Yes	No
overcast	28	No	Yes
rain	21	Yes	No
rain	20	No	Yes
rain	18	Yes	No
overcast	17	Yes	Yes
sunny	22	No	No

$$H(windy) = -\frac{1}{4}\log_2\frac{1}{4} - \frac{3}{4}\log_2\frac{3}{4} \approx 0.8113$$

$$H(\neg windy) = -\frac{3}{4}\log_2\frac{3}{4} - \frac{1}{4}\log_2\frac{1}{4} \approx 0.8113$$

$$\begin{aligned} G(Windy?) &= H(S) - \frac{4}{8}H(windy) - \frac{4}{8}H(\neg windy) \\ &\approx 0.1887 \end{aligned}$$

Example: first iteration

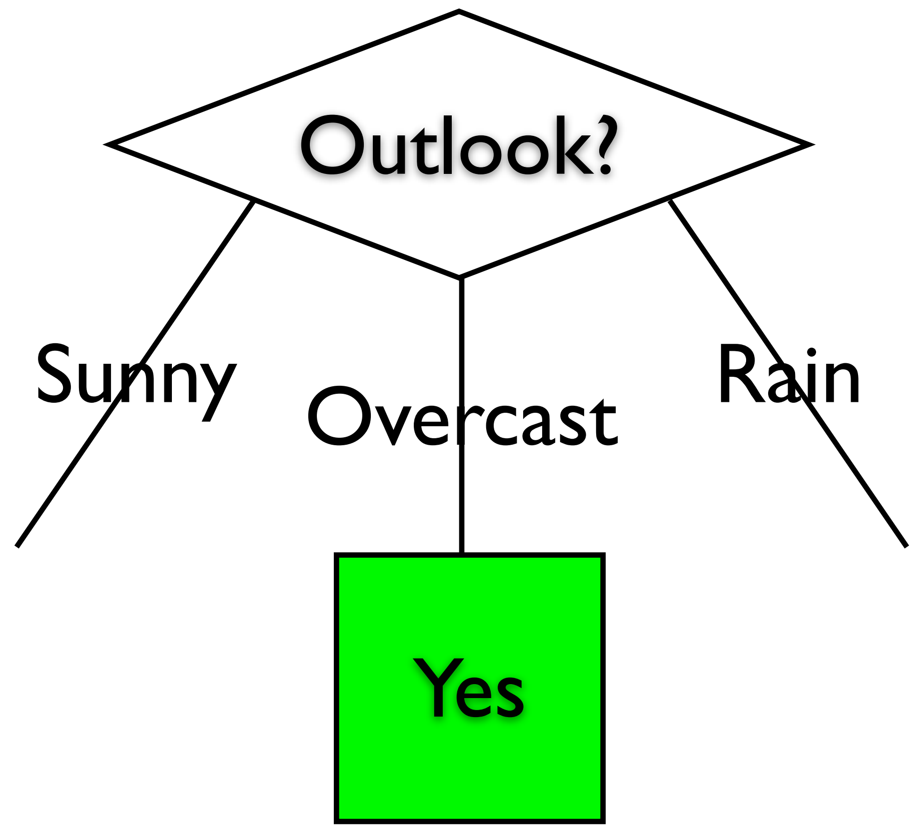
Example: first iteration

- The best splits were either Outlook or t=27.5°C

Example: first iteration

Queen Mary University of London

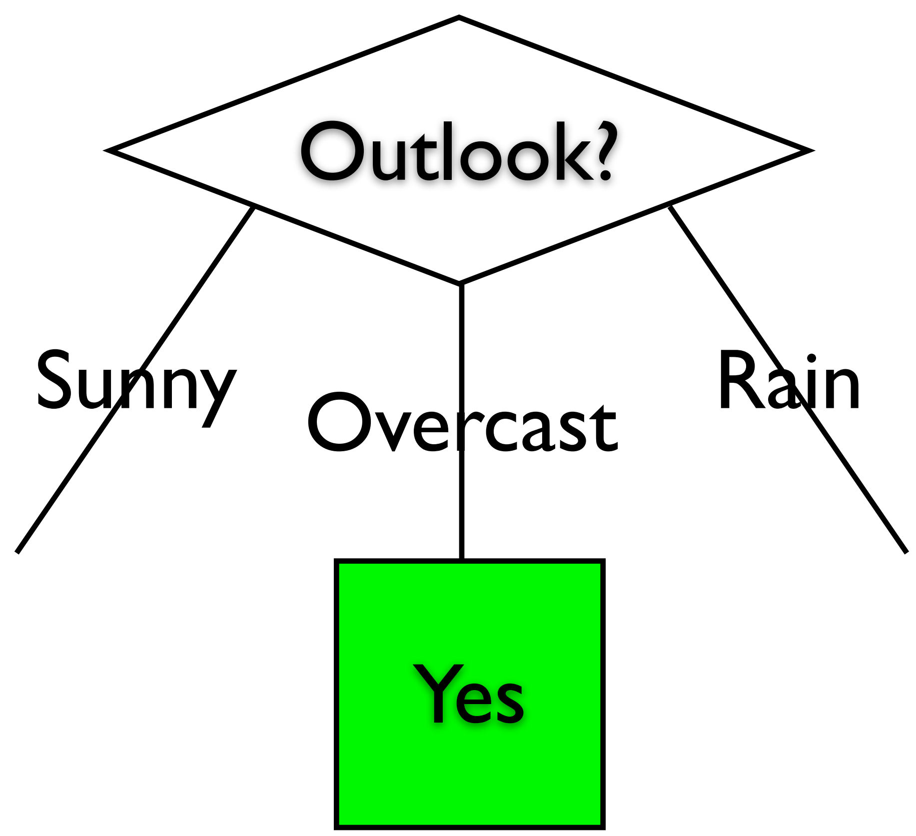
- The best splits were either Outlook or t=27.5°C
- Let us assume we split on Outlook:



Example: first iteration

Queen Mary University of London

- The best splits were either Outlook or t=27.5°C
- Let us assume we split on Outlook:



- We must then apply the algorithm to the Sunny and Rain branches

Example: the Sunny branch

Outlook	Temperature	Windy	Attend?
sunny	29	No	Yes
sunny	27	Yes	No
sunny	22	No	No

A split at t=28°C classifies the branch

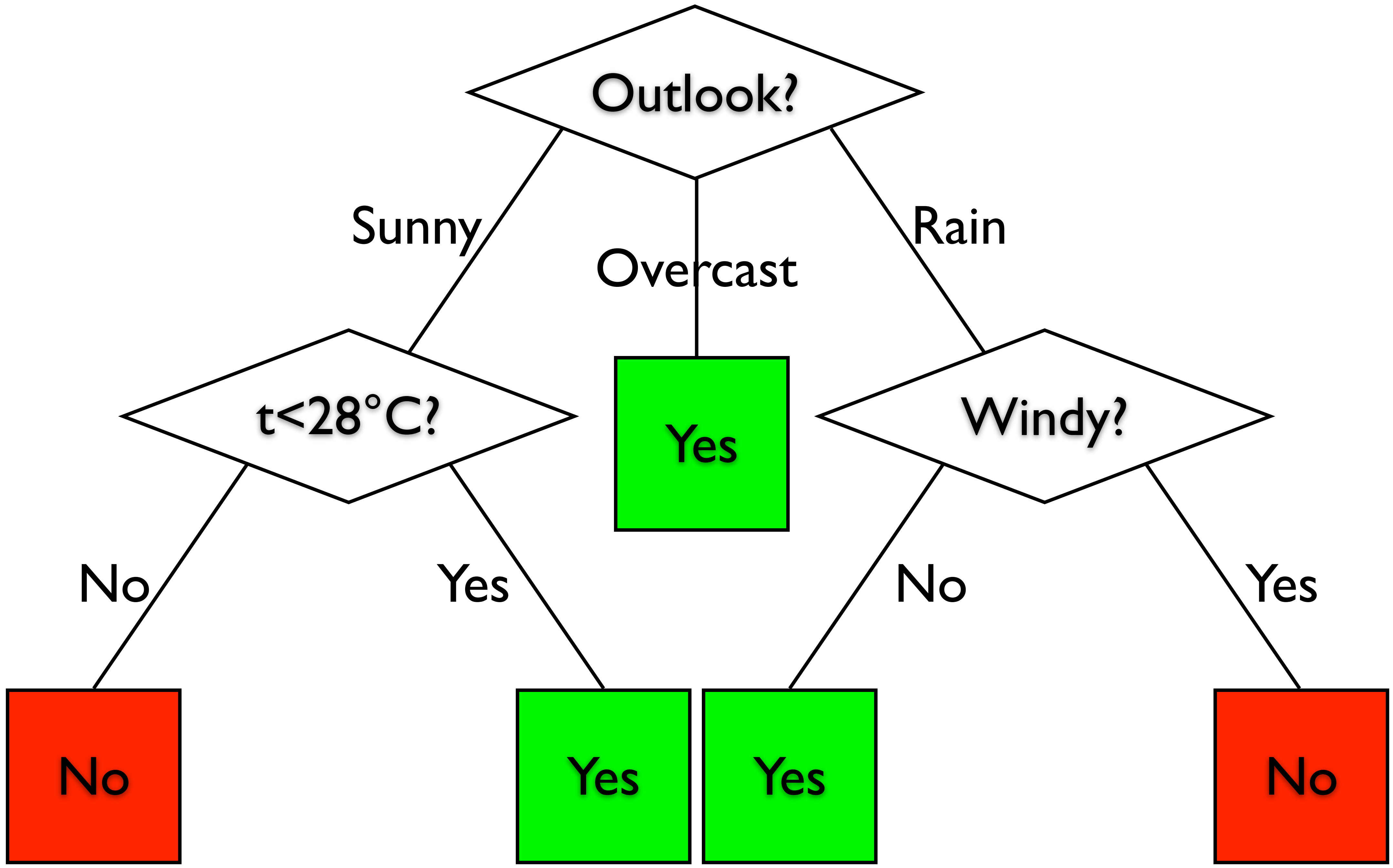
Example: the Rainy branch

Queen Mary
University of London

Outlook	Temperature	Windy	Attend?
rain	21	Yes	No
rain	20	No	Yes
rain	18	Yes	No

A split along Windy classifies the branch

Example: the final tree



Strong and weak points of ID3

Strong and weak points of ID3

- Each question only checks one feature:
  - works well with attributes of different types (e.g. size and gender)
  - designs rectangular regions only, even if it does not fit the data

Strong and weak points of ID3

Queen Mary University of London

- Each question only checks one feature:
  - works well with attributes of different types (e.g. size and gender)
  - designs rectangular regions only, even if it does not fit the data
- Generates the shortest possible tree:
  - fast classification after learning
  - helps to generalise the model
  - helps to understand the generated rules

Strong and weak points of ID3

- Each question only checks one feature:
  - works well with attributes of different types (e.g. size and gender)
  - designs rectangular regions only, even if it does not fit the data
- Generates the shortest possible tree:
  - fast classification after learning
  - helps to generalise the model
  - helps to understand the generated rules
- Training is expensive:
  - try all possible questions at each node
  - for each question, compute the answer with each data point of the subset
  - infinite range attributes generate lots of different questions

A word on overfitting

A word on overfitting

- Noise blurs the boundaries between classes

A word on overfitting

- Noise blurs the boundaries between classes
- Sampling can alter the perception of the problem
  - selection bias
  - rare events

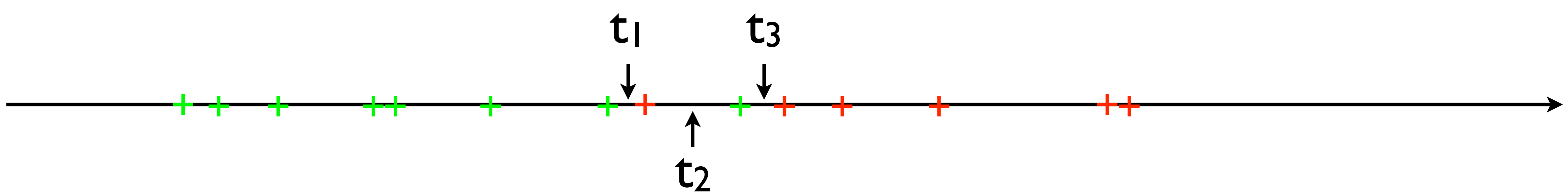
A word on overfitting

Queen Mary University of London

- Noise blurs the boundaries between classes
- Sampling can alter the perception of the problem
  - selection bias
  - rare events
- General solution: more data! (not always possible)

A word on overfitting

- Noise blurs the boundaries between classes
- Sampling can alter the perception of the problem
  - selection bias
  - rare events
- General solution: more data! (not always possible)
- **Overfitting**
  - learning a tree that models properties specific to the training set
    - the tree works on the training set but not on the problem data
    - perfect classification of the training set is usually too much



Reduced-error pruning

Reduced-error pruning

- Split the training set in two:
  - one for learning
  - one for pruning

Reduced-error pruning

- Split the training set in two:
  - one for learning
  - one for pruning
- Goal: learn a tree that classifies the pruning set well although it was not trained on it

Reduced-error pruning

Queen Mary University of London

- Split the training set in two:
  - one for learning
  - one for pruning
- Goal: learn a tree that classifies the pruning set well although it was not trained on it
- First build a tree with ID3 on the training set (learning subset)

Reduced-error pruning

- Split the training set in two:
  - one for learning
  - one for pruning
- Goal: learn a tree that classifies the pruning set well although it was not trained on it
- First build a tree with ID3 on the training set (learning subset)
- Then greedily prune nodes to improve the classification of the pruning set

Reduced-error pruning

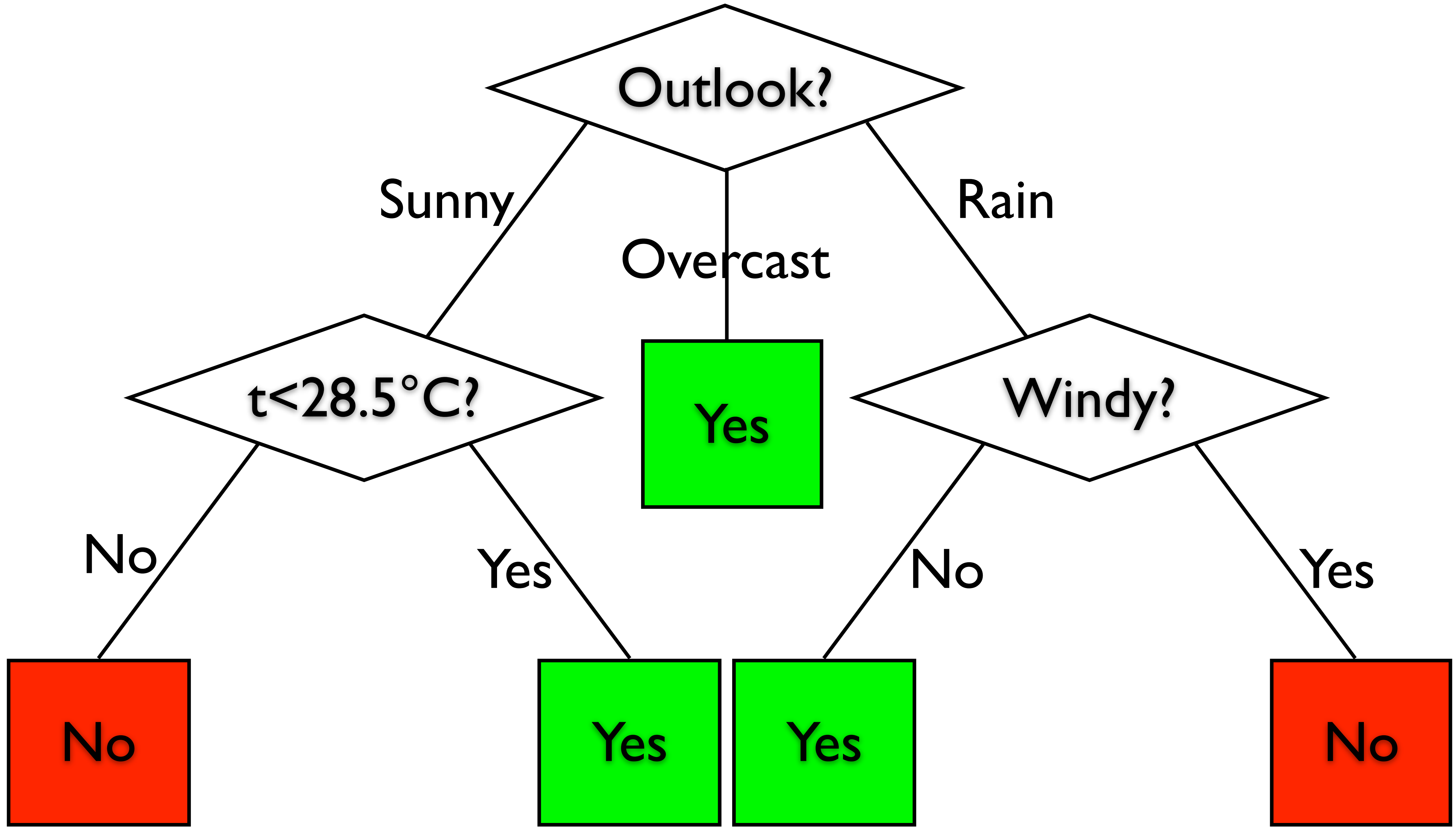
Queen Mary
University of London

- Split the training set in two:
  - one for learning
  - one for pruning
- Goal: learn a tree that classifies the pruning set well although it was not trained on it
- First build a tree with ID3 on the training set (learning subset)
- Then greedily prune nodes to improve the classification of the pruning set
- What if we don't have enough data to make a pruning set?

Rule pruning

Rule pruning

- The tree can be written as a set of logical rules
  - each path is a single conjunction (“and”) of tests
  - there is a disjunction (“or”) between different paths leading to the same class



Rule post-pruning

Rule post-pruning

Queen Mary University of London

- First convert the tree into rules

Rule post-pruning

- First convert the tree into rules
- Then greedily prune the rules by removing conditions

Rule post-pruning

Queen Mary
University of London

- First convert the tree into rules
- Then greedily prune the rules by removing conditions
- Only prune if the precision of the rule does not decrease
  - Precision = fraction of points of the correct class among the points picked by the rule

Rule post-pruning

Queen Mary
University of London

- First convert the tree into rules
- Then greedily prune the rules by removing conditions
- Only prune if the precision of the rule does not decrease
  - Precision = fraction of points of the correct class among the points picked by the rule
- The pruning is done on the training set directly
  - no need to split the set in two
  - more used in practice

Summary

Summary

Queen Mary
University of London

- Information as a measurable quantity
  - information as structure in data

Summary

Queen Mary
University of London

- Information as a measurable quantity
  - information as structure in data
- ID3: a method of using information to decide on structure

Summary

- Information as a measurable quantity
  - information as structure in data
- ID3: a method of using information to decide on structure
- Avoid overfitting
  - pruning to re-generalise