Machine Learning
CS7052
Lecture 7, Decision Trees

Dr. Elaheh Homayounvala

Week 7

Outline of today's lecture

- Review week 5
- Supervised learning, Decision Trees

Review last weeks

k-Nearest Neighbours (kNN)

Linear Models (and Gradient Descent)

What we covered so far

- Supervised learning
  1. kNN
  2. Linear Models

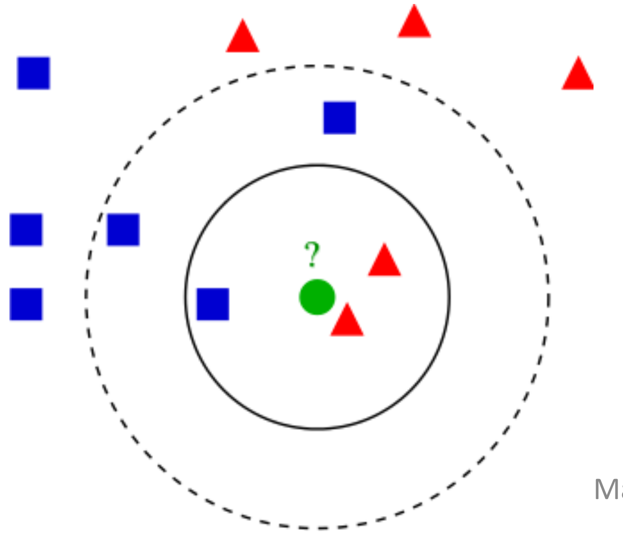
  kNN and linear models can be applied in classification as well as regression

- Unsupervised learning
  - Clustering

k-Nearest Neighbours Algorithm

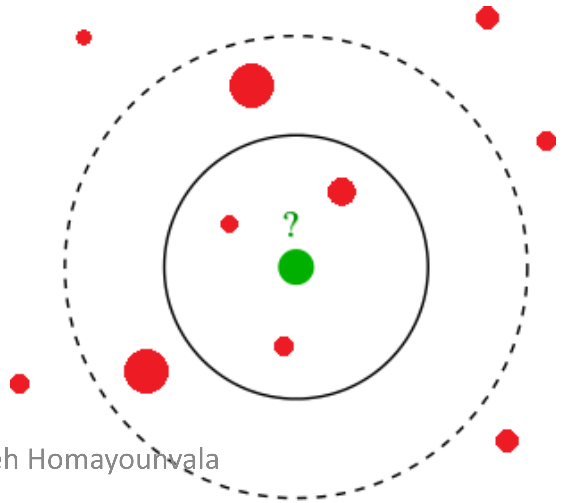
Classification:

1. Find k closest objects to the predicted object x in the training set.
2. Associate x the most frequent class among its k neighbours.



Regression:

1. Find k closest objects to the predicted object x in the training set.
2. Associate x average output of its k neighbours.



Linear Models

- Linear models:

$$\hat{y} = w[0] * x[0] + w[1] * x[1] + ... + w[p] * x[p] + b$$

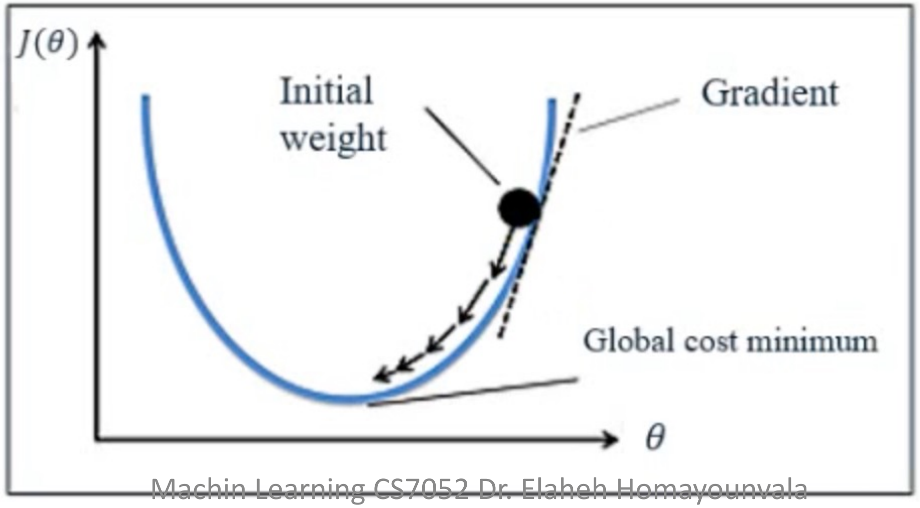
- Simple form of cost or loss function:

$$\sum_{i=1}^{M}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{M}\left(y_i - \sum_{j=0}^{p} w_j \times x_{ij}\right)^2$$

The data-set has M instances and p features

Linear Models, Gradient Descent

- We look for optimising *w* and *b* such that it minimises the cost function
- Gradient Decent is one of the ways for cost minimisation
- What are the others?



LONDON METROPOLITAN UNIVERSITY

Practical skills in ML

Jupyter Notebook

numpy, pandas, matplotlib, scikitlearn libraries

Practical ML

- Choose a dataset and a model and a tool
- Construct/Learn a model based on training data
- Use the model to make predictions for test data
- Evaluate the model (accuracy)
- Tune the model or choose another if necessary

Decision Trees

What is a decision tree?

How to build a decision tree? Attribute selection measures

Information Gain, Gain Ratio and Gini index

Parameters, Strengths and Weaknesses

Decision Trees

- Widely used models for classification and regression tasks
- They learn a hierarchy of if/else questions, leading to a decision

Decision trees

Distinguish between the following four animals

- Bears
- Hawks
- Penguins
- dolphins.

- Muller and Guido’s book, page 73
- Animal pictures from pixabay.com

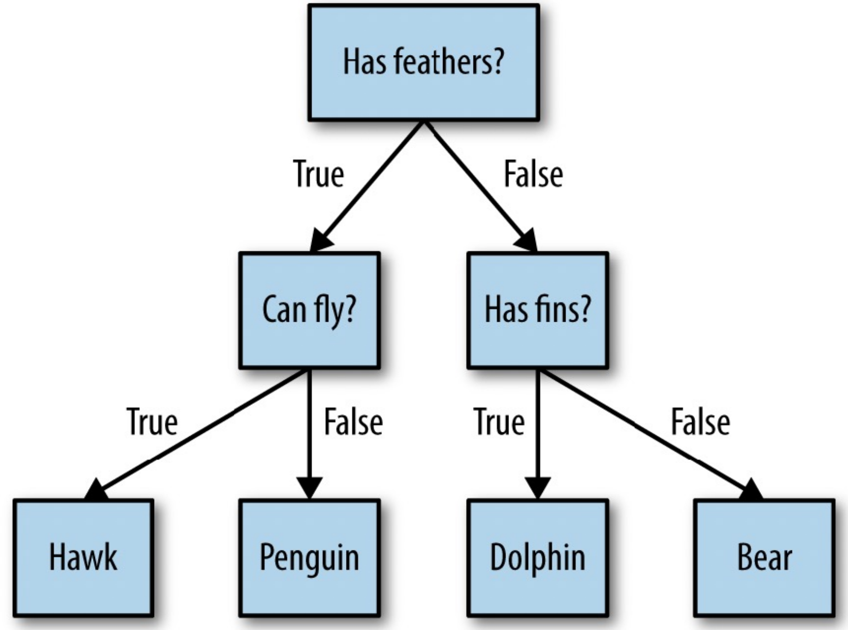


Figure 2-22. A decision tree to distinguish among several animals

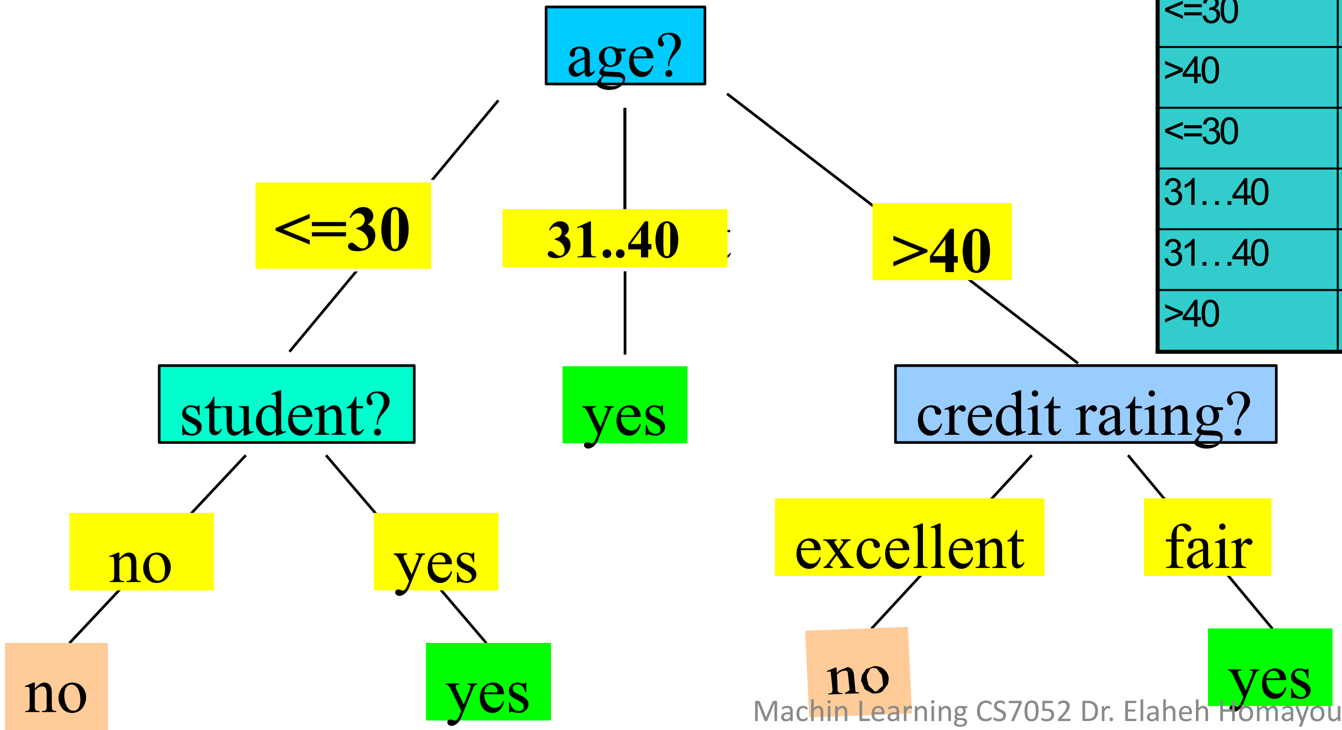



Nodes and Edges in Decision Trees

- Each node in the tree either represents
  - a question or
  - a terminal node (also called a leaf) that contains the answer
- The edges connect the answers to a question with the next question you would ask.
- It is an upside-down tree, root at top!

Decision Tree, An Example

- Training dataset: Buys_computer
- Decision Tree:



age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Process 1, model construction (induction)

Training Data

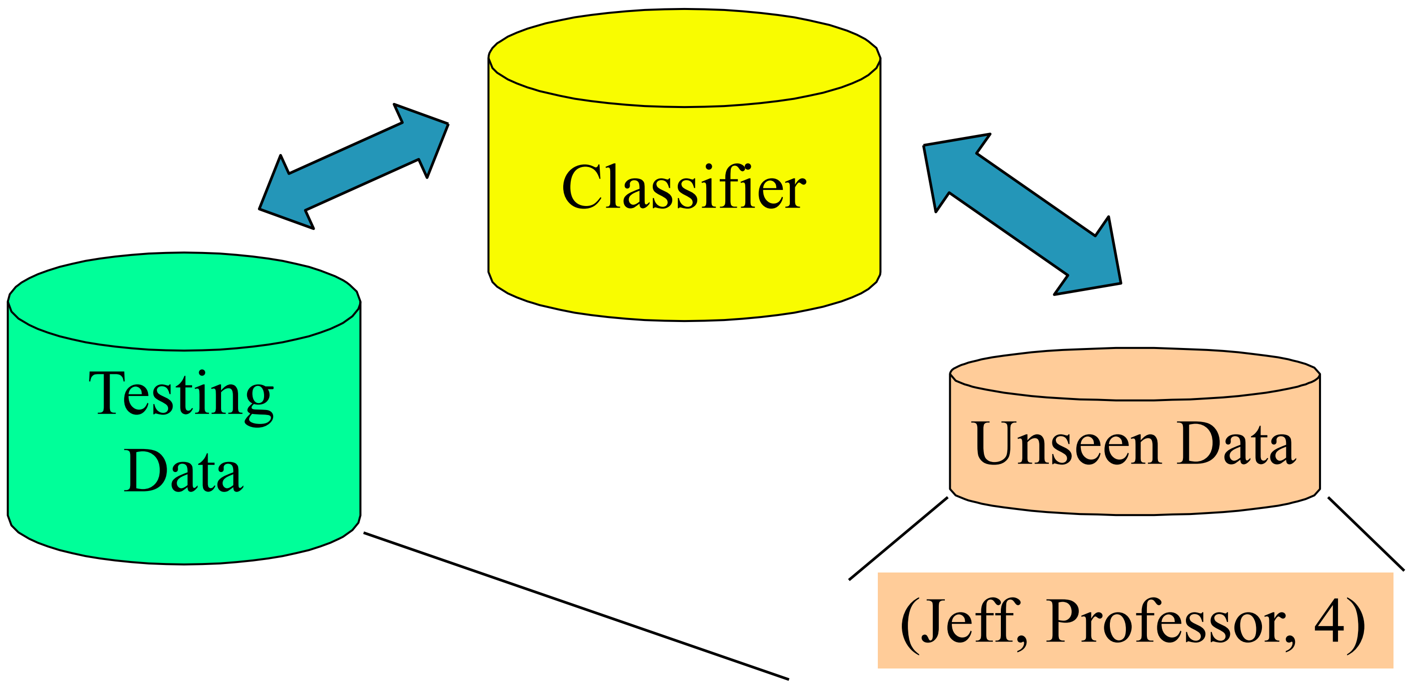
Classification Algorithms

Classifier (Model)

NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

IF rank = ‘professor’
OR years > 6
THEN tenured = ‘yes’

Process 2, using the model in prediction



NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

Tenured?

Yes

Building/Learning a Decision Tree

- Learning a decision tree means learning the sequence of if/else questions that gets us to the true answer most quickly.

- In ML if/else questions are called “tests”.

Learning a Decision Tree

- To build a tree, the algorithm searches over all possible tests and finds the one that is most informative about the target variable.
- Continue and choose the next test recursively.
- The recursive partitioning of the data is repeated until
  - Each region in the partition (each leaf) only contains a single target value.
- A leaf of the tree that contains data points that all share the same target value is called pure.

Building a Decision Tree, example

- Muller and Guido's book, page 74

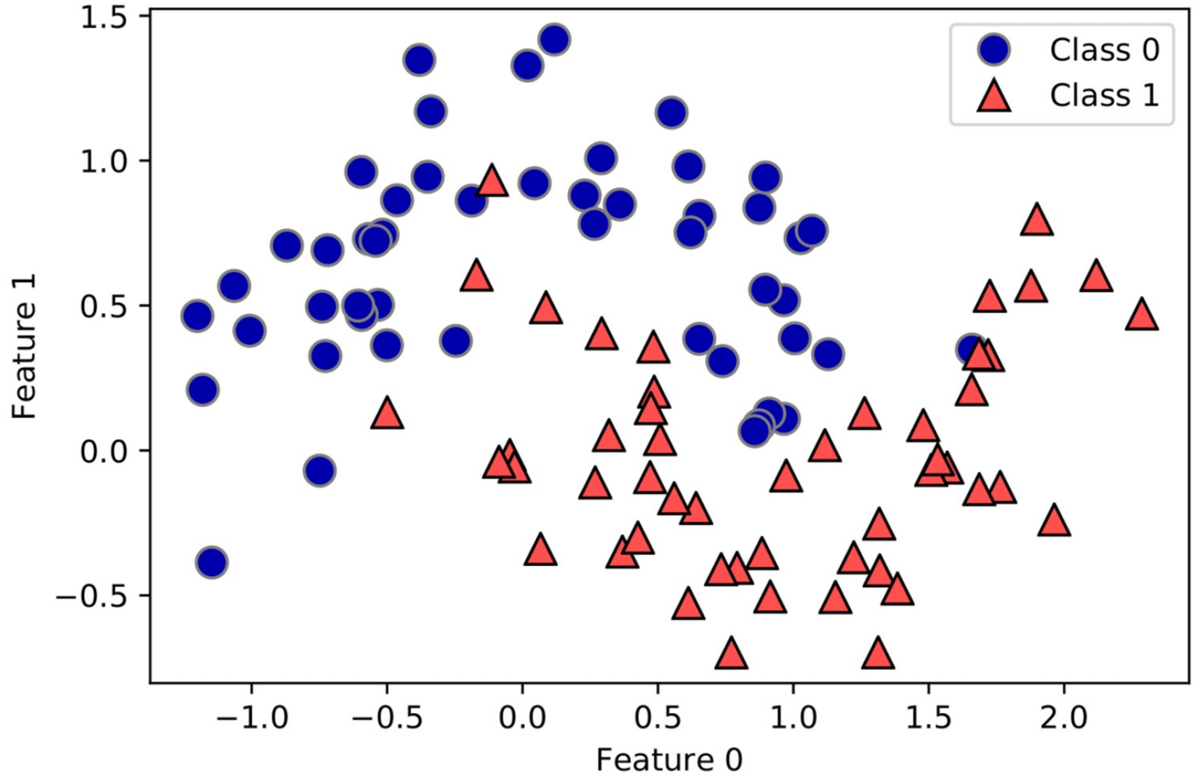
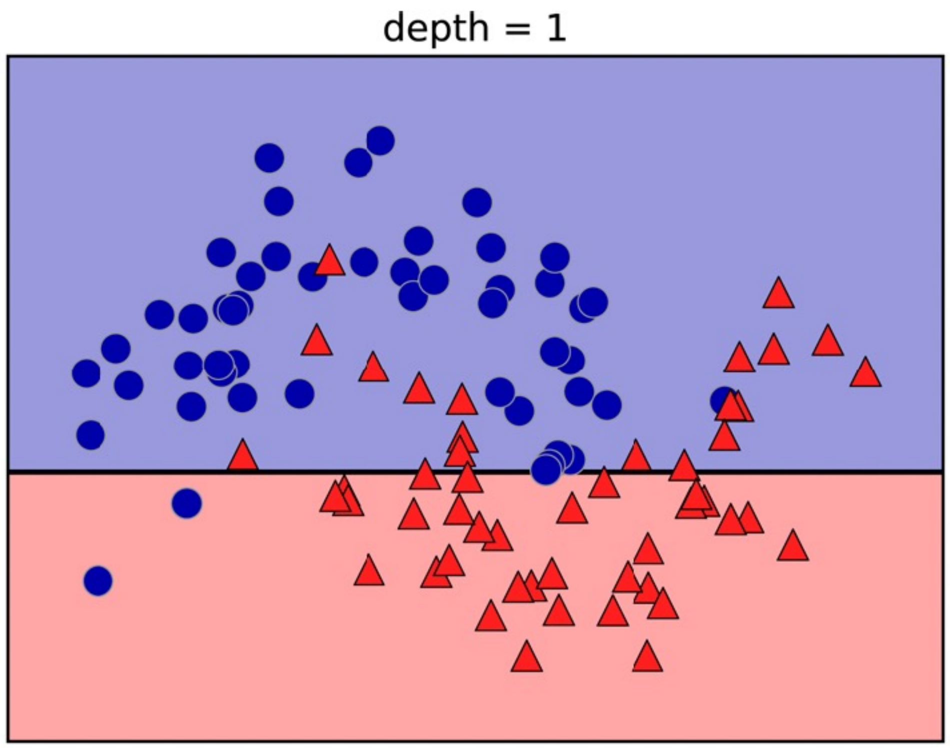


Figure 2-23. Two-moons dataset on which the decision tree will be built

Decision Boundary of Tree, depth 1

- Muller and Guido’s book, page 74



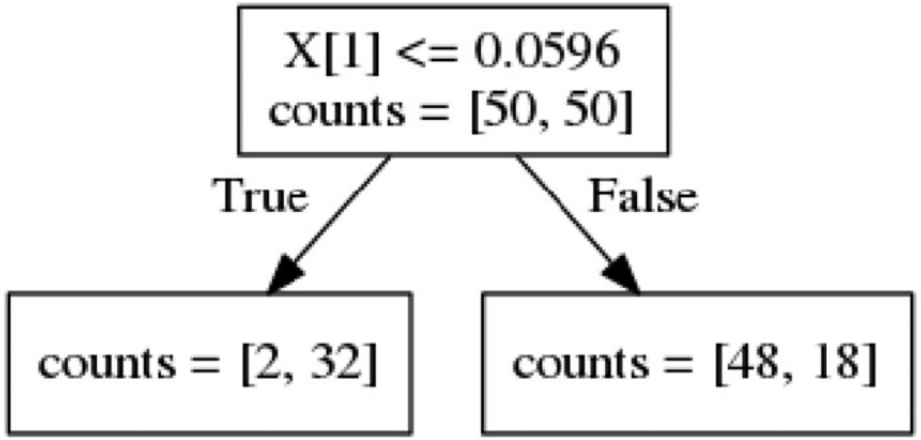
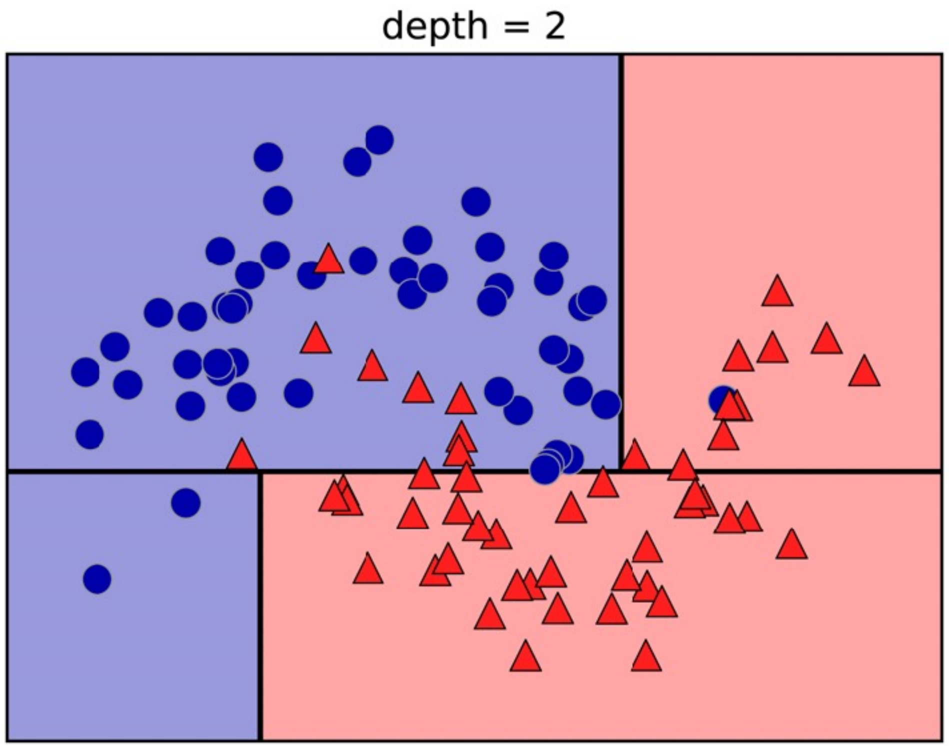


Figure 2-24. Decision boundary of tree with depth 1 (left) and corresponding tree (right)

Decision Boundary of Tree, depth 2



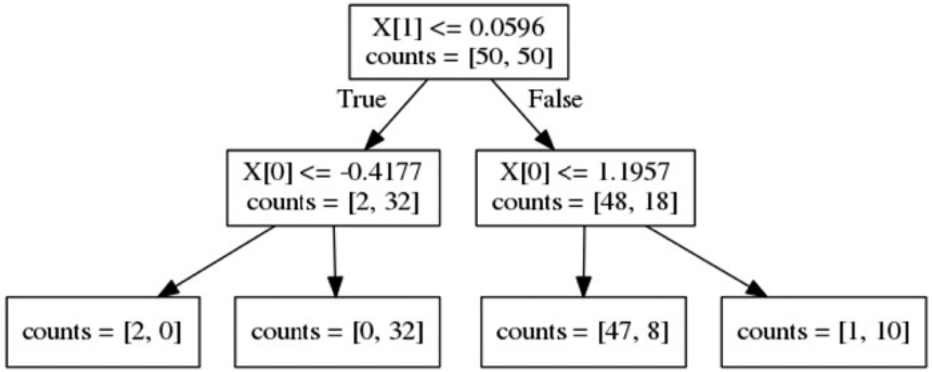


Figure 2-25. Decision boundary of tree with depth 2 (left) and corresponding decision tree (right)

Algorithm for Decision Tree Induction, more detailed

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down **recursive divide-and-conquer** manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretised in advance)
  - Examples are partitioned recursively based on <u>selected</u> attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., **information gain**)

Algorithm for Decision Tree Induction, cont.

Conditions for stopping partitioning

- All samples for a given node belong to the same class
- There are no remaining attributes for further partitioning **majority voting** is employed for classifying the leaf
- There are no samples left

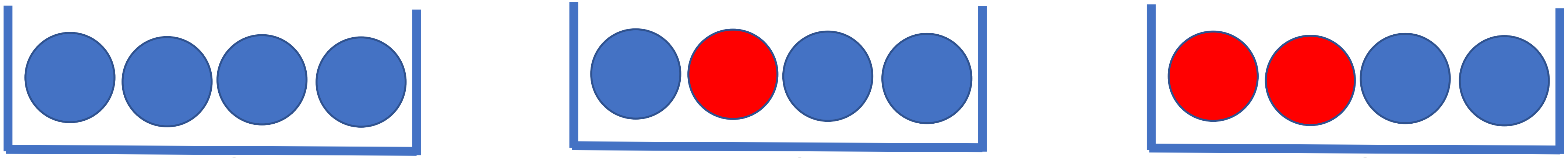
How do we choose the order of test attributes?

- To build a tree, the algorithm searches over all possible tests and finds the one that is most informative about the target variable.
- How can we identify the most informative test?
- Test attributes are selected on the basis of a heuristic or statistical measure (e.g., **information gain**)
- What are those heuristic or statistical measures?
- What is information gain?
- First let's learn a new concept: Entropy

Entropy

- A measure of uncertainty for a random variable

  - Calculation: For a discrete random variable Y taking m distinct values $\{y_1, \ldots, y_m\}$,
    - $H(Y) = -\sum_{i=1}^{m} p_i \log(p_i)$, where $p_i = P(Y = y_i)$



Bucket 1

Entropy = 0

Bucket 2

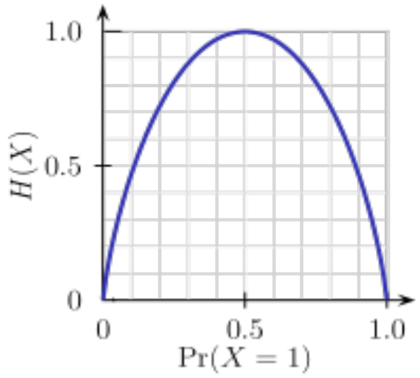
Entropy= 0.81125

Bucket 3

Entropy = 1

Brief review of Entropy

- Entropy (Information Theory)
  - A measure of uncertainty associated with a random variable
  - Calculation: For a discrete random variable *Y* taking *m* distinct values $\{y_1, \ldots, y_m\}$,
    - $H(Y) = -\sum_{i=1}^{m} p_i \log(p_i)$, where $p_i = P(Y = y_i)$
  - Interpretation:
    - Higher entropy => higher uncertainty
    - Lower entropy => lower uncertainty
- Conditional Entropy
  - $H(Y|X) = \sum_x p(x) H(Y|X = x)$



m = 2

Attribute selection measure Information Gain (ID3/C4.5)

- Select the attribute with the highest information gain
- Let p_i be the probability that an arbitrary tuple in D belongs to class C_i, estimated by $|C_{i,D}| / |D|$
- Missing information (entropy) needed to classify a tuple in D:

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

- Missing Information after using A to split D into v partitions :

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

- Information gained by branching on attribute A

$$Gain(A) = Info(D) - Info_A(D)$$

Attribute selection, Information Gain

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"

$$Info(D) = I(9,5) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940$$

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
31…40	4	0	0
>40	3	2	0.971

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

$$Info_{age}(D) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2) = 0.694$$

$\frac{5}{14}I(2,3)$ means "age <=30" has 5 out of 14 samples, with 2 yes'es and 3 no's. Hence

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

Similarly,

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit\_rating) = 0.048$$

Computing Information Gain for Continuous-valued attributes

- Let attribute A be a continuous-valued attribute
- Must determine the *best split point* for A
  - Sort the value A in increasing order
  - Typically, the midpoint between each pair of adjacent values is considered as a possible *split point*
    - $(a_i+a_{i+1})/2$ is the midpoint between the values of a_i and a_{i+1}
  - The point with the *minimum expected information requirement* for A is selected as the split-point for A
- Split:
  - D1 is the set of tuples in D satisfying A ≤ split-point, and D2 is the set of tuples in D satisfying A > split-point

Gain ratio for attribute selection

- Information gain measure is biased towards attributes with a large number of values
- C4.5 (a successor of ID3) uses gain ratio to overcome the problem (normalisation to information gain)

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

  - GainRatio(A) = Gain(A)/SplitInfo(A)
- Example $SplitInfo_{income}(D) = -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right) = 1.557$
  - gain_ratio(income) = 0.029/1.557 = 0.019
- The attribute with the maximum gain ratio is selected as the splitting attribute

Gini Index (Cart, IBM Intelligent Miner)

- If a data set *D* contains examples from *n* classes, gini index, *gini*(*D*) is defined as

$$gini(D) = 1 - \sum_{j=1}^{n} p_j^2$$

where p_j is the relative frequency of class *j* in *D*

- If a data set *D* is split on A into two subsets D_1 and D_2, the *gini* index *gini*(*D*) is defined as

$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$$

- Reduction in Impurity:

$$\Delta gini(A) = gini(D) - gini_A(D)$$

- The attribute provides the smallest $gini_{split}(D)$ (or the largest reduction in impurity) is chosen to split the node (*need to enumerate all the possible splitting points for each attribute*)

Computation of Gini index

- Example: D has 9 tuples in buys_computer = “yes” and 5 in “no”

$$gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

- Suppose the attribute income partitions D into 10 in D_1: {low, medium} and 4 in D_2

$$gini_{income \in \{low, medium\}}(D) = \left(\frac{10}{14}\right)Gini(D_1) + \left(\frac{4}{14}\right)Gini(D_2)$$

$$= \frac{10}{14}\left(1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2\right) + \frac{4}{14}\left(1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2\right)$$

$$= 0.443$$

$$= Gini_{income \in \{high\}}(D).$$

$Gini_{\{low,high\}}$ is 0.458; $Gini_{\{medium,high\}}$ is 0.450. Thus, split on the {low,medium} (and {high}) since it has the lowest Gini index

Comparing attribute selection methods

- The three measures in general return good result.
  - **Information gain**:
    - biased towards multivalued attributes
  - **Gain ratio**:
    - tends to prefer unbalanced splits in which one partition is much smaller than the others
  - **Gini index**:
    - biased to multivalued attributes
    - has difficulty when number of classes is large
    - tends to favour tests that result in equal-sized partitions and purity in both partitions

Decision Trees for Regression

- Similar to decision trees for classification but
- Decision Tree Regressor is not is not able to extrapolate (make predictions outside of the range of the training data)
- Muller and Guido’s book, page 84

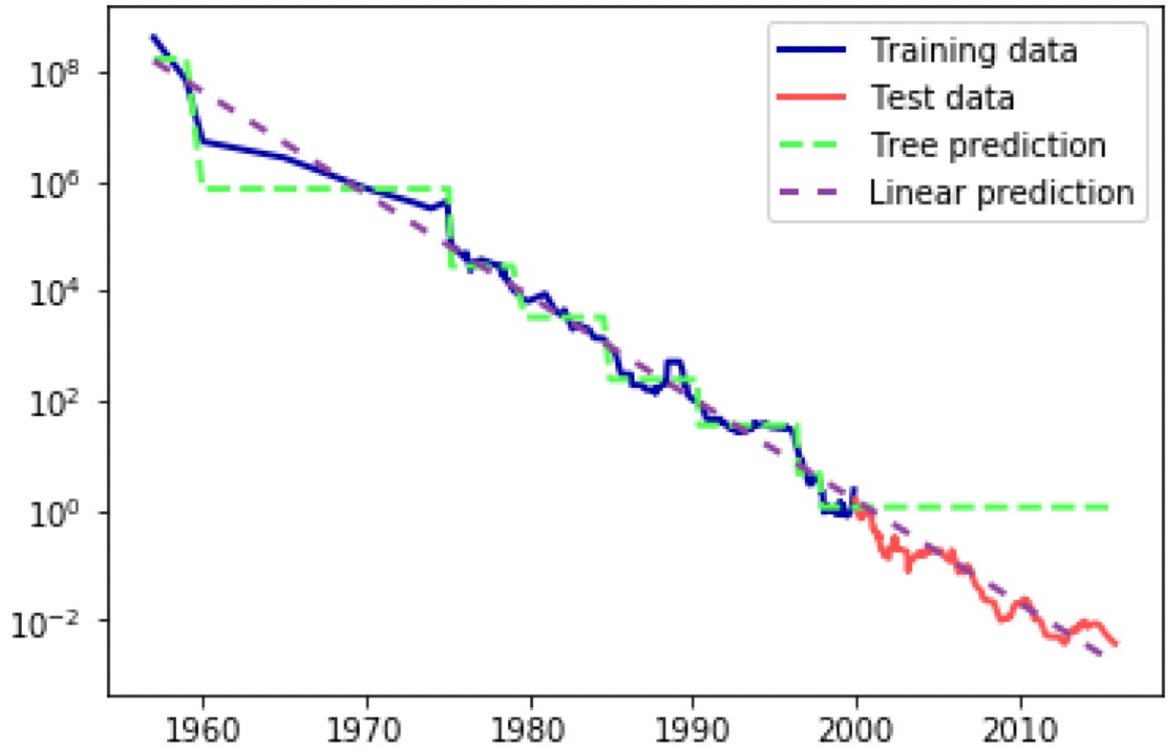


Figure 2-32. Comparison of predictions made by a linear model and predictions made by a regression tree on the RAM price data

Overfitting and Tree Pruning

- Overfitting: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples
- Two approaches to avoid overfitting
  - Pre-pruning: *Halt tree construction early* - do not split a node if this would result in the goodness measure falling below a threshold
    - Difficult to choose an appropriate threshold
  - Post-pruning: *Remove branches* from a "fully grown" tree— removing or collapsing nodes that contain little information

LONDON
METROPOLITAN
UNIVERSITY

Strengths, weaknesses and parameters

Parameters

- pre-pruning parameters such as:
  - maximum depth of the tree
  - maximum number of leaves
  - a minimum number of points in a node to keep splitting it

Strengths

- Can be easily visualized and understood by non-experts
- The algorithms are completely invariant to scaling of the data
  - no pre-processing like normalization or standardization of features is needed for decision tree algorithms.
- Work well when you have features that are on completely different scales or a mix of binary and continuous features

Weaknesses

- They tend to overfit (even with pre-pruning)
- Poor generalisation performance
- Ensemble methods are used instead of single trees

Ensemble of Decision Trees

Further Reading and possible topics for coursework

- Random Forest
- Gradient boosted regression trees