dm25s1

Topic 10 : Classification2

Part 01 : DecisionTree

Dr Bernard Butler

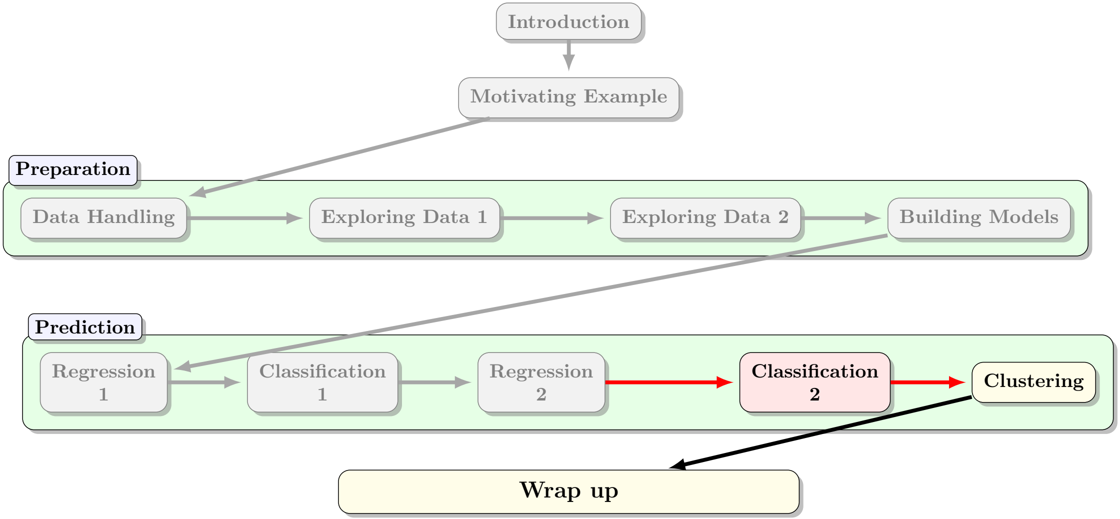
Department of Computing and Mathematics, WIT.
(bernard.butler@setu.ie)

Autumn Semester, 2025

Outline

- How Decision Trees work
- How Decision Trees are used

Data Mining (Week 10)



Outline

Objectives

In this topic, you are introduced to some more advanced techniques used for classification. Remember: you have already met *logistic regression* and *k nearest neighbours* in Topics 8 and 2, respectively.
The new approaches are:

Objectives

In this topic, you are introduced to some more advanced techniques used for classification. Remember: you have already met *logistic regression* and *k nearest neighbours* in Topics 8 and 2, respectively.
The new approaches are:

- A technique that uses a series of questions to classify a data set (Decision Trees)

Objectives

In this topic, you are introduced to some more advanced techniques used for classification. Remember: you have already met *logistic regression* and *k nearest neighbours* in Topics 8 and 2, respectively.
The new approaches are:

- A technique that uses a series of questions to classify a data set (Decision Trees)
- A technique that uses probability directly to classify a data set (Naive Bayes)

Objectives

In this topic, you are introduced to some more advanced techniques used for classification. Remember: you have already met *logistic regression* and *k nearest neighbours* in Topics 8 and 2, respectively.
The new approaches are:

- A technique that uses a series of questions to classify a data set (Decision Trees)
- A technique that uses probability directly to classify a data set (Naive Bayes)

These are two of the Top 10 algorithms in data mining (**WuKumarRossQuinlanEtAl2008**), each with its own strengths and weaknesses.

Outline

Motivation

Twenty Questions is a powerful way of learning (identifying something)

Can it be used to predict categorical variables?

Motivation

Twenty Questions is a powerful way of learning (identifying something)

Can it be used to predict categorical variables?

- Assume we have a set of l labels to assign to n_{test} data observations

Motivation

Twenty Questions is a powerful way of learning (identifying something)

Can it be used to predict categorical variables?

- Assume we have a set of l labels to assign to n_{test} data observations
- During training we repeatedly partition the training set using a sequence of ever finer rules

Motivation

Twenty Questions is a powerful way of learning (identifying something)

Can it be used to predict categorical variables?

- Assume we have a set of l labels to assign to n_{test} data observations
- During training we repeatedly partition the training set using a sequence of ever finer rules
- The resulting decision tree generates a mapping from *attributes* of an item (the mapping is defined by a path from root to leaf) to conclusions about the item's target value (represented in the leaves).

Motivation

Twenty Questions is a powerful way of learning (identifying something)

Can it be used to predict categorical variables?

- Assume we have a set of l labels to assign to n_{test} data observations
- During training we repeatedly partition the training set using a sequence of ever finer rules
- The resulting decision tree generates a mapping from *attributes* of an item (the mapping is defined by a path from root to leaf) to conclusions about the item's target value (represented in the leaves).
- The rules which generate the binary splits are applied in a greedy fashion and are intended to reduce the *impurity* in each nodes' children as quickly as possible

Motivation

Twenty Questions is a powerful way of learning (identifying something)

Can it be used to predict categorical variables?

- Assume we have a set of l labels to assign to n_{test} data observations
- During training we repeatedly partition the training set using a sequence of ever finer rules
- The resulting decision tree generates a mapping from *attributes* of an item (the mapping is defined by a path from root to leaf) to conclusions about the item's target value (represented in the leaves).
- The rules which generate the binary splits are applied in a greedy fashion and are intended to reduce the *impurity* in each nodes' children as quickly as possible
- the algorithm proceeds top-down from the root (all data), recursively generating rules as it goes

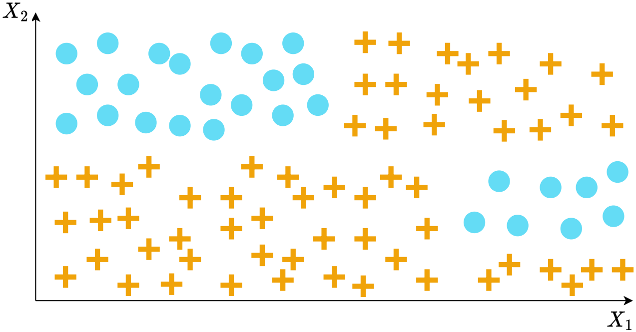
Motivation

Twenty Questions is a powerful way of learning (identifying something)

Can it be used to predict categorical variables?

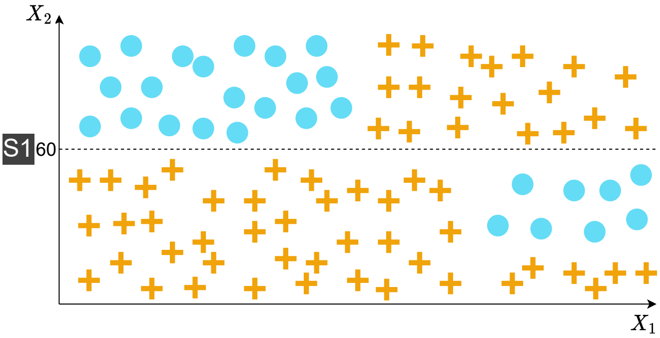
- Assume we have a set of l labels to assign to n_{test} data observations
- During training we repeatedly partition the training set using a sequence of ever finer rules
- The resulting decision tree generates a mapping from *attributes* of an item (the mapping is defined by a path from root to leaf) to conclusions about the item's target value (represented in the leaves).
- The rules which generate the binary splits are applied in a greedy fashion and are intended to reduce the *impurity* in each nodes' children as quickly as possible
- the algorithm proceeds top-down from the root (all data), recursively generating rules as it goes
- Prediction is simple: the rules are applied along the path from root to leaf. The predicted class value is either the most frequent value at the leaf, or the leaf's probability vector.

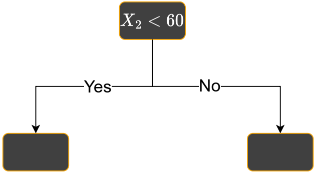
Classification tree: Example Data



Task: learn from this training data, to classify new data as either orange cross or blue disk

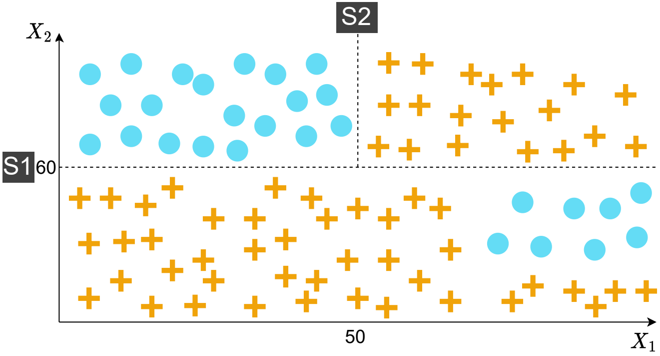
Classification tree: Example Data - First Split

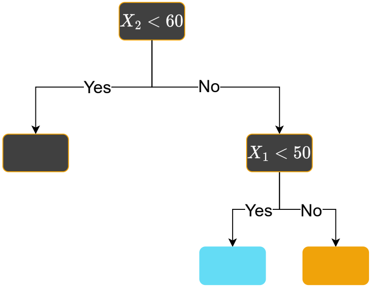




First split is on X_2; purity is improved (less mixing in each subset)

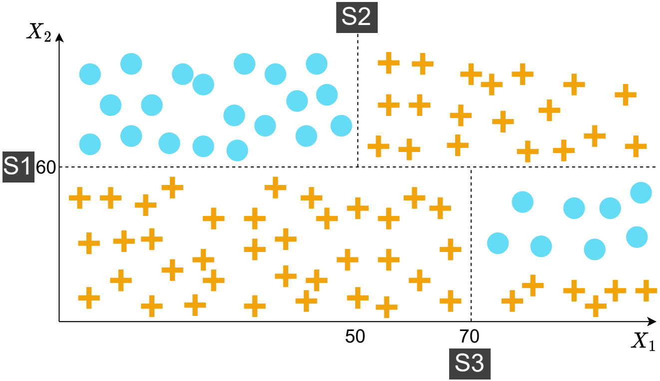
Classification tree: Example Data - Second Split

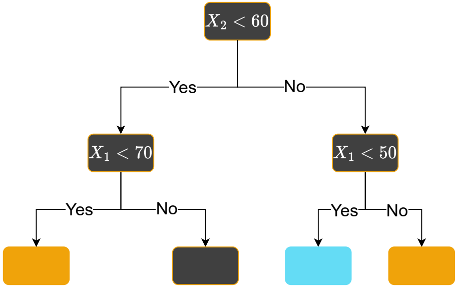




Second split is on X_1 so one region is pure (all blue disks) - can continue.

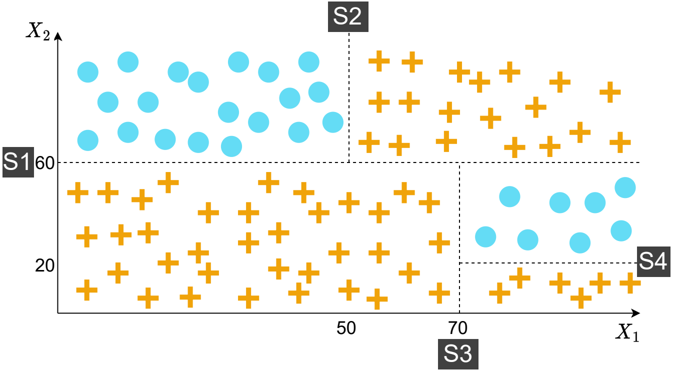
Classification tree: Example Data - Third Split

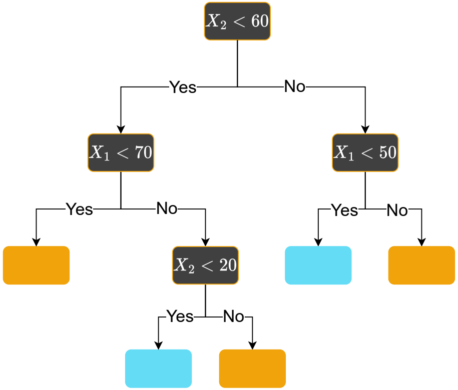




Third split on X_1 adds two extra pure regions.

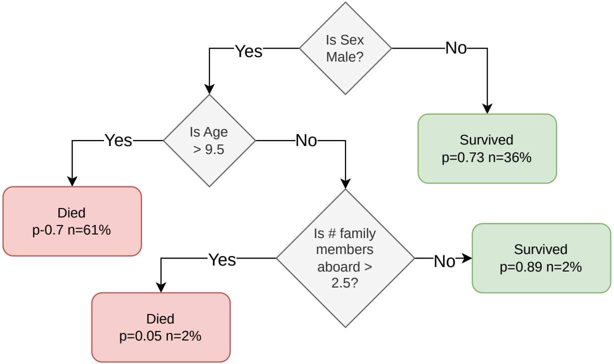
Classification tree: Example Data - Fourth Split





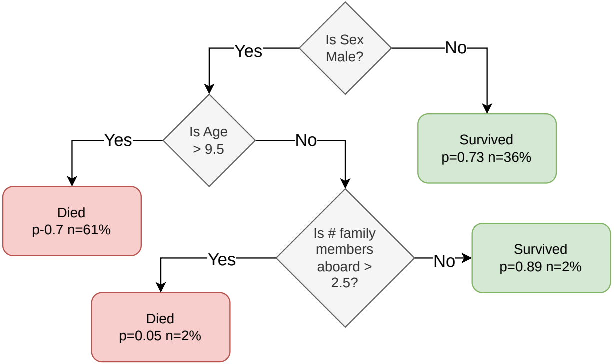
After fourth split on X_2, all regions are pure, so we stop.

Classification tree example: Titanic survival



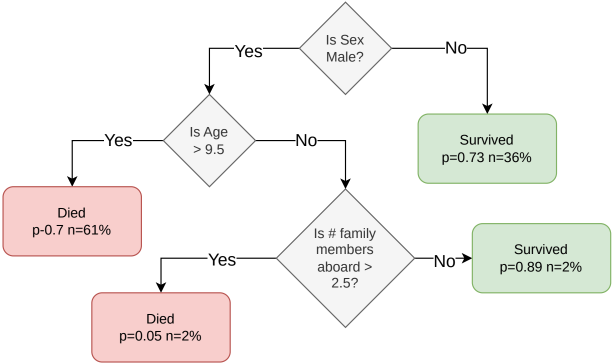
- First split is on Sex, as that attribute was the most important predictor of survival.

Classification tree example: Titanic survival



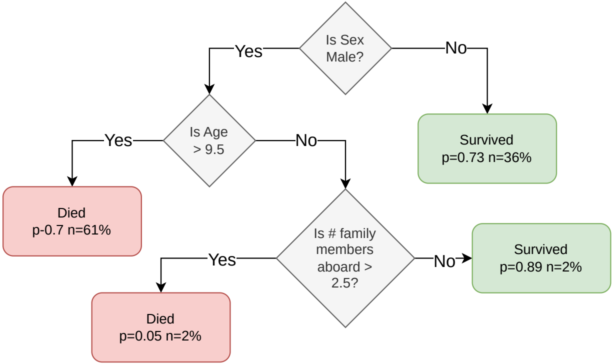
- First split is on Sex, as that attribute was the most important predictor of survival.
- Leaves show the probability of survival and the percentage of training observations in the leaf.

Classification tree example: Titanic survival



- First split is on Sex, as that attribute was the most important predictor of survival.
- Leaves show the probability of survival and the percentage of training observations in the leaf.
- Percentages sum to 100% (approximately), as the leaves partition the training data.

Classification tree example: Titanic survival



- First split is on Sex, as that attribute was the most important predictor of survival.
- Leaves show the probability of survival and the percentage of training observations in the leaf.
- Percentages sum to 100% (approximately), as the leaves partition the training data.
- Leaf colour indicates p(survival) ≈ 1 (green) or p(survival) ≈ 0 (red)

Classification tree notes

- At each step, we choose a feature and split a data subset using that feature

Classification tree notes

- At each step, we choose a feature and split a data subset using that feature
- Each split is parallel to one of the feature axes

Classification tree notes

- At each step, we choose a feature and split a data subset using that feature
- Each split is parallel to one of the feature axes
- Aim of splitting is to maximise progress to purity at each step

Classification tree notes

- At each step, we choose a feature and split a data subset using that feature
- Each split is parallel to one of the feature axes
- Aim of splitting is to maximise progress to purity at each step
- But how do we choose the feature to split? Or the value of that feature for splitting?

Classification tree notes

- At each step, we choose a feature and split a data subset using that feature
- Each split is parallel to one of the feature axes
- Aim of splitting is to maximise progress to purity at each step
- But how do we choose the feature to split? Or the value of that feature for splitting?
- The algorithm needs a suitable metric and criterion that can be calculated

Classification tree notes

- At each step, we choose a feature and split a data subset using that feature
- Each split is parallel to one of the feature axes
- Aim of splitting is to maximise progress to purity at each step
- But how do we choose the feature to split? Or the value of that feature for splitting?
- The algorithm needs a suitable metric and criterion that can be calculated
  - For each impure set

Classification tree notes

- At each step, we choose a feature and split a data subset using that feature
- Each split is parallel to one of the feature axes
- Aim of splitting is to maximise progress to purity at each step
- But how do we choose the feature to split? Or the value of that feature for splitting?
- The algorithm needs a suitable metric and criterion that can be calculated
  - For each impure set
  - For each candidate split

Information Entropy: intuition

Probability	Information	Perceived as
Low	High	Surprising
High	Low	Unsurprising

Information Entropy: intuition

Probability	Information	Perceived as
Low	High	Surprising
High	Low	Unsurprising

Example

If it is winter in Ireland, and Met Eireann forecast that tomorrow will have a temperature of 10 degrees C, nobody would pay much attention. If, on the other hand, they said it would be 40 degrees C, everyone would notice!

Information Entropy: intuition

Probability	Information	Perceived as
Low	High	Surprising
High	Low	Unsurprising

Example

If it is winter in Ireland, and Met Eireann forecast that tomorrow will have a temperature of 10 degrees C, nobody would pay much attention. If, on the other hand, they said it would be 40 degrees C, everyone would notice!

Entropy is average amount of information conveyed by an event, considering all possible outcomes.

Information Entropy: intuition

Probability	Information	Perceived as
Low	High	Surprising
High	Low	Unsurprising

Example

If it is winter in Ireland, and Met Eireann forecast that tomorrow will have a temperature of 10 degrees C, nobody would pay much attention. If, on the other hand, they said it would be 40 degrees C, everyone would notice!

Entropy is average amount of information conveyed by an event, considering all possible outcomes.

How information is measured

Information is measured in bits, and is computed from the probability $P(x)$ using $h(x) = -\log_2(P(x))$.

Information Entropy: Applied to classification

Classification and entropy

- Given a set of observations with the same label, we wish to make membership of that set as unsurprising as possible as possible, given the training set.

Information Entropy: Applied to classification

Classification and entropy

- Given a set of observations with the same label, we wish to make membership of that set as unsurprising as possible as possible, given the training set.
- Can we partition a set of observations to achieve this, so that it is “obvious” that the label assignments are correct? In other words, we want as many elements as possible to have a label that matches the label given to the set partition to which they belong. This increases the *purity* of that partition.

Information Entropy: Applied to classification

Classification and entropy

- Given a set of observations with the same label, we wish to make membership of that set as unsurprising as possible as possible, given the training set.
- Can we partition a set of observations to achieve this, so that it is "obvious" that the label assignments are correct? In other words, we want as many elements as possible to have a label that matches the label given to the set partition to which they belong. This increases the *purity* of that partition.

Definition 1 ((Information) Entropy)

$$H(X) = -\sum_{i=1}^{n} P(x_i)\log_2(P(x_i))$$

where $X = \{x_i\}$. If all probabilities are equal (X is uniformly distributed), $H(X) = 1$. If they differ, $H(X) < 1$. Remember the weather forecasting example!

Information Entropy: Applied to classification

Classification and entropy

- Given a set of observations with the same label, we wish to make membership of that set as unsurprising as possible as possible, given the training set.
- Can we partition a set of observations to achieve this, so that it is "obvious" that the label assignments are correct? In other words, we want as many elements as possible to have a label that matches the label given to the set partition to which they belong. This increases the *purity* of that partition.

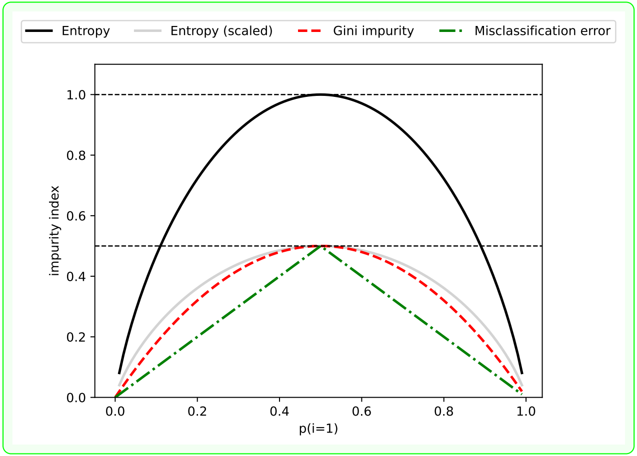
Definition 1 ((Information) Entropy)

$$H(X) = -\sum_{i=1}^{n} P(x_i)\log_2(P(x_i))$$

where $X = \{x_i\}$. If all probabilities are equal (X is uniformly distributed), $H(X) = 1$. If they differ, $H(X) < 1$. Remember the weather forecasting example!

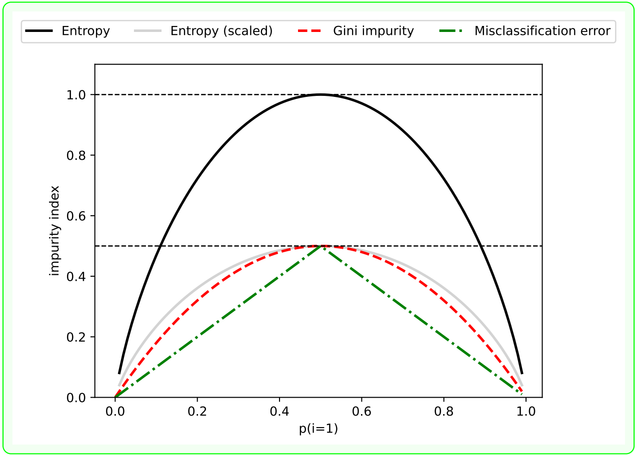
A decision tree recursively partitions a set so as to increase the purity (equivalently: reduce the mixing) of the set of observations X at each node as we move from the root to the leaves.

Classification tree metrics for rule building



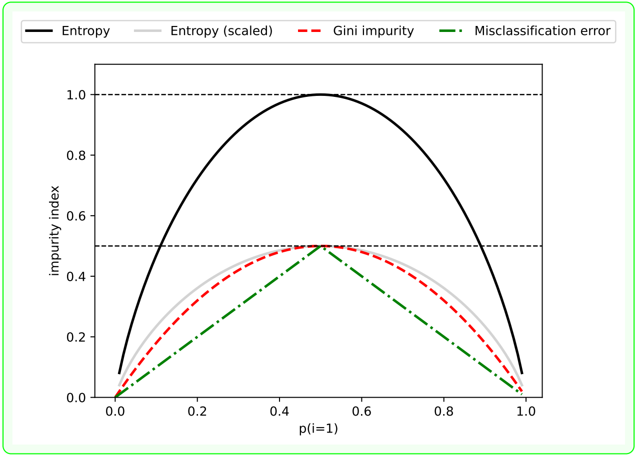
- Let p_i be the probability of an item with label $1 < i < J$ being chosen.

Classification tree metrics for rule building



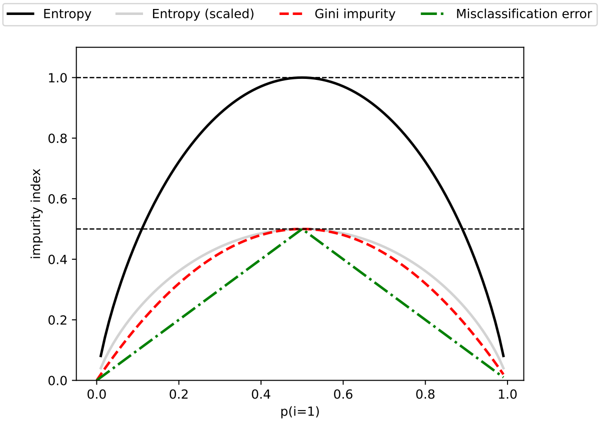
- Let p_i be the probability of an item with label $1 < i < J$ being chosen.
- Then the *GINI* impurity is $1 - \sum_{i=1}^{J} p_i^2$.

Classification tree metrics for rule building



- Let p_i be the probability of an item with label $1 < i < J$ being chosen.
- Then the *GINI* impurity is $1 - \sum_{i=1}^{J} p_i^2$.
- Worst case (maximum impurity): labels are uniformly distributed (entropy of the set is maximum: each value of the label appears the same number of times in the set of observations at that node).

Classification tree metrics for rule building



- Let p_i be the probability of an item with label $1 < i < J$ being chosen.
- Then the *GINI* impurity is $1 - \sum_{i=1}^{J} p_i^2$.
- Worst case (maximum impurity): labels are uniformly distributed (entropy of the set is maximum: each value of the label appears the same number of times in the set of observations at that node).
- We wish to minimise this entropy.

Sidebar: Entropy

Definition 2 (Entropy)

- Entropy is a concept from *thermodynamics* and *information theory*.

Sidebar: Entropy

Definition 2 (Entropy)

- Entropy is a concept from *thermodynamics* and *information theory*.
- Here it measures the *impurity* of a collection of items.

Sidebar: Entropy

Definition 2 (Entropy)

- Entropy is a concept from *thermodynamics* and *information theory*.
- Here it measures the *impurity* of a collection of items.
- It ranges from 0 (only 1 item, possibly repeated) to 1 (equal numbers of each item type).

Sidebar: Entropy

Definition 2 (Entropy)

- Entropy is a concept from *thermodynamics* and *information theory*.
- Here it measures the *impurity* of a collection of items.
- It ranges from 0 (only 1 item, possibly repeated) to 1 (equal numbers of each item type).
- Mathematically, it is defined for *one attribute* T as $H(T) = -\sum_{j=1}^{J} p_j \log_2 p_j$, in a collection of size N where there are J unique elements of T, hence $p_j = \frac{n_j}{N}$ where there are n_j elements of type j.

Sidebar: Entropy

Definition 2 (Entropy)

- Entropy is a concept from *thermodynamics* and *information theory*.
- Here it measures the *impurity* of a collection of items.
- It ranges from 0 (only 1 item, possibly repeated) to 1 (equal numbers of each item type).
- Mathematically, it is defined for *one attribute* T as $H(T) = -\sum_{j=1}^{J} p_j \log_2 p_j$, in a collection of size N where there are J unique elements of T, hence $p_j = \frac{n_j}{N}$ where there are n_j elements of type j.
- For *two attributes* T and X, $H(T,X) = \sum_{c \in X} P(c)E(c)$ where each c represents a level of the X attribute.

Sidebar: Information Gain

Definition 3 (Information Gain)

- Information Gain measures the decrease in entropy (equivalently: increase in purity) after a dataset is split on an attribute.

Sidebar: Information Gain

Definition 3 (Information Gain)

- Information Gain measures the decrease in entropy (equivalently: increase in purity) after a dataset is split on an attribute.
- It is defined as $G(T,X) = H(T) - H(T,X)$, where

Sidebar: Information Gain

Definition 3 (Information Gain)

- Information Gain measures the decrease in entropy (equivalently: increase in purity) after a dataset is split on an attribute.
- It is defined as $G(T,X) = H(T) - H(T,X)$, where
  - $H(T)$ is the entropy at the parent node, and

Sidebar: Information Gain

Definition 3 (Information Gain)

- Information Gain measures the decrease in entropy (equivalently: increase in purity) after a dataset is split on an attribute.
- It is defined as $G(T,X) = H(T) - H(T,X)$, where
  - $H(T)$ is the entropy at the parent node, and
  - $H(T,X)$ is the entropy after the split by candidate attribute X.

Example: PlayTennis example data

outlook	temp	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Source: Mitchell, *Machine Learning*, 1997.

PlayTennis example calculations

Example 4 (H(play))

$$
\begin{aligned}
H(\text{play}) &= -\left(p(\text{play} = \text{yes}) \log_2 p(\text{play} = \text{yes}) + p(\text{play} = \text{no}) \log_2 p(\text{play} = \text{no})\right) \\
&= H_{9,5} \\
&= -\left(\frac{9}{14} \log_2\left(\frac{9}{14}\right) + \frac{5}{14} \log_2\left(\frac{5}{14}\right)\right) \approx 0.94
\end{aligned}
$$

PlayTennis example calculations

Example 4 (H(play))

$$\begin{aligned} H(\text{play}) &= -\left(p(\text{play}=\text{yes})\log_2 p(\text{play}=\text{yes}) + p(\text{play}=\text{no})\log_2 p(\text{play}=\text{no})\right) \\ &= H_{9,5} \\ &= -\left(\frac{9}{14}\log_2\left(\frac{9}{14}\right) + \frac{5}{14}\log_2\left(\frac{5}{14}\right)\right) \approx 0.94 \end{aligned}$$

Example 5 (H(play,outlook))

$$\begin{aligned} H(\text{play},\text{outlook}) &= p(\text{outlook}=\text{sunny})H(\text{play}\&(\text{outlook}=\text{sunny})) + \dots \\ &= p(\text{outlook}=\text{sunny})H_{3,2} + p(\text{outlook}=\text{overcast})H_{4,0} + \dots \\ &\approx \frac{5}{14}0.97 + \frac{4}{14}0 + \frac{5}{14}0.97 \\ &\approx 0.69 \end{aligned}$$

PlayTennis example calculations

Example 4 (H(play))

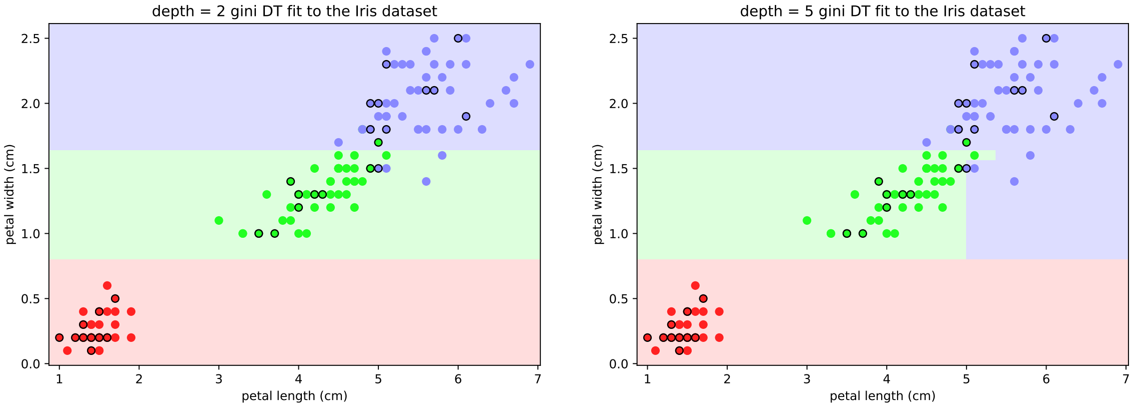
$$\begin{aligned} H(\text{play}) &= -\left(p(\text{play}=\text{yes})\log_2 p(\text{play}=\text{yes}) + p(\text{play}=\text{no})\log_2 p(\text{play}=\text{no})\right) \\ &= H_{9,5} \\ &= -\left(\frac{9}{14}\log_2\left(\frac{9}{14}\right) + \frac{5}{14}\log_2\left(\frac{5}{14}\right)\right) \approx 0.94 \end{aligned}$$

Example 5 (H(play,outlook))

$$\begin{aligned} H(\text{play},\text{outlook}) &= p(\text{outlook}=\text{sunny})H(\text{play}\&(\text{outlook}=\text{sunny})) + \dots \\ &= p(\text{outlook}=\text{sunny})H_{3,2} + p(\text{outlook}=\text{overcast})H_{4,0} + \dots \\ &\approx \frac{5}{14}0.97 + \frac{4}{14}0 + \frac{5}{14}0.97 \\ &\approx 0.69 \end{aligned}$$

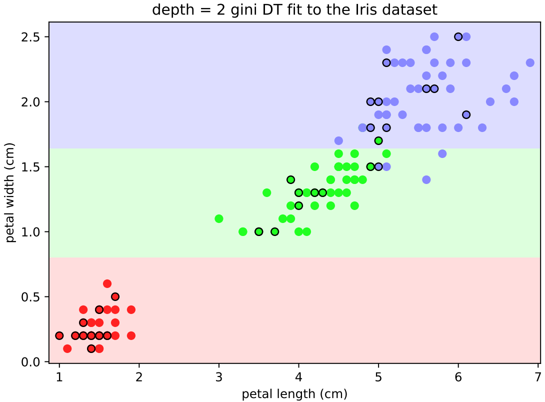
When growing decision trees, at a given node we search over the attributes for splitting, and choose the one that gives the maximum information gain, until we reach a leaf, which has an entropy of zero.

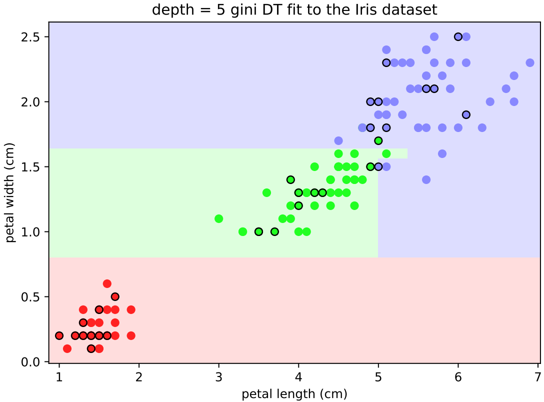
Classification tree examples: Iris Data



Note the rectangular regions (because each split is over one variable) and the greater complexity when the maximum depth of the tree increases.

Classification tree examples: Iris Data

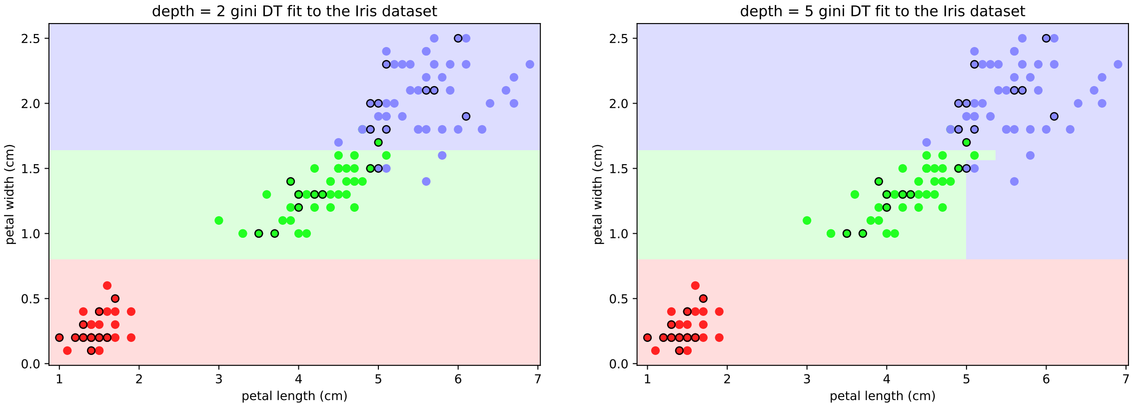




Note the rectangular regions (because each split is over one variable) and the greater complexity when the maximum depth of the tree increases.

Points within a dark circle represent test data, with the main colour of the point indicating its species label.

Classification tree examples: Iris Data



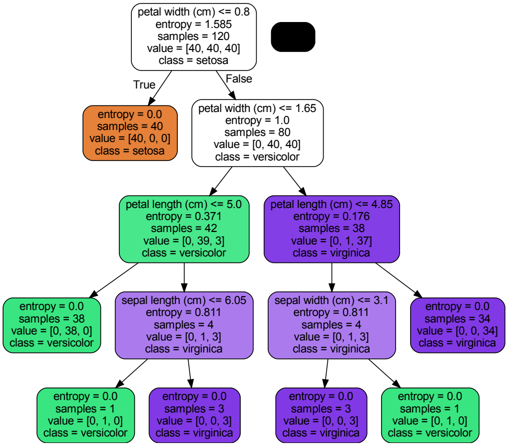
Note the rectangular regions (because each split is over one variable) and the greater complexity when the maximum depth of the tree increases.

Points within a dark circle represent test data, with the main colour of the point indicating its species label.

The choice of metric (Gini impurity or Information Gain) makes only slight changes to fit.

Classification tree view: Iris Data

- Note that the leaf nodes are pure (entropy=0) and are coloured according to predicted value (species label): brown for *I. setosa*, green for *I. versicolor* and purple for *I. virginica*.
- Also, the maximum entropy occurs at the root, where there are 40 of each of the 3 species, resulting in entropy = $\log_2(3)$.



Classification tree: Use for Prediction

Now that we have a decision tree, how do we use it to predict the label?

Classification tree: Use for Prediction

Now that we have a decision tree, how do we use it to predict the label?

Example: estimating the species of an iris plant

Classification tree: Use for Prediction

Now that we have a decision tree, how do we use it to predict the label?

Example: estimating the species of an iris plant

- Let Petal_Width = 1.5cm and Petal_Length = 5cm. The other (sepal) dimensions are ignored by the decision tree because they were not as useful for classification.

Classification tree: Use for Prediction

Now that we have a decision tree, how do we use it to predict the label?

Example: estimating the species of an iris plant

- Let Petal_Width = 1.5cm and Petal_Length = 5cm. The other (sepal) dimensions are ignored by the decision tree because they were not as useful for classification.
- The first split (Petal_Width ≤ 0.8) is `False` so we take the *right* branch.

Classification tree: Use for Prediction

Now that we have a decision tree, how do we use it to predict the label?

Example: estimating the species of an iris plant

- Let Petal_Width = 1.5cm and Petal_Length = 5cm. The other (sepal) dimensions are ignored by the decision tree because they were not as useful for classification.
- The first split (Petal_Width $\leq 0.8)$ is `False` so we take the *right* branch.
- The second split (Petal_Width ≤ 1.75 is `True` so we take the *left* branch.

Classification tree: Use for Prediction

Now that we have a decision tree, how do we use it to predict the label?

Example: estimating the species of an iris plant

- Let Petal_Width = 1.5cm and Petal_Length = 5cm. The other (sepal) dimensions are ignored by the decision tree because they were not as useful for classification.
- The first split (Petal_Width ≤ 0.8) is `False` so we take the *right* branch.
- The second split (Petal_Width ≤ 1.75 is `True` so we take the *left* branch.
- The third split (Petal_Length ≤ 4.95 is `False` so we take the *right* branch.

Classification tree: Use for Prediction

Now that we have a decision tree, how do we use it to predict the label?

Example: estimating the species of an iris plant

- Let Petal_Width = 1.5cm and Petal_Length = 5cm. The other (sepal) dimensions are ignored by the decision tree because they were not as useful for classification.
- The first split (Petal_Width ≤ 0.8) is `False` so we take the *right* branch.
- The second split (Petal_Width ≤ 1.75 is `True` so we take the *left* branch.
- The third split (Petal_Length ≤ 4.95 is `False` so we take the *right* branch.
- The fourth split (Petal_Width ≤ 1.6 is `True` so we take the *left* branch.

Classification tree: Use for Prediction

Now that we have a decision tree, how do we use it to predict the label?

Example: estimating the species of an iris plant

- Let Petal_Width = 1.5cm and Petal_Length = 5cm. The other (sepal) dimensions are ignored by the decision tree because they were not as useful for classification.
- The first split (Petal_Width ≤ 0.8) is `False` so we take the *right* branch.
- The second split (Petal_Width ≤ 1.75 is `True` so we take the *left* branch.
- The third split (Petal_Length ≤ 4.95 is `False` so we take the *right* branch.
- The fourth split (Petal_Width ≤ 1.6 is `True` so we take the *left* branch.
- We have reached a leaf (entropy = 0) and cannot split any further.

Classification tree: Use for Prediction

Now that we have a decision tree, how do we use it to predict the label?

Example: estimating the species of an iris plant

- Let Petal_Width = 1.5cm and Petal_Length = 5cm. The other (sepal) dimensions are ignored by the decision tree because they were not as useful for classification.
- The first split (Petal_Width ≤ 0.8) is `False` so we take the *right* branch.
- The second split (Petal_Width ≤ 1.75 is `True` so we take the *left* branch.
- The third split (Petal_Length ≤ 4.95 is `False` so we take the *right* branch.
- The fourth split (Petal_Width ≤ 1.6 is `True` so we take the *left* branch.
- We have reached a leaf (entropy = 0) and cannot split any further.
- Depending on where we stop, we would assign the prevalent label for that node (`versicolor`, `versicolor`, `virginica` or `virginica` if the `max_depth` was 2, 3, 4 or 5, respectively).

Classification tree: Use for Prediction

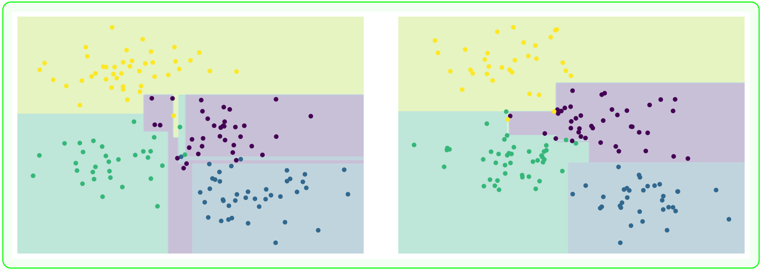
Now that we have a decision tree, how do we use it to predict the label?

Example: estimating the species of an iris plant

- Let Petal_Width = 1.5cm and Petal_Length = 5cm. The other (sepal) dimensions are ignored by the decision tree because they were not as useful for classification.
- The first split (Petal_Width ≤ 0.8) is `False` so we take the *right* branch.
- The second split (Petal_Width ≤ 1.75 is `True` so we take the *left* branch.
- The third split (Petal_Length ≤ 4.95 is `False` so we take the *right* branch.
- The fourth split (Petal_Width ≤ 1.6 is `True` so we take the *left* branch.
- We have reached a leaf (entropy = 0) and cannot split any further.
- Depending on where we stop, we would assign the prevalent label for that node (`versicolor`, `versicolor`, `virginica` or `virginica` if the `max_depth` was 2, 3, 4 or 5, respectively).

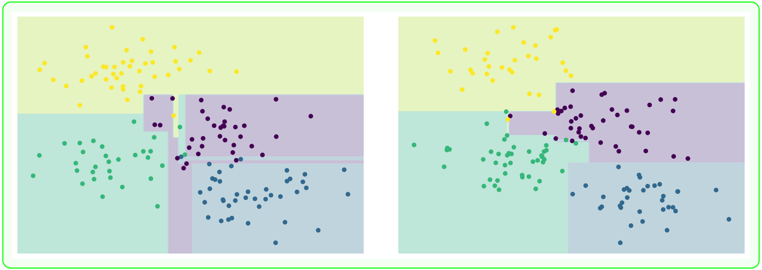
Python can extract paths from the root to each leaf as a set of if-then-else rules, to explain decisions.

Be careful of overfitting. . .



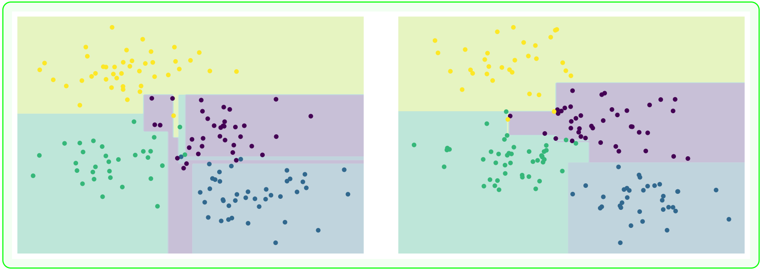
- Given two samples from the same noisy data set, the Decision Tree model fitted to each has no restrictions, so all its leaves are *pure*.

Be careful of overfitting. . .



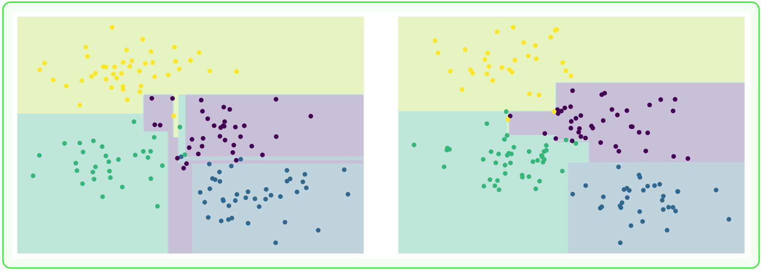
- Given two samples from the same noisy data set, the Decision Tree model fitted to each has no restrictions, so all its leaves are *pure*.
- The resulting decision trees look very different, especially in the middle.

Be careful of overfitting. . .



- Given two samples from the same noisy data set, the Decision Tree model fitted to each has no restrictions, so all its leaves are *pure*.
- The resulting decision trees look very different, especially in the middle.
- This sensitivity to the noise in the data is characteristic of *overfitting* (high variance).

Be careful of overfitting...



- Given two samples from the same noisy data set, the Decision Tree model fitted to each has no restrictions, so all its leaves are *pure*.
- The resulting decision trees look very different, especially in the middle.
- This sensitivity to the noise in the data is characteristic of *overfitting* (high variance).
- Control by a) limiting depth or b) limiting number of leaves.

Classification trees in python

```python
criterion = "entropy"
treeDepth = 5
tree = DecisionTreeClassifier(criterion=criterion, max_depth=treeDepth, random_state=0)
tree.fit(Xtrain, ytrain)
y_treeTest = tree.predict(Xtest)
print(accuracy_score(ytest, y_treeTest))
print(confusion_matrix(ytest, y_treeTest))
print(classification_report(ytest, y_treeTest, digits=3))
```

After creating the classifier object, fit the training data and then use the fit to predict yTest from xTest. I have also shown how to get some diagnostic output. Similar diagnostics can be obtained for other classifiers.

Classification Trees - summary

- Classification trees learn recursive feature splits to predict categorical targets
- After training, they are relatively easy to use and to interpret (white-box, not black-box)
- Use stopping criteria (e.g., max depth of tree) to control bias and variance: e.g., early stopping results in shallower trees, resulting in higher bias and lower variance
- The goal is to maximise the purity in the leaves, as measured by the *entropy* of the distribution of the target class labels at each leaf.
- This *entropy at each leaf* is related, but different to, the *cross-entropy of the classifier model* (not just the leaves)
    - A parent node is split into children if the sum of their entropy scores is less than the entropy of the parent node. This occurs when the child nodes have less mixing of labels and so are more pure.
    - Each leaf node takes its predicted value (of the target) from the majority class label of the subset of training observations resulting in that node.
    - Cross-entropy loss measures the difference between the distributions of the true and predicted targets and can be calculated for any classifier.
- Classification tree "stumps" are commonly used as base models in ensemble techniques like bagging, boosting and stacking. For example, RandomForest models combine large numbers of classification trees using *bagging* and use random feature selection per base model.