dm24s1

Topic 10 : Classification2

Part 01 : Classification2-Overview

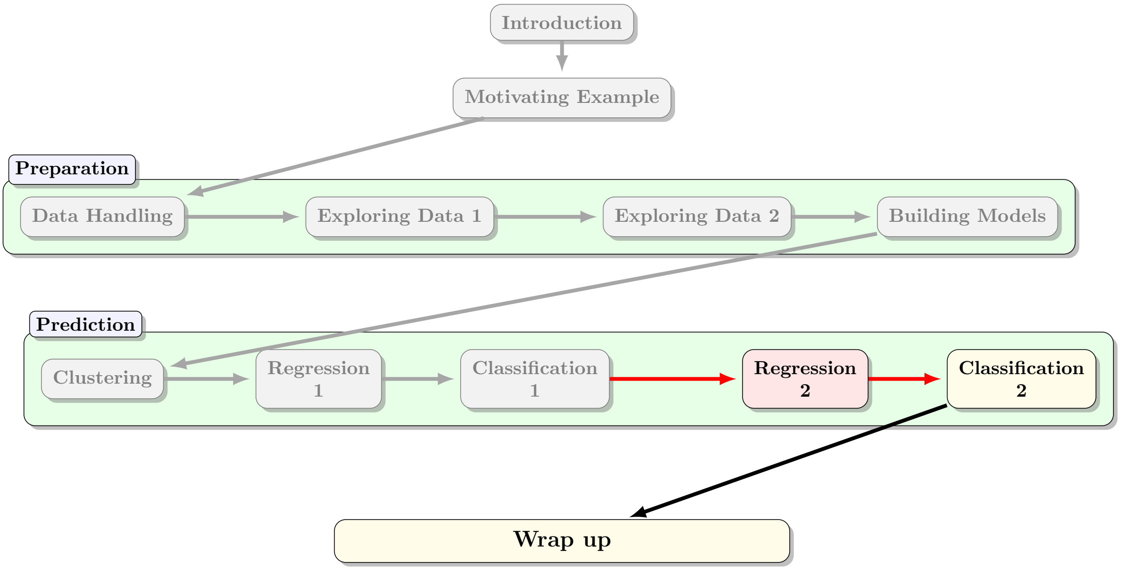
Dr Bernard Butler
Department of Computing and Mathematics, WIT.
(bernard.butler@setu.ie)

Autumn Semester, 2024

Outline

- Decision Trees
- Naive Bayes

Data Mining (Week 10)



Classification2-Overview — Summary

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

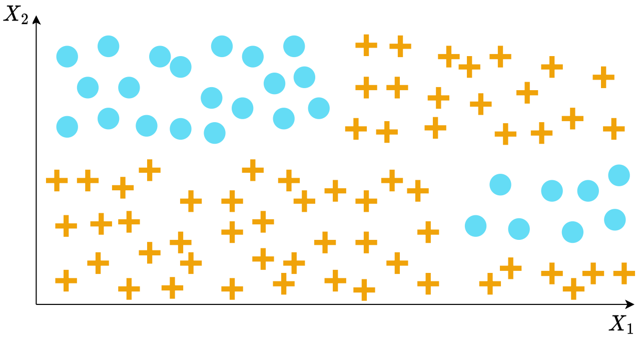
Motivation

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

Can it be used to predict categorical variables?

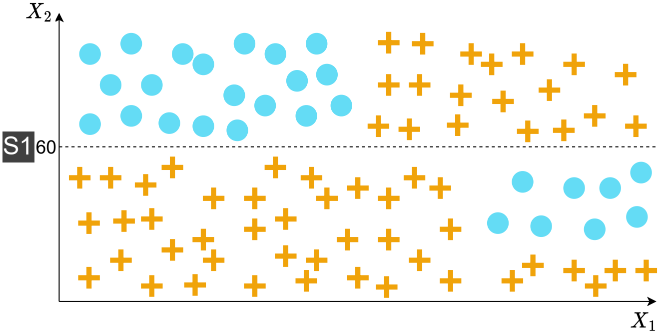
- Assume we have a set of l labels to assign to n_{test} data observations
- During training we repeatedly partition the training set using a sequence of ever finer rules
- The resulting decision tree generates a mapping from *attributes* of an item (the mapping is defined by a path from root to leaf)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

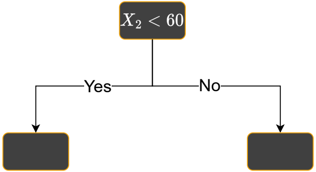
Classification tree: Example Data



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

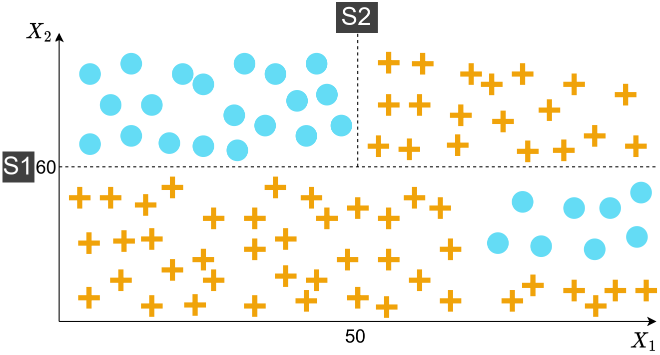
Classification tree: Example Data - First Split

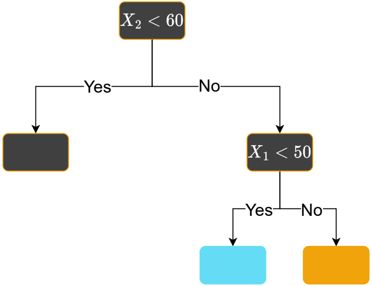




First split is on X_2; purity is improved (less mixing in each subset)

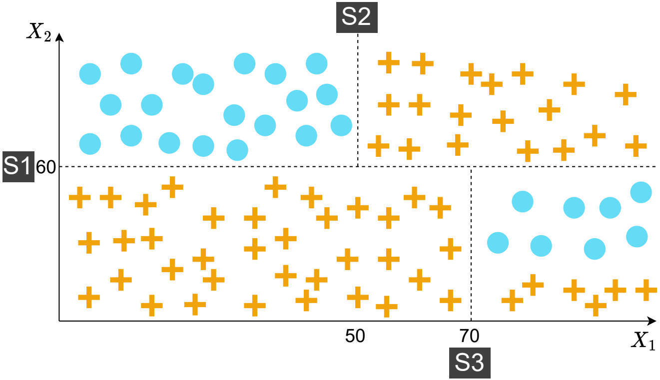
Classification tree: Example Data - Second Split

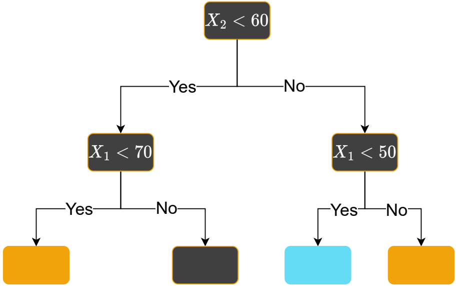




Second split is on X_1 so one region is pure (all blue disks) - can continue.

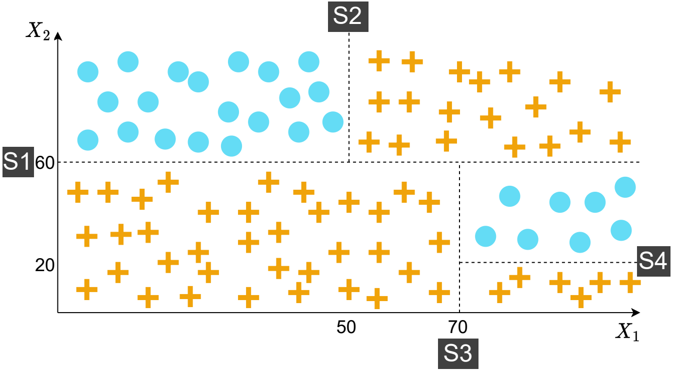
Classification tree: Example Data - Third Split

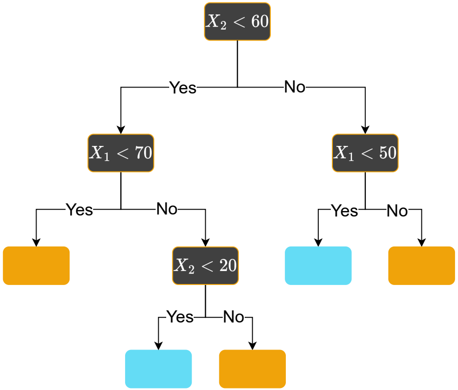




Third split on X_1 adds two extra pure regions.

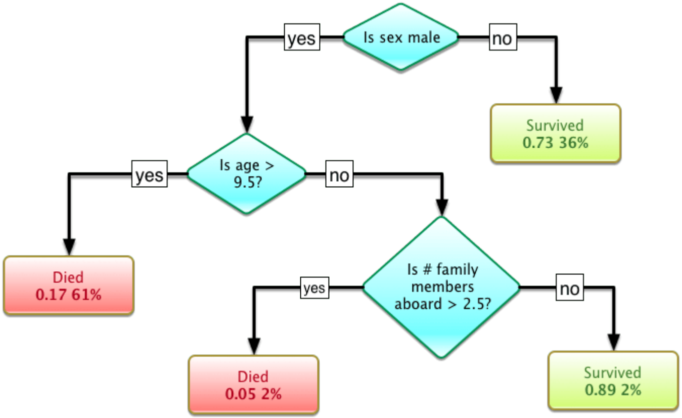
Classification tree: Example Data - Fourth Split





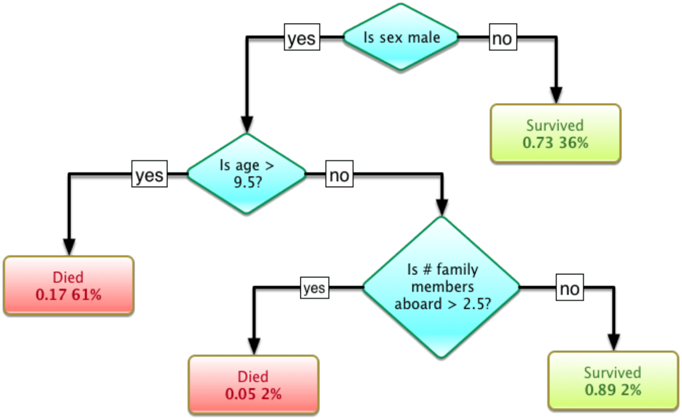
After fourth split on X_2, all regions are pure, so we stop.

Classification tree example: Titanic survival



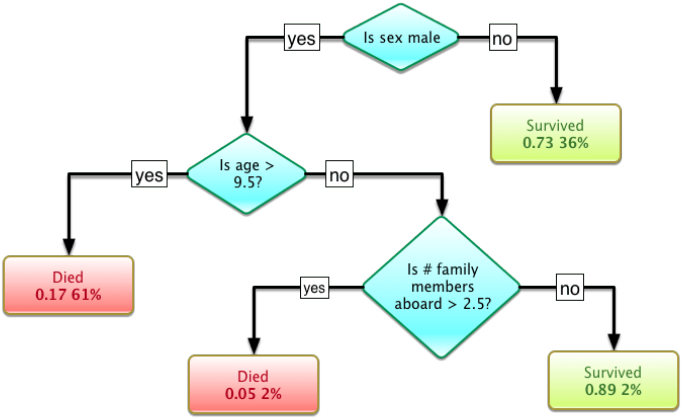
- First split is on Sex, as that attribute was the most important predictor of survival.

Classification tree example: Titanic survival



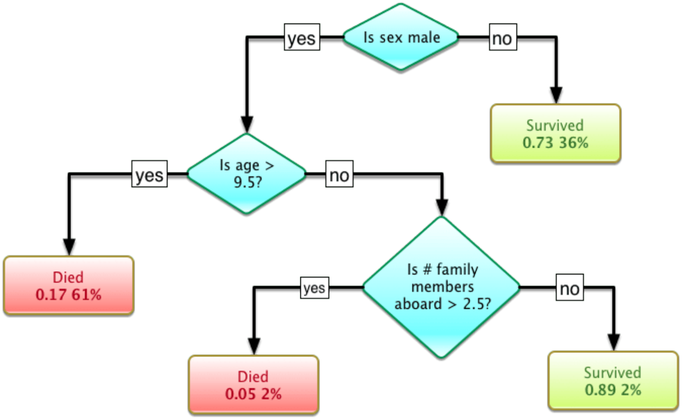
- First split is on Sex, as that attribute was the most important predictor of survival.
- Leaves show the probability of survival and the percentage of training observations in the leaf.

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) $\approx$ 1 (green) or p(survival) $\approx$ 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.

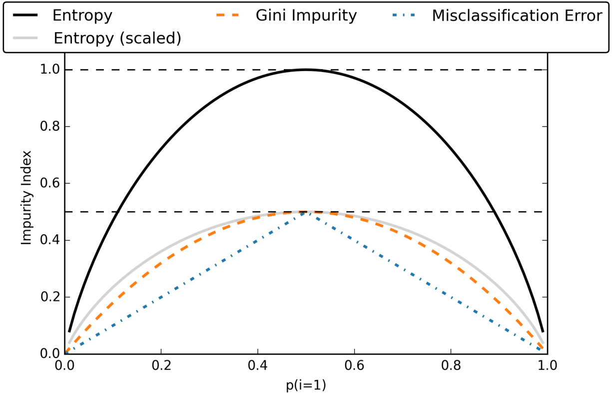
Definition 1 ((Information) Entropy)

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

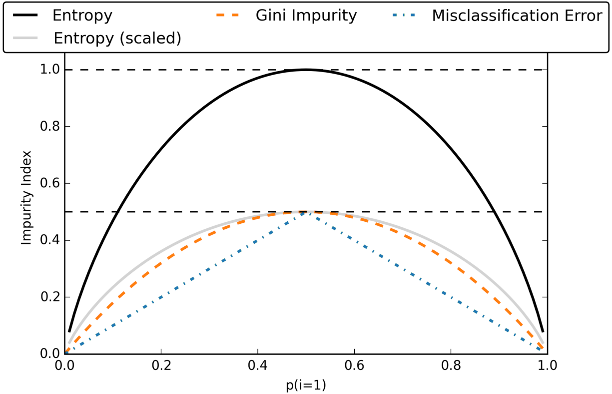
where $X = \{x_i\}$. If all probabilities are equal (X is uniformly distributed), $H(X) = 1$. If they differ, $H(X) < 1$. Remember the weather forecasting example!

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

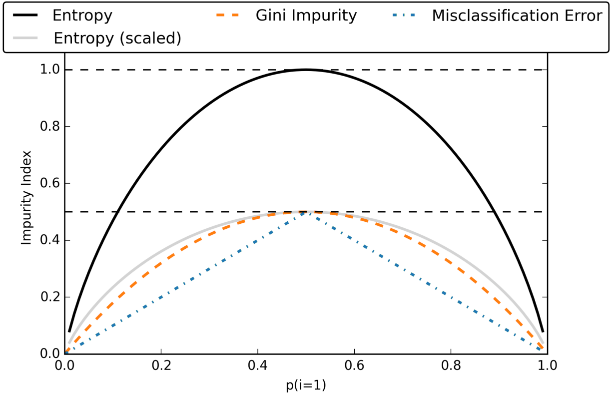


Classification tree metrics for rule building



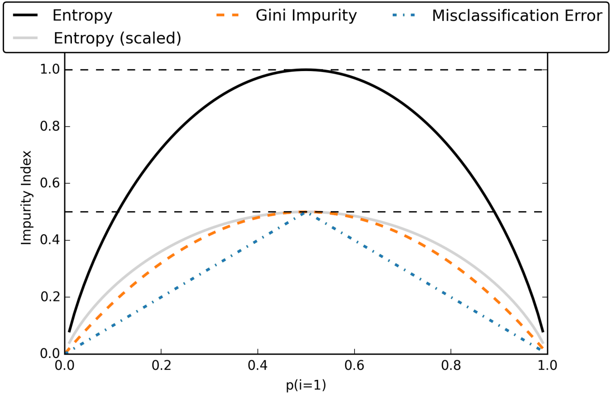
Let p_i be the probability of an item with label $1 < i < J$ being chosen.

Classification tree metrics for rule building



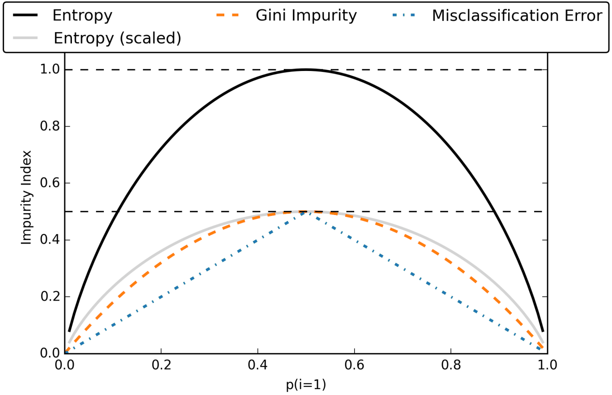
Let p_i be the probability of an item with label $1 < i < J$ being chosen. Then the *GINI* impurity is $1 - \sum_{i=1}^{J} p_i^2$.

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))

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

PlayTennis example calculations

Example 4 (H(play))

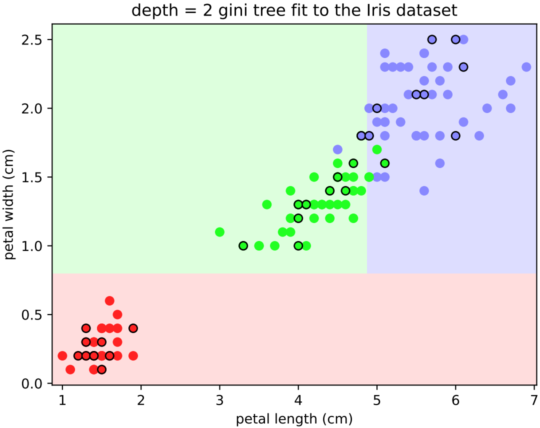
$$\begin{aligned} H(\text{play}) &= -\left(p(\text{play} = \text{yes}) \log_2 p(\text{play} = \text{yes}) + p(\text{play} = \text{no}) \log_2 p(\text{play} = \text{no})\right) \\ &= H_{9,5} \\ &= -\left(\frac{9}{14} \log_2\left(\frac{9}{14}\right) + \frac{5}{14} \log_2\left(\frac{5}{14}\right)\right)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

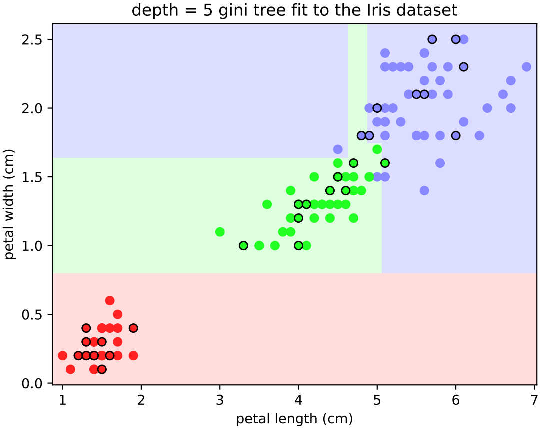
Example 5 (H(play,outlook))

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

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.

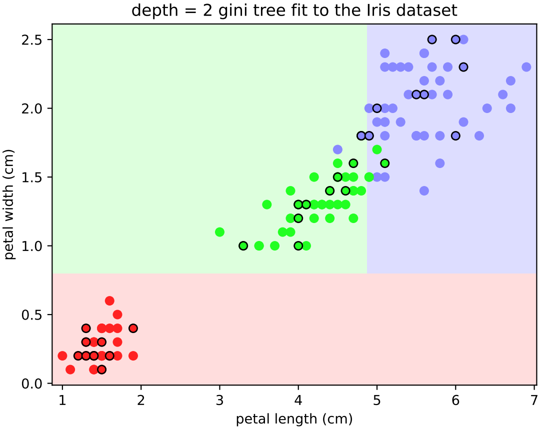
Classification tree examples: Iris Data

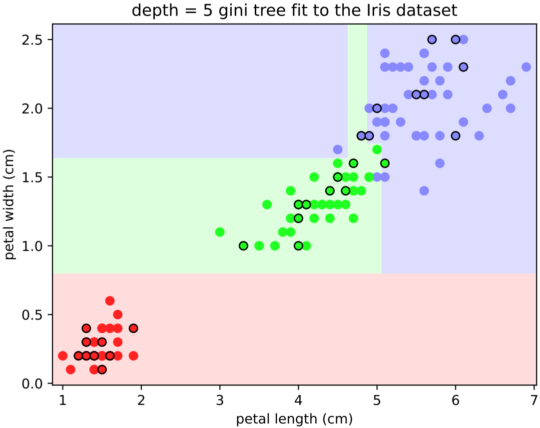




Note the rectangular regions (because each split is over one variable) and the greater complexity when the maximum depth of the tree increases.

Classification tree examples: Iris Data

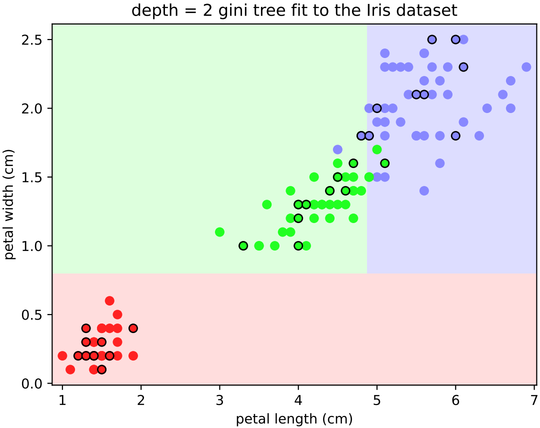


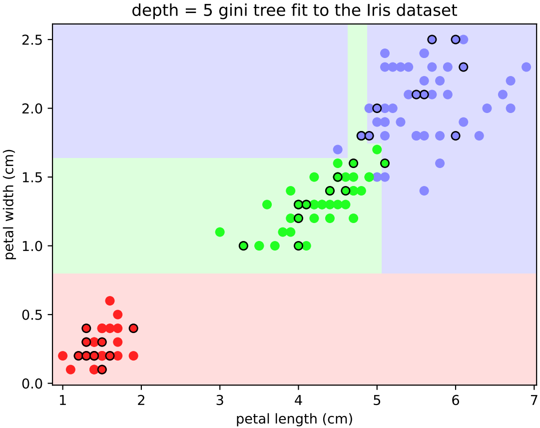


Note the rectangular regions (because each split is over one variable) and the greater complexity when the maximum depth of the tree increases.

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

Classification tree examples: Iris Data





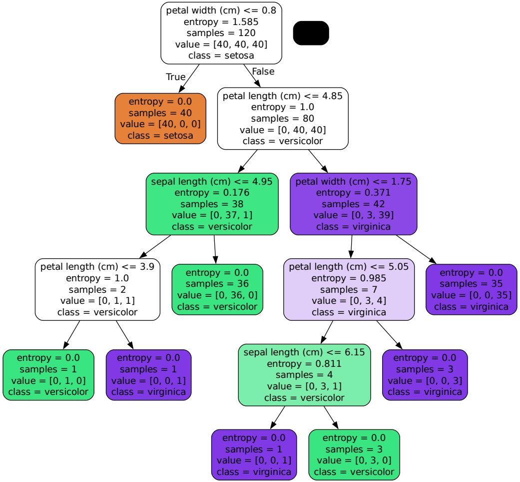
Note the rectangular regions (because each split is over one variable) and the greater complexity when the maximum depth of the tree increases.

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 ≤ 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

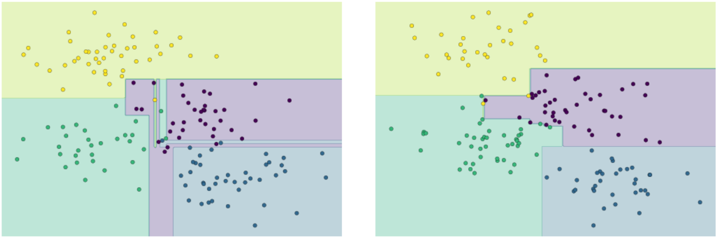
Now that we have a decision tree, how do we use it to predict the label?

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).

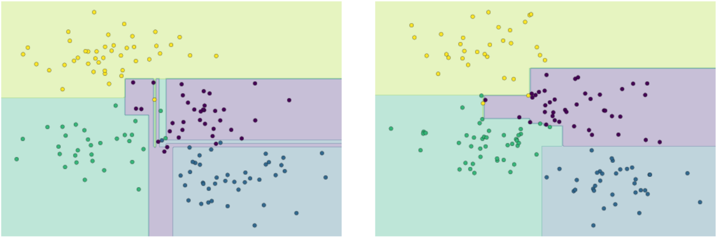
Python can extract paths from the root to each leaf as a set of if-then-else rules, to explain decisions.

Be careful of overfitting. . .



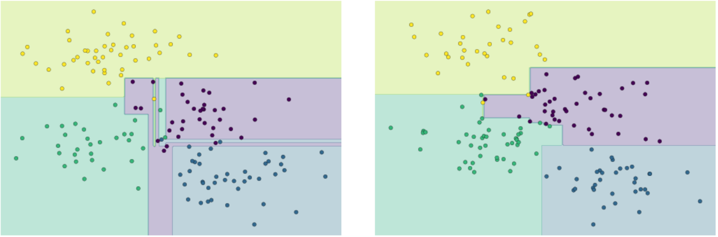
- Given two very similar (generated) data sets, all leaves in each fitted decision tree are *pure*.

Be careful of overfitting. . .



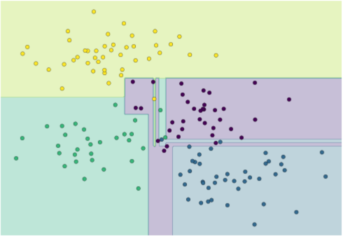
- Given two very similar (generated) data sets, all leaves in each fitted decision tree are *pure*.
- The resulting trees look very different.

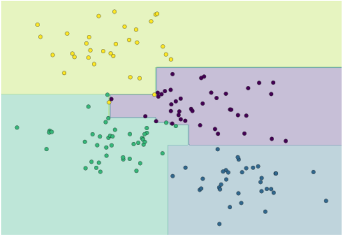
Be careful of overfitting...



- Given two very similar (generated) data sets, all leaves in each fitted decision tree are *pure*.
- The resulting trees look very different.
- This sensitivity to the noise in the data is characteristic of *overfitting* (high variance).

Be careful of overfitting. . .





- Given two very similar (generated) data sets, all leaves in each fitted decision tree are *pure*.
- The resulting trees look very different.
- 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

```
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.

Outline

Rev. Bayes and his theorem

Rev. Thomas Bayes, 1702–1761

Usage

Given $P(E|H)$ (Probability of Evidence (attributes) given the Hypothesis (the known classes) in the *training* set), Bayes theorem shows how to invert this relationship to get $P(H|E)$ (Probability of the Hypothesis (class) given the evidence (attributes) with an (unseen) *test case*).

Rev. Bayes and his theorem

Rev. Thomas Bayes, 1702–1761

Bayes' Theorem

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{1}$$

where $P(X|Y)$ is the conditional probability of X given that we know Y is true.

Usage

Given $P(E|H)$ (Probability of Evidence (attributes) given the Hypothesis (the known classes) in the *training* set), Bayes theorem shows how to invert this relationship to get $P(H|E)$ (Probability of the Hypothesis (class) given the evidence (attributes) with an (unseen) *test case*).

Rev. Bayes and his theorem

Rev. Thomas Bayes, 1702–1761

Usage

Given $P(E|H)$ (Probability of Evidence (attributes) given the Hypothesis (the known classes) in the *training* set), Bayes theorem shows how to invert this relationship to get $P(H|E)$ (Probability of the Hypothesis (class) given the evidence (attributes) with an (unseen) *test case*).

Bayes' Theorem

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{1}$$

where $P(X|Y)$ is the conditional probability of X given that we know Y is true.

Application to classification

By convention, $A = H$ and $B = E$, where H is the hypothesis and E is the evidence in support of that hypothesis.

With this interpretation, the Bayes identity can be used to predict class probabilities (hypothesis) from features (evidence).

Conditional probabilities and Bayes terminology

Definition 6 (Conditional Probability)

If A and B are events, the Probability of A, given that B is true (has happened), written $P(A|B)$ is defined as follows:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{2}$$

where $P(A \cap B)$ is the probability that both A and B are true.

Conditional probabilities and Bayes terminology

Definition 6 (Conditional Probability)

If A and B are events, the Probability of A, given that B is true (has happened), written $P(A|B)$ is defined as follows:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{2}$$

where $P(A \cap B)$ is the probability that both A and B are true.

Given Bayes Theorem 1, we have

Term	Description
$P(A)$	Class prior; Prior probability
$P(B)$	Predictor prior; evidence
$P(A\|B)$	Posterior probability; Updated Class Prior
$P(B\|A)$	Likelihood

Conditional probabilities and Bayes terminology

Definition 6 (Conditional Probability)

If A and B are events, the Probability of A, given that B is true (has happened), written $P(A|B)$ is defined as follows:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{2}$$

where $P(A \cap B)$ is the probability that both A and B are true.

Given Bayes Theorem 1, we have

Term	Description
$P(A)$	Class prior; Prior probability
$P(B)$	Predictor prior; evidence
$P(A\|B)$	Posterior probability; Updated Class Prior
$P(B\|A)$	Likelihood

- In courtrooms, B might represent the available evidence in favour of guilt, and A might represent the verdict “is guilty”.

Conditional probabilities and Bayes terminology

Definition 6 (Conditional Probability)

If A and B are events, the Probability of A, given that B is true (has happened), written $P(A|B)$ is defined as follows:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{2}$$

where $P(A \cap B)$ is the probability that both A and B are true.

Given Bayes Theorem 1, we have

Term	Description
$P(A)$	Class prior; Prior probability
$P(B)$	Predictor prior; evidence
$P(A\|B)$	Posterior probability; Updated Class Prior
$P(B\|A)$	Likelihood

- In courtrooms, B might represent the available evidence in favour of guilt, and A might represent the verdict “is guilty”.
- In data mining, B might represent the features (derived from the Data) for a given instance and A might represent the *predicted label* for these features.

Conditional probabilities and Bayes terminology

Definition 6 (Conditional Probability)

If A and B are events, the Probability of A, given that B is true (has happened), written $P(A|B)$ is defined as follows:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{2}$$

where $P(A \cap B)$ is the probability that both A and B are true.

Given Bayes Theorem 1, we have

Term	Description
$P(A)$	Class prior; Prior probability
$P(B)$	Predictor prior; evidence
$P(A\|B)$	Posterior probability; Updated Class Prior
$P(B\|A)$	Likelihood

- In courtrooms, B might represent the available evidence in favour of guilt, and A might represent the verdict "is guilty".
- In data mining, B might represent the features (derived from the Data) for a given instance and A might represent the *predicted label* for these features.
- If A and B are independent events, $P(A \cap B) \equiv P(A)P(B)$, so $P(A|B) = P(A)$ and $P(B|A) = P(B)$.

Extended and Bayes

In practice, there would be multiple features/evidence so $B = \{B_1, B_2, \ldots, B_n\}$

.

Definition 7 (Extended Bayes Theorem)

The extended form, when $\{B_j\}$ partition B, so $B = \cup_j B_j$ and $B_p \cap B_q \equiv \emptyset$ unless $p = q$, is

$$P(A|\{B_i\}) = \frac{P(\{B_i\}|A)P(A)}{P(\{B_i\})} \tag{3}$$

which is the component-wise version of the standard Bayes Theorem.

Side note: Prosecutor's Fallacy

Note that $P(A|B) \neq P(B|A)$ in general. If the ratio $\frac{P(A)}{P(B)}$ is not close to 1, lawyers can mislead jurors regarding guilt or innocence. *Probability of Guilt given the evidence is not the same as the probability of the evidence assuming the defendant is guilty.*

Naive Extended Bayes

Definition 8 (Naive Bayes)

If the features B are assumed to be independent of each other, it can be shown that

$$P(B) = P(B_1 \cap B_2 \cap \ldots B_n) = \prod_i P(B_i) \tag{4}$$

$$P(B|A_j) = \prod_k P(B_i|A_j) \tag{5}$$

The naïve form of Bayes theorem becomes

$$P(A_j|B) = \frac{\prod_i P(B_i|A_j)P(A_j)}{\prod_i P(B_i)} \tag{6}$$

Naive Bayes classifier

Definition 9 (Naive Bayes classifier)

- The Naive Bayes algorithm predicts a class, given a set of set of features, using probability principles.

Naive Bayes classifier

Definition 9 (Naive Bayes classifier)

- The Naive Bayes algorithm predicts a class, given a set of set of features, using probability principles.
- It is naive because it assumes the features are statistically independent (rarely true in practice).

Naive Bayes classifier

Definition 9 (Naive Bayes classifier)

- The Naive Bayes algorithm predicts a class, given a set of set of features, using probability principles.
- It is naive because it assumes the features are statistically independent (rarely true in practice).
- The training data is used to estimate the probabilities and likelihood.

Naive Bayes classifier

Definition 9 (Naive Bayes classifier)

- The Naive Bayes algorithm predicts a class, given a set of set of features, using probability principles.
- It is naive because it assumes the features are statistically independent (rarely true in practice).
- The training data is used to estimate the probabilities and likelihood.
- Bayes theorem is used to predict the observation's membership of each class (associated with a label).

Naive Bayes classifier

Definition 9 (Naive Bayes classifier)

- The Naive Bayes algorithm predicts a class, given a set of set of features, using probability principles.
- It is naive because it assumes the features are statistically independent (rarely true in practice).
- The training data is used to estimate the probabilities and likelihood.
- Bayes theorem is used to predict the observation's membership of each class (associated with a label).
- It is then assigned to the class for which its conditional probability is greatest.

Naive Bayes Interpretation

- Both the simple and extended Bayes Theorem provide a formula to flip between "B given A" (data given labels) to "A given B" (labels given data).

Naive Bayes Interpretation

- Both the simple and extended Bayes Theorem provide a formula to flip between “B given A” (data given labels) to “A given B” (labels given data).
- From the Training set, we can calculate

Naive Bayes Interpretation

- Both the simple and extended Bayes Theorem provide a formula to flip between “B given A” (data given labels) to “A given B” (labels given data).
- From the Training set, we can calculate
    - $P(A_j)$ (prior probability of each class label),

Naive Bayes Interpretation

- Both the simple and extended Bayes Theorem provide a formula to flip between "B given A" (data given labels) to "A given B" (labels given data).
- From the Training set, we can calculate
  - $P(A_j)$ (prior probability of each class label),
  - $P(B_i)$ (the predictor prior) and

Naive Bayes Interpretation

- Both the simple and extended Bayes Theorem provide a formula to flip between “B given A” (data given labels) to “A given B” (labels given data).
- From the Training set, we can calculate
  - $P(A_j)$ (prior probability of each class label),
  - $P(B_i)$ (the predictor prior) and
  - $P(B_i|A_j)$ (the evidence that the feature valued B_i predicts the class label A_j).

Naive Bayes Interpretation

- Both the simple and extended Bayes Theorem provide a formula to flip between "B given A" (data given labels) to "A given B" (labels given data).
- From the Training set, we can calculate
  - $P(A_j)$ (prior probability of each class label),
  - $P(B_i)$ (the predictor prior) and
  - $P(B_i|A_j)$ (the evidence that the feature valued B_i predicts the class label A_j).
- From this, we can use the naive version of the extended Bayes Theorem 6 to predict $P(A_j|B)$, the posterior probability of class label A_j given all the evidence from the features B.

Overview of Naïve Bayes algorithm

- In the *Training Phase*, the method precomputes various probabilities $P(A_j)$ and conditional probabilities $P(B_k|A_j)$.

Overview of Naïve Bayes algorithm

- In the *Training Phase*, the method precomputes various probabilities $P(A_j)$ and conditional probabilities $P(B_k|A_j)$.
- In the *Prediction Phase*

Overview of Naïve Bayes algorithm

- In the *Training Phase*, the method precomputes various probabilities $P(A_j)$ and conditional probabilities $P(B_k|A_j)$.
- In the *Prediction Phase*
  - the extended form of Bayes' Theorem 6 is used to compute the posterior class probabilities $P(A_j|B)$ for the given observation.

Overview of Naïve Bayes algorithm

- In the *Training Phase*, the method precomputes various probabilities $P(A_j)$ and conditional probabilities $P(B_k|A_j)$.
- In the *Prediction Phase*
  - the extended form of Bayes' Theorem 6 is used to compute the posterior class probabilities $P(A_j|B)$ for the given observation.
  - according to the *Maximum A Posteriori* (MAP) criterion, the observation is assigned to the class with the highest probability from $P(A_j|B)$.

Overview of Naïve Bayes algorithm

- In the *Training Phase*, the method precomputes various probabilities $P(A_j)$ and conditional probabilities $P(B_k|A_j)$.
- In the *Prediction Phase*
  - the extended form of Bayes' Theorem 6 is used to compute the posterior class probabilities $P(A_j|B)$ for the given observation.
  - according to the *Maximum A Posteriori* (MAP) criterion, the observation is assigned to the class with the highest probability from $P(A_j|B)$.
- One of the features of Naïve Bayes (with $P(A|B)$), like decision trees (with $P(A \cap B)$), is the direct role played by probability

Overview of Naïve Bayes algorithm

- In the *Training Phase*, the method precomputes various probabilities $P(A_j)$ and conditional probabilities $P(B_k|A_j)$.
- In the *Prediction Phase*
  - the extended form of Bayes' Theorem 6 is used to compute the posterior class probabilities $P(A_j|B)$ for the given observation.
  - according to the *Maximum A Posteriori* (MAP) criterion, the observation is assigned to the class with the highest probability from $P(A_j|B)$.
- One of the features of Naïve Bayes (with $P(A|B)$), like decision trees (with $P(A \cap B)$), is the direct role played by probability
- When training Naïve Bayes, it is convenient to compute a table of *marginal counts*, as seen in the next slide, and to use these for prediction.

Fruit classification example

Example: Fruit classification

Type	Long	¬Long	Sweet	¬Sweet	Yellow	¬Yellow	*Total*
Banana	400	100	350	150	450	50	*500*
Orange	0	300	150	150	300	0	*300*
Other	100	100	150	50	50	150	*200*
Total	*500*	*500*	*650*	*350*	*800*	*200*	*1000*

Source: stackoverflow

Fruit classification example

Example: Fruit classification

Type	Long	¬Long	Sweet	¬Sweet	Yellow	¬Yellow	*Total*
Banana	400	100	350	150	450	50	*500*
Orange	0	300	150	150	300	0	*300*
Other	100	100	150	50	50	150	*200*
Total	*500*	*500*	*650*	*350*	*800*	*200*	*1000*

Source: stackoverflow

Fruit classification : Precalculations

Fruit classification example

Example: Fruit classification

Type	Long	¬Long	Sweet	¬Sweet	Yellow	¬Yellow	*Total*
Banana	400	100	350	150	450	50	*500*
Orange	0	300	150	150	300	0	*300*
Other	100	100	150	50	50	150	*200*
Total	*500*	*500*	*650*	*350*	*800*	*200*	*1000*

Source: stackoverflow

Fruit classification : Precalculations

P(<Fruit>) = Total_<Fruit> / Total_*
P(<Feature>) = Total_<Feature> / Total_*
P(<Feature> | <Fruit>) = <Fruit,Feature> / Total_<Fruit>

$\rightarrow$ P(Other) = 200/1000 = 0.2
$\rightarrow$ P(Sweet) = 650/1000 = 0.65
$\rightarrow$ P(Sweet | Other) = 150/200 = 0.75

Fruit classification example: prediction

Given observation: Long=L, Sweet=S, Yellow=Y fruit, which is it?

Fruit classification example: prediction

Given observation: Long=L, Sweet=S, Yellow=Y fruit, which is it?

Banana - B

$P(\mathrm{B}|L,S,Y) =$

$$\frac{P(\mathrm{L}|\mathrm{B})P(\mathrm{S}|\mathrm{B})P(\mathrm{Y}|\mathrm{B})P(\mathrm{B})}{P(\mathrm{L})P(\mathrm{S})P(\mathrm{Y})}$$

$$= \frac{0.8 \times 0.7 \times 0.9 \times 0.5}{0.5 \times 0.65 \times 0.8}$$

$= 0.97$

Fruit classification example: prediction

Given observation: Long=L, Sweet=S, Yellow=Y fruit, which is it?

Banana - B

$P(\mathrm{B}|L, S, Y) =$

$$\frac{P(\mathrm{L}|\mathrm{B})P(\mathrm{S}|\mathrm{B})P(\mathrm{Y}|\mathrm{B})P(\mathrm{B})}{P(\mathrm{L})P(\mathrm{S})P(\mathrm{Y})}$$

$$= \frac{0.8 \times 0.7 \times 0.9 \times 0.5}{0.5 \times 0.65 \times 0.8}$$

$= 0.97$

Orange - O

$P(\mathrm{O}|L, S, Y) =$

$$\frac{P(\mathrm{L}|\mathrm{O})P(\mathrm{S}|\mathrm{O})P(\mathrm{Y}|\mathrm{O})P(\mathrm{O})}{P(\mathrm{L})P(\mathrm{S})P(\mathrm{Y})}$$

$$= \frac{0.0 \times 0.5 \times 1.0 \times 0.3}{0.5 \times 0.65 \times 0.8}$$

$= 0$

Fruit classification example: prediction

Given observation: Long=L, Sweet=S, Yellow=Y fruit, which is it?

Banana - B

$P(\mathrm{B}|L,S,Y) =$

$$\frac{P(\mathrm{L}|\mathrm{B})P(\mathrm{S}|\mathrm{B})P(\mathrm{Y}|\mathrm{B})P(\mathrm{B})}{P(\mathrm{L})P(\mathrm{S})P(\mathrm{Y})}$$

$$= \frac{0.8 \times 0.7 \times 0.9 \times 0.5}{0.5 \times 0.65 \times 0.8}$$

$= 0.97$

Orange - O

$P(\mathrm{O}|L,S,Y) =$

$$\frac{P(\mathrm{L}|\mathrm{O})P(\mathrm{S}|\mathrm{O})P(\mathrm{Y}|\mathrm{O})P(\mathrm{O})}{P(\mathrm{L})P(\mathrm{S})P(\mathrm{Y})}$$

$$= \frac{0.0 \times 0.5 \times 1.0 \times 0.3}{0.5 \times 0.65 \times 0.8}$$

$= 0$

Other = R

$P(\mathrm{R}|L,S,Y) =$

$$\frac{P(\mathrm{L}|\mathrm{R})P(\mathrm{S}|\mathrm{R})P(\mathrm{Y}|\mathrm{R})P(\mathrm{R})}{P(\mathrm{L})P(\mathrm{S})P(\mathrm{Y})}$$

$$= \frac{0.5 \times 0.75 \times 0.25 \times 0.2}{0.5 \times 0.65 \times 0.8}$$

$= 0.07$

Fruit classification example: prediction

Given observation: Long=L, Sweet=S, Yellow=Y fruit, which is it?

Banana - B

$P(\mathrm{B}|L, S, Y) =$

$$\frac{P(\mathrm{L}|\mathrm{B})P(\mathrm{S}|\mathrm{B})P(\mathrm{Y}|\mathrm{B})P(\mathrm{B})}{P(\mathrm{L})P(\mathrm{S})P(\mathrm{Y})}$$

$$= \frac{0.8 \times 0.7 \times 0.9 \times 0.5}{0.5 \times 0.65 \times 0.8}$$

$= 0.97$

Orange - O

$P(\mathrm{O}|L, S, Y) =$

$$\frac{P(\mathrm{L}|\mathrm{O})P(\mathrm{S}|\mathrm{O})P(\mathrm{Y}|\mathrm{O})P(\mathrm{O})}{P(\mathrm{L})P(\mathrm{S})P(\mathrm{Y})}$$

$$= \frac{0.0 \times 0.5 \times 1.0 \times 0.3}{0.5 \times 0.65 \times 0.8}$$

$= 0$

Other = R

$P(\mathrm{R}|L, S, Y) =$

$$\frac{P(\mathrm{L}|\mathrm{R})P(\mathrm{S}|\mathrm{R})P(\mathrm{Y}|\mathrm{R})P(\mathrm{R})}{P(\mathrm{L})P(\mathrm{S})P(\mathrm{Y})}$$

$$= \frac{0.5 \times 0.75 \times 0.25 \times 0.2}{0.5 \times 0.65 \times 0.8}$$

$= 0.07$

According to the MAP criterion, the observation (mystery fruit) is a banana!

Fruit classification example: prediction

Given observation: Long=L, Sweet=S, Yellow=Y fruit, which is it?

Banana - B

$P(\mathrm{B}|L,S,Y) =$

$$\frac{P(\mathrm{L}|\mathrm{B})P(\mathrm{S}|\mathrm{B})P(\mathrm{Y}|\mathrm{B})P(\mathrm{B})}{P(\mathrm{L})P(\mathrm{S})P(\mathrm{Y})}$$

$$= \frac{0.8 \times 0.7 \times 0.9 \times 0.5}{0.5 \times 0.65 \times 0.8}$$

$= 0.97$

Orange - O

$P(\mathrm{O}|L,S,Y) =$

$$\frac{P(\mathrm{L}|\mathrm{O})P(\mathrm{S}|\mathrm{O})P(\mathrm{Y}|\mathrm{O})P(\mathrm{O})}{P(\mathrm{L})P(\mathrm{S})P(\mathrm{Y})}$$

$$= \frac{0.0 \times 0.5 \times 1.0 \times 0.3}{0.5 \times 0.65 \times 0.8}$$

$= 0$

Other = R

$P(\mathrm{R}|L,S,Y) =$

$$\frac{P(\mathrm{L}|\mathrm{R})P(\mathrm{S}|\mathrm{R})P(\mathrm{Y}|\mathrm{R})P(\mathrm{R})}{P(\mathrm{L})P(\mathrm{S})P(\mathrm{Y})}$$

$$= \frac{0.5 \times 0.75 \times 0.25 \times 0.2}{0.5 \times 0.65 \times 0.8}$$

$= 0.07$

According to the MAP criterion, the observation (mystery fruit) is a banana!

Given the 3 binary-valued attributes, there are $2^3 = 8$ possible combinations - Naïve Bayes will classify each of these 8 combinations as one of the 3 fruit classes.

Naïve Bayes Classifier summary

- Advantages

Naïve Bayes Classifier summary

- Advantages
  - Conceptually simple and easy to implement

Naïve Bayes Classifier summary

- Advantages
  - Conceptually simple and easy to implement
  - Fast and scalable

Naïve Bayes Classifier summary

- Advantages
  - Conceptually simple and easy to implement
  - Fast and scalable
  - Variants exist for numeric (Gaussian-), binary (Bernoulli-) and multi-class (mulinomial-) featured Naïve Bayes

Naïve Bayes Classifier summary

- Advantages
  - Conceptually simple and easy to implement
  - Fast and scalable
  - Variants exist for numeric (Gaussian-), binary (Bernoulli-) and multi-class (mulinomial-) featured Naïve Bayes
  - Particularly good for text classification and email spam detection

Naïve Bayes Classifier summary

- Advantages
  - Conceptually simple and easy to implement
  - Fast and scalable
  - Variants exist for numeric (Gaussian-), binary (Bernoulli-) and multi-class (mulinomial-) featured Naïve Bayes
  - Particularly good for text classification and email spam detection
- Disadvantages

Naïve Bayes Classifier summary

- Advantages
  - Conceptually simple and easy to implement
  - Fast and scalable
  - Variants exist for numeric (Gaussian-), binary (Bernoulli-) and multi-class (mulinomial-) featured Naïve Bayes
  - Particularly good for text classification and email spam detection
- Disadvantages
  - Ignores feature relationships, so #(feature) + #($\neg$feature) is the same for all features

Naïve Bayes Classifier summary

- Advantages
  - Conceptually simple and easy to implement
  - Fast and scalable
  - Variants exist for numeric (Gaussian-), binary (Bernoulli-) and multi-class (mulinomial-) featured Naïve Bayes
  - Particularly good for text classification and email spam detection
- Disadvantages
  - Ignores feature relationships, so #(feature) + #($\neg$feature) is the same for all features
  - Be careful of zero-valued conditional probabilities and numerical underflow

Naïve Bayes Classifier summary

- Advantages
  - Conceptually simple and easy to implement
  - Fast and scalable
  - Variants exist for numeric (Gaussian-), binary (Bernoulli-) and multi-class (mulinomial-) featured Naïve Bayes
  - Particularly good for text classification and email spam detection
- Disadvantages
  - Ignores feature relationships, so #(feature) + #($\neg$feature) is the same for all features
  - Be careful of zero-valued conditional probabilities and numerical underflow
  - Prone to underfitting (high bias)

Naïve Bayes Classifier summary

- Advantages
  - Conceptually simple and easy to implement
  - Fast and scalable
  - Variants exist for numeric (Gaussian-), binary (Bernoulli-) and multi-class (mulinomial-) featured Naïve Bayes
  - Particularly good for text classification and email spam detection
- Disadvantages
  - Ignores feature relationships, so #(feature) + #($\neg$feature) is the same for all features
  - Be careful of zero-valued conditional probabilities and numerical underflow
  - Prone to underfitting (high bias)
- For prediction $F(X) \rightarrow Y$

Naïve Bayes Classifier summary

- Advantages
  - Conceptually simple and easy to implement
  - Fast and scalable
  - Variants exist for numeric (Gaussian-), binary (Bernoulli-) and multi-class (mulinomial-) featured Naïve Bayes
  - Particularly good for text classification and email spam detection
- Disadvantages
  - Ignores feature relationships, so #(feature) + #($\neg$feature) is the same for all features
  - Be careful of zero-valued conditional probabilities and numerical underflow
  - Prone to underfitting (high bias)
- For prediction $F(X) \to Y$
  - Most classifiers are discriminative—they learn $P(Y|X)$: “Given data X, what class Y is it?”

Naïve Bayes Classifier summary

- Advantages
  - Conceptually simple and easy to implement
  - Fast and scalable
  - Variants exist for numeric (Gaussian-), binary (Bernoulli-) and multi-class (mulinomial-) featured Naïve Bayes
  - Particularly good for text classification and email spam detection
- Disadvantages
  - Ignores feature relationships, so #(feature) + #(¬feature) is the same for all features
  - Be careful of zero-valued conditional probabilities and numerical underflow
  - Prone to underfitting (high bias)
- For prediction $F(X) \to Y$
  - Most classifiers are discriminative—they learn $P(Y|X)$: "Given data X, what class Y is it?"
  - Naïve Bayes is generative—it learns $P(X|Y)$ and $P(Y)$: "Given classes Y, what data X fits each class, and how often do those classes occur?"

Naïve Bayes Classifier summary

- Advantages
  - Conceptually simple and easy to implement
  - Fast and scalable
  - Variants exist for numeric (Gaussian-), binary (Bernoulli-) and multi-class (mulinomial-) featured Naïve Bayes
  - Particularly good for text classification and email spam detection
- Disadvantages
  - Ignores feature relationships, so #(feature) + #(¬feature) is the same for all features
  - Be careful of zero-valued conditional probabilities and numerical underflow
  - Prone to underfitting (high bias)
- For prediction $F(X) \rightarrow Y$
  - Most classifiers are discriminative—they learn $P(Y|X)$: "Given data X, what class Y is it?"
  - Naïve Bayes is generative—it learns $P(X|Y)$ and $P(Y)$: "Given classes Y, what data X fits each class, and how often do those classes occur?"
- Implementations exist in `sklearn`: `from sklearn.naive_bayes import GaussianNB`, etc.

Outline

Ordinal regression vs Ordinal Classification

- Should ordinal targets be predicted using regression?

Ordinal regression vs Ordinal Classification

- Should ordinal targets be predicted using regression?
  - Yes, because like numbers, they have a natural order...

Ordinal regression vs Ordinal Classification

- Should ordinal targets be predicted using regression?
  - Yes, because like numbers, they have a natural order…
  - No, because differences don't work the same way…

Ordinal regression vs Ordinal Classification

- Should ordinal targets be predicted using regression?
    - Yes, because like numbers, they have a natural order…
    - No, because differences don't work the same way…
- Should ordinal targets be predicted using classification?

Ordinal regression vs Ordinal Classification

- Should ordinal targets be predicted using regression?
  - Yes, because like numbers, they have a natural order…
  - No, because differences don't work the same way…
- Should ordinal targets be predicted using classification?
  - Yes, because the targets are categories, not numbers…

Ordinal regression vs Ordinal Classification

- Should ordinal targets be predicted using regression?
  - Yes, because like numbers, they have a natural order…
  - No, because differences don't work the same way…
- Should ordinal targets be predicted using classification?
  - Yes, because the targets are categories, not numbers…
  - No, because the difference between two categories depends on their order, and classification ignores this

Ordinal regression vs Ordinal Classification

- Should ordinal targets be predicted using regression?
  - Yes, because like numbers, they have a natural order…
  - No, because differences don't work the same way…
- Should ordinal targets be predicted using classification?
  - Yes, because the targets are categories, not numbers…
  - No, because the difference between two categories depends on their order, and classification ignores this
- scikit-learn does not offer Ordinal Regression/Ordinal Classification directly

Ordinal regression vs Ordinal Classification

- Should ordinal targets be predicted using regression?
  - Yes, because like numbers, they have a natural order…
  - No, because differences don't work the same way…
- Should ordinal targets be predicted using classification?
  - Yes, because the targets are categories, not numbers…
  - No, because the difference between two categories depends on their order, and classification ignores this
- scikit-learn does not offer Ordinal Regression/Ordinal Classification directly
- But there are proposals to wrap existing classifiers and to solve an extended problem that predicts the target while considering ordinal target values.

Ordinal regression vs Ordinal Classification

- Should ordinal targets be predicted using regression?
  - Yes, because like numbers, they have a natural order...
  - No, because differences don't work the same way...
- Should ordinal targets be predicted using classification?
  - Yes, because the targets are categories, not numbers...
  - No, because the difference between two categories depends on their order, and classification ignores this
- scikit-learn does not offer Ordinal Regression/Ordinal Classification directly
- But there are proposals to wrap existing classifiers and to solve an extended problem that predicts the target while considering ordinal target values.

In the meantime, either Regression or Classification is used, with caveats...

Outline

Summary

- Classification is one of the main tasks in data mining, and is a mature and well-studied field

Summary

- Classification is one of the main tasks in data mining, and is a mature and well-studied field
    - k-nearest-neighbours is conceptually simple (based on voting) and the lack of a model (lazy learning) means it responds better to data drift

Summary

- Classification is one of the main tasks in data mining, and is a mature and well-studied field
    - k-nearest-neighbours is conceptually simple (based on voting) and the lack of a model (lazy learning) means it responds better to data drift
    - Logistic regression is an extension of linear regression and benefits from its strengths

Summary

- Classification is one of the main tasks in data mining, and is a mature and well-studied field
  - k-nearest-neighbours is conceptually simple (based on voting) and the lack of a model (lazy learning) means it responds better to data drift
  - Logistic regression is an extension of linear regression and benefits from its strengths
  - Decision trees learn a representation that is often easily interpretable, but works better with linear boundaries

Summary

- Classification is one of the main tasks in data mining, and is a mature and well-studied field
  - k-nearest-neighbours is conceptually simple (based on voting) and the lack of a model (lazy learning) means it responds better to data drift
  - Logistic regression is an extension of linear regression and benefits from its strengths
  - Decision trees learn a representation that is often easily interpretable, but works better with linear boundaries
  - Naïve Bayes offers a probability-based generative model, able to work from data summaries, ideal for text and email classification

Summary

- Classification is one of the main tasks in data mining, and is a mature and well-studied field
  - k-nearest-neighbours is conceptually simple (based on voting) and the lack of a model (lazy learning) means it responds better to data drift
  - Logistic regression is an extension of linear regression and benefits from its strengths
  - Decision trees learn a representation that is often easily interpretable, but works better with linear boundaries
  - Naïve Bayes offers a probability-based generative model, able to work from data summaries, ideal for text and email classification
- More advanced classifiers have their own advantages, especially in relation to high dimensional data:

Summary

- Classification is one of the main tasks in data mining, and is a mature and well-studied field
    - k-nearest-neighbours is conceptually simple (based on voting) and the lack of a model (lazy learning) means it responds better to data drift
    - Logistic regression is an extension of linear regression and benefits from its strengths
    - Decision trees learn a representation that is often easily interpretable, but works better with linear boundaries
    - Naïve Bayes offers a probability-based generative model, able to work from data summaries, ideal for text and email classification
- More advanced classifiers have their own advantages, especially in relation to high dimensional data:
    - Ensemble methods (such as RandomForest and AdaBoost) were state of the art (2000-2012, say) and sacrifice interpretability for good performance with high dimensional data

Summary

- Classification is one of the main tasks in data mining, and is a mature and well-studied field
  - k-nearest-neighbours is conceptually simple (based on voting) and the lack of a model (lazy learning) means it responds better to data drift
  - Logistic regression is an extension of linear regression and benefits from its strengths
  - Decision trees learn a representation that is often easily interpretable, but works better with linear boundaries
  - Naïve Bayes offers a probability-based generative model, able to work from data summaries, ideal for text and email classification
- More advanced classifiers have their own advantages, especially in relation to high dimensional data:
  - Ensemble methods (such as RandomForest and AdaBoost) were state of the art (2000-2012, say) and sacrifice interpretability for good performance with high dimensional data
  - SVM was state of the art (1985-2000, say) and is still extremely effective for very high dimensional problems like document classification

Other considerations

- KNN uses *lazy* learning, all other techniques above use *eager* learning (derive model from training data)

Other considerations

- KNN uses *lazy* learning, all other techniques above use *eager* learning (derive model from training data)
- Naive Bayes uses *generative* learning to learn how the data was generated, all other techniques above use *discriminative* learning to derive the function that assigns class labels

Other considerations

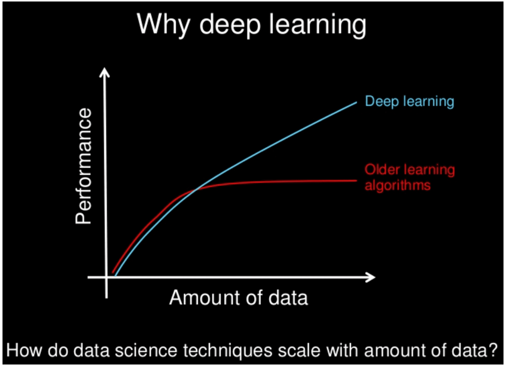
- KNN uses *lazy* learning, all other techniques above use *eager* learning (derive model from training data)
- Naive Bayes uses *generative* learning to learn how the data was generated, all other techniques above use *discriminative* learning to derive the function that assigns class labels
- For KNN and Decision Trees, the representation grows with the size of the data - that is not generally true in all other techniques above

Other considerations

- KNN uses *lazy* learning, all other techniques above use *eager* learning (derive model from training data)
- Naive Bayes uses *generative* learning to learn how the data was generated, all other techniques above use *discriminative* learning to derive the function that assigns class labels
- For KNN and Decision Trees, the representation grows with the size of the data - that is not generally true in all other techniques above

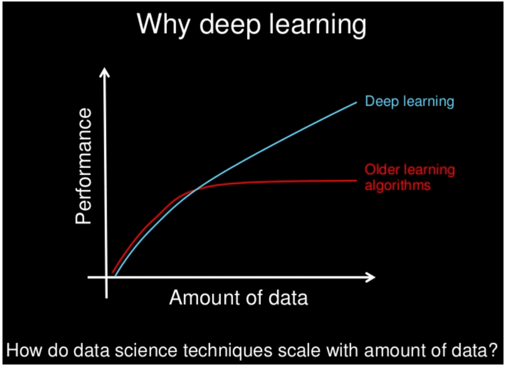
Classification is sometimes confused with clustering - will cover this clustering next lecture.

But is that the last word on Classification?



Source: Andrew Ng, *Why Deep Learning*

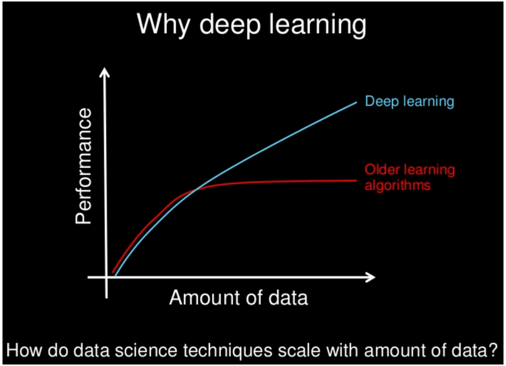
But is that the last word on Classification?



Source: Andrew Ng, *Why Deep Learning*

Learning from big data

But is that the last word on Classification?

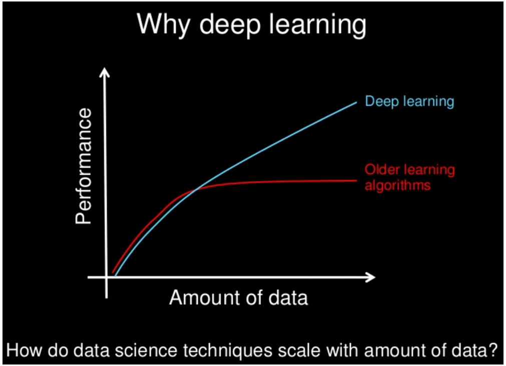


Source: Andrew Ng, *Why Deep Learning*

Learning from big data

- Traditional classification algorithms eventually run out of steam as data size increases

But is that the last word on Classification?

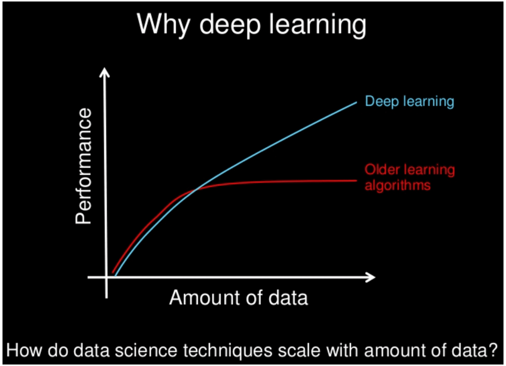


Source: Andrew Ng, *Why Deep Learning*

Learning from big data

- Traditional classification algorithms eventually run out of steam as data size increases
- Shallow neural networks had been discounted in the 1980s and 1990s when trained with small data

But is that the last word on Classification?

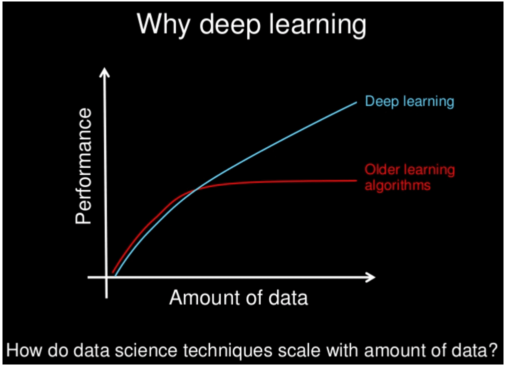


Source: Andrew Ng, *Why Deep Learning*

Learning from big data

- Traditional classification algorithms eventually run out of steam as data size increases
- Shallow neural networks had been discounted in the 1980s and 1990s when trained with small data
- Deep learning to the rescue!

But is that the last word on Classification?

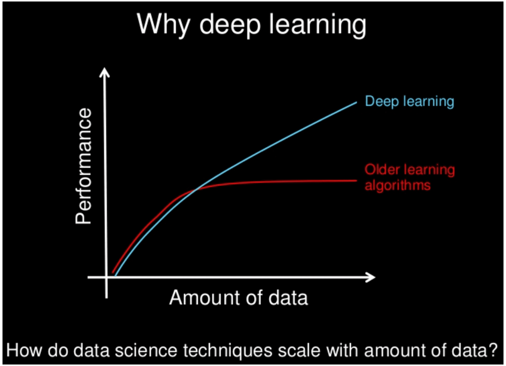


Source: Andrew Ng, *Why Deep Learning*

Learning from big data

- Traditional classification algorithms eventually run out of steam as data size increases
- Shallow neural networks had been discounted in the 1980s and 1990s when trained with small data
- Deep learning to the rescue!
- Kernel SVM and logistic regression lead nicely to perceptron models, hence ANNs, hence deep learning

But is that the last word on Classification?

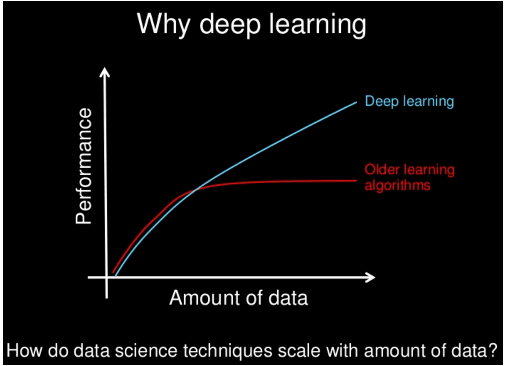


Source: Andrew Ng, *Why Deep Learning*

Learning from big data

- Traditional classification algorithms eventually run out of steam as data size increases
- Shallow neural networks had been discounted in the 1980s and 1990s when trained with small data
- Deep learning to the rescue!
- Kernel SVM and logistic regression lead nicely to perceptron models, hence ANNs, hence deep learning
- Deep learning requires lots of data but the models can scale better to take account of extra data

But is that the last word on Classification?



Source: Andrew Ng, *Why Deep Learning*

Learning from big data

- Traditional classification algorithms eventually run out of steam as data size increases
- Shallow neural networks had been discounted in the 1980s and 1990s when trained with small data
- Deep learning to the rescue!
- Kernel SVM and logistic regression lead nicely to perceptron models, hence ANNs, hence deep learning
- Deep learning requires lots of data but the models can scale better to take account of extra data

Deep Learning will probably be covered in semester 2…

General References