Decision trees and random forests

Dr Gianluca Campanella

7th June 2016

Contents

Decision trees

Exercise: should we wait?

Problem

You're out with friends and need to decide whether to wait for a table at a busy restaurant.

You have the following information:

- Whether there's an alternative restaurant nearby
- Whether the restaurant has a bar
- How busy the restaurant is (empty, some people, packed)
- Whether you're hungry (not at all, peckish, starving)
- Whether it's raining
- Type of restaurant (British, Chinese, Italian or Thai)
- Whether it's Friday or Saturday night

Exercise: should we wait?

Idea

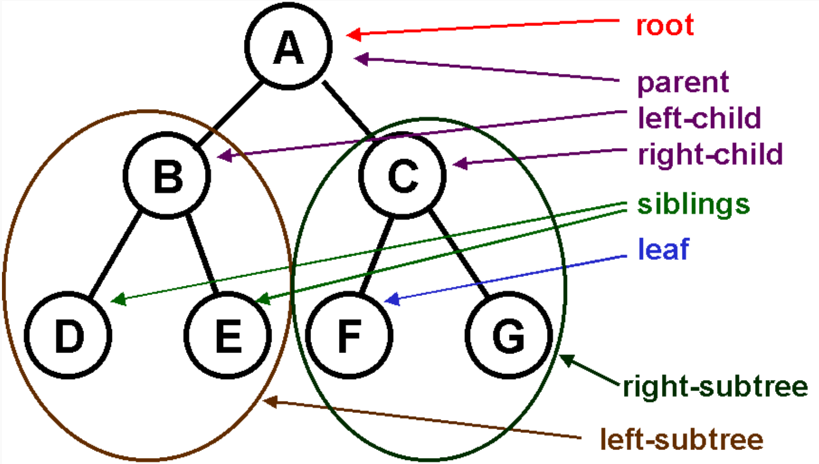
Imagine taking a sequence of decisions:

- If the restaurant is packed…
    - …and we're starving…
        - …but there's no alternative in the area, then wait

Question

How 'tall' should the decision tree grow?

Terminology



Expressiveness

- Decision trees can express **any** function of the predictors (using one leaf per sample)
- We want to find **structure** in the data, not overfit

$\rightarrow$ **Compact** trees

Expressiveness

- Decision trees can express **any** function of the predictors (using one leaf per sample)
- We want to find **structure** in the data, not overfit

$\rightarrow$ **Compact** trees

Idea

- Choose 'most significant' attribute as (sub)root

$\rightarrow$ Ideally achieving perfect separation of categories

- Repeat (recursively)

Comparison with logistic regression

- Logistic regression is a linear model
- Each predictor ‘acts’ independently of all others (its marginal effect is the regression coefficient)

Comparison with logistic regression

- Logistic regression is a linear model
- Each predictor ‘acts’ independently of all others (its marginal effect is the regression coefficient)

This doesn’t always work:

- If the restaurant is packed...
  - ...and we’re starving...
    - ...but **there is** an alternative, do we still wait?

Decision trees automatically contain **interactions**, since each question depends on the previous one

Training

- Start with the entire dataset
- Find the question that best segregates the samples based on the outcome $\rightarrow$ purity
- Repeat (recursively) until:
    - You have asked as many questions as you wanted
    - The gain in purity of possible splits is negligible
    - Leaves are completely pure

Prediction

- Answer each question
- Once you reach a leaf, take the majority label of the (training) samples in that leaf

Purity metrics

Gini impurity

- Measures how often a randomly chosen sample would be incorrectly labelled if it was labelled at random given class proportions p_i
- $\sum_k p_k (1 - p_k)$

Information gain

- Reduction in (Shannon) entropy $-p \log(p)$
- Difference between entropy of parent and (weighted) entropy of children

Overfitting

- Decision trees can easily ‘memorise’ the data

$\rightarrow$ Overfitting

Overfitting

- Decision trees can easily ‘memorise’ the data

$\rightarrow$ Overfitting

Solutions

Impose a limit on the...

- Maximum number of questions (depth)
- Minimum number of samples in each leaf

Pros and cons

Pros

- Can be used for regression or classification
- Can be visualised $\rightarrow$ easy to interpret
- Correspond to a series of ‘rules’
- Learn interactions and irrelevant predictors
- Don’t need scaled predictors

Cons

- Prone to overfitting and sensitive to small variations
- May not be globally optimal because of ‘greedy’ recursive binary splitting
- Don’t work well with unbalanced classes or small datasets

Random forests

Bagging

Imagine a situation where…

- You have many different models
- Each predicts your outcome with some accuracy
- Each also makes (independent) errors

How could you improve your prediction?

Bagging

Imagine a situation where...

- You have many different models
- Each predicts your outcome with some accuracy
- Each also makes (independent) errors

How could you improve your prediction?

Idea
Let all classifiers predict and take the majority vote
(or the mean for continuous outcomes)

Random forests

- A collection (**ensemble**) of decision trees
- Built randomly...
  - On a subset of the data
  - Using a subset of predictors

  ...to avoid overfitting
- For prediction, each tree contributes an answer, and the final model prediction is the majority vote (or the mean for continuous outcomes)

Boosting

Imagine a situation where...

- You are training many models of the same type **sequentially**
- Each predicts your outcome...
    - Correctly for some samples
    - Incorrectly for some other samples

How could you improve your prediction?

Boosting

Imagine a situation where...

- You are training many models of the same type **sequentially**
- Each predicts your outcome...
  - Correctly for some samples
  - Incorrectly for some other samples

How could you improve your prediction?

Idea

- At each step, 'refine' the model by giving more weight to incorrectly predicted samples (the 'hard' ones)
- For prediction, compute a weighted vote/mean of **all** predictions