THE UNIVERSITY *of* EDINBURGH | usher institute

Decision tree learning

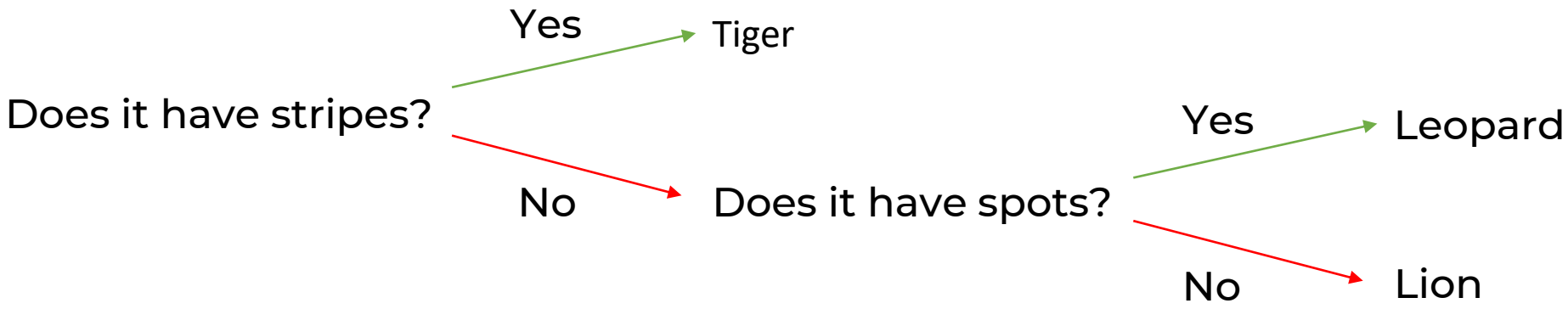
Steven Kerr

Supported by

THE UNIVERSITY *of* EDINBURGH | DDI Data-Driven Innovation

Decision trees

- A decision tree is a flow-chart that has a tree-like structure.
- Decision trees can be used for prediction tasks.
- Example:



Decision trees

- There are many machine learning algorithms for training decision trees.
- A typical algorithm works by iteratively choosing which feature to query next in order to maximise some metric.
- A metric that is commonly used is the *information gain*. It is a formal mathematical quantity that captures our intuition that some features are more informative for prediction than others.
- For example, when classifying animals, the question “Does it have stripes” is more informative than “Can it learn quantum mechanics?”

Decision trees

- At each stage, the algorithm chooses which of the remaining features is the most informative.
- This is called a *greedy* algorithm – it makes an optimal choice at each stage.
- This results in an ordered list of queries.
- The prediction at a node of the decision tree can be made in many ways. Common methods are:
  - Probability of class membership at that node.
  - Majority vote at that node.

Classification and regression trees

- Classification tree – when the variable we are trying to predict is categorical.
- Regression tree - when the variable we are trying to predict is continuous.

Pruning

- When there is a large number of features, the decision tree can become very complex.
- We risk *overfitting*. This is when an algorithm performs much better on the data it was trained on compared to new data.
- This may result from querying features that are not sufficiently informative.
- Pruning consists of removing sections of the decision tree in order to avoid overfitting. There are many algorithms for doing this.

Ensemble learning

- Ensemble learning combines many models into one.

- Bootstrap aggregating (Bagging).
  - Randomly re-sample from our original dataset with replacement.
  - Use this re-sampled data to train a model.
  - Repeat this many times, and take the average of the predictions from all models.
  - This combats overfitting.
  - Bootstrap aggregating applied to decision trees is called a *random forest*.

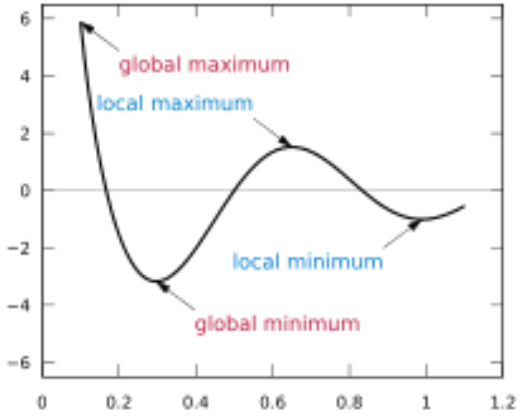
- Gradient boosted trees.

Gradient boosted trees

- Let x_i be a feature vector for observation i.
- Let $h_m(x)$ be a set of models, $m = 1 \ldots M$. Given a feature vector x, $h_m(x)$ returns a prediction.
- We wish to create a model that is a weighted average,
$$F(x) = \Sigma_m w_m h_m(x).$$
- Let $L(y_i, \hat{y}_i)$ be a loss function for an individual prediction.
- Choosing w_m, h_m to minimise the average loss is a difficult problem. We will try to solve it using an iterative optimisation process.

Gradient descent

- Gradient descent is a method for finding local minima/maxima of a function.
- For example, let's say you are stuck in a mountain range on a foggy day where visibility is low, and you're trying to get down.
- One way to proceed is to find the direction of steepest descent at your current location, and take a step in that direction. Repeat.
- Gradient descent does this for functions.
- Caution: It may only find local minima/maxima!



Gradient boosted trees

- Gradient boosted trees work as follows:
  - Initialise the model with a constant $\mathrm{F}_0(\mathrm{x}) = c$
  - Compute the *pseudo-residuals* r_i for each observation. This is the direction of steepest descent of the loss function for that observation.
  - Fit a model $h_m(x)$ that predicts the pseudo-residuals.
  - Choose an optimal weight w_m for this model.
  - Add the models together: $F_m = F_{m-1}(x) + w_m h_m(x)$.
  - Repeat.

- Gradient boosting is something like applying gradient descent method in function space.

sparklyr

- In sparklyr, use either of the following commands to fit a gradient boosted classification tree:

  ml_gradient_boosted_trees(data, formula, type = 'classification')

  ml_gbt_classifier(data, formula)

- Use either of the following commands to fit a gradient boosted regression tree:

  ml_gradient_boosted_trees(data, formula, type = 'regression')

  ml_gbt_regressor(data, formula)