Decision Trees

Splitting datasets one feature at a time

Jyotirmaya Shivottam

December 20, 2021

S-Lab, NISER

Table of contents

Overview

What are Decision Trees?

- Non-parametric supervised learning methods
- White-box - They mimic human approach to decision-making by identifying features or choices that are relevant and breaking them down to smaller sub-choices. This makes Decision Trees easy to interpret.
- Made up of conditional control (*if-else*) statements
- Can be used for classification (e.g. ID3) or regression (e.g. CART, C4.5, C5)
- One of the most popular predictive analysis tools. Often used in *Expert Systems*.

Tree Construction Algorithms

- Some of the well-known algorithms are ID3, C4.5, C5 and CART.
- Generally, the algorithms are *greedy* or *top-down*, looking for a locally-optimal choice.
- Input is of the form $(\mathbf{x}, Y) = (x_1, \ldots, x_k, Y)$, where x_i are the feature values and Y denotes target labels.
- Tree learners are recursive.
- At each iteration, the goal is to find the best split based on some metric, that measures the *homogeneity of the target variable within the subsets*.
- Two of the most commonly used measures are *Gini Impurity* and *Information Gain*.

Metrics - Gini Impurity

- Gini Impurity, used in the CART algorithm for classification, is a measure of the probability that a randomly selected example would be incorrectly classified. It is given as:

$$\mathrm{I_G}(X) = 1 - \sum_{i=1}^{n} \mathrm{P}(x_i)^2$$

  where, $\mathrm{P}(x_i)$ is the probability of choosing an item, x with label, i.
- $\mathrm{I_G}$ becomes 0, when the split is pure or when all cases come under a single class.
- Shows how the model differs from a pure split.
- Provides a criterion to minimize the probability of misclassification $\rightarrow$ Minimizing Gini Impurity per split.

Metrics - Information Gain i

- Some terminology:
  - (Self-)Information (I_X) quantifies the level of *surprise* of a particular outcome. It is given as:

$$\mathrm{I}(x_i) = -\log \mathrm{P}(x_i)$$

  - (Shannon) Entropy ($H(X)$) is defined as the expected value of information:

$$\begin{aligned} \mathrm{H}(X) &= \mathrm{E}[\mathrm{I}_X] \\ &= -\sum_{i=1}^{n} \mathrm{P}(x_i) \log \mathrm{P}(x_i) \end{aligned}$$

- Now, we can define Information Gain ($\mathrm{IG}(X)$) for a particular split over a feature, a:

$$\mathrm{IG}(X) = \mathrm{H}(X) - \mathrm{H}(X|a)$$

Metrics - Information Gain ii

- Information Gain, as used in the ID3, C4.5 & C5 algorithms, is a measure of the amount of information gained about a random variable from observing another random variable.
- Alternatively, it signifies the reduction in entropy or disorder on splitting the data in a certain way.
- Here, the goal becomes maximizing Information Gain, per split.

Metrics

- Both are information-theoretic measures and special cases of the *Tsallis Entropy*.
- In general, the choice of measure does not affect tree construction.
- Where things differ, is in application or in terms of the kind of trees, we want to construct.
- Gini Impurity only allows 2-way or binary splits, while Information Gain allows n-way or polyadic splits.
- Both metrics are useful for classification or regression, but Gini impurity is less suited to regression tasks, while Information Gain is viable for both.

Example - ID3 Algorithm

- ID3 or the *Iterative Dichotomiser 3* was described by Ross Quinlan in his seminal paper in 1986 [3].
- It is a recursive algorithm.
- It uses Information Gain as the measure of homogeneity.
- The algorithm goes as follows:
  - Calculate the entropy of every feature, a, of the data set, S.
  - Split S into subsets using the feature for which the entropy after splitting is minimum; or, equivalently, information gain is maximum.
  - Make a decision tree node containing that feature.
  - Recurse on subsets till $S - \{a\}$ is exhausted.

An example in code

An example in code

- Working with the Iris dataset.
- We will use scikit-learn's [2] *`DecisionTreeClassifier`* to classify between three classes or labels.
- We will look into ways to visualize the tree using *`graphviz`* and *`dtreeviz`*.

Refer to the .ipynb notebook for the code.

Conclusion

Advantages

- Interpretablity - Easy to understand and can be explained using Boolean logic.
- Built-in feature selection.
- Minimal data preparation is required.
- Efficiency - the computation cost increases in terms of the logarithm of the number of examples or data-points.
- Can handle numerical and categorical variables.
- Can handle multi-output problems.

Disadvantages

- Prone to *overfitting* or *under-generalization* especially when the training data is unbalanced. $\rightarrow$ Can be addressed via *Pruning*.
- Instability - small variations in data might result in completely different splits and hence trees $\rightarrow$ Can be solved via *Ensemble Learning*.
- Cannot interpolate or extrapolate as the predictions are piecewise constant approximations.
- In general, there is no Optimality Guarantee - the final tree may or may not be the most optimal tree. $\rightarrow$ Ensemble learning can help with this.

Summary

- Decision Trees are like work-flow diagrams or flow-charts, with leaf nodes or terminating blocks representing decisions, while other nodes represent sub-decisions.
- Tree construction algorithms are usually greedy recursive heuristics that search for the best local choice.
- The optimization criteria is usually either minimizing Gini Impurity or maximizing Information Gain.
- Decision Trees work with categorical and numerical data, and provide an interpretable learning approach.
- Various disadvantages of Decision Trees can be overcome using techniques such as Pruning or Ensemble Learning.

References

P. Harrington.
Machine Learning in Action.
Manning Publications Co., USA, 2012.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay.
Scikit-learn: Machine learning in Python.
Journal of Machine Learning Research, 12:2825–2830, 2011.

J. R. Quinlan.
Induction of decision trees.
Machine Learning, 1(1):81–106, Mar 1986.

Thank You! Questions?