Data Mining Lectures - Decision trees

Piotr Wąsiewicz

Institute of Computer Science

pwasiewi@elka.pw.edu.pl

25 maja 2017

Decision trees

- A hierarchical structure representing dataset/domain partitioning nodes: splits based on attribute-value conditions and leaves: class labels or probability distributions.
- Prediction by descending the tree at each node dispatching along a branch at each leaf a class label or probability determined.

Splits for discrete attributes

- Value-based: split outcomes correspond to single attribute values.
- Equality based: split outcomes correspond to binary equality test results.
- Partition-based: split outcomes correspond to attribute value subsets.
- Membership-based: split outcomes correspond to binary membership test results.

Splits for numeric attributes

- Inequality based: split outcomes correspond to binary inequality test results.
- Interval-based: split outcomes correspond to attribute value interval.

Decision tree growing

1. create the root node and mark it as open;
2. assign all training instances from T to the root node;
3. while there are open nodes:
   A. select an open node n;
   B. calculate class distribution $T(d|n)$ based on T[n];
   C. assign class label $argmax[d]T(d|n)$ to n;
   D. if stop criteria are satisfied for n mark n as a closed leaf; else
      1. select a split t for n;
      2. for each outcome r of split t:
      A. create a descendant node n[r] corresponding to r and mark it as open;
      B. assign all instances from T[n,t=r] to n[r];
      C. mark n as a closed node;

Stop criteria

- Uniform class: all training instances in the node are of the same class.
- No instances left: the set of training instances assigned to the node is empty.
- No splits left: there is no split that can be applied to further partition the current subset of training instances.
- Can be relaxed:
    1. most instances of the same class (low class impurity),
    2. less than a specified minimum number of instances,
    3. the best available split is not sufficiently good.

Split selection

- Strict stop criteria guarantee training set error minimization.
- Split selection responsible for overfitting avoidance.
- Ockham's razor: among trees with the same training set error prefer smaller ones and it can be achieved by minimizing class impurity e.g. its entropy.

Pruning and probability classification

- In pruning the complexity parameter (cp) controls the tradeoff between error and size
- Class probability distribution at leaves enables probabilistic prediction
- Can be used to minimize misclassification costs; instead of predicting the most probable class predict the class with the minimum expected cost,
- Can be used to adjust the operating point for binary classification e.g. obtaining the ROC curve

Growing the exemplary decision tree

An exemplary test dataset T

a	b	c	class (C)	nodemap (S)
a_1	b_1	c_1	1	**1**
a_2	b_1	c_1	1	**1**
a_1	b_2	c_1	0	**1**
a_2	b_2	c_2	1	**1**
a_2	b_2	c_2	1	**1**

- The dataset T has three input attributes a, b, c and target class labels C and a nodemap S - current tree terminal node indices which are attached the given instances to. At the beginning of creating (growing) the tree we have only one node with a number 1 and it is a root and a temporary leaf simultaneously, so the nodemap has only one node.

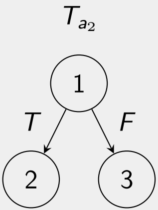
Growing the exemplary decision tree

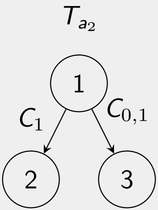
The impurity measure of splits

- In order to select a split condition first the impurity measure for all atrribute value splits e.g. $a(x) = a_1$ are calculated based on entropy e.g. for $a(x) = a_1$:
  $E_{a,a_1}(T) = -\frac{|T^0_{a,a_1}|}{|T_{a,a_1}|}\log_2(\frac{|T^0_{a,a_1}|}{|T_{a,a_1}|}) - \frac{|T^1_{a,a_1}|}{|T_{a,a_1}|}\log_2(\frac{|T^1_{a,a_1}|}{|T_{a,a_1}|}) =$
  $-\frac{1}{2}\log_2(\frac{1}{2}) - \frac{1}{2}\log_2(\frac{1}{2}) = 1$
-
  -

a	b	c	class (C)	nodemap (S)
$\boldsymbol{a}_1$	b_1	c_1	$\mathbf{1}$	1
a_2	b_1	c_1	1	1
$\boldsymbol{a}_1$	b_2	c_1	$\mathbf{0}$	1
a_2	b_2	c_2	1	1
a_2	b_2	c_2	1	1

Creating new nodes based on the selected split - disjoint instances subsets

- The split with the smallest entropy equal to 0 or as close as possible to 0 is selected. If several entropies are equal to 0, the split e.g. with the largest number of instances is chosen: $a(x) = a_2$ has three instances with the same class 1 this means $C_{a(x)=a_2} = 1$ or $C(a_2)_1 = True$ and stop criteria for the same target class values are applied successfully.
- $E_{a,a_2}(T) = -\frac{|T^0_{a,a_2}|}{|T_{a,a_2}|}\log_2(\frac{|T^0_{a,a_2}|}{|T_{a,a_2}|}) - \frac{|T^1_{a,a_2}|}{|T_{a,a_2}|}\log_2(\frac{|T^1_{a,a_2}|}{|T_{a,a_2}|}) = -\frac{0}{3}\log_2(\frac{0}{3}) - \frac{3}{3}\log_2(\frac{3}{3}) = 0$
- 



Growing the exemplary decision tree

Creating a leaf after the stop criterion is satisfied

- Based on the chosen split condition $a(x) = a_2$ the new nodemap is generated. The instances with the following value attributes $a(x) = a_2$ belong to the branch of the node 2 and as it was earlier mentioned the node 2 becomes the tree leaf (a terminal node with one class value) and the rest of instances belongs to the node 3. Thus, not necessary all target classes in the branch subsets should be equal to 1, sometimes the majority of the same instances class values above the given threshold is the admisible criterion for creating leaves.
-

a	b	c	class (C)	nodemap (S)
a_1	b_1	c_1	1	3
a_2	b_1	c_1	**1**	**2**
a_1	b_2	c_1	0	3
a_2	b_2	c_2	**1**	**2**
a_2	b_2	c_2	**1**	**2**

Growing the exemplary decision tree

The next layer of child binary tree nodes: $2n$ and $2n+1$

- In the next steps for every next node instance subset the split with the smallest entropy equal to 0 or as close as possible to 0 is also chosen.
- But for the node 2 instances denoted in the nodemap the stop criterion is met and it can become a leaf which accurately classifies the attached test dataset instances and hopefully will achieve good accuracy for new similar instances chosen by the condition $a(x) = a_2$ (there will be a few false positives).
- For the node 3 branch instance subset the attribute b values are appriopriate: for each target class it has different values. It creates two nodes with binary tree child nodes numbering $2n = 6$ and $2n+1 = 7$.
  $E_{b,b_1}(T) = -\frac{|T^0_{b,b_1}|}{|T_{b,b_1}|}\log_2(\frac{|T^0_{b,b_1}|}{|T_{b,b_1}|}) - \frac{|T^1_{b,b_1}|}{|T_{b,b_1}|}\log_2(\frac{|T^1_{b,b_1}|}{|T_{b,b_1}|}) = -\frac{0}{1}\log_2(\frac{0}{1}) - \frac{1}{1}\log_2(\frac{1}{1}) = 0$
  $E_{b,b_2}(T) = 0$
  $E_{c,c_1}(T) = -\frac{1}{2}\log_2(\frac{1}{2}) - \frac{1}{2}\log_2(\frac{1}{2}) = 1$

Growing the exemplary decision tree

Visualization of the final classifier generation