Machine Learning Study

Takumi

1 Algorithms

1.1 ID3 - Information Gain

Algorithm

1. ID3 uses greedy algorithm and picks the feature associated with the max information gain as the root node.
2. A top-down approach dives deeper by working on a sub-sample space of each branch resulting from the original split; again uses a greedy algorithm to pick features as subsequent internal nodes.
3. Repeat until there is a unique label in the subset, assigned as leaf node

Mathematical Representation

$$H(D) = \sum_k \frac{|C_k|}{|D|} \cdot log_2 \left(\frac{|C_k|}{|D|} \right)$$

where $|D|$ is the size of the full sample and $|C_k|$ is the size of sample with label k

For any feature A, the conditional information entropy is,

$$\begin{aligned} H(D|A) &= \sum_i (\frac{|D_i|}{|D|}) \cdot H(D_i|A) \\ &= \sum_i (\frac{|D_i|}{|D|}) \left(\sum_k \frac{|D_{ik}|}{|D_i|} \cdot log_2 \left(\frac{|D_{ik}|}{|D_i|} \right) \right) \end{aligned}$$

$H(D|A)$ is the weighted average impurity resulting from splitting on the feature (A); $H(D)$ is the impurity of the full sample.

$$\text{Information Gain} = H(D) - H(D|A)$$

Greedily, pick feature A that reduces impurity to the largest degree as the root node. Suppose $A = \{A_1, A_2\}$; then on A_1 or A_2 subset separately, greedily pick the best feature to be the child node.

Cons

- No prune, overfitting
- ID3 can only deal with discrete features but not continuous ones.

- ID3 favors features that take more values (eg. ID) since impurity tends to be smaller for those features and thus information gain is greater.
- no pre-processing of nan
- multinomial tree, log computations, costly

1.2 C4.5 - Information Gain Ratio

Algorithm

Similar to ID3, but use information gain ratio as the determinant metric instead. Beyond, C4.5 uses pessimistic error pruning, a post-pruning technique, to avoid overfitting. It also discretizes continuous features.

Mathematical Representation

$$\text{Information Gain Ratio} = \frac{\text{Information Gain of feature } A}{\text{Entropy of feature } A}$$

ID3 favors features that take more values since those features split the sample into granular buckets where each bucket tends to have unique label. However, such features are impure in its prior distribution. C4.5 penalizes that by dividing the information gain by the information entropy of the feature itself.

Discretization

For continuous feature A which takes N value, pick a point to split the sample into binary subsets.

1. sort the N values of feature A in ascending order; there are N - 1 options for the decision boundary; each is the average between its left and right value.
2. pick the boundary that results in the greatest information gain. Any value smaller than the boundary goes to the left branch and any value greater goes to the right branch.

PEP (pessimistic error pruning)

(Recap)

Bernoulli distribution:

$$\mathbb{E}[X] = p$$
$$Var[X] = p \cdot (1 - p)$$

$Y = \sum_i X_i$ where X_i is n i.i.d Bernoulli random variable.

$$\mathbb{E}[Y] = n \cdot p$$

$$Var[Y] = n \cdot p \cdot (1-p)$$

When population parameter p is unknown, use $\hat{p}$, the sample ratio, to estimate. Sample ratio is an unbiased estimator of the true p.

For a subtree, suppose it has L leaf nodes, then the aggregate error ratio of that subtree is

$$ErrorRatio = \frac{\sum_{i=1}^{L} e_i + 0.5L}{\sum_{i=1}^{L} n_i}$$

where n_i is the sample size and e_i is the number of FP/FN of leaf node i. '0.5' is the penalize factor.

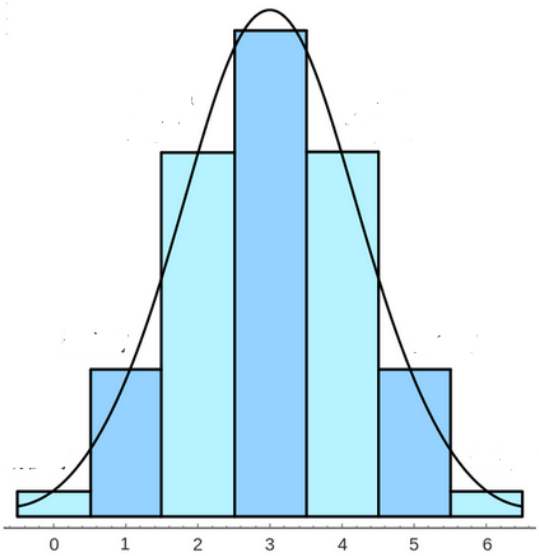

Figure 1: Normal Curve on Sum of Bernoulli r.v.

As shown in the figure above, $\mathbb{P}(e < a)$ is actually equal to $\mathbb{P}(Z < \frac{a+0.5-nq}{\sqrt{npq}})$ since e, at the end of the day, is discrete but Z is continuous. Therefore, add '0.5' to adjust.

Suppose whether this subtree correctly classifies a sample is subject to Bernoulli distribution and it's i.i.d between samples. Then $1_{FN/FP} \sim Bernoulli(q)$. When true q is unknown, we can use the sample error ratio, the above equation, to estimate.

According to some version of the CLT, as $np > 5$ and $n(1-p) > 5$, the sample ratio approaches the Gaussian distribution.

$$\hat{e} \sim N(nq, npq)$$
$$\widehat{errorRatio} \sim N(q, \frac{pq}{n})$$

Therefore, the mean of the error (number of) is

$$ErrorMean = ErrorRatio \cdot \sum_{i}^{L} n_i$$

the standard deviation of the error (number of) is

$$ErrorSTD = \sqrt{ErrorRatio \cdot (1 - ErrorRatio) \cdot \sum_{i}^{L} n_i}$$

Then, suppose we prune the subtree and replace it with a leaf node and assign label under majority voting, the error ratio is

$$ErrorRatio' = \frac{\sum_i^L e'_i + 0.5}{\sum_i^L n_i}$$

the mean is

$$ErrorMean' = ErrorRatio' \cdot \sum_{i}^{L} n_i$$

The decision boundary is to prune the subtree if $ErrorMean' < ErrorMean + ErrorSTD$.

Pros

- deal with both discrete and continuous features
- use post-pruning (pessimistic error pruning) to mitigate overfitting
- handle incomplete data

Cons

- log and continuous features are computationally heavy
- only handle classification problem

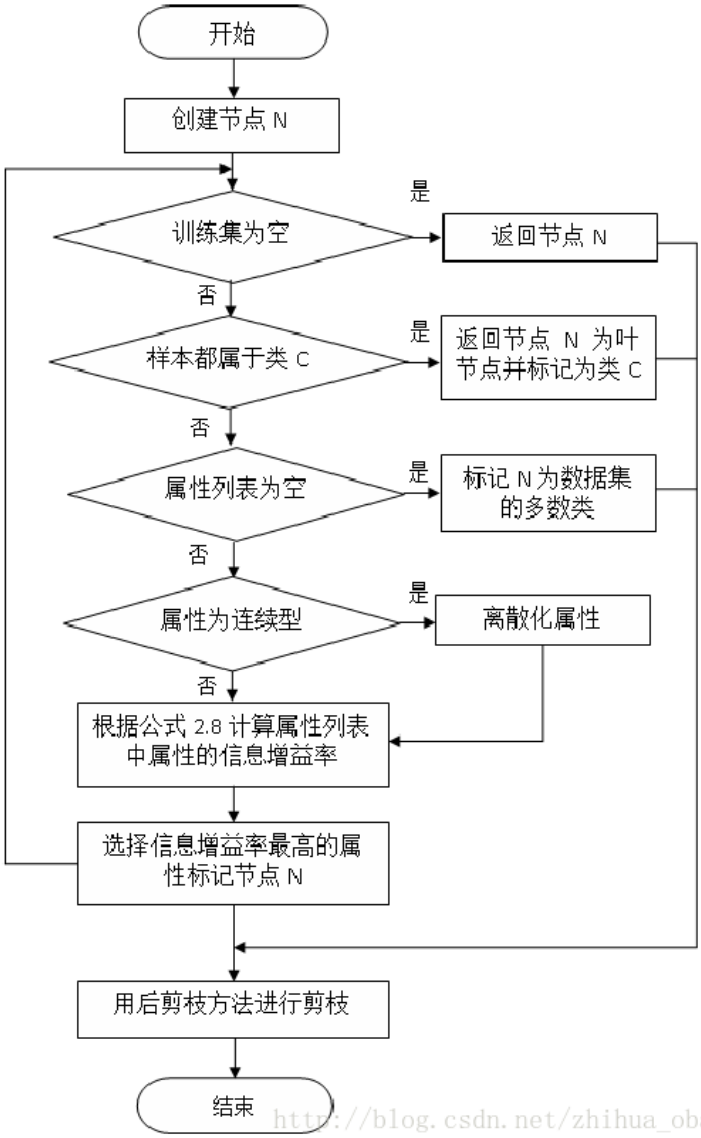

Figure 2: Diagram