Applied Machine Learning Final Project Proposal

Amir Mehrpanah

June 2020

Todo list

Binary Tree Integrated Ensemble Models

Abstract

There are cases in which we can't achieve a reliable(or accurate) prediction based on just using one complex function. This leads us to use an ensemble of simple classifiers, namely $\Gamma = \{h_i\}_{i\in\mathbb{N}_k}$ for some $k \in \mathbb{N}$, which has practically proved to perform better than a single classifier. There are many ways to compute the final result of an ensemble. One of which is voting or weighted voting. In this section we propose a better way of combining parts into a whole, which accelerates the reaction time with little compromise on the accuracy of our ensemble.

Direction

Intuitively speaking, whenever a classifier h_i (for simplicity suppose each $h_i \in \Gamma$ is a binary classifier) predicts 1 or 0 it doesn't tell us anything about the confidence of this prediction. As an example let's suppose h_i for some specific i defined as follows:

$$h_i(x) = \begin{cases} +1 & \langle x, w_i\rangle + b \geq 0 \\ -1 & o.w \end{cases}$$

This function will predict one for either 0.1 or 10, although this prediction, definitely, does not share the same confidence. We know the lack of confidence in some predictions may be compensated by the final voting over all possible values for i.

Method

To solve both confidence problem and gain a faster reaction, we can rearrange the members of Γ with a binary tree. First we assume that Γ is a sequence ($\Gamma = (h_i)_{i\in\mathbb{N}_k}$) and $h_i \trianglelefteq h_j$ means it is less probable to have low-confidence

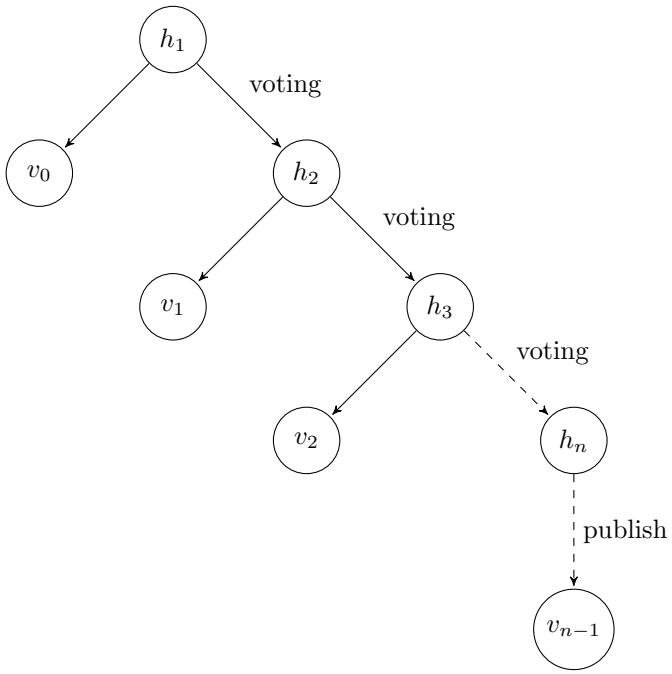

Figure 1: A Binary Tree decides whether it is necessary to engage the next classifier or not. Here v_i stands for voting result in $i'th$ step

predictions for h_i in contrast to h_j on a given S. Following pseudo-code of the binary tree and Figure 1 provided for clarification. The aim is to decrease both the expected algorithm complexity and expected response time.

```
1  Gamma = [h1, h2,..., hk] # with hi <= hj iff i <= j
2  def predict(x):
3      for h in Gamma:
4          [v, c] = h(x) # v is the value of prediction probe ↩
               and c is confidence
5          if c is very high: # do some kinda thresholding
6              return sign(v) # h is confident stop the voting
7          else: # h is not confident enough
8              vote += sign(v) # do some kinda voting
9      return sign(vote) # no one's sure about x so we publish↩
           voting result
```

include relevant literature

Perspective

This problem can have interesting extensions where the heterogeneous computing has also taken into account. Consider the choice of j for engagement of h_j where h_i is currently being computed. Then we need to minimize expected computation time where trying to maximize accuracy parameters in each step.