CS446: Machine Learning **Spring 2017**

Problem Set 2 Solutions

Handed Out: Feb 3^{th}, 2017 *Handed In: Feb* 15^{nd}, 2017

1. Learning Decision Trees

   (a) Data Set, $D \rightarrow [35+, 15-]$
   Values(Holiday) $=\{yes, no\}$
   $D_{Holiday=yes} \rightarrow [20+, 1-]$
   $D_{Holiday=no} \rightarrow [15+, 14-]$
   $D_{ExamTomorrow=yes} \rightarrow [10+, 5-]$
   $D_{ExamTomorrow=no} \rightarrow [25+, 10-]$

   Information Gain for Holiday:

   $Gain(D, Holiday) = Entropy(D) - \sum_{h\in\{yes,no\}} \frac{|D_h|}{D} * Entropy(D_h)$
   $= Entropy(D) - (\frac{21}{50})Entropy(D_{Holiday=yes}) - (\frac{29}{50})Entropy(D_{Holiday=no})$

   Now, $Entropy(D) = Entropy([35+, 15-])$
   $= -(\frac{35}{50})lg(\frac{35}{50}) - (\frac{15}{50})lg(\frac{15}{50})$
   $= 0.8813$

   $Entropy(D_{Holiday=yes}) = Entropy([20+, 1-])$
   $= -(\frac{20}{21})lg(\frac{20}{21}) - (\frac{1}{21})lg(\frac{1}{21})$
   $= 0.2762$

   $Entropy(D_{Holiday=no}) = Entropy([20+, 14-])$
   $= -(\frac{15}{29})lg(\frac{15}{29}) - (\frac{14}{29})lg(\frac{14}{29})$
   $= 0.9991$

   Therefore, $Gain(D, Holiday) = 0.8813 - (\frac{21}{50}) * 0.2762 - (\frac{29}{50}) * 0.9991$
   $= 0.1858$

   Information Gain for Exam Tomorrow:

   $Gain(D, ExamTomorrow) = Entropy(D) - \sum_{h\in\{yes,no\}} \frac{|D_h|}{D} * Entropy(D_h)$
   $= Entropy(D) - (\frac{15}{50})Entropy(D_{ExamTomorrow=yes})$
   $-(\frac{35}{50})Entropy(D_{ExamTomorrow=no})$

   Now, $Entropy(D) = 0.8813$

   $Entropy(D_{ExamTomorrow=yes}) = Entropy([10+, 5-])$
   $= -(\frac{10}{15})lg(\frac{10}{15}) - (\frac{5}{15})lg(\frac{5}{15})$

$= 0.9183$

$Entropy(D_{ExamTomorrow=no}) = Entropy([25+, 10-])$
$= -(\frac{25}{35})lg(\frac{25}{35}) - (\frac{10}{35})lg(\frac{10}{35})$
$= 0.8631$

Therefore, $Gain(D, ExamTomorrow) = 0.8813 - (\frac{15}{50}) * 0.9183 - (\frac{35}{50}) * 0.8631$
$= 0.0016$

As ID3 interprets *best attribute* as the attribute with highest information gain, $Holiday$ is the best attribute in this case and therefore, is selected as the root attribute

(b) Heres a tree based on the misclassification heuristic. The rule to break the tie is to use the left most attribute in the table.

```
if Color = Blue :
  if Size = Small :
    class = F
  if Size = Large :
    if Act = Dip :
      class = T
    if Act = Stretch :
      if Age = Adult :
        class = F
      if Age = Child :
        class = T
if Color = Red :
  if Size = Small:
    if Act = Dip :
      class = T
    if Act = Stretch :
      if Age = Adult :
        class = F
      if Age = Child :
        class = T
  if Size = Large:
    if Act = Dip :
      class = T
    if Act = Stretch :
      if Age = Adult :
        class = F
      if Age = Child :
        class = T
```

`Note`: Suppose a data set has two labels, + and -, with n^+ and n^- examples for corresponding labels, where $n^+ > n$. If, after splitting on an attribute, we find

that for every child node, $n^+ > n$, then the information gain measured using the majority error will be zero. (Why?) However, information gain using the entropy can provide a better estimate of impurity in these cases.

(c) The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts. Consequently, practical decision-tree learning algorithms, e.g., ID3, C 4.5, are based on heuristics such as the greedy algorithm where locally-optimal decisions are made at each node. In case of ID3, the greedy choice is made based on the value of information gain. As there is no backtracking involved, such algorithms cannot guarantee to return the globally-optimal decision tree.

2. Decision Trees as Features

We provide a brief description of the target concept and report the results of one particular run of the different approaches.

The target concept is kind of sweet; or at least we think it is. If the beginning five letters in last name contains 'm', then the label is the boolean variable indicating whether the first name contains any alphabet from 'love' (i.e., 'l', 'o', 'v', or 'e'). Else if the beginning five letters in last name contains 'l', then the label is the boolean variable indicating whether the first name contains any alphabet from 'like'. Otherwise, the label is the boolean variable indicating whether the first name contains any alphabet from 'foo' AND the last name contains any alphabet from 'pod'. For example, the label for name "eric baum" is "+" according to the concept above.

While this is a relatively simple concept to describe, in terms of the features we gave you, it is not easy, though theoretically possible, to represent as a tree. This is because the features create too sparse of a space and a learner can easily overfit the data.

Using the features described in the homework, the results obtained by one run of the algorithms are summarized in Table 1. For all the SGD methods, the learning rate was chosen by cross-validation from the set $\{0.00001, 0.0001, 0.001, 0.01\}$.

Algorithm	**Accuracy (%)**	**Std. Dev.**	**Parameters**
Simple SGD (LMS)	69.7 ± 8.6	4.2	Learning rate = 10^{-5}
Full decision tree	73.8 ± 11.1	5.4	
Decision stump	62.8 ± 12.6	6.1	Depth = 4
Decision stump	68.9 ± 8.9	4.3	Depth = 8
Stumps+SGD	75.3 ± 10.5	5.1	Depth = 8, Learning rate = 10^{-5}

Table 1: Performance of various algorithms on the badges data

We see that, for this data set, the best performance is achieved using SGD over decision stumps. The reason for this lies in the expressiveness. The underlying concept which generated the data is complex — complex enough to not be linearly separable in our feature space. Thus, SGD with simple features is unable to obtain low error as the space of all linear classifiers with our "base" features is not expressive enough to express the underlying concept. On the other hand, decision stumps provide a more expressive

Algorithm	Accuracy (%)	Std. Dev.	Parameters
Simple SGD (LMS)	73.5 ± 7.6	3.7	Learning rate = 10^{-5}
Full decision tree	84.0 ± 7.2	3.5	
Decision stump	83.0 ± 7.4	3.6	Depth = 4
Decision stump	84.3 ± 7.8	3.8	Depth = 8
Stumps+SGD	86.4 ± 5.8	2.8	Depth = 8, Learning rate = 10^{-5}

Table 2: Performance of various algorithms on the badges data with additional 'alphabet-present' features.

representation. For instance, the `XOR` function cannot be expressed as a linear function but it can expressed as a decision tree (or stump) of depth 2. Given the nature of our underlying concept, the 100-dimensional expressive decision stump feature space is better equipped to fit to the data. Note that this not very different conceptually from using feature conjunctions – in both cases, we transform the data from a lower dimensional feature representation to a higher dimensional one. The key difference, though, is that in this case the transformation itself is learned using a different learning algorithm (ID3).

After adding additional features which indicate the presence of a character in the first and the last name, we can improve the accuracy of the classifiers. These are mentioned in Table 2. Notably, with these features all the algorithms are improved. The best performance now is still SGD over decision stumps. This can be explained by the fact that our underlying concept can be better represented by the "presence of a character" feature. Using the paired t-test, we note that the improvement of Stumps+SGD over Decision stump (depth=8) is significant with $p < 0.10$[1].

[1]http://www.statstutor.ac.uk/resources/uploaded/paired-t-test.pdf. A note on how to calculate the t-statistic. Thanks to Bhopesh Bassi for pointing this resource to us.