Supervised Learning

Writeup for Assignment 01 - CS 6741

Magahet Mendiola

ABSTRACT

An analysis of two machine learning classification problems, including an evaluation of various learning algorithms and an exploration into the

1. DATA

Two classification datasets have been chosen based on the following criteria: Each dataset includes a minimum of 3,000 instances to insure sufficient data to evaluate each learning algorithm. Test sets were created by taking a uniform random sample of 33% of the original instances without replacement. The remaining 66% were then used as a training superset. Smaller subsets were pulled from this partition as needed.

2. CLASSIFICATION - MUSHROOMS

The first dataset is titled, Mushroom [1] and was chosen from the UCI ML Repository. The classification task for this dataset is to determine whether a given mushroom is edible or poisonous based on the specimen's physical attributes. There are 22 recorded attributes which describe the physical appearance and olfactory perception of each sample. A full description of attributes can be found at http://archive.ics.uci.edu/ml/datasets/Mushroom.

This dataset allows for experimentation of various learning algorithms against attributes with only discrete values. The recorded attributes also require minimal domain knowledge to interpret, which reduces complexity in conceptualizing relationships and evaluating each learner's behavior.

2.1 Attribute characteristics

By examining the value distributions as seen in Figure 1, we first observe that the two classifications are fairly evenly distributed (48% edible, 52% poisonous). This allows for the evaluation of learning algorithm accuracy based on the percentage of classification mistakes made by the model on our test set. This metric will not be unduly weighted by a greater occurrence of one of the classes. It is worth considering the use case for such a classification task though. For example, if this model were to be used to decide whether to serve a given mushroom to a class full of students, we would prefer a hypothesis that favored misclassification of edible over that of poisonous. This idea will be examined while exploring the test results.

Each attribute includes a finite and limited set of values (min: 2, max:12). This indicates that information gain would provide a suitable metric for decision tree attribute selection, as the variance in value ranges is not significant and, similar to our reasoning for class distribution, we will not need to account for attributes with greater than average ranges of values.

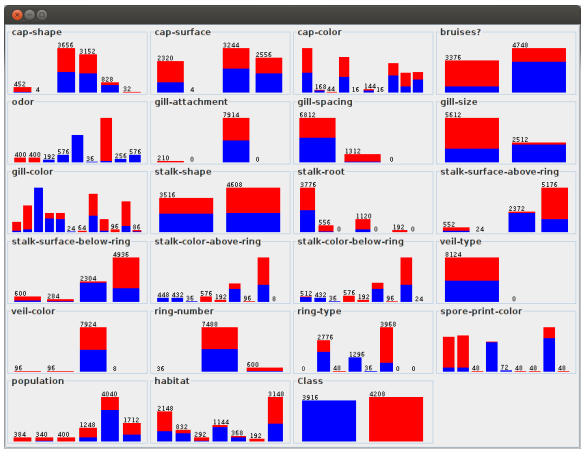

Figure 1: mushroom attribute distribution and classification

With all the attributes taking discrete (nominal) values, we will be somewhat constrained in learning algorithms options. For example, neural network network input nodes will need to be setup for each attribute value pair; separating them into binary attributes for each value. Similarly, nominal attributes will limit the capabilities of SVM. Non-linear kernels will not be effective in increasing the, already linearity separable, feature space. The behavior of k-nn will also be constrained as well. Distance metrics for such attributes will either be simplistic (testing for exact matches) or less intuitive than measuring euclidean distance.

It can also be observed visually that a number of attributes provide clear separability between the target classes. This supports the intuition that this set of attributes does have some functional relationship to the classes. In fact it appears, from the figure, that the attribute "odor" by itself could provide a fairly clear indicator of how a sample should be classified. "Spore-print-color" appears to provide decent class separation as well. It can be predicted that these attributes will be heavily weighted during the attribute selection or weighting process.

2.2 Algorithm Evaluations

Each leaning algorithm has been applied to the training

data with the same procedure. As described earlier, a subset of data was partitioned and preserved as a training set. The remain instances were then sample from to create smaller training sets for use in evaluating each learning algorithms' performance and behavior.

2.2.1 Learning Curves

Evaluation began by choosing conservative starting values for each learners' parameters. Then each learner was trained against a range of training set sizes between 10 and 2000 instances (increments of 10). After each training phase, the original test data (1/3 of the original dataset) was used to evaluate the learner's performance. In this case classification error was used as the performance metric. The plot in Figure 2 shows each learning algorithm's results.

A number of assumptions are made for this initial evaluation. The goal of the exercise being to obtain a general intuition regarding each algorithms' performance, and not to present ideal conditions. Further parameter changes are discussed in subsequent sections of this analysis. The following describes the initial conditions for each learning algorithm.

Decision Tree Weka's open source version of the C4.5 algorithm (J48) was chosen, which uses normalized information gain ratio to select attributes. Post-pruning was done with a minimum leaf size of 2 instances a confidence factor of 0.25.

Boosting Although it would be beneficial to compare our single iteration C4.5 to the same learner with boosting enable, unfortunately that was not possible with this dataset. As can be observed in the learning curve plots below, the training error for J48 was negligible. This caused the AdaBoost algorithm to compute a zero classification error tree in most cases. This resulted in the boosting process returning after a single decision tree is built. This result persisted even with more aggressive post-pruning parameters. In order to evaluate boosting, a weaker learner was chosen in place of C4.5. Boosting was performed on a decision stump learner, which is a simplistic decision tree algorithm that creates a single node/attribute tree.

Neural Network A multilayer perceptron network was used with one hidden layer using a sigmoid activation function. Input nodes were created for each attribute/value pair, and 63 hidden nodes were used, which is the number of input nodes plus the number of output nodes divided by two. The algorithm performs gradient decent to adjust weights, and includes both a learning rate and a momentum factor to slow down and smooth weight changes. For this test, both terms are held static (learning rate = 0.3, momentum = 0.2).

k-NN For the initial tests, the nearest neighbor algorithm was set to k = 1. This matches the single closest instance from the training set to determine a classification. The distance metric used is euclidean distance. The plot titles for k-NN are IBk, which is the Weka implementation of k-NN.

SVM SVM was run initially with a linear kernel. Both polynomial and radial basis function kernels are evaluated later. The plot title for SVM is SMO, which is Weka's implementation of SVM.

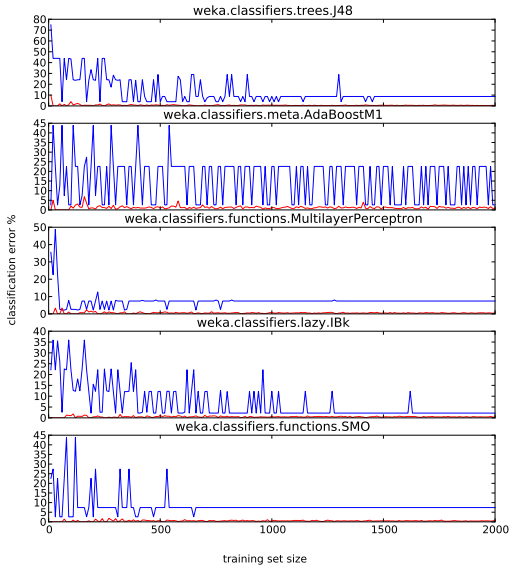

Figure 2: Mushroom - Classification Error by Training Size

The results show training error (red lines) for every learner remain below 10%. In all cases, except decision trees, this error was never higher than 5% even with only 10 samples to train from. This indicates that our learners were able to converge to a hypothesis that fit the training data very well for even small sample sizes. The overall dataset seems to have a little variance and each learner is able to closely model the training data.

The test error plots also exhibit low error for most learners, with the exception of boosting. As expected, smaller training sample sizes show higher misclassification rates. The error rates quickly drop to below 10% for training sets around 1,000 samples. Comparing each learner, we see that our perceptron network was able to reach this low level of classification error with the smallest number of training examples (100). However, the runtime plot in Figure 3 shows that this learner was the slowest to train by an order of magnitude. Depending on the use case, a trade-off can be made for which learner to implement.

Another interesting result is from our boosting algorithm, which struggled to reach the same level of classification error as the others. The intuition for this is that as there are certain attributes with much greater performance metric results which unduly influenced the ability of the weak learner to find suitable attributes other than those with the most influence on the classification.

2.2.2 Training Times

The elapsed time required for each learner to consume the training set has been plotting in Figure 3. Results from this data indicate that C4.5 performed well empirically. Training time for decision trees appeared to run two orders of magnitude faster than our neural network, and one order greater than SVM. The perceptron network seemed to growing linearly and remained far above any other learner. The one surprise in the results was k-NN, which seemed to take

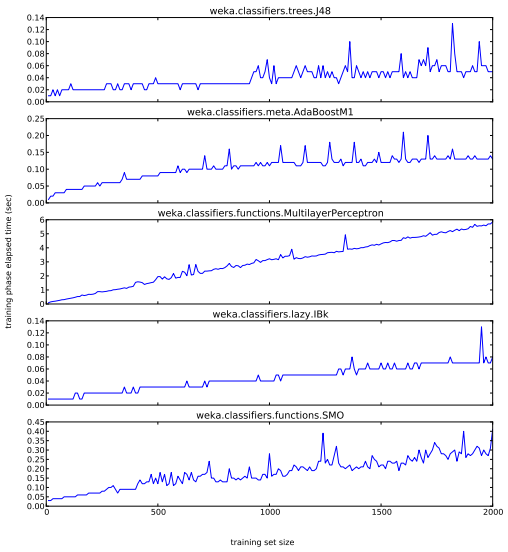

Figure 3: Mushroom - Training Time by Training Size

longer to train than decision trees. This may be due to the preprocessing of samples to prepare them for distance measurements, such as pre-sorting.

2.2.3 *Decision Trees*

Further experimentation into the decision tree learner was performed on subsets of the original data. As before, 1/3 of the data was partitioned for testing. The training set was then taken as 1,000 instances from the remaining set.

As a baseline, the learner was run with initial settings, which included post-pruning with a 0.25 confidence factor and a minimum leaf size of 2 instances. These settings produced the tree in Figure 4. This tree produced no training classification error and 12% error on the test set. The majority of training instances were classified based on two attributes, odor and stalk-shape.

The discrepancy in training and testing error indicates that the learner may be overfitting the training data. This is further supported by the small number of instances categorized by deeper leaves. To try to archive better generalize, the same process was repeated with more aggressive post-pruning parameters. The results of setting the confidence factor to 0.01 is shown in Figure 5.

Although a negligible number of instances were miscategorized from the training set, the classification error on the test set dropped to 4%. The resulting tree is made up of a single attribute node with nine leaves. This confirms the initial hypothesis that overfitting was occurring with the initial parameters. It can also be seen that, as expected, the odor attribute does in fact exert a great deal of influence on correct classification. To validate this further, Weka's attribute evaluator tool was used to calculate the information gain for this attribute, which came out at 0.83. This confirms further the strength of this attribute in classification.

It should be noted that Adjusting the post-pruning parameters further would have no effect on the tree, as it cannot be pruned further. When run with no post-pruning performed, the same tree is produced as with the initial settings.

```
odor = a: e (69.0)
odor = l: e (77.0)
odor = c: p (37.0)
odor = y: p (6.0)
odor = f: p (234.0)
odor = m: e (0.0)
odor = n
|   stalk-shape = e
|   |   habitat = g: p (3.0)
|   |   habitat = l
|   |   |   bruises? = t: p (3.0)
|   |   |   bruises? = f: e (4.0)
|   |   habitat = m: p (4.0)
|   |   habitat = p: e (0.0)
|   |   habitat = u: e (16.0)
|   |   habitat = w: e (18.0)
|   |   habitat = d
|   |   |   stalk-surface-above-ring = f: p (0.0)
|   |   |   stalk-surface-above-ring = y: p (0.0)
|   |   |   stalk-surface-above-ring = k: p (4.0)
|   |   |   stalk-surface-above-ring = s: e (4.0)
|   stalk-shape = t: e (468.0)
odor = p: p (51.0)
odor = s: p (2.0)

Number of Leaves  :     20

Size of the tree :  25
```

Figure 4: c4.5 decision tree with 0.25 confidence interval

2.3 Neural networks

2.4 Boosting

2.5 Support Vector Machines

2.6 k-nearest neighbor

3. CLASSIFICATION - INCOME

Our second dataset was also chosen from the UCI ML Repository and is titled, Adult [1]. The classification task in this case is to determine if one's household income exeeds $50,000/yr based on 14 biographical attributes. The data was collected from a census database from 1994. Attributes include the subject's age, level of education, marital-status, occupation, race, sex, etc. The full description of attributes can be found at http://archive.ics.uci.edu/ml/datasets/Adult.

4. ANALYSIS

4.1 Overview of Results

4.2 Algorithm Comparison

5. REFERENCES

[1] K. Bache and M. Lichman UCI Machine Learning Repository 2013 http://archive.ics.uci.edu/ml

```
odor = a: e (69.0)
odor = l: e (77.0)
odor = c: p (37.0)
odor = y: p (6.0)
odor = f: p (234.0)
odor = m: e (0.0)
odor = n: e (524.0/14.0)
odor = p: p (51.0)
odor = s: p (2.0)

Number of Leaves  :     9

Size of the tree :  10
```

Figure 5: c4.5 decision tree with 0.01 confidence interval

University of California, Irvine, School of Information and Computer Sciences