Online Applied Machine Learning Programming Assignment

(Due April 11, 2017 @ 11:59PM)

For this programming/Weka assignment you will compare the performance of SVM, Random Forest, and Adaboost on the Adult data set from the UCI repository (http://archive.ics.uci.edu/ml/datasets/Adult). The prediction task associated with this data set is to predict whether or not a person makes more than $50K a year using census data.

To complete this assignment, you need to do one of two things for each of the four classifiers you will be examining as a part of this assignment. You need to either implement the machine learning algorithm in Python (you may use Numpy, Scipy, and/or SKlearn) or use Weka to learn and evaluate the model. You may implement all three algorithms in Python, use Weka to run each of the algorithms on the data, or implement some of the algorithms in Python and use Weka to run the other algorithms on the data. The choice is yours, but make sure to clearly indicate whether you used Weka or implemented the algorithms in Python.

For each classifier you evaluate on the UCI dataset, you will need to report any preprocessing you did on the data before learning a prediction model for (ie. did you discretize the features? If so, how?), the performance of the learned model on the training set and on the test set (this is how we can check for overfitting), and why you chose to evaluate the model's performance the way you did. A brief outline the steps you should follow can be found below.

1. Use adult.data as your training set and adult.test as your test set.
2. You may want to consider creating a validation set from the training set to help you select hyper-parameters. Hyper-parameters you may want to tune are:
   a. the value of C in SVM;
   b. the depth of the trees in Adaboost and Random Forest;
   c. the number of trees in Adaboost and Random Forest.
3. Take a look at the data and get a feel for the types of feature values you observe.
4. Preprocess the data. Make sure to explain how you preprocessed the data and why.
   a. It is important to note that some of the features are missing values (indicated with a "?"). How will you handle missing feature values? Explain your solution and why you chose it. You can treat a missing feature value as its own feature value, but this might not be the best solution, so you should look into others.
   b. Some of the feature values are continuous and some are categorical. Do you want to discretize the continuous features? If so, why? Do you want to discretize the feature values for all of the classifiers? It is okay to preprocess the data differently for each of the classifiers, just make sure to explain what you did and why.
5. If you are using Weka, then convert the CSV into the appropriate ARFF format. If you are using Python, then I recommend looking into Numpy's loadtxt() function.
6. If you are using Python/SKlearn, then implement the learning algorithm.
7. Learn the model using the training set and tune hyper-parameters on the validation set.
8. Evaluate the learned model on the test set.
   a. If you used a validation set to tune your hyper-parameters, then make sure to use the model with the best performance on the test set.
9. Report the performance of the model.
   a. Make sure to use an appropriate performance measure (ie. accuracy vs. ROC vs. F1-score). Explain why you chose that performance measure.

    b. Explain the effect different hyper-parameters have on the performance of the algorithm. For example, what effect does the depth of the tree have on the performance of the decision tree?

10. Briefly compare the performance of the different learning algorithms on the test set and make some hypothesizes about why you might have observed these differences.

You can download the data set from http://archive.ics.uci.edu/ml/machine-learning-databases/adult/ and find the ReadMe for the data set at http://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.names . In the ReadMe you can find current performance results of a number of different classification algorithms. This should give you a point of comparison for your own classifiers. However, you should play around with the different learning parameters and see if you can beat their results!! I will ask students who are particularly creative and able to achieve really interesting results to share what they did with everyone else.