Ryan C. Adams
Applied Machine Learning

Project 3 Report

Introduction:

Our program analyzes the run time an accuracy of the following classifiers:

1. Perceptron
2. Support vector machine (linear and rbf kernels)
3. Decision tree
4. K-nearest neighbor
5. Logistic regression

We leave all parameters set to their default categories to have a baseline understanding of the 'out of the box' results without any tuning.

We utilize two datasets:

1. The digits dataset, which is a built in library from scikit
2. The Human Activity Recognition Using Smartphones Data Set, availabe on kaggle at https://www.kaggle.com/mboaglio/simplifiedhuarus/home

Please note that the Human Activity Recognition Using Smartphones Data set uses a rolling time window of 2.5 seconds

Discussion:

Digit Dataset:

We obtain the following results for the digits data set with all classifiers set to default settings, using a test size of .33 and a random state of 25.

Two fastest results - perceptron, decision tree
Two with highest accuracy - knn, linear svm

```
================================================
Generating report for Classifiers
Note times are in the form min:second.nanoseconds
================================================

                    accuracy         time
name
perceptron          0.929293  00:00.010745
decision tree       0.840067  00:00.012083
knn                 0.991582  00:00.073500
logistic regression 0.959596  00:00.143461
linear_svm          0.981481  00:00.039630
rbf_svm             0.547138  00:00.313792
```

Human Activity Recognition Using Smartphones Dataset:

Human Activity Recognition Using Smartphones with all classifiers set to default settings, using a test size of .33 and a random state of 25.

Two fastest results - perceptron, linear_svm
Two with highest accuracy - logistic regression, linear svm

```
================================================
Generating report for Classifiers
Note times are in the form min:second.nanoseconds
================================================

                    accuracy         time
name
perceptron          0.904282  00:00.083927
decision tree       0.896725  00:00.841329
knn                 0.934509  00:01.852434
logistic regression 0.972292  00:01.599959
linear_svm          0.973132  00:00.752981
rbf_svm             0.914358  00:02.421335
```

Decision Tree Classifier:

Strategies Pre Pruning:

The main thing I took away from this is that the pruning happens in the fit function, and its simply using the values that you pass into the constructor to decide whether or not to split or create a leaf. This results in pruning because in the logic of the program it can either halt, or allow for more leaves.

1. min_samples_split :Per the api, the minimum number of samples required to split a node.
    a. /tree/tree.py
        i. We see this as a parameter in the initialization of the DecisionTree object on line 87 ,100 of /tree/tree.py
        ii. Additionally we see it in the fit function 189 through 205, and this is where the pruning is taking place because as it fits its checking to see if the minimum sample split criteria is met.
        iii. Its then passed into a DepthFirstTreeBuilder 350, 357

2. min_weight_fraction_leaf: Per the api, the minimum weighted fraction of the sum of total of weights required to be a leaf node.
    a. /tree/tree.py
        i. Constructor : 89, 102
        ii. Fit function: 238, 271, 272, 275
        iii. It turn into min_weight_leaf
        iv. Gets passed to the splitter 342
        v. Then in the Tree builder 352, 359