1. **Introduction**

For this assignment, I wanted to compare how the classification algorithms behave when the data sets contain attributes with the same type (e.g. all are numeric), but the output class is provided by a computer, on one set, and by human, on the other one.

The instances of the first data set (sat.arff) are multi-spectral values of pixels in 3x3 neighbourhoods in a satellite image, and the classification associated with the central pixel in each neighbourhood.

The attributes of the second data set (white-wine.arff) define chemical properties/substances recorded in wine (sulphates, pH etc) and the class output specifies the quality of the wine according to at least 3 evaluations made by wine experts.

From the above description of the data sets, one’s intuition is that predictions should be better obtained for the first computer-aided data set, as humans are more error prone.

I have chosen weka software during the assignment.

1. **Approach**

* 2.1 Model Complexity Experiments: run each algorithm to identify its best parameters, where best means the least error/highest accuracy.

The parameters selected are:

* Decision Tree: using **pruning** or not; **confidence factor** used for aggressive pruning
* Boosting: choosing learning **classifier**: one-level decision tree **stump** classifier or a full **decision tree**, with **pruning** or **confidence factor** set; number of bagging iterations
* KNN: **k,** number of nearest neighbor; **weight distance** method to use
* Neural Networks: **learning rate**; **momentum**; number of **hidden layers** and number of **units** in each (in weka only the sigmoid activation function can be used; because of time constraints, I didn’t implement one (e.g. tanh))
* Support Vector Machines: **kernel function**: polynomial (with different exponent) or radial (with gamma parameter);
* 2.2 Compare best algorithms to select the one that can generalize the most
* 2.3 Learning Curve Experiments: gain more insights, understand bias VS variance tradeoffs, overfitting vs underfitting.

1. **Model Complexity Experiments**

As the algorithms are being executed multiple times, with different parameters, using cross validation for training the algorithm takes a lot of time.

Moreover, as the data set is being shuffled before being split into training set and test set, I have observed that the accuracy is almost the same when using/not using cross validation on my data sets.

Each algorithm has an associated “best” class which is configured with the selected parameters and 70%-30% for training and test set size respectively. Further, the “best” class can be executed a number of iterations in order to make the average of the error.

* 1. **Results**

For Decision Trees, the optimal configuration/result obtained after 100 executions is the following:

**sat.arff**

==================DecisionTree Best========================

BestResult [option=dt:pruning=true,cf=0.1, errorRate=13.4127 ==================DecisionTree Best========================

TODO: write nicer

So, a smaller number of nodes is better in this case than the usual number.

**white-wine.arff**

1. **Comparisons of algorithms**
2. **Learning Curve**

For the learning curve, 30% of the data is kept for the test set. The input to the learning curve is comprised of:

* n: #of iterations used to build and evaluate the classifier
* step: the incremental size of the training set; it also provides the initial training set size

Learning curve graph has the training size on the OX axes and training and test error on the OY axis. At each training size, the classifier is being built and evaluated on the training set and test set respectively.

1. **Execution Time**

A data set of the step size is created and used as a training set to build the classifier and a test set to evaluate the performance. The time needed to train and test the same data set is recorded for each algorithm and plotted for the two sets (sat.arff amd wine-white.arff)

It can be seen that ……………….