Comparison of Supervised Learning Algorithms

Qingyu Shen **Author**
Computer Science Undergraduate Affiliation
University of California, San Diego Address
9450, Gilman Dr *email*
gis025@ucsd.edu

Abstract

This paper will analyze four datasets with five supervised machine learning classifiers. Through this analysis, the goal is to compare and contrast the benefits of each classifier and rank them accordingly. All datasets are large and greater than 4500 to improve accuracy when training and testing. Word length of this report is 1637, 6 pages.

1. Introduction

This paper aims to analyze and compare the performance of five supervised learning algorithms, namely: K nearest neighbors (KNN), Decision Trees (DT), Neural Networks (ANN), Logistic Regression (LR), and Random Forests (RF). Datasets used for training and testing of these algorithms were obtained from the UCI Machine Learning Repository. The four datasets used were Cov_type Adults, Abalone, and White wine quality.

2. Methodology

2.1 Learning Algorithms

We begin by creating three partitions for the training and validation set with ratios 90:10, 50:50 and 10:90 for training and testing set size. Next, we perform RandomizedSearch cross-validation with five folds to hyper-tune the parameters for each partition. RandomizedSearch is used over GridSearch due to efficiency and maximization whilst still maintaining near optimum results. For further information regarding the benefits of RandomizedSearch, refer to the paper by Bergstra and Bengio, http://jmlr.csail.mit.edu/papers/volume13/bergstra12a/bergstra12a.pdf. After finding the best parameters for each partition, I calculate the accuracy of predictions for the testing set as well as training and validation accuracies. An average accuracy score is also calculated for all datasets on each classifier to obtain performance ranking.

KNN: 26 values of K ranging from K=1 to K = |trainset| were used in hyper parameter tuning. Euclidean distance is used as the measure of proximity. Euclidean distance is used for the Minkowski metric. Uniform weights for all points in each neighborhood.

Decision Trees: Varying depths from 1 to 10 are hyper-tuned in the cross validation. Gini impurity is used to measure the quality of the split. Minimum number of samples required to split a node is 2.

ANN: Momentum is varied from {0,0.2,0.5,0.9} and hidden units from {1,2,4,8,16,32,64,128} during cross validation. Solver is stochastic gradient descent (sgd). An optimizer in the family of quasi-Newton methods (lbfgs) was also briefly used for comparison purposes although values are not reported due to the time complexity of the full execution. Activation was logistic as is standard.

Random Forests: During cross validation, the maximum size of the feature set was considered at values {1,2,4,6,8,12,16,20} under the condition that the value is less than or equal to size of the feature set. Gini impurity is used to measure the quality of the split.

Logistic Regression: I varied the regularization parameter from 10^-8 to 10^4 by factors of 10 during cross validation. Norm used in the penalization is l2 and uniform weight given to all samples. Liblinear algorithm is used as the solver.

2.2 Performance Metrics

To measure performance, I calculate the average accuracy across three different partitions of training/testing data splits. Other data recorded include training accuracy, validation accuracy, and individual testing accuracy.

2.3 Data Sets

The five classifiers are used on four different data sets all of which are greater than 4500 in sample size. **Please note, Wine Quality data set is NOT the same as the significantly smaller Wine data set.** All the data sets were converted to binary problems. For abalone, nine is the threshold where greater than that value is labelled positive and less than nine is labelled zero. For wine quality, a rating of six is labelled positive while any other rating is given a label of zero. For Cov_type, the largest class in the sample is taken as positive and any other class is labelled zero. For adult, I follow the original binary classification with positive label for greater than 50000 income and zero otherwise. Samples with missing values were removed from the data set prior to classification.

3. Experiment

All test accuracies are obtained from executing RandomizedSearchCV in Sklearn's library five times and averaging the result. Table 1 contains the resulting averages for the respective partitions. An asterisk is placed in each column for the highest accuracy model of that column. The MEAN column represents an averaging across the three partitions used. Random Forests significantly outperforms the other models consistently across all three partitions while Neural Nets ranks last in accuracy among all partitions. We can see that as the training set size decreases, the testing accuracy also decreases. Also, if a model is ranked k, where k is an arbitrary value from 1 to 5 inclusive, in MEAN accuracy then it will also be ranked k across the other partitions.

Neural nets proved to be a relative outlier as the accuracies obtained were lower than the other classifiers. In addition, the outcome of training accuracies for Neural Nets differed significantly from the expected values which should be higher. Further experiments and research suggest that a reason for this is due to the solver algorithm used. Lbfgs, an optimizer in the family of quasi-Newton methods, gave results that were much closer to expected such as training accuracies above 0.9 and near 1. However, the time complexity of lbfgs simply is not feasible at the current stage for data sets so large and therefore sgb (stochastic gradient descent) is used in preference for time constraint purposes. It is important, however, to realize the contrast in results for Neural Networks depending on the solver algorithm used.

In table 2, I identified the average testing accuracy for each classifier on each data set. The experiments to obtain these results are the same as the experiments used to obtain results for table 1 through cross validation of five folds with the result being averaged over five executions. The best score in each column has been highlighted by an asterisk. We can see that while Random Forests has highest overall performance in terms of accuracy, it is not always the most accurate for each individual data set.

Table 1: Testing Accuracy w.r.t. Different Partition Sizes of training vs validation set

	90:10	**50:50**	**10:90**	**MEAN**
DT	0.759	0.745	0.719	0.747
KNN	0.747	0.707	0.647	0.700
RF	0.802*	0.770*	0.725*	0.766*
LR	0.735	0.722	0.712	0.723
ANN	0.660	0.648	0.632	0.647

Table 2: Average Testing Accuracy of classifiers for given Data Sets

	Abalone	**Wine Quality**	**Cov_type**	**Adult**
RF	0.780	0.670*	0.776*	0.836
DT	0.777	0.624	0.750	0.837*
LR	0.800*	0.566	0.747	0.778
KNN	0.772	0.604	0.685	0.740
ANN	0.727	0.555	0.562	0.742

In Figure 1 and Figure 2, the testing accuracy is made visible with ranking of the five classifiers across the four data sets in the order: RF, DT, LR, KNN, and ANN. Figure 3 shows the ranking of each classifier with respect to the individual data sets. Finally, figures 4 and 5 show the training and validation accuracies during cross validation. A clear trend is visible for validation and testing where a higher ratio partition of training vs validation sample size increases the overall test accuracy compared to smaller ratios.

Figure 1: Testing Accuracy

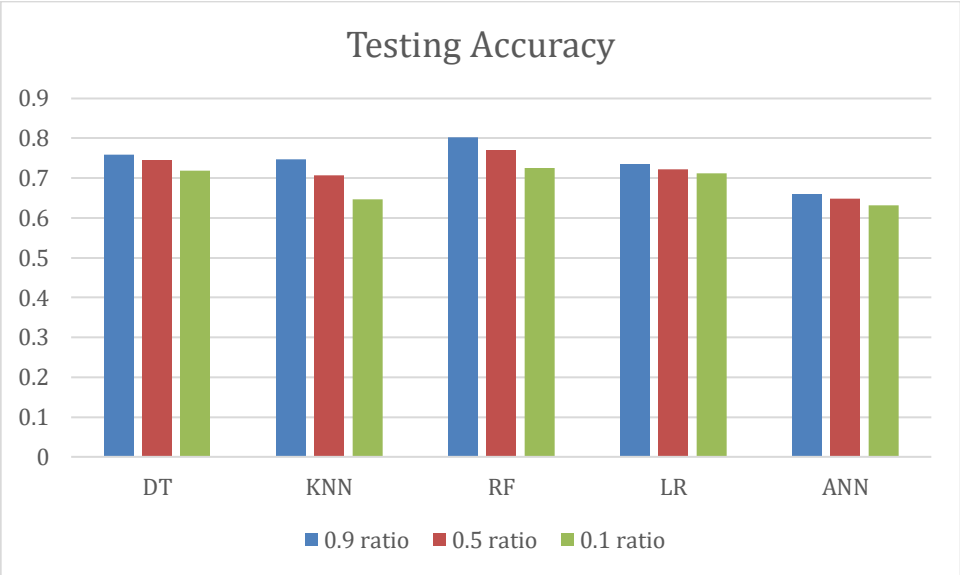


Figure 2: Testing Accuracy Stacked

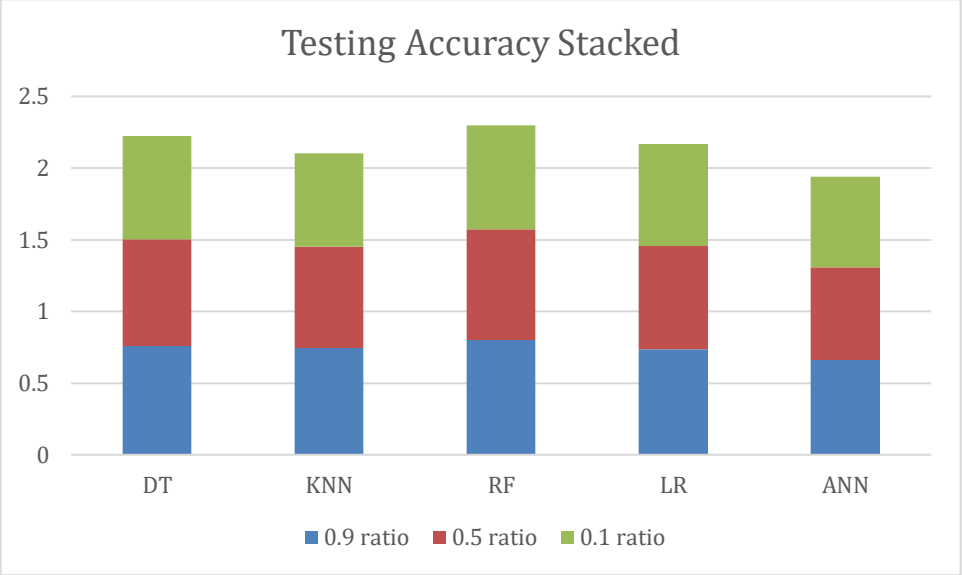


Figure 3: Classifier ranking w.r.t. Data Sets

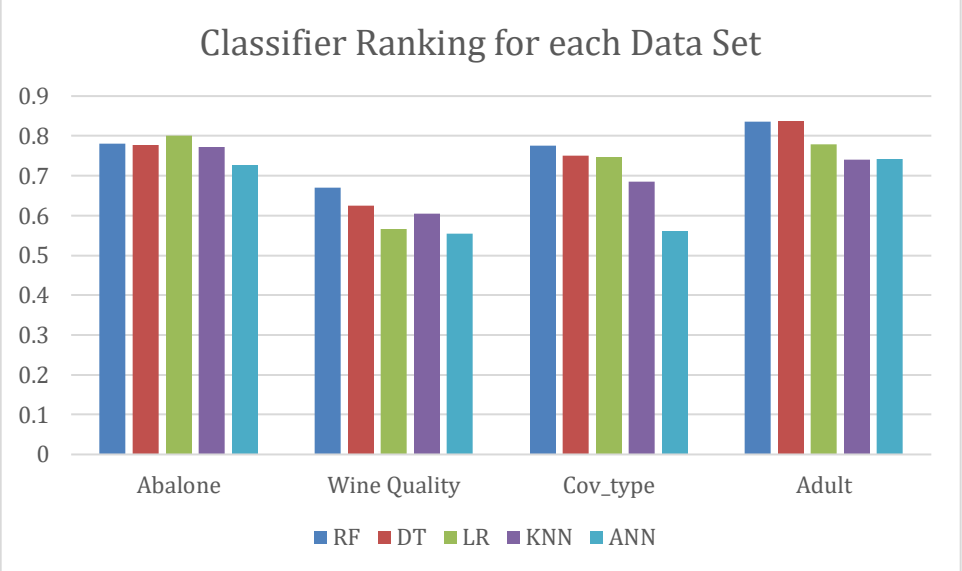


Figure 4: Training Accuracy

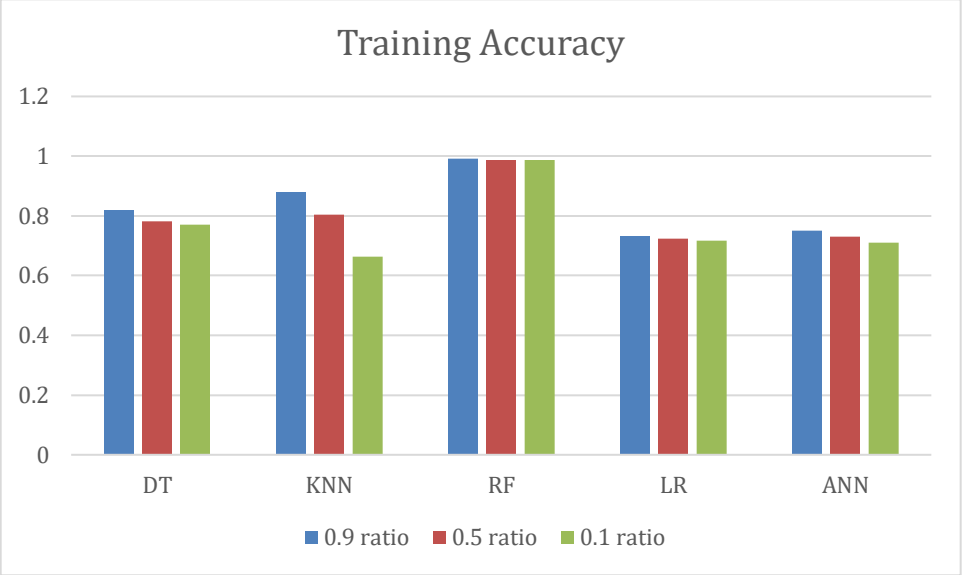
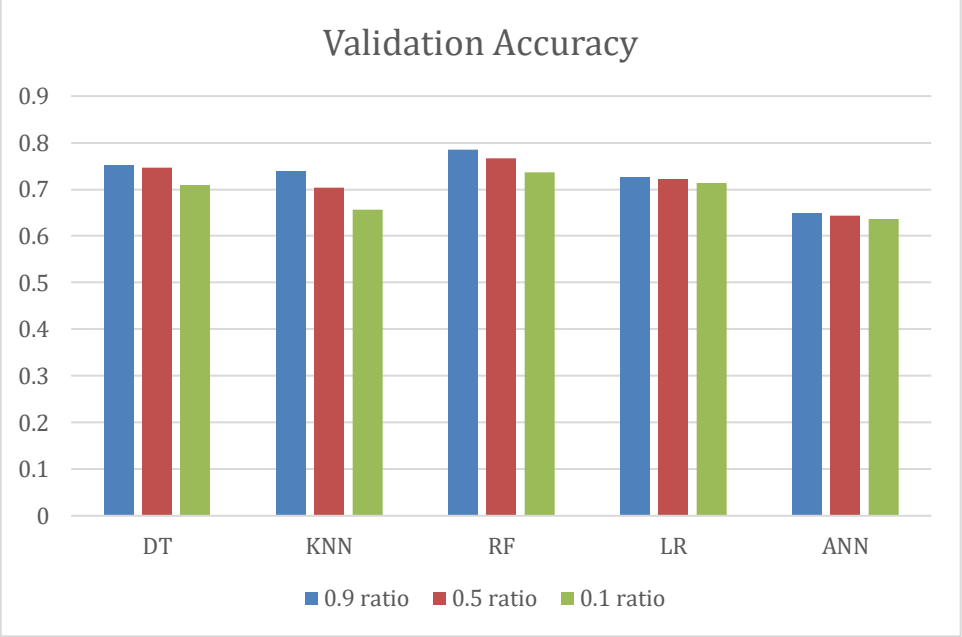


Figure 5: Validation Accuracy



4. Conclusion

There is a clear trend of decrease in accuracy when the partition ratio decreases for train : validation split. These results are consistent with the expected outcome. We can observe that Random Forests perform best overall while ANN performs worst in terms of accuracy. However, as stated earlier, ANN's performance is heavily affected by the solver algorithm. Using lbfgs, ANN may very well be a top performer and have a significantly higher testing accuracy as a result.

Overall the results were fairly consistent with the expected performance outcome. I used many unique hyper parameters, large data sets, and averaged over 5 runs so the results to obtain consistent results. An interesting result obtained is the high performance of Random Forest despite it not being the best classifier across all four data sets. This shows that even the best models perform poorly on same problems. Overall, KNN is the algorithm most affected by partition size and shows a relative sharp decrease in accuracy score when partition ratio is decreased. Random Forest also has consistently high training scores across the 3 partitions although the validation accuracy does show signs of decreasing. Logistic Regression algorithm, on the other hand, shows the most consistent validation accuracies across all 3 partitions.

5. Bonus Points

To obtain bonus points, I implemented 5 classifiers on 4 different datasets. This exceeds the expected 3 classifiers and 3 dataset requirements. I also executed each partition 5 times before calculating the average. This exceeds the required averaging across only 3 iterations.

In addition, I also deserve bonus points because all my datasets are greater than 4500 in sample size. In fact, my adult data set is 12000 and my Cov_type data set is 8000. Such large data sets are time-intensive to compute but obtain more accurate results due to the ability for larger training sets.

Lastly, I further deserve bonus points because I covered data sets both related to and unrelated to the sample paper. Adult and Cov_type data sets are same as in the sample paper, but I also used Abalone and Wine Quality data sets. This provides new results and details that extend beyond the given scope and create new comparisons that previously were not clearly visible. I also analyzed graphs of different axes to show trend lines such as the performance of each classifier with respect to individual data sets and not just overall performance.

6. References

J. Bergstra, Y. Bengio. *Random Search for Hyper Parameter Optimization*. 2012. Online available at http://jmlr.csail.mit.edu/papers/volume13/bergstra12a/bergstra12a.pdf.

R. Caruana, A. Niculescu-Mizil. *An Empirical Comparison On Supervised Machine Learning Algorithms.* Online available at https://www.cs.cornell.edu/~caruana/ctp/ct.papers/caruana.icml06.pdf

Blake, C., Merz, C. (1998). UCI repository of machine learning databases.

Acknowledgments

Professor Zhouwen and TAs of COGS 118A Winter 18’