Supervised Learning: Analysis of Adult Income & Wine Quality Dataset

Woojae Kim (jkim7657@gmail.com)

1. Introduction

The purpose of this report is to explore data-driven machine learning techniques, specifically utilizing supervised machine learning algorithms. Five classification algorithms—**decision tree, neural networks, AdaBoosted decision trees, support vector machines, k-nearest neighbors**—were implemented on two different datasets: adult income dataset and wine quality dataset. After some data preprocessing, the algorithms were trained on both datasets through k-fold (k=10) cross-validation method. During the training, learning curves and complexity curves were plotted. Then the trained models were tested on the unseen, held-out testing sets (20% of the entire datasets), and the resulting performances were analyzed utilizing various metrics.

2. Dataset Description

2.1 Ault Income Dataset:

This is an abbreviated version of 1994 census bureau dataset. From the original census data, Ronny Kohavi and Barry Becker extracted a set of reasonably clean records using the following conditions: ((AAGE>16) && (AGI>100) && (AFNLWGT>1) && (HRSWK>0)) **[1, 2]**. The classification task is to predict whether a person makes over $50K a year or not, based on 15 attributes (6 continuous or 9 categorical). These attributes describe personal information, including age, education level, marital status, occupation, race, sex, working hours and native country.

The target attribute values (>50K and <=50K) were transformed into a Boolean values (0 and 1, respectively). The resulting dataset, including target attribute, had a dimension of 30,704 x 51. The classes were highly imbalanced (negative label: 75% / positive: 25%).

2.2 Wine Quality Dataset:

This data was a result of a chemical analysis of Portuguese “Vinho Verde” white wines [2]. The dataset contains the quantities of 11 different chemical properties: fixed acidity, volatile acidity, citric acid, residual sugar, and etc. The target variable is *quality* of wine, which is an integer score between 0 (poor) and 10 (excellent).

This multi-class classification problem was transformed into a binary classification problem by defining “regular” wines (negative label with score < 7) and “good” wines (positive label with score >= 7). Since there were much more regular wines than good wines, the target labels were highly imbalanced (negative label: 78% / positive: 22 %). The resulting dataset had a dimension of 4,898 x 12.

2.3 *Why are they interesting?*

Both problems are binary classification problems with very similar target label distributions which are imbalanced. The adult dataset has the number of samples that is six times than that of the wine dataset. The number of attributes for adult dataset, however, is about five times than that of the wine dataset. Together, the two datasets should shed some light on the relationship between the problem complexity (which for now is assumed to be the attribute size) and learnability (the number required training samples to learn a hypothesis).

3. Problem Formulation

3.1 Description of Classification Tasks:

The goal of the classification tasks for both datasets is to come up with a best classifier which is able to “correctly”—will clarify later—classify the target labels, within a reasonable amount of training time which

scales well with the size of the training set. Unlike other imbalanced classification tasks, such as spam or disease detection problems, there is no specific emphasis on a certain label. For example, correctly classifying a person with low income (>$50K) is as important as classifying someone as high income (<= $50K). The same logic applies for wine quality predictions. In this context, "correctly" means the following: When a classifier makes a prediction for a given subject, the resulting label has high probability to be right.

3.2 Evaluation Metric:

Since both datasets are highly imbalanced, simple accuracy will not be a good evaluating metric. In both datasets, a classifier which consistently predicts the mode of the target attribute will result in over 75% accuracy. Precision is a great metric for a highly imbalanced dataset. Nonetheless, by definition, precision has bias towards the positive label in a binary classification setting and was not chosen as a metric.

True Positive Rate (TPR or Recall) is a good metric because it fits our classification goals. Loosely speaking, TPR represents the proportion of true positive identified correctly. Nonetheless, TPR alone will not be good enough. Since we equally value making correct predictions for negative labels, we can consider TPR and False Positive Rate (FPR) together and use Area Under the ROC Curve (AUCROC or AUC). One can generally think of AUC as the probability that the classifier will correctly classify a given subject into one of the two labels. Also, AUC is insensitive imbalanced class labels.

As a result, AUC was chosen as the evaluating metric, and all classifiers and its hyperparameters were tuned to maximize AUC.

4. Analysis Method

4.1 Data Preprocessing:

For adult income dataset, each continuous attribute was standardized to have zero mean and unit variance. In general, categorical variables were one-hot encoded. *Fnlwgt* attribute, which was included only for the sampling purpose, was completely deleted. Few examples with unknown (*"?"* category) entries were deleted.

Some of the categorical attributes had too many minor categories with small frequencies. To avoid the curse of dimensionality, if deemed appropriate, some minor categories were merged into a single, combined category. When combining these minor categories, domain knowledge and statistical impacts were considered. For example, *education* attribute had 16 categories, but 1-4th, 5-6th, 7-8th, 9th, 10th, 11^{th,} and 12th categories were combined into *"below-HS"* category, totaling 15% of the entire dataset. Likewise, *native country* attribute initially had 43 categories, but all non-US countries were merged into Non-US category, totaling 10% of the entire dataset (90% were US).

For wine quality dataset, all attributes representing the chemical properties of a given wine were continuous numerical variables. These attributes were standardized to have zero mean and unit variance.

4.2 Model Selection and Testing:

K-fold cross-validation (CV) was utilized throughout the model selection process. The original datasets were split into train (80%) and test (20%) sets, and 10-fold CV was performed on the train set for training and hyperparameter tuning. For each supervised algorithm, an extensive grid search was utilized to find the best hyperparameters which maximized AUC.

During the training and hyperparameter tuning, learning curves (x-axis: number of training samples, y-axis: AUC) and complexity curves (x-axis: chosen hyperparameter, y-axis: AUC) were plotted to check convergence and performance of each learning algorithms.

After the best hyperparameters were determined, all classification models were trained on the entire train set (80%), and the models were then tested on the unseen, held-out test set (20%). The resulting performance metrics were compared.

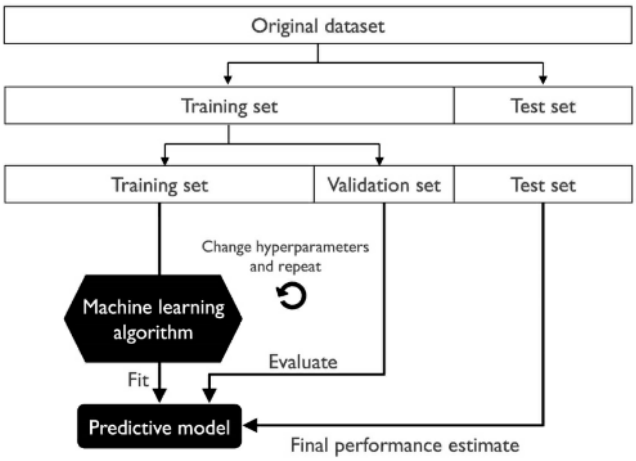


Figure 1. Summary of model selection process, adapted from [3]

5. Results

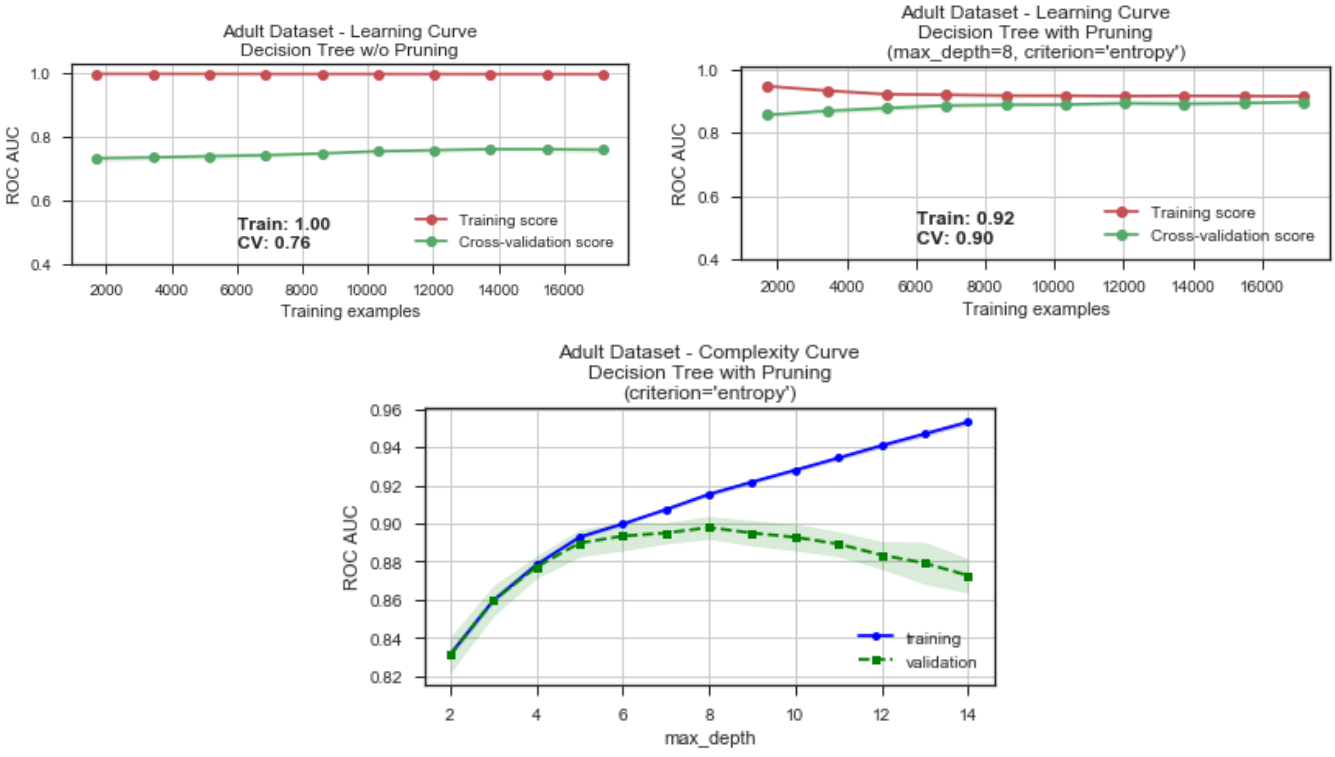
The following sections present the experimental results, including but not limited to, learning curves and complexity curves for **decision tree, neural networks, AdaBoosted decision trees, support vector machines, k-nearest neighbors** algorithms on adult income and wine quality datasets. For brevity of the report, not all complexity curves (for all available hyperparameters) were presented.

5.1 Training and Hyperparameter Tuning - Adult Income Dataset

In general, most models exhibited great balance between variance and bias. Inspecting the learning curves, it is unlikely that any additional data will help to further improve the AUC scores for all models (thus converged).

Decision Tree:

For decision tree, Classification and Regression Trees (CART) algorithm by Brieman et al. was utilized **[4]**. CART algorithm is similar to C4.5, but it supports numerical target variables and does not compute rule sets. Like many other decision trees, it has an inductive bias for trees with shorter depths. CART constructs binary trees using the given attribute and threshold which maximizes information gain at each node.

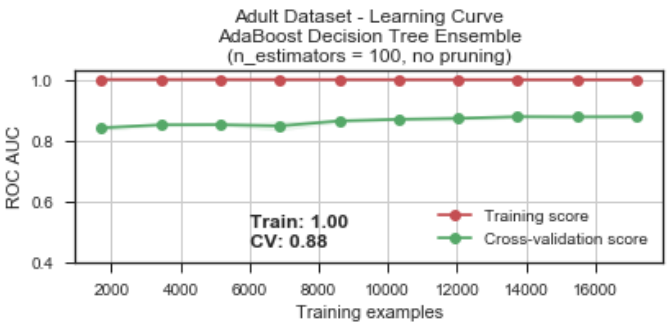


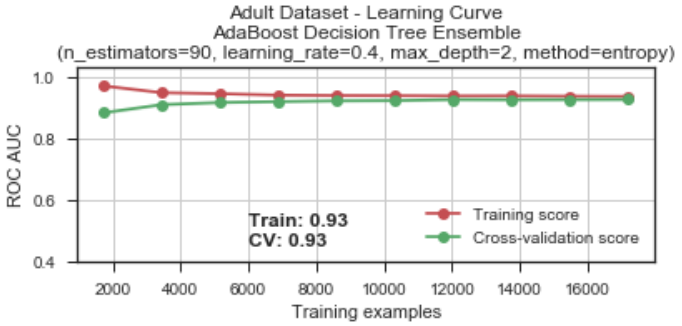
Nonetheless, without pruning, decision trees are prone to overfitting the train set and will not generalize. Overfitting can be mitigated by "pruning" method. In Scik-kit Learn, the maximum depth of the tree (*max_depth* hyperparameter) was tuned via 10-fold CV. Learning curves above illustrate the effect of pruning on the evaluation metric, AUC.

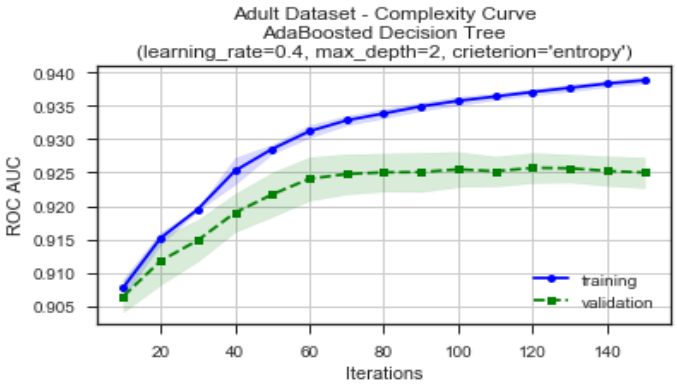
After an extensive grid search, hyperparameter tuning, maximum tree depth=8 and entropy split criterion were utilized (over gini index). One can see from the complexity curve that max_depth=8 is optimal for the given dataset. As the max_depth further increases, the decision tree's model complexity also increases and fails to generalize to the CV set, implied by the widening gap between training and validation scores.

AdaBoosted Decision Trees:

Boosting algorithm introduced by Freund and Schapire **[5]** was utilized to form an adaboosted decision trees. For each boosting iteration, the training examples that were incorrectly classified by the boosted model will have their weights increased.







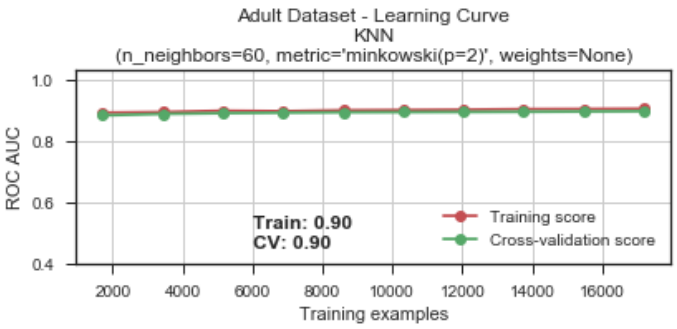
Because a boosted model consists of "weak" learners, one can be more aggressive about pruning of individual decision trees. Therefore, one needs to optimize both boosting model and base weak learner hyperparameters. After extensive grid search, the number of iterations =90 (n_estimators), learning rate=0.4, decision tree depth=2, and entropy criterion were used.

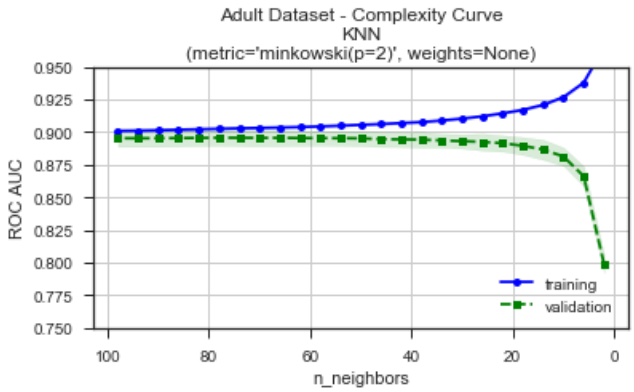
Two interesting observation can be made. If "strong" learners are used (no pruning), the model tend to overfit, regardless of the number of training examples provided. Also, observe in the complexity plot that the AUC score does not drop as the number of iteration increases, which implies that there is no overfitting.

K-Nearest Neighbors (KNN):

Likewise, KNN hyperparameters were tuned. KNN is a "lazy" algorithm which does not require any training. If any, it just needs to store all available training examples in the memory. Once queried, it makes a prediction based on pre-specified set of hyperparameters, such as number of nearest neighbors (k), distance metric, and point weighting method (based on distance). After grid search, k=60, Euclidian distance metric (Minkowski with p=2), and no weighting were determined to be optimal for this dataset. One can see from the complexity curve that there is essentially no significant gain in further increasing n_neighbors over k=60, as it will only increase the query time.

Learning curve for KNN further corroborates the fact that this is a lazy algorithm. Since there is no learning, the number of training examples after a certain threshold does not affect the AUC score.



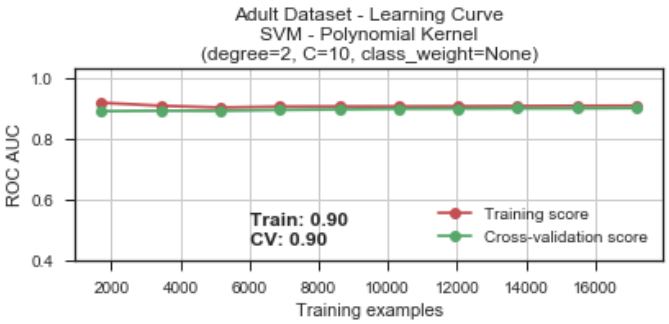


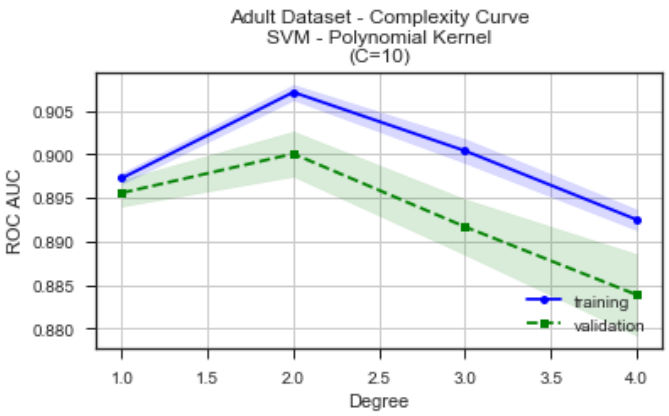
It should be noted here that the specific search algorithm of the nearest neighbors was optimally chosen between the brute force method (with $\boldsymbol{O}(DN^2)$ where D and N are number of dimensions and samples), K-D tree with $\boldsymbol{O}(D\ N)$ **[6]** and ball tree with $\boldsymbol{O}(D\ \log(N))$ for $D > 20$ **[7]**. The specific algorithm for the model selection can be found in **[8].**

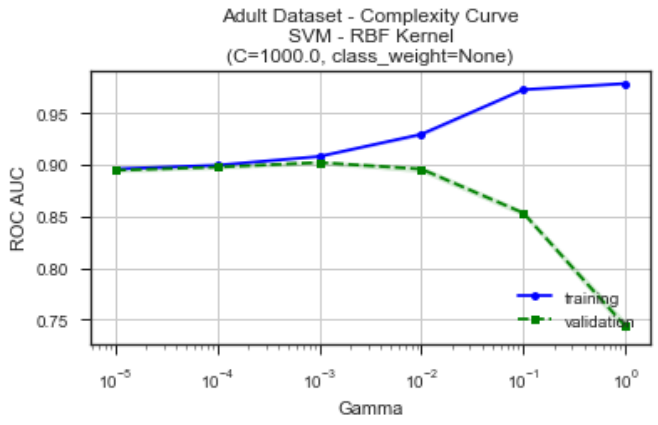
Support Vector Machine (Polynomial and RBF Kernels):

For SVM, two kernels were used: polynomial with degree=2 and RBF kernel. The following formulas were utilized for the kernel functions:

- Polynomial: $(\gamma\langle x, x'\rangle + r)^d$ where d = 2
- RBF: $\exp(-\gamma\|x - x'\|^2)$ where gamma(γ) = 0.001







Likewise, SVM model hyperparameters for both kernels were determined through extensive grid search. For polynomial kernel, degree of 2 was determined to be optimal, as one can see from the complexity curve above. The regularization parameters (C) for polynomial and rbf kernels were determined to be C=10 and C=1000, respectively.

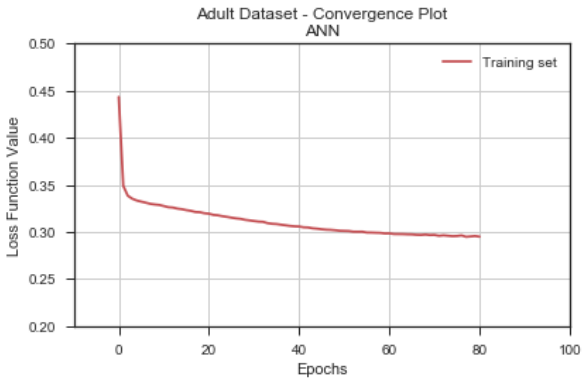
Specific mathematical formulation of the implemented SVM can be found in **[9]-[10]**.

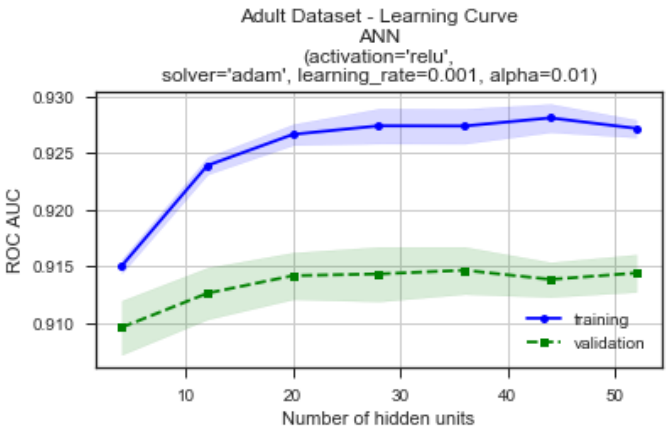
Artificial Neural Networks (ANN):

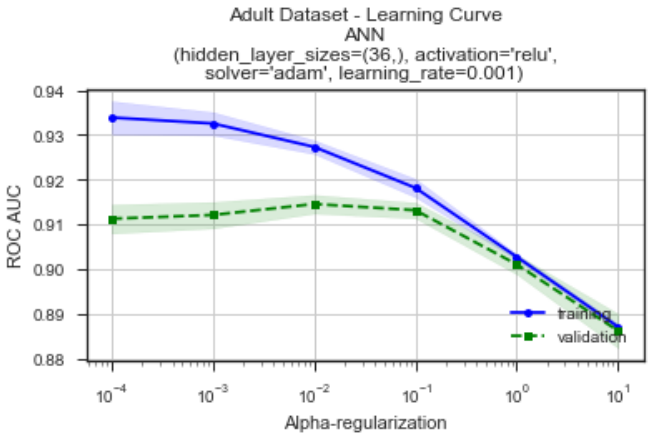
For ANN implementation, multi-layer perceptron (MLP) classifier with a single hidden layer was utilized. During the training the MLP optimizes the log-loss function using a stochastic gradient descent method called "Adam" (see **[11]**). As the ANN continuously adjusts its weights, it goes through the entire dataset several times in "batches" until there is no significant change in the log-loss function value. Each run through the entire train set is call an "epoch."

A typical learning curve does not suffice to show a convergence of ANN algorithm. By analyzing a convergence plot (see log-loss vs. epoch plot below), one can clearly visualize the rate of convergence of the model as well as the required training time.

After a grid search, the following optimal hyperparameters were set: number of hidden units=36, activation function=Rectified Linear Units (ReLU), learning rate=0.001 and L2 penalty (alpha) = 0.01. As one can see from the complexity curves below, increasing the number of hidden units in the hidden layer does not affect AUC score. This shows the effect of applied L2 regularization. One may choose a higher number of hidden units, but L2 regularization effectively reduces the number of hidden units in the layer by decreasing the magnitudes of weights connected to the irrelevant hidden units. Alpha-regularization complexity plot, one can also see that alpha=0.001 is also optimal.







For specific mathematical formulation of the implemented algorithm, see **[12]–[13]**.

5.2 Training and Hyperparameter Tuning - White Wine Dataset

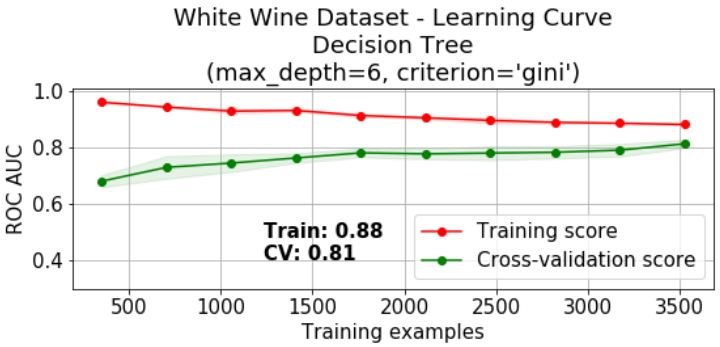
Following similar methodologies delineated in Section 5.1, training and hyperparameter tuning of supervised algorithms for white wine dataset were performed. For each algorithm, a learning curve and a complexity curve which clearly shows the optimality condition of a chosen hyperparameter were supplied. For brevity of the report, the implementation details of each algorithm, which was delineated in Section 5.1, will not be repeated.

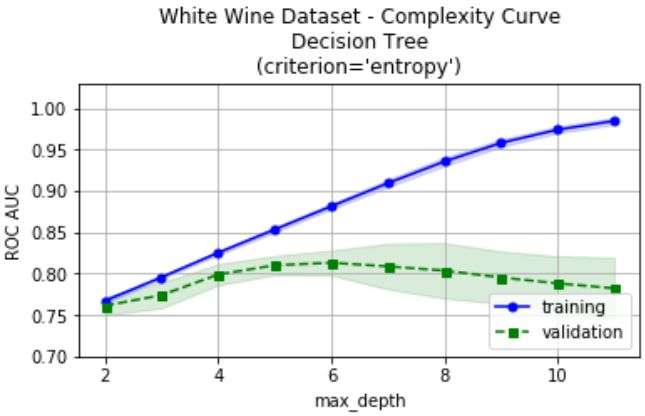
Some models showed reasonable convergence (KNN and ANN), but most models suffered from high

variance problem, and more training examples will likely to help here.

Decision Tree:

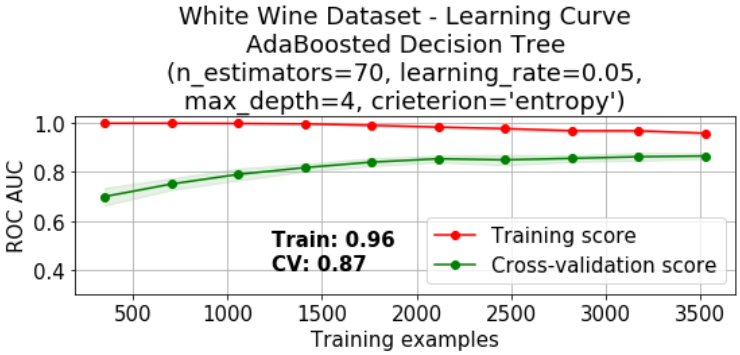
Despite relatively high AUC scores (low "error"), the gap between training and CV scores from learning curve imply a high variance problem. Since the gap is continuously narrowing as the train size increases, additional train cases are likely to help.

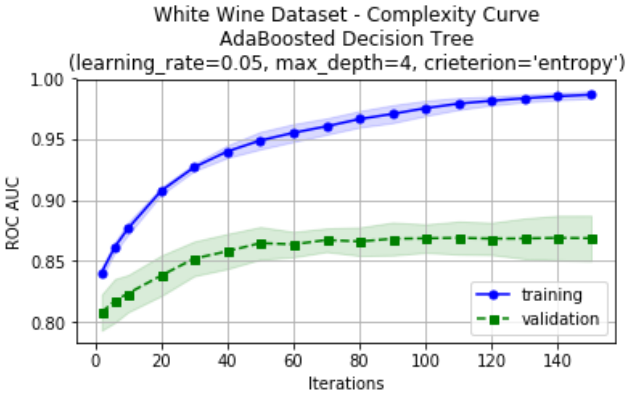




AdaBoosted Decision Trees:

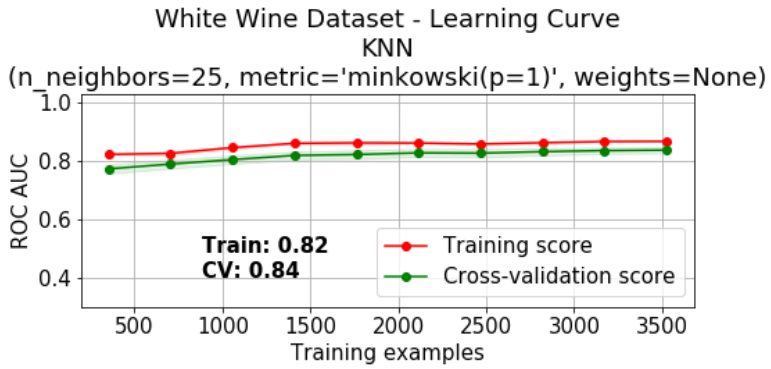
Likewise, the model is suffering from a high variance problem, and more training samples will likely to help.

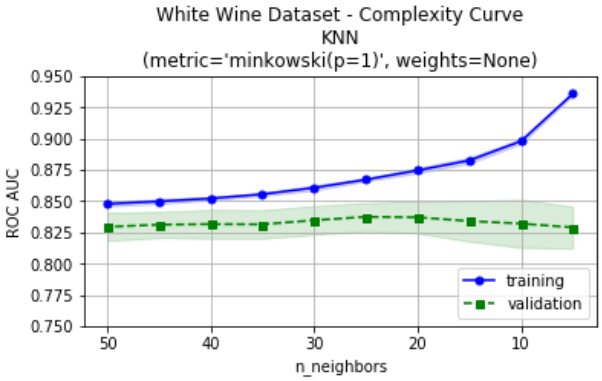




K-Nearest Neighbors (KNN):

KNN, being a lazy model, does not require training, and the performance of classifier is unlikely to change after its hyperparameters have been tuned for certain size of training examples.



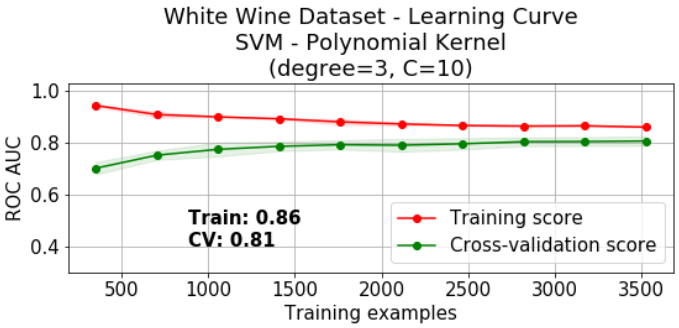


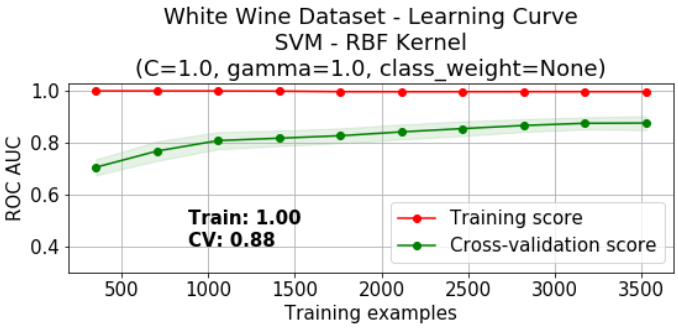
Support Vector Machine (Polynomial and RBF Kernels):

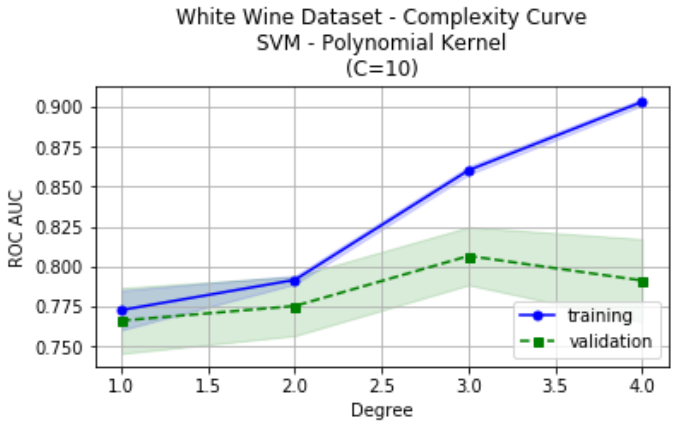
Interestingly, the SVM with rbf kernel suffered from significant overfitting problem after an extensive grid search to tune its hyperparameters. Clearly, the rbf SVM is eagerly learning the noise present in the training samples and it is failing to generalize to CV sets. Compared to rbf SVM, the SVM with polynomial kernel

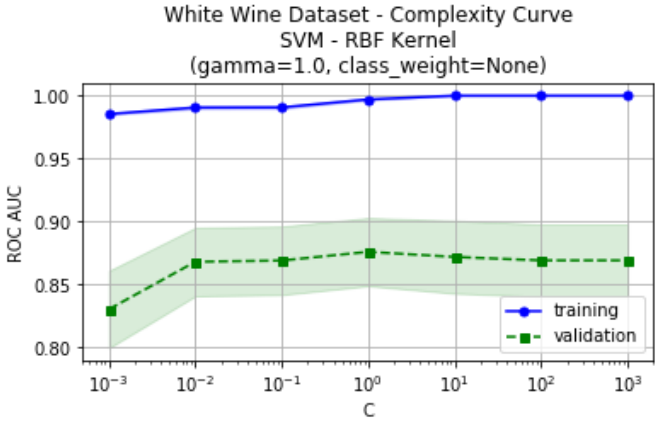
(degree=3) suffers from a minor variance problem.

In both cases, adding additional training samples is likely to help the learning processes. For rbf SVM, it is interesting to note that the model with the highest AUC was not the model which generalizes well. Even the model is clearly suffering from an overfitting problem, it still maximized AUC for the CV sets. To make rbf SVM better, one might decrease alpha to enforce higher level of regularization.



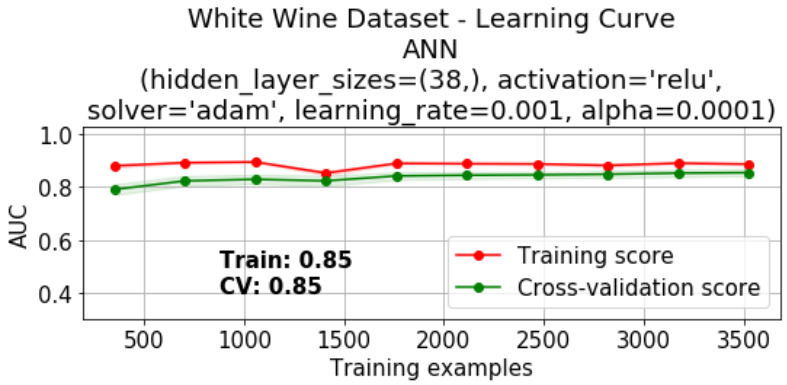


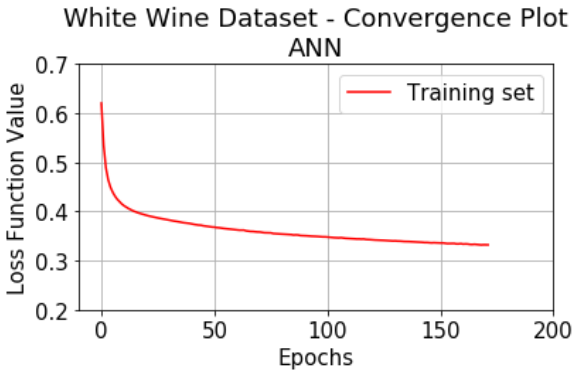


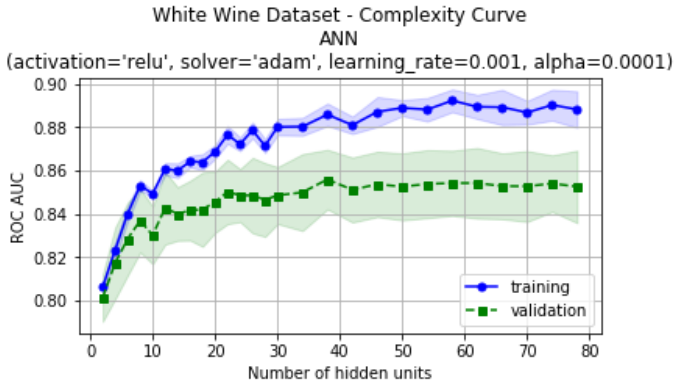


Artificial Neural Networks (ANN):

After hyperparameter tuning, ANN exhibited consistent behavior. Supplying the model with additional training set did not enhance the AUC score after a certain point, implying that what remains is the irreducible error. Convergence plot was also utilized to monitor the convergence of the ANN algorithm.







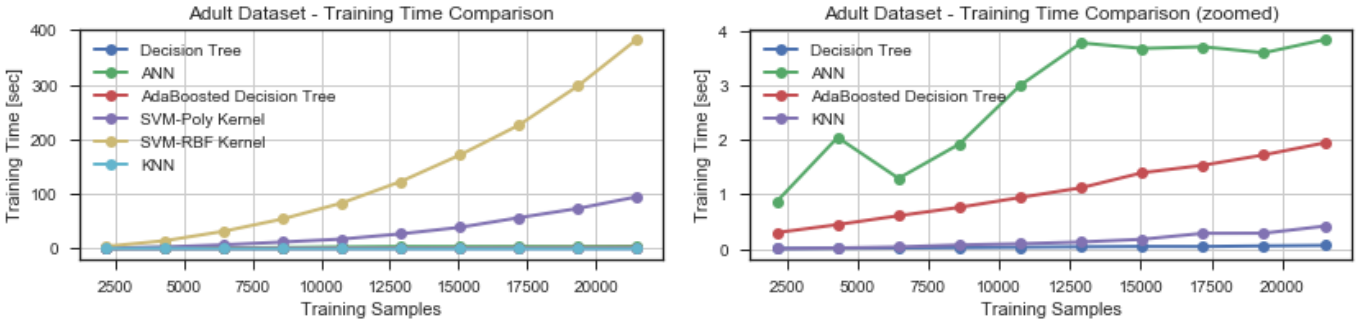
6. Discussion

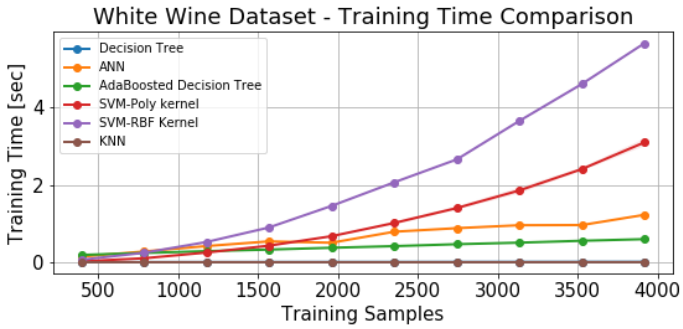
In order to pick the "best" classifiers—as defined in Section 3.1—scalability of training and testing time as well as generalization performance on unseen data should be analyzed.

6.1 Scalability of Algorithms (Training & Testing):

In addition to the AUC score, the training time scalability of the supervised learning algorithms was analyzed to determine the "best" learning algorithm for the classification problems. For each algorithm, the training and testing time (in seconds) were plotted against increasing number of samples (see below). Since the general trend is the same, the following discussion will be mostly based on the adult dataset, which has significantly higher number of training samples.

As one can clearly see, SVM classifiers exhibited significantly higher training time that were orders of magnitude of higher than the other algorithms. As the number of training size increased, this gap further widened. Due to the higher complexity associated with RBF kernel function calculation, RBF kernel performed even worse than the polynomial kernel. This result can be explained by the fact that the core of SVM algorithm implemented here is a quadratic programming (QP) problem, separating support vectors from the rest. The QP solver used in this algorithm scales between $\boldsymbol{O}(DN^2)$ and $\boldsymbol{O}(DN^3)$.

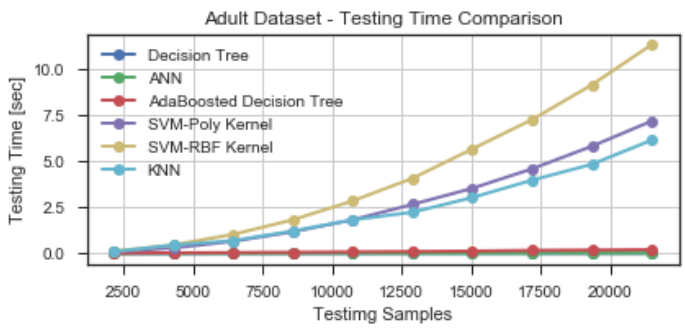


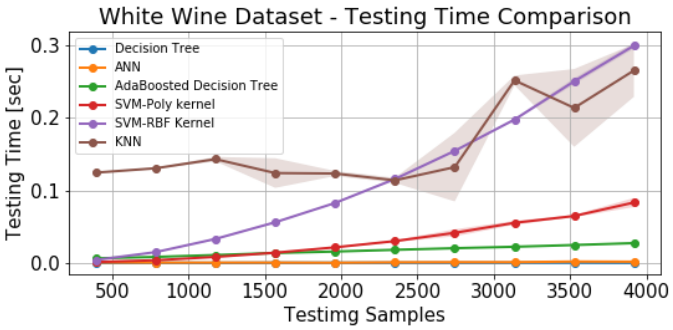


Once the SVM classifiers are excluded from the plot (see zoomed plot above), all other classifiers— including decision tree, adaboosted trees, ANN and KNN—scales fairly well with a relatively large size of training set (over 20,000). Even though some algorithms (such as decision tree) scale better than the other (adaboosted trees and ANN), these discrepancies can be neglected in practice when compared to SVM classifiers.

Even though there is an increasing trend in training time of ANN as the number of samples increases, the behavior is rather erratic. This is due to the required internal iterations. For a given sample set size, ANN may converge at different rates. It is also interesting to observe that KNN—which is a lazy algorithm—requires more training time than decision tree. This may be due to the internal algorithm to facilitate the selection of distance search algorithm, given training samples.

As of training time, SVM classifiers and KNN do not scale very well with the increasing test sample size. The result for KNN and SVM Poly Kernel make sense especially if one consider the aforementioned complexity of KNN ($\boldsymbol{O}(DN^2)$ in the worst case). Apart from those SVMs and KNN, all other algorithms scale very well with the increasing test size, and their testing time won't be an issue in practice.





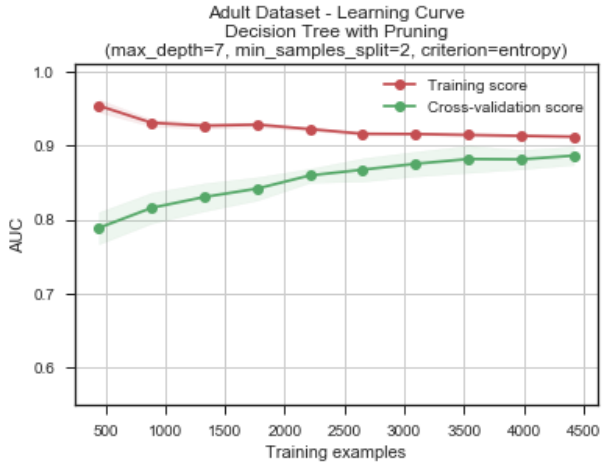
6.2 Generalization (Performance on Unseen Data):

Finally, the tuned classifiers were trained on the entire train set (80%) and tested on the held-out, unseen test set (20%). Confusion matrices, as well as ROC curves for all algorithms were plotted for both datasets (see next page).

ROC Curves:

Based on the ROC curves, the AUC scores for the adult dataset is consistently higher than those of the wine dataset. The number of attributes in the adult dataset (50) is about five times than that of the wine dataset (11). Coincidentally, The ratio between the number of provided training samples for two datasets is in the similar realm (about 20,000 for the adult set and about 3500 for the wine dataset). Nonetheless, only the algorithms trained on the wine dataset suffered from variance problems. Clearly, the required number of training samples to learn a proper hypothesis does not scale linearly with the problem complexity (the number of attributes).

For an additional simple testing, the adult dataset was downsized by (20%) and then tested using 10-fold CV. The following plot shows the result for decision tree. Around similar training sample size (about 3,500), one can clearly see that the classifier had a high variance problem and exhibits similar learning curves as that of the full-sized wine dataset. Relatively high dimensional problems can be successfully solved by most supervised learning algorithms if provided with sufficiently large training set.

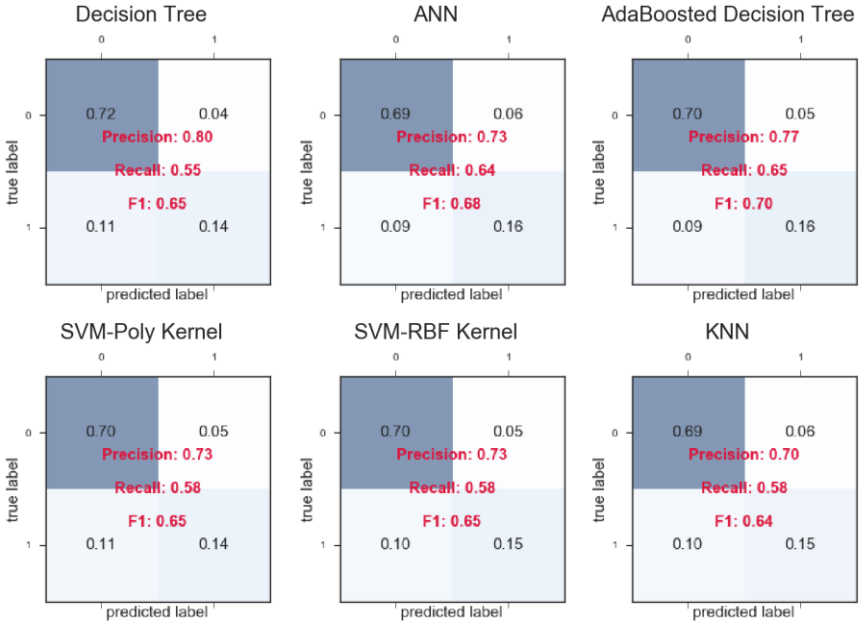


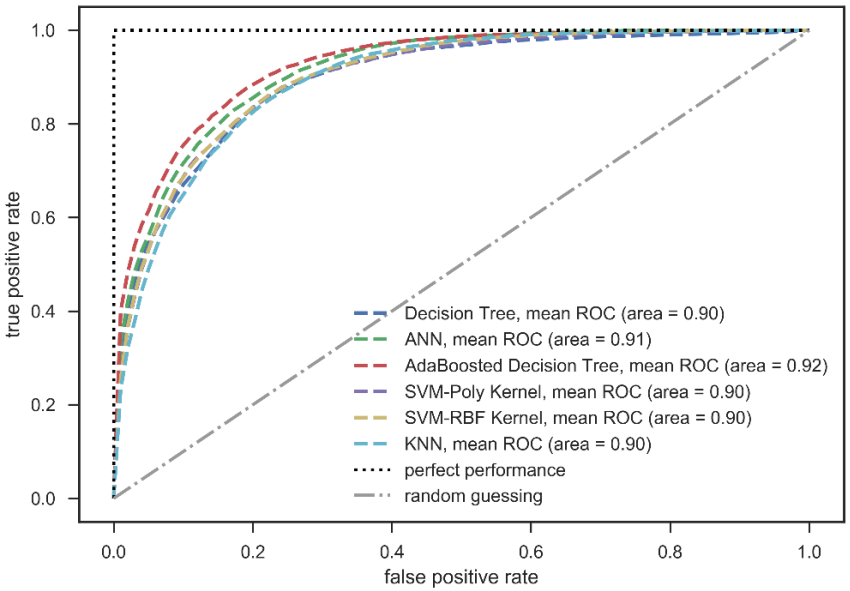
If provided with enough training samples, one can see from the adult ROC curve that ANN and adaboosted decision trees slightly outperformed the other algorithms. With the wine dataset, given limited number of training examples, SVM classifiers and ANN exhibited excellent behaviors.

Confusion Matrix:

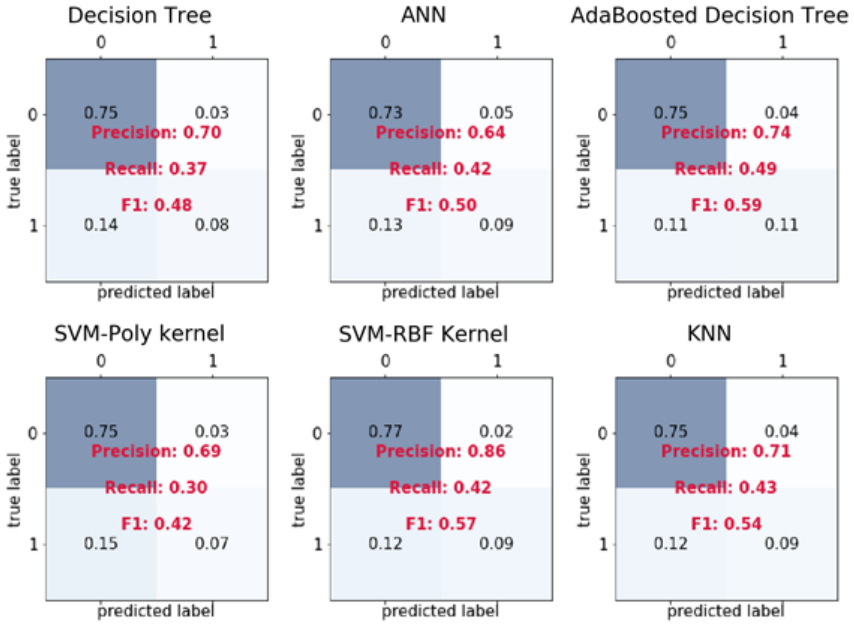
Another interesting result was the actual predictive behaviors of the supervised algorithms on the unseen dataset, which is summarized in confusion matrices (see next page). For the adult dataset, SVM classifiers (poly and rbf) and KNN produced almost identical confusion matrices. This is consistent with ROC results as SVMs and KNN all share the same AUC score. The fact that ANN and adaboosted trees (with high AUCs) have similar matrices with higher precision and recall is also consistent with the ROC result.

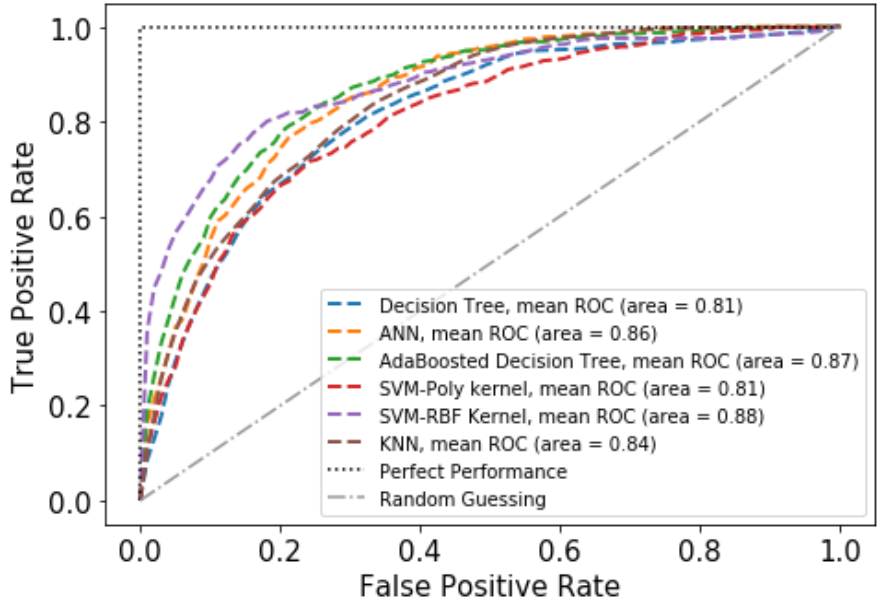
Adult Income Dataset Results





White Wine Dataset Results





The f1 scores of the algorithms exhibited linear relationship with AUC scores. Another interesting result is the high precision score of decision tree classifier, which may be due to the fact that categorical variables are more easily represented by decision trees.

As of white wine dataset, the most interesting aspect was the high precision score of SVM with rbf kernel. This can be attributed to the fact that SVM—provided with the right kernel— can fit complex hypothesis very well even with limited number of training samples.

The f1 scores again showed some correlation between AUC scores, which implies that f1 score could have been a reasonable metric for evaluation.

7. Conclusion

In this report, five different classification algorithms were implemented on two different datasets with similar, imbalanced target label distributions. Each learning algorithm's hyperparameters were tuned via grid search, which was validated by complexity curves.

The tuned models were trained on the training set (80%) with 10-fold CV. Learning curves were also plotted with the fixed hyperparameters to assess the convergence of the learning algorithms and troubleshoot potential bias/variance problems.

The relationship between problem complexity (no. of attributes) and learnability (no. of training samples to learn a hypothesis) were examined through learning curves, without explicitly calculating VC dimensions. It was found that the classifiers for the adult dataset, despite the adult dataset's higher problem complexity, consistently achieved a better convergence and learning behaviors.

Furthermore, the scalability of the chosen classifiers was analyzed by measuring the wall clock times for training and testing operations over various sample sizes. It was found that train and test time for SVM classifiers (polynomial and RBF) scaled very poorly with respect to a large sample size (over 10k). As expected, KNN took no time to train, but its testing time scaled very poorly—almost comparable to SVM classifiers—with increasing sample size.

Finally using the unseen, test sets (20%), performance metrics of all classification algorithms were compared. It was found that adaboosted decision trees and ANN consistently performed well on both datasets. SVM classifiers also performed fairly well in both datasets, but they were more sensitive to the complexity of the given problems and to the choice of kernels. Based on the results, it can be inferred that AUC is a reasonable evaluation metric which is suited for our classification task.

If the time scalability aspect is factored into the selection of "best" algorithm, along with AUC score, the best algorithms vary for the two datasets. For a dataset with large samples, adaboosted decision trees will provide the best balance between computational costs and performance. If one is provided with a relatively small dataset (less than 5k), it makes more sense to use classification algorithms that can fit more complex hypotheses, such as SVM classifiers and ANN because computational costs are relatively small.

References

[1] Ron Kohavi, "Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid", *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 1996.

[2] https://archive.ics.uci.edu/ml/datasets/wine

[3] Raschka, Sebastian. Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow, 2nd Edition (Kindle Location 3544). Packt Publishing. Kindle Edition.

[4] L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. Wadsworth, Belmont, CA, 1984.

[5] Y. Freund, and R. Schapire, "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting", 1997.

[6] Bentley, J.L., "Multidimensional binary search trees used for associative searching", Communications of the ACM (1975)

[7] Omohundro, S.M., Five balltree construction algorithms", International Computer Science Institute Technical Report (1989)

[8] http://scikit-learn.org/stable/modules/neighbors.html#neighbors

[9] C. Cortes, V. Vapnik, "Support-vector networks", Machine Learning, 20, 273-297 (1995).

[10] http://scikit-learn.org/stable/modules/svm.html#svm-mathematical-formulation

[11] Kingma, Diederik, and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

[12] Hinton, Geoffrey E. "Connectionist learning procedures." Artificial intelligence 40.1 (1989): 185-234.

[13] http://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html