2019 Spring CS7641 Assignment 1: Supervised Learning

Zheng Fu (zfu66@gatech.edu)

Abstract: In this project 5 classifiers (decision tree, SVM, neural network, boosting and KNN) were preformed to predict human wine taste preferences on the basis of 11 analytical tests. The performances of these classifiers were evaluated using ROC and learning curves along with other metrics. A white noise dataset is also generated to gauge the abilities of these classifiers on dealing with false positive rate.

I. Introduction

For dataset 1, the red wine quality dataset was selected from Kaggle. This dataset is related to the Portuguese "Vinho Verde" wine taste. The independent variable of this dataset is the human wine taste preferences scaled from 0 to 10. The dependent variables are 11 features based on physicochemical tests. Since the independent variable is continuous, the interesting part of this dataset is the user might set up an arbitrary cutoff to define “Good” or “Bad” wine quality, and such cutoff may have impact on classifier performance. Also it is a small dataset with only 1599 observations, which means I could focus more on tuning up the classifiers rather than being worried about the time complexity of the algorithms themselves. For dataset 2, it is similar with dataset 1 and the only difference is the labels of independent variable were randomly permutated. Hence the entire dataset becomes a white noise. Such dataset is very useful in real world machine learning tasks, since beside the accuracy of the classifier, we are also interested in the false positive rate in prediction (i.e. cancer diagnosis). If a machine learning model has good performance on white noise dataset, it is necessary to perform a further investigation on the algorithm as well as the data itself.

II. Methods

1. Data pre-processing

The wine quality dataset was directly downloaded from Kaggle website. Here I used quality score = 6.5 as a cutoff to assign “Good” and “Bad” labels to the independent variables. Then the categorical labels were transformed to numerical variables via sklearn.preprocessing.LabelEncoder function (‘Good’ = 1 and ‘Bad’ = 0). Note that using quality score = 6.5 (> 6.5 is ‘Good’ otherwise ‘Bad’) as the threshold will generate 217 ‘Good’ samples and 1382 ‘Bad’ samples that makes the dataset imbalanced. Numpy.random.permutation function was applied to randomly permute the independent variables’ labels and generate dataset 2. All dependent variables were standardized by removing the mean and scaling to unit variance (Z ~ N(0, 1)). Both dataset 1 and 2 were split to 80% training set and 20% for testing set.

2. Classifier

Decision trees: sklearn.tree.DecisionTreeClassifier was applied to perform the binary classification and Gini impurity was used to measure the quality of the split. Three parameter sets need to be optimized: the number of data points placed in a node before the node is split (min_samples_split: from 0.1 to 1.0 step 0.1), the maximum depth of the tree (max_depth: from 1 to 20 step 1) and the number of features while searching the best split (max_feature: from 1 to 11 step 1). Note that scikit-learn decision tree classifier currently does not support pruning mechanism, so here I vary the min_samples_split parameter to better guide the training and get rid of the problematic leaves.

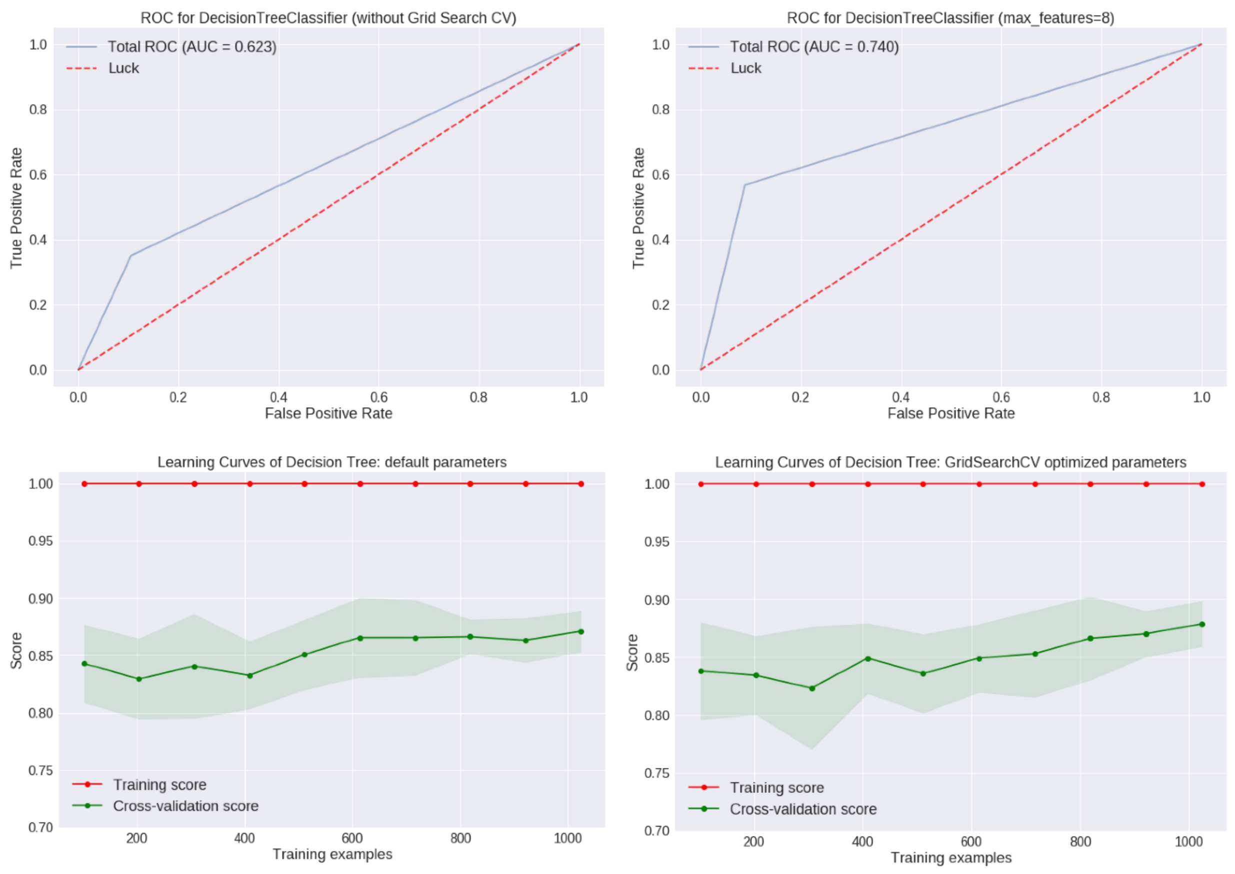

Figure 1. ROC and learning curves of decision tree classifier with default and GridSearchCV optimized parameters(max_features=8, all other parameters are default) on dataset 1.

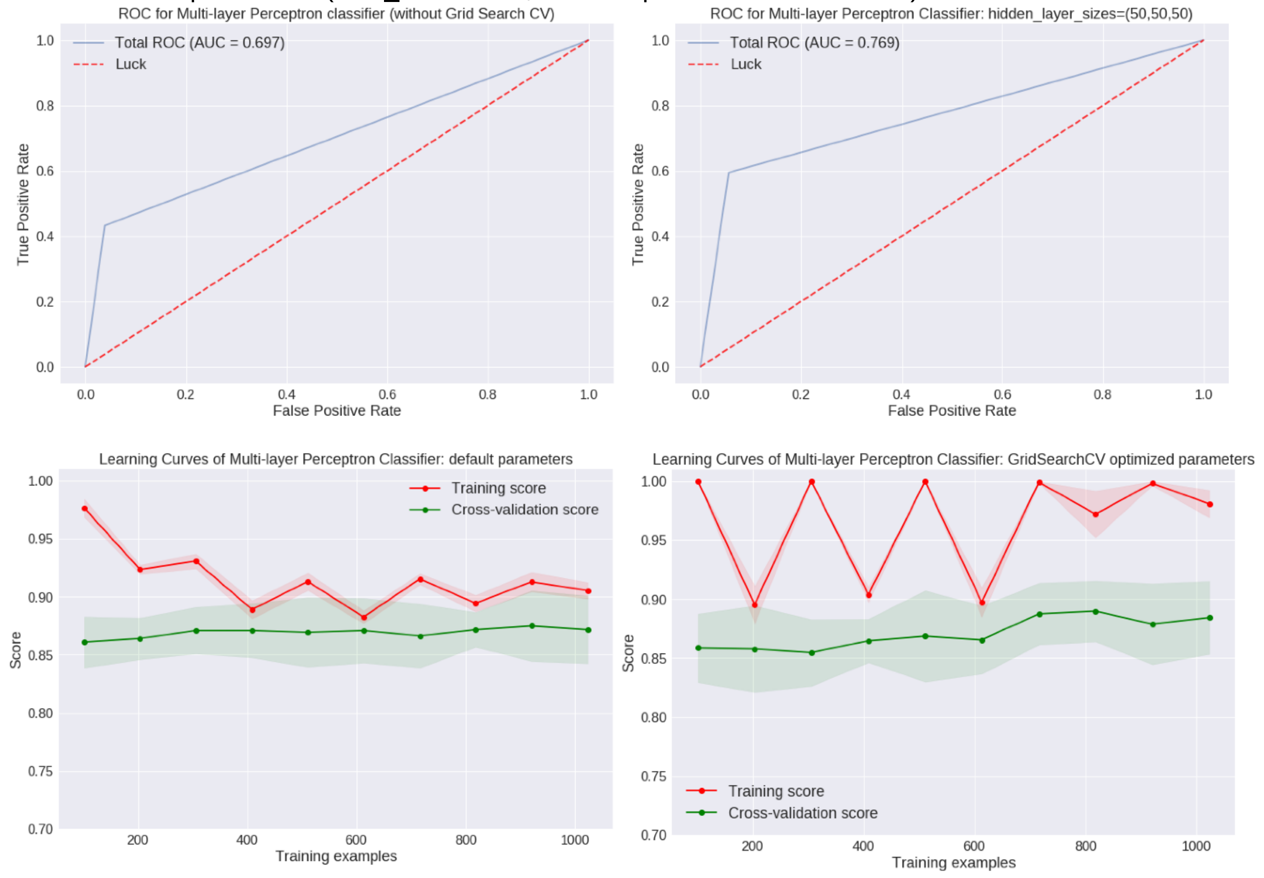

Figure 2. ROC and learning curves of neural network classifier with default and GridSearchCV optimized parameters(hidden_layer_size=(50,50,50), all other parameters are default) on dataset 1.

Neural networks: sklearn.neural_network.MLPClassifier was applied to perform the binary classification. Five parameter sets need to be optimized: the number of neurons in each hidden layer (hidden_layer_sizes: (25,), (50,), (100,), (25,50), (50,100), (25,50,25), (50,50,50), (50,100,50)), the activation function of neurons (activation: logistic sigmoid, hyperbolic tan and rectified linear unit functions), the solver for weight optimization (solver: quasi-Newton methods and stochastic gradient descent), L2 regularization term (alpha: from 0.001 to 100, each step times 10) and the learning rate schedule for weight updates (learning_rate: *'constant', 'invscaling' and 'adaptive'*).

Boosting: sklearn.ensemble.AdaBoostClassifier was applied to perform the binary classification and the boosted ensemble is built on sklearn.tree.DecisionTreeClassifier with optimized hyper-parameters. Two parameter sets need to be optimized: the maximum number of classifiers at which boosting is terminated (n_estimators: from 1 to 100 step 10) and the learning rate by which the contribution of each classifier will be reduced (learning_rate: from 0.001 to 100, each step times 10).

Support Vector Machines: sklearn.svm.SVC was applied to perform the binary classification. Three parameter sets need to be optimized: the kernel used in SVM algorithm (kernel: linear and radial basis function), penalty parameter of the error term (C: from 0.1 to 1.4 step 0.1) and the kernel coefficient (gamma: from 0.1 to 1.4 step 0.1).

***k*-Nearest Neighbors:** sklearn.neighbors.KNeighborsClassifier was applied to perform the binary classification. Two parameter sets need to be optimized: the number of neighbors at a point (n_neighbors: from 1 to 30 step 1) and the power parameter used in Minkowski metric to compute the distances among points (p: from 1 to 5 step 1).

3. Tuning up the hyper-parameters

The classifier was first carried out with default parameters to get a baseline idea of its performance. For each parameter set, it will be tuned up in terms of classification accuracy with 5-fold cross-validation. And then a parameter grid will be formed with all possible hyper-parameters and feed to sklearn.model_selection.GridSearchCV function to perform an exhaustive search with 5-fold cross-validation.

4. Learning curve

The 10%, 20%, ...100% of the total number of samples were used to generate the learning curve with sklearn.model_selection.learning_curve function. For each sample size, 20% data were randomly selected as a validation set and 20 iterations were performed to get smoother mean test and train score curves with 5-fold cross-validation within each iteration.

III. Dataset 1 results

Decision Tree: The AUC score of decision tree classifier is 0.623 using default parameter sets, and it increases to 0.740 using GridSearchCV optimized parameter sets. And either using default or optimized parameter sets, the gap between the validation and training learning curve is wide. It indicates high variance, and the model fits training data too well and will have trouble generalizing data not seen in the training set (Figure 1).

Neural network: The AUC score of neural network classifier is 0.697 using default parameter sets, and it increases to 0.769 using GridSearchCV optimized parameter sets. By using default parameter sets, the gap between the validation and training learning curve becomes narrower when the sample size increase, which indicates low variance and lower possibility of overfitting. The same pattern also occurs using optimized parameter sets (Figure 2).

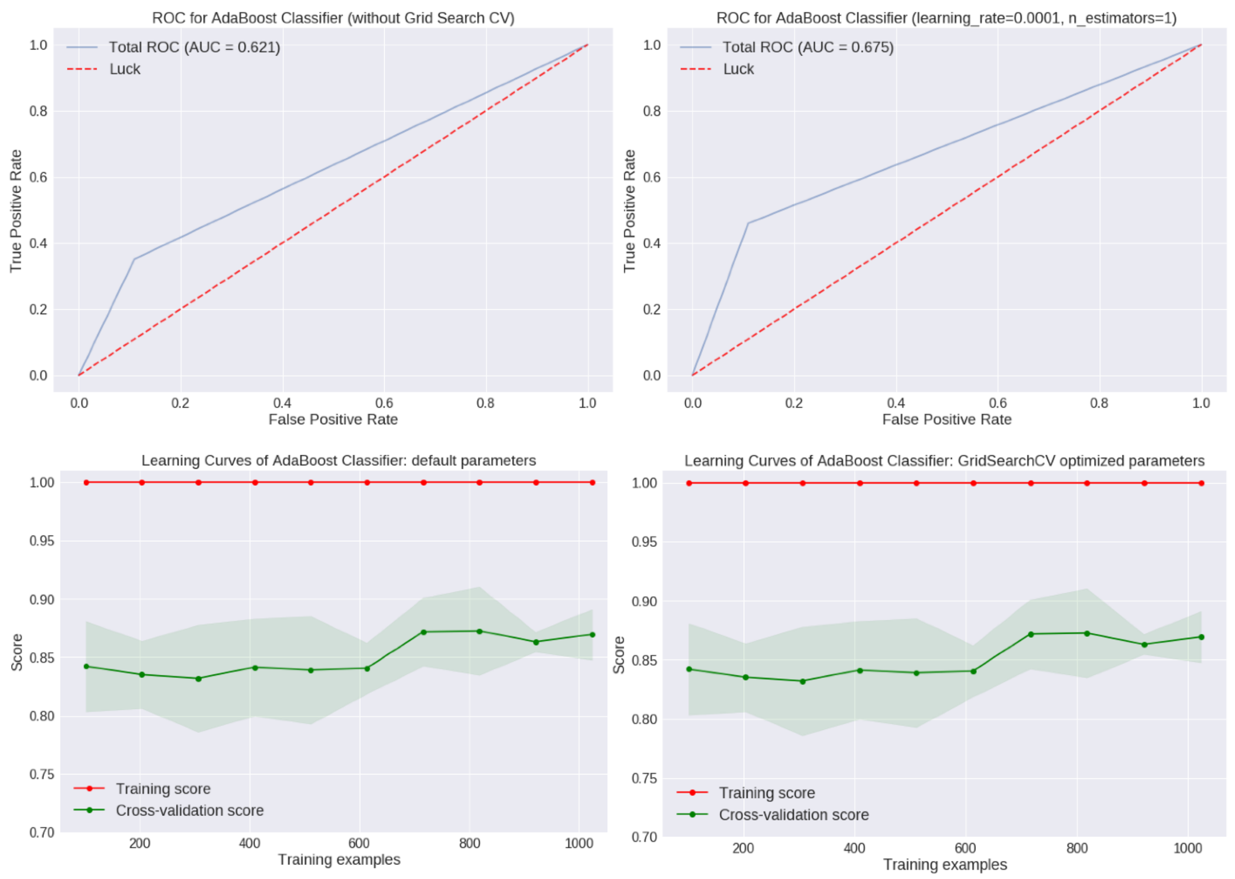

Figure 3. ROC and learning curves of decision tree boosting classifier with default and GridSearchCV optimized parameters(learning_rate=0.0001, n_estimators=1, all other parameters are default) on dataset 1.

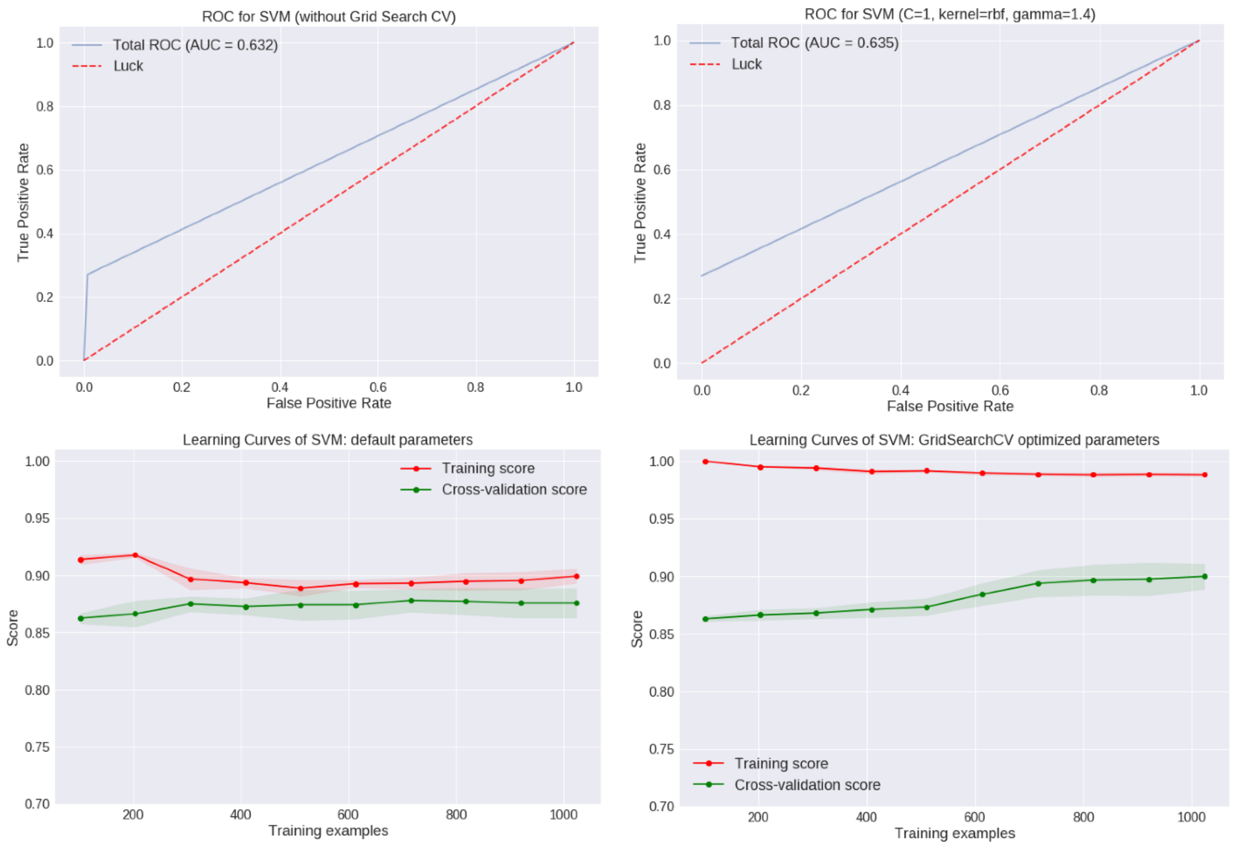

Figure 4. ROC and learning curves of SVM classifier with default and GridSearchCV optimized parameters(C=1, kernel=rbf, gamma=1.4, all other parameters are default) on dataset 1.

Boosting: The AUC score of decision tree boosting classifier is 0.621 using default parameter sets, and it increases to 0.675 using GridSearchCV optimized parameter sets. The learning curves are similar with those of decision tree classifier that indicates a high possibility of overfitting (Figure 3).

SVM: The AUC score of SVM classifier is 0.632 using default parameter sets, and it increases to 0.635 using GridSearchCV optimized parameter sets (radial basis function kernel). I also tried linear kernel and it had AUC score = 0.500 that indicates it may not be proper for solving this classification question. The gap between the validation and training learning curve becomes narrower while the sample size increasing using default parameter sets, which indicates low variance and less possibility of overfitting. However such gap is quite large while using GridSearchCV optimized parameter sets that implies generalization issue (Figure 4).

KNN: The AUC score of KNN classifier is 0.636 using default parameter sets, and it increases to 0.650 using GridSearchCV optimized parameter sets. Both default and optimized parameter sets have narrower gaps between their corresponding training and validation learning curves while sample size increasing, which indicates low variance and less possibility of overfitting (Figure 5).

IV. Discussion the performances of classifiers on dataset 1

How fast were they in terms of wall clock time?
The neural network classifier has the longest wall clock time (1.260 seconds) and the decision tree classifier has the shortest wall clock time (0.005 seconds) (Table 1).

Which algorithm performed best? How do you define best?
For binary classification question, AUC score is a good metric to measure the classifier performance. Higher AUC means better performance. Thus the neural network classifier has the best performance with the highest AUC score 0.769. Note that this dataset is highly unbalanced hence the predictive accuracy on the “Good” class should also be considered. Again the neural network classifier has the best predictive accuracy on the “Good” class with precision = 0.58, recall = 0.59 and f1-score = 0.59 (Table 1).

Would cross validation help?
Comparing with no cross-validation results, the accuracy score of decision tree classifier and boosting classifier are increased after 5-fold cross-validation, whereas that of neural network, SVM and KNN classifiers are reduced after 5-fold cross-validation. Again the accuracy score is not a good metric to measure the classifier performance for this dataset since it is highly unbalanced. Even predicting all testing samples are “Bad” will still have a good accuracy score (Table 1).

How much performance was due to the problems you chose?
As it was mentioned in the introduction part, the arbitrary cutoff to determine “Good” and “Bad” quality might have impact on the performance of classifiers. Like Kaggle said, if we setup the cutoff = 7 that makes the dataset more unbalanced, even a simple decision tree classifier will have AUC = 0.88. Thus I would say not 100% of the classifiers’ performances were due to the problem I chose.

V. Dataset 2 results

Decision Tree: The AUC score of decision tree classifier is 0.524 using default parameter sets, and it reduces to 0.500 using GridSearchCV optimized parameter sets. The gap between the validation and training learning curve is wide while using default parameter sets, and it becomes much narrower

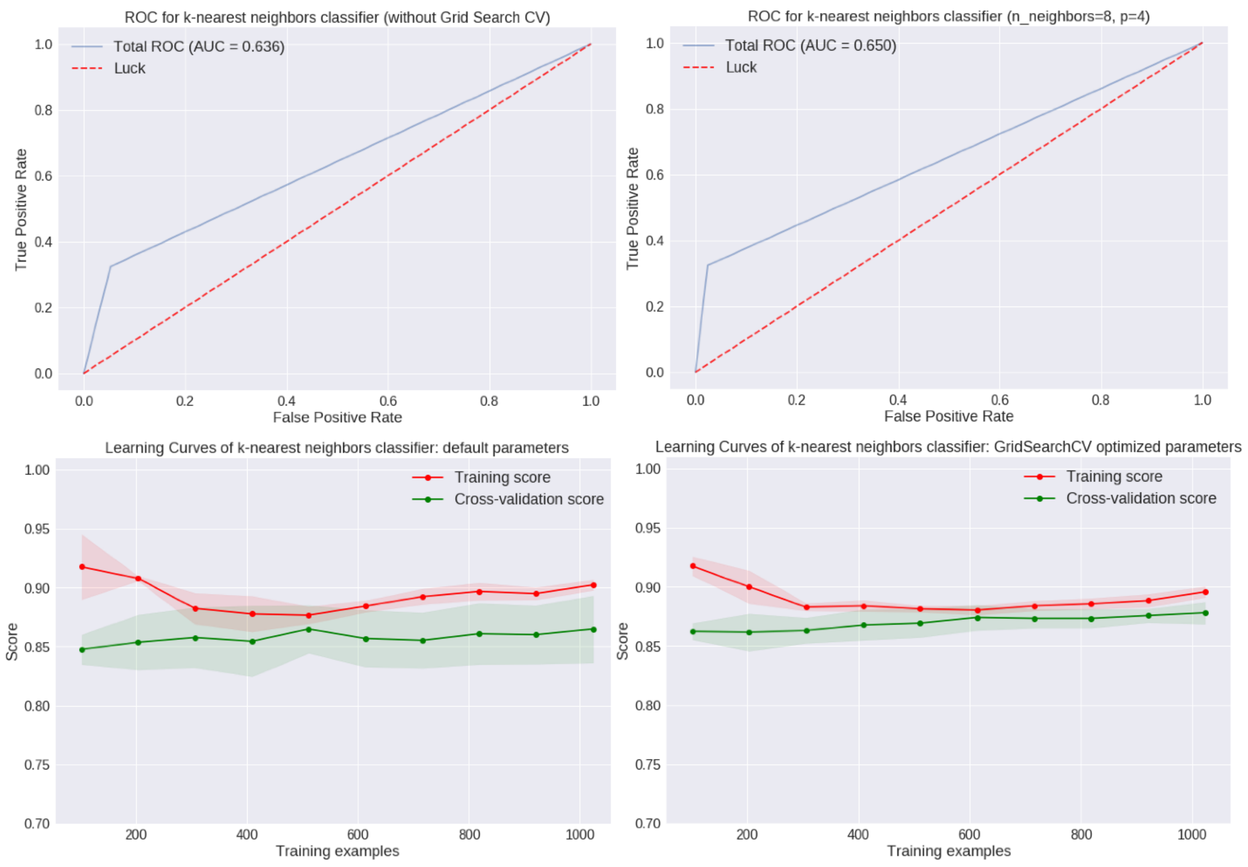

Figure 5. ROC and learning curves of KNN classifier with default and GridSearchCV optimized parameters(n_neighbors=8, p=4, all other parameters are default) on dataset 1.

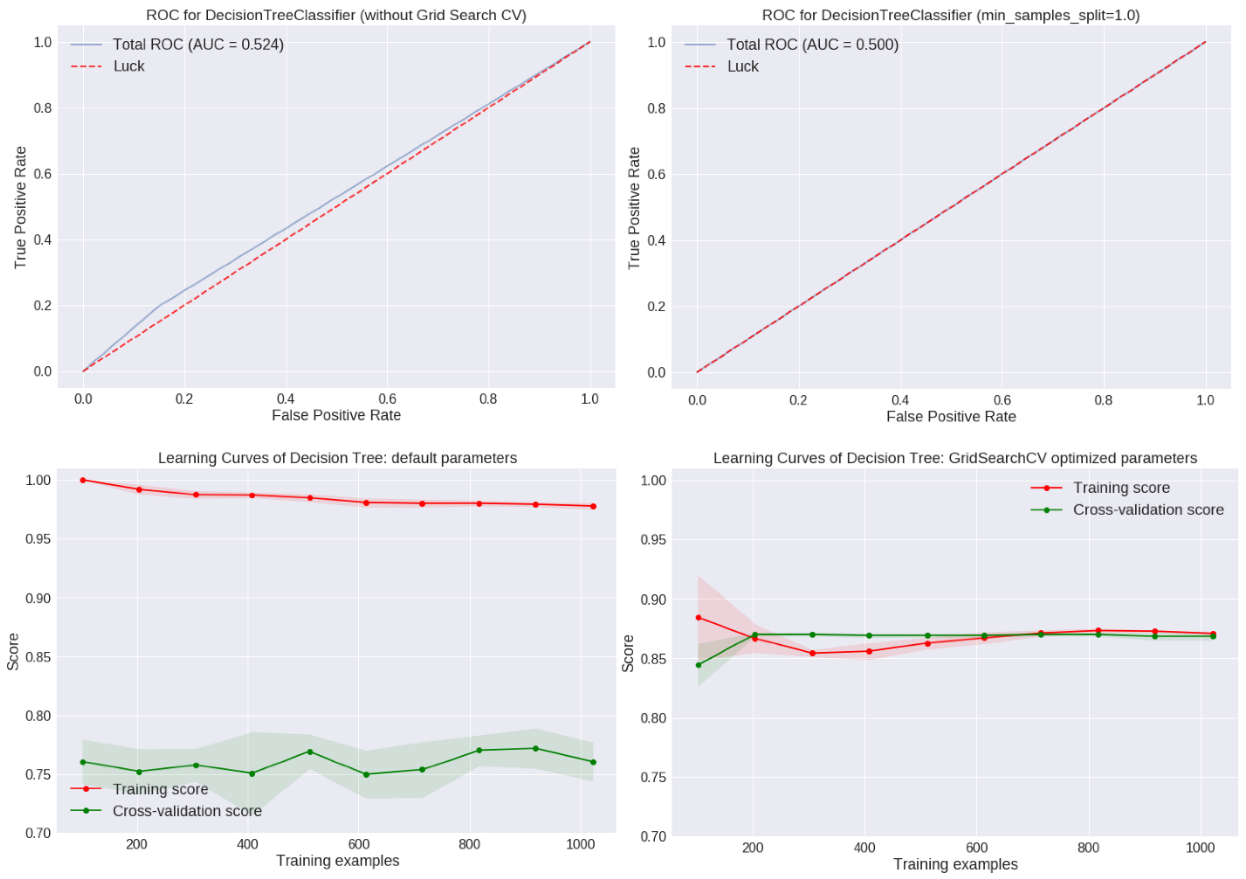

Figure 6. ROC and learning curves of decision tree classifier with default and GridSearchCV optimized parameters(min_samples_split=1.0, all other parameters are default) on dataset 2.

while using GridSearchCV optimized parameter sets that indicates lower variance and less possibility of overfitting (Figure 6).

Neural network: Both default and GridSearchCV optimized parameter sets have AUC=0.500 and extremely narrow gaps between its corresponding training and validation learning curves when sample size increases, which indicates low variance and quite high bias (Figure 7).

Boosting: Both default and GridSearchCV optimized parameter sets have AUC=0.500. The gap between the validation and training learning curve becomes narrower while the sample size increasing using default parameter sets, which indicates low variance and less possibility of overfitting. And the gap between the validation and training learning curve is extremely narrow and not associated with the increment of the sample size while using GridSearchCV optimized parameter sets (Figure 8).

SVM: Both default and GridSearchCV optimized parameter sets (linear kernel) have AUC=0.500 and extremely narrow gaps between their corresponding training and validation learning curves that are not associated with the increment of the sample size. I also tried radial basis function kernel and it had AUC score = 0.500 as well (Figure 9).

KNN: The AUC score of KNN classifier is 0.493 using default parameter sets, and it increases to 0.500 using GridSearchCV optimized parameter sets. Both parameter sets have extremely narrow gaps between their corresponding training and validation learning curves that are not associated with the increment of the sample size (Figure 10).

VI. Discussion the performances of classifiers on dataset 2

How fast were they in terms of wall clock time?
The neural network classifier has the longest wall clock time(0.302 seconds) and the decision tree classifier has the shortest wall clock time(0.005 seconds) (Table 2).

Which algorithm performed best? How do you define best?
Since this is a white noise dataset, we expect the classifier has AUC score no more than 0.500. Neural network classifier has a AUC score a little bit higher than 0.500 (0.524) while using default parameter sets. And the rest classifiers have AUC score no more than 0.500 no matter what parameter sets were used (Table 2).

Would cross validation help?
Comparing with no cross-validation results, the accuracy score of all 5 classifiers are increased after 5-fold cross-validation. Again this is a white noise dataset and with or without cross-validation, the AUC score of classifiers should be no more than 0.500 (Table 2).

How much performance was due to the problems you chose?
This is a dataset with labels randomly permutated. Hence the performances of the classifiers are 100% due to the problems I chose and they should always report AUC score no more than 0.500.

VII. References

1. Kaggle: red wine quality dataset, https://www.kaggle.com/uciml/red-wine-quality-cortez-et-al-2009

2. Scikit-learn package, https://scikit-learn.org/stable/index.html

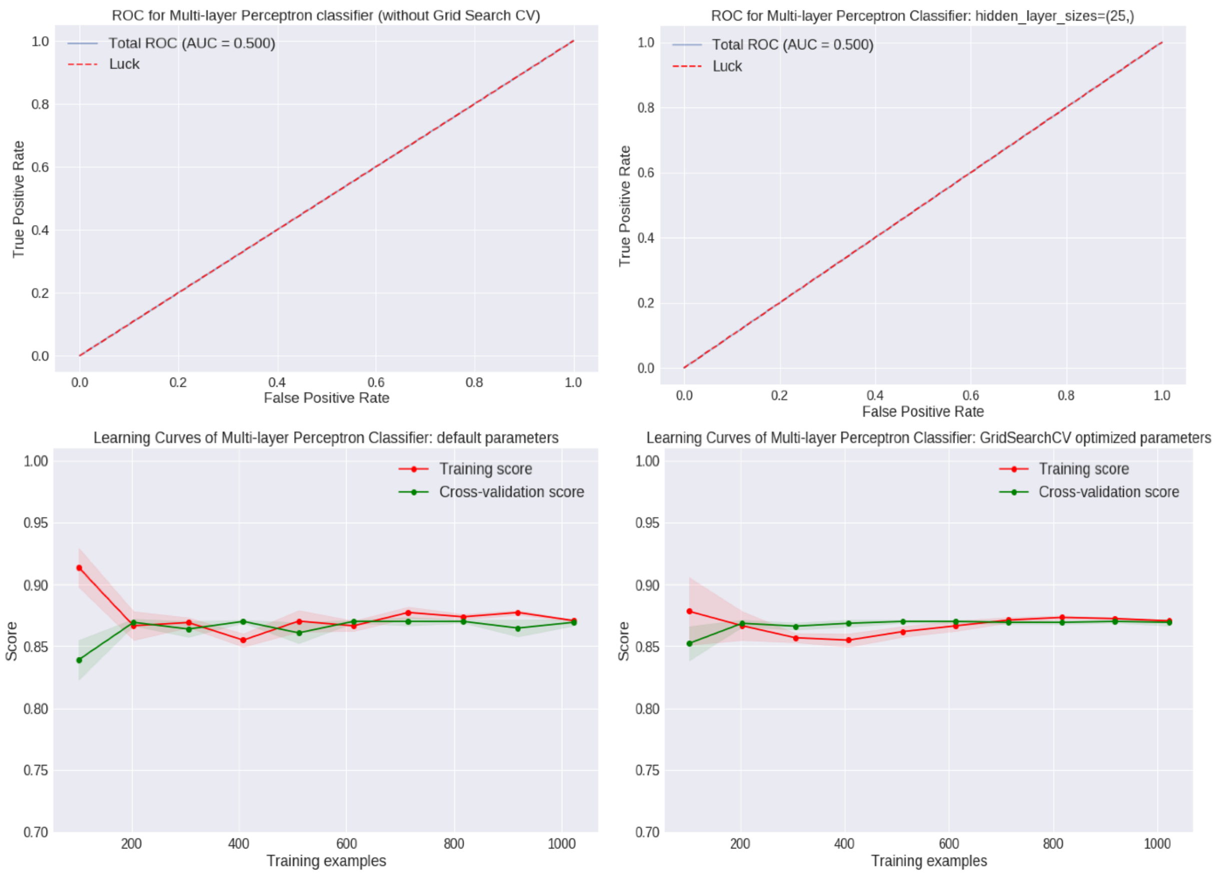

Figure 7. ROC and learning curves of neural network classifier with default and GridSearchCV optimized parameters(hidden_layer_size=(25,), all other parameters are default) on dataset 2.

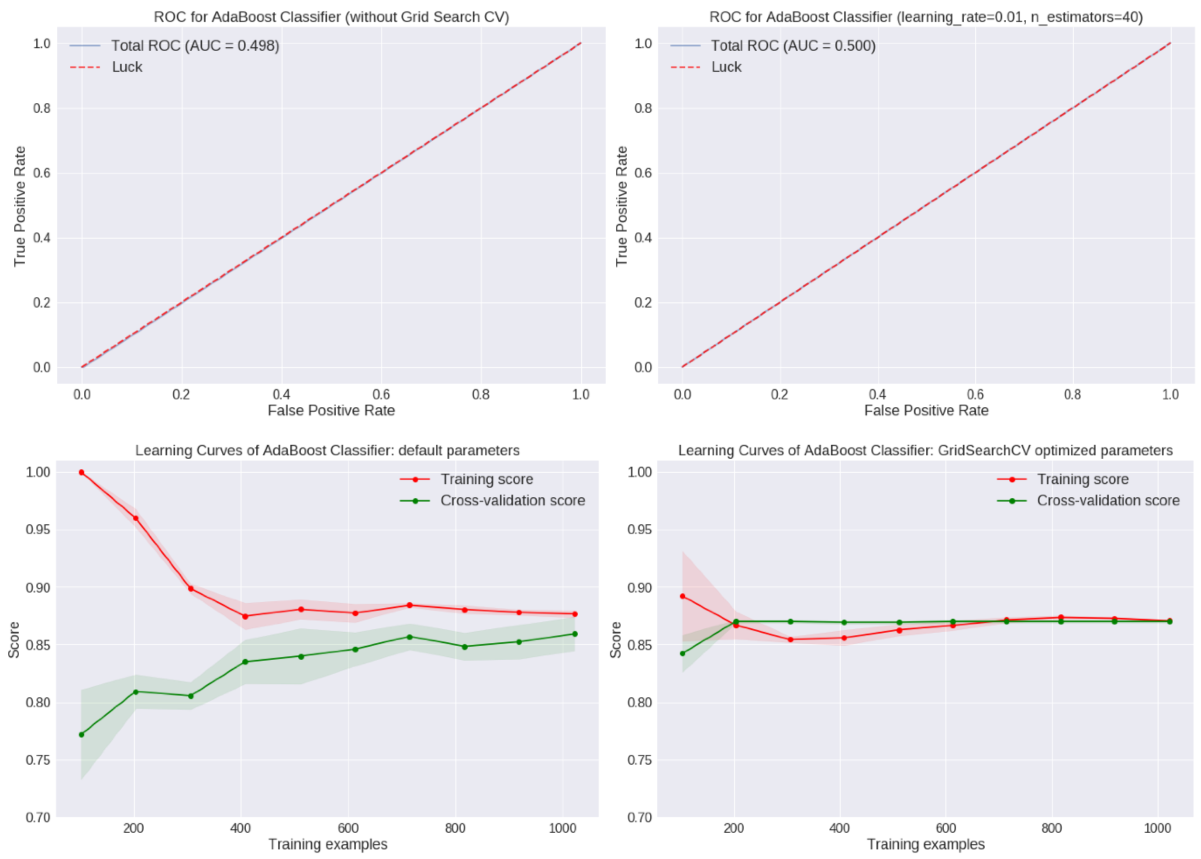

Figure 8. ROC and learning curves of decision tree boosting classifier with default and GridSearchCV optimized parameters(learning_rate=0.01, n_estimators=40, all other parameters are default) on dataset 2.

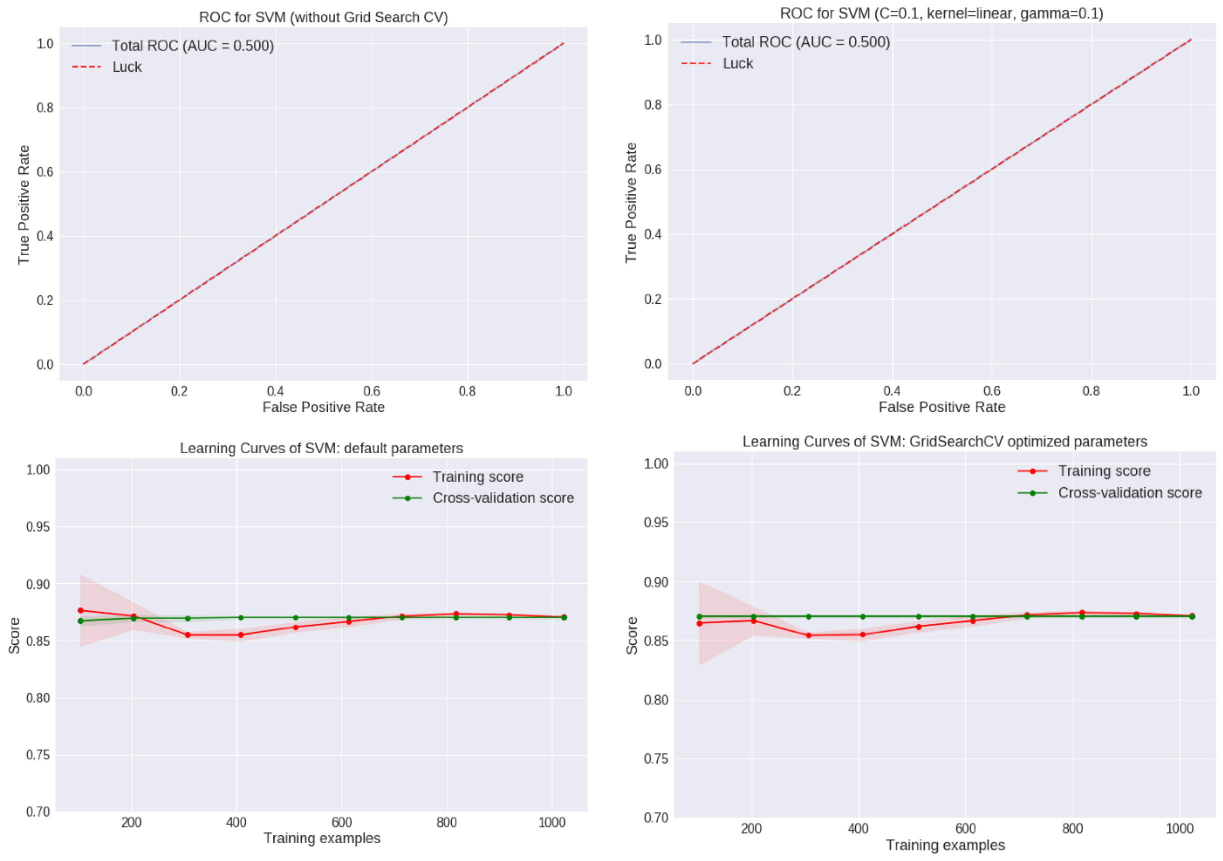

Figure 9. ROC and learning curves of SVM classifier with default and GridSearchCV optimized parameters(C=0.1, kernel=linear, gamma=0.1, all other parameters are default) on dataset 2.

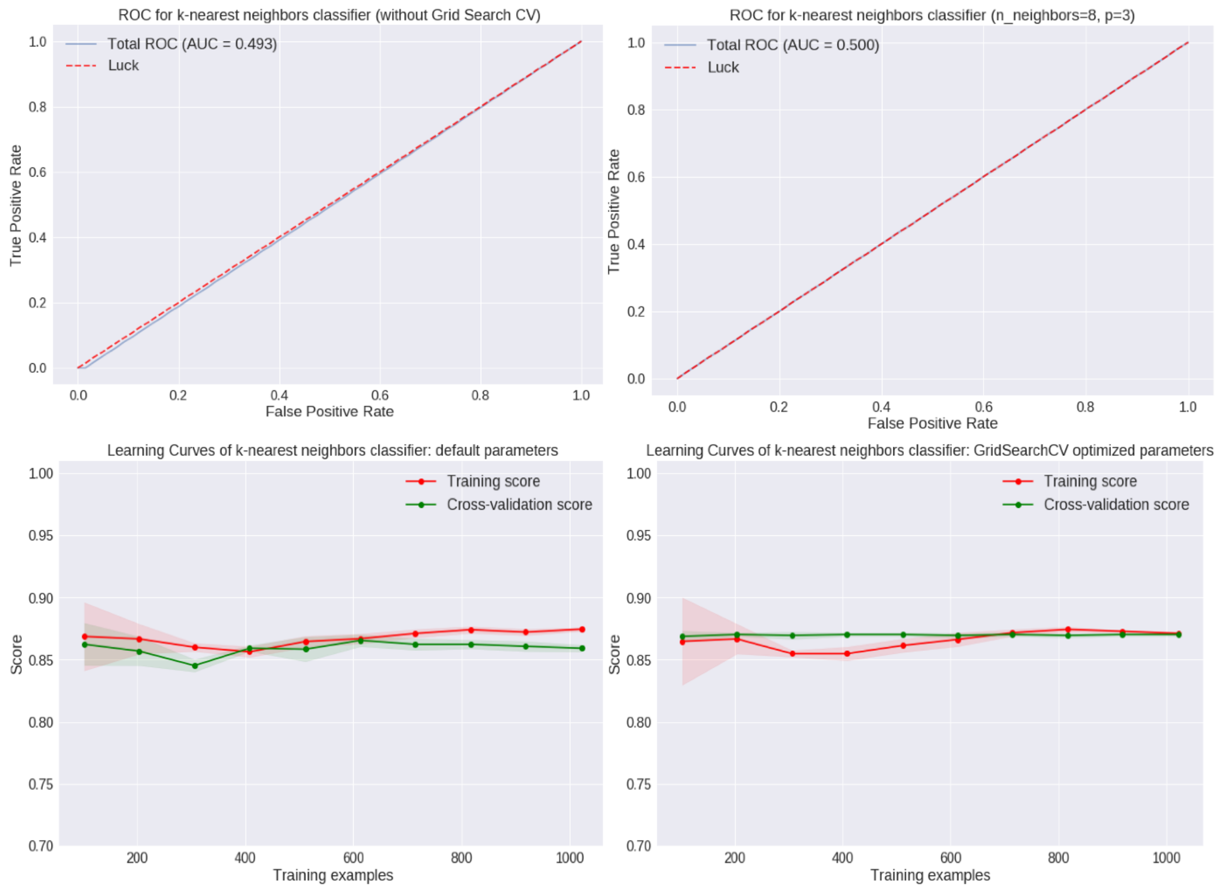

Figure 10. ROC and learning curves of KNN classifier with default and GridSearchCV optimized parameters(n_neighbors=8, p=3, all other parameters are default) on dataset 2.

Metric		Decision Tree	Neural Network	Boosting	SVM	KNN
Precision	"Bad"	0.940	0.950	0.930	0.910	0.920
	"Good"	0.460	0.580	0.350	1.000	0.630
	Total	0.890	0.900	0.860	0.920	0.880
Recall	"Bad"	0.910	0.940	0.890	1.000	0.980
	"Good"	0.570	0.590	0.460	0.270	0.320
	Total	0.870	0.900	0.840	0.920	0.900
F1-Score	"Bad"	0.930	0.940	0.910	0.950	0.950
	"Good"	0.510	0.590	0.400	0.430	0.430
	Total	0.880	0.900	0.850	0.890	0.890
Accuracy(no CV)		0.872	0.903	0.841	0.916	0.900
Accuracy(5-fold CV)		0.878	0.886	0.869	0.900	0.878
AUC		0.740	0.769	0.675	0.635	0.650
Wall Clock Time(seconds)		0.005	1.260	0.051	0.061	0.141

Table 1. Comparing the performances of five classifiers with parameter sets having best AUC on dataset 1.

Metric		Decision Tree	Neural Network	Boosting	SVM	KNN
Precision	"Bad"	0.850	0.840	0.840	0.840	0.840
	"Good"	0.200	0.000	0.000	0.000	0.000
	Total	0.750	0.710	0.710	0.710	0.710
Recall	"Bad"	0.850	1.000	1.000	1.000	1.000
	"Good"	0.200	0.000	0.000	0.000	0.000
	Total	0.750	0.840	0.840	0.840	0.840
F1-Score	"Bad"	0.850	0.910	0.910	0.910	0.910
	"Good"	0.200	0.000	0.000	0.000	0.000
	Total	0.750	0.770	0.770	0.770	0.770
Accuracy(no CV)		0.747	0.841	0.841	0.841	0.841
Accuracy(5-fold CV)		0.869	0.870	0.870	0.871	0.870
AUC		0.524	0.500	0.500	0.500	0.500
Wall Clock Time(seconds)		0.005	0.011	0.302	0.080	0.009

Table 2. Comparing the performances of five classifiers with parameter sets having best AUC on dataset 2.