Final Report

Student names
Minxi Yang, Sanghyun Joo, Yan Sun

Definition of the problem
Classification of events into “tau-tau decay of a Higgs boson” versus “background noises”.

Description of the dataset to be used.
50k events with ID, label, weight, and 30 features for each sample.
Labels are either ‘s’, signal data, or ‘b’, background data.

URL where the above dataset(s) is(are) available.
https://www.kaggle.com/c/higgs-boson/data

Three machine learning algorithms to be used
Neural Network, SVM, and Adaptive Boosted Decision Tree.

Data Description and Preprocessing
All the features except ‘EventId’, ‘Weight’, ‘PRI_jet_num’ are standardized to mean 0 and variance 1 for the valid values. The missing value is filled with the minimum of the standardized value minus 1. Some of the features are distributed in a very large range (from 0 to several hundreds), but most of the values are less than 100, and in high range the data points are distributed very sparsely (for example: https://github.com/minxiyang/CS_578_Project/blob/main/plots/plots_raw_log/DER_sum_pt.pdf).
For those features, we perform a transformation x=ln(x+1) and clipping the high values before the standardization.

The ‘PRI_jet_num’ is a discrete feature, we didn’t change it. ‘EventId’ is the irrelevant feature and wouldn’t be used for training and prediction.

For Neural Network, we rescale the sum of the weight of the signal to equal the sum of the weight of the background by multiplying a global constant.

The preprocessing.py file is used for data preprocessing

Cross Validation

1. Neural Network
   K-fold cross validation with k=5 on 30K training set is used. We train the neural network using stochastic gradient descent(SGD) with momentum for at most 300 epochs with mini-batch size 128. We use fixed learning rate 0.001 and momentum parameter 0.9. We use early stopping by looking at the AUC on the validation set(we stop the training if the AUC on validation set doesn’t improve for 10 epochs and use the model with best validation AUC for prediction).

2. SVM

K-fold cross validation with k=5 on 30K training set is used. For each epoch, the SVM with radial basis kernel is trained on training folds with the given gamma and C, and the model is tested on the testing fold. When testing, AUC is calculated. After the k-fold cross validation is finished, the mean and variance of five AUCs are calculated and returned.

3. Adaptive Boosted Decision Tree
   K-fold cross validation with k=5 on 30K training set is used. Decision tree is used as the weak classifier for the AdaBoost.

Hyperparameter Tuning

1. Neural Network
   We use a fully connected neural network with 30 hidden units on each hidden layer. The hyper parameters to be tuned are the number of layers and activation function. The number of layers are chosen from {1,2,3}. The activation function is chosen from ReLU(rectified linear unit) and tanh(hyperbolic tangent function).

2. SVM
   The hyperparameters to be tuned are gamma for radial basis kernel and slack penalty C. It tries k-fold cross validation for all combinations of gammas = {0.01, 0.1, 1, 10, 100} and C = {0.01, 0.1, 1, 10, 100}, and finds the gamma and C that give the highest mean AUC with moderate variance (less than 1).

3. Adaptive Boosted Decision Tree
   The hyperparameters are the number of the tree [1,3,5,10,100,500], number of the maximum layers [3,4,5,6,7], and the learning rate [0.01, 0.1, 1]. All the combinations are tried and we try to find the classifier with the greatest mean of AUC.

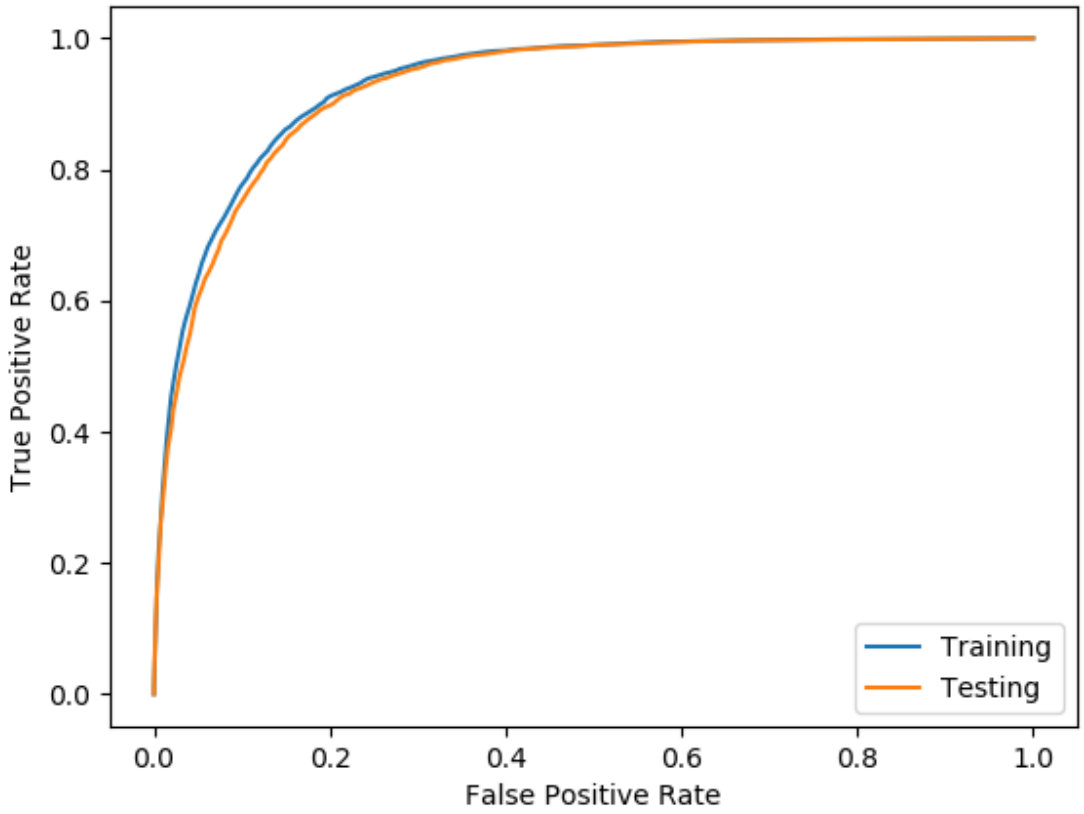
Experimental Results

1. Neural Network
   For our 5-fold cross-validation, we get the mean and standard deviation of the AUC on training and validation data set for each hyper-parameter combination. The results are summarized in the following two tables(standard deviations are given in parentheses)

Training AUC	hidden layer = 1	hidden layer = 2	hidden layer = 3
tanh	0.9316(0.0014)	0.9318(0.0001)	0.9317(0.0021)
relu	0.9295(0.0017)	0.9302(0.0018)	0.9314(0.0014)

Validation AUC	hidden layer = 1	hidden layer = 2	hidden layer = 3
tanh	0.9261(0.0026)	0.9258(0.0038)	0.9250(0.0059)
relu	0.9240(0.0042)	0.9246(0.0015)	0.9243(0.0022)

The mean of the AUC on the validation set of different combinations of hyper-parameters are very close. (tanh, hidden layer = 1) has the largest mean of AUC but it is not significantly better than the combination with second largest mean value, i.e. (tanh, hidden layer = 2). The results are not very sensitive to the choice of number of hidden layers and activation function. In what follows, we choose the hyper-parameter with the largest mean AUC on validation set, i.e. we use tanh as activation function and use 1 hidden layer network.
We retrain the model using all 30k training samples, we use 10% of the data as validation to determine early stopping. The following figure shows the ROC curve on the training and test set. And the AUC on the training set is 0.9265, the AUC on the test set is 0.9233

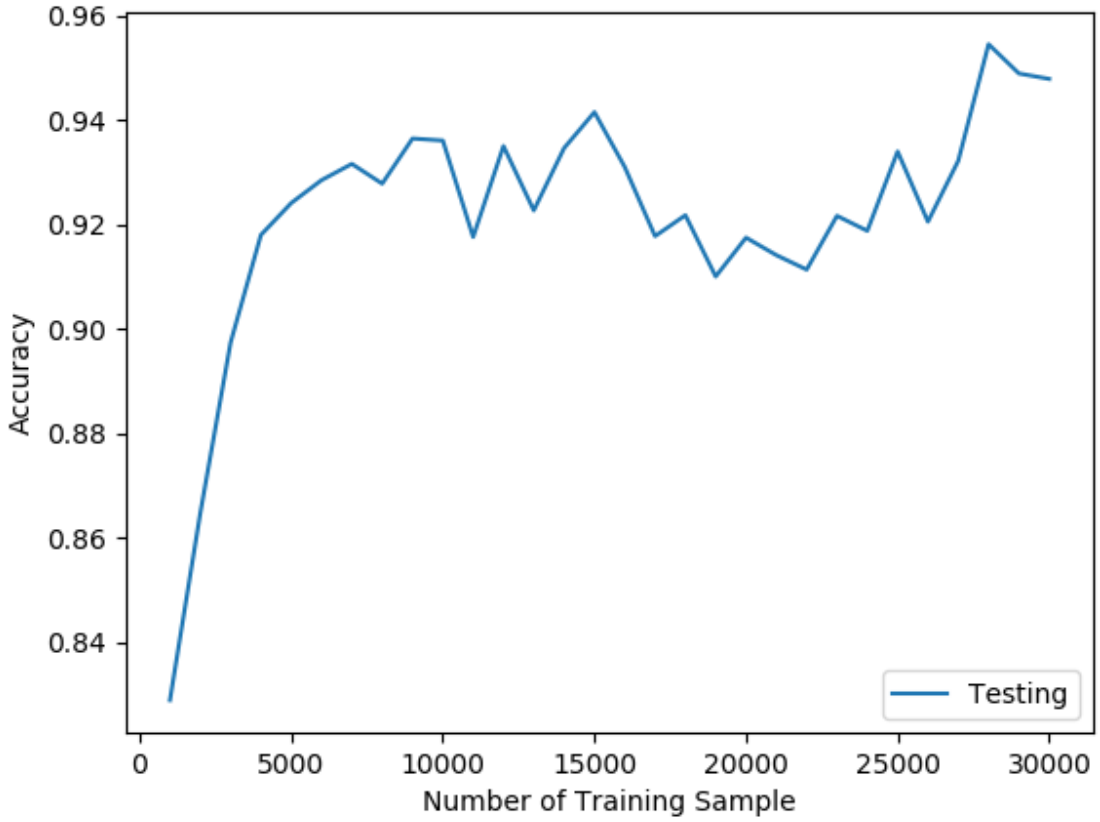


We use AMS(*approximate median significance*)(Please refer to https://www.kaggle.com/c/higgs-boson/overview/evaluation for details) on the training set to determine the threshold for making predictions. The best threshold we use is 0.7471 and the AMS on the test set is 1.2059. The following table shows the confusion matrix on the test set.

		Actual Value	
		Positive	Negative
Predicted	Positive	TP = 50.4045	FP = 1402.3164

Value	Negative	FN = 29.7287	TN = 23288.3415

We also tried training with different training sample size(from 1K to 30K), For each training data, we use 10% of the data as validation to determine early stopping. And use AMS to determine the threshold for making predictions. The following figure shows the prediction accuracy on the test set when training with different training sample size.



2. SVM
   a. Mean and variance of AUCs
   For SVM, two hyperparameters, gamma and C, are tuned. 5-fold cross validation for all combinations of gammas = {0.01, 0.1, 1, 10, 100} and C = {0.01, 0.1, 1, 10, 100} are tried, and mean and variance of them are computed. Since the variances are very low, most of them are 0.0 when round up to 4th decimal points.

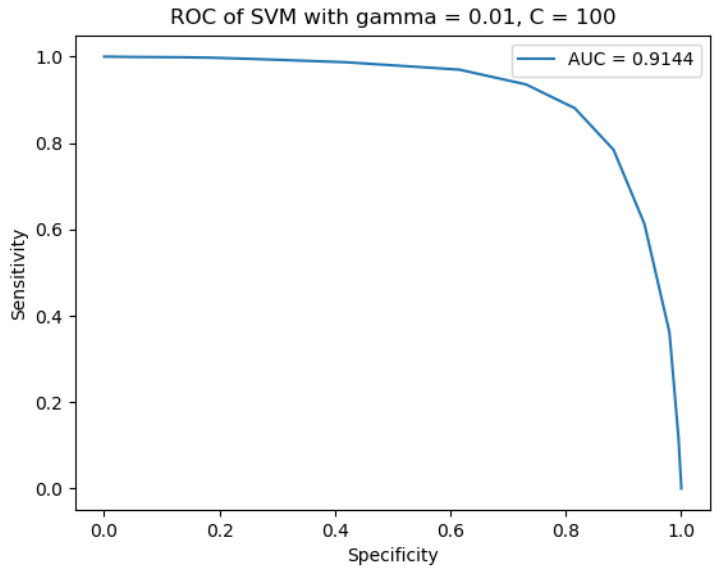
	C = 0.01	C = 0.1	C = 1	C = 10	C = 100
γ = 0.01	0.7932, 0.0	0.8247, 0.0	0.8563, 0.0	0.8716, 0.0	**0.873, 0.0**

$\gamma = 0.1$	0.7821, 0.0001	0.837, 0.0	0.8603, 0.0	0.8538, 0.0	0.8275, 0.0
$\gamma = 1$	0.661, 0.0	0.6645, 0.0	0.6998, 0.0001	0.7093, 0.0001	0.7172, 0.0002
$\gamma = 10$	0.5037 0.0	0.5036, 0.0	0.506, 0.0	0.5038, 0.0	0.5058, 0.0
$\gamma = 100$	0.3999, 0.04	0.3999, 0.04	0.4999, 0.0	0.4999, 0.0	0.4999, 0.0

Considering moderate variance (less than 1), C = 100 and $\gamma = 0.01$ give the highest mean of AUCs which is 0.873.

b. ROC and mean AUC

With C = 100 and $\gamma = 0.01$, the model is trained using the 30K training set. In order to plot the ROC curve and compute the AUC, we computed specificity and sensitivity on the 20K testing set using the trained model. The AUC is 0.9144



The image of ROC with its AUC is saved in "./plots/SVM/SVM_ROC.png"

c. Best threshold

With C = 100 and $\gamma = 0.01$, the model is trained using the 30K training set. In order to find the best threshold, we computed unnormalized TP and FP rates on the same 30K training set depending on different thresholds using the trained model. Then, found the threshold that maximizes the AMS(*approximate median significance*) which is calculated with unnormalized TP and FP rates. The best threshold is 0.6656.

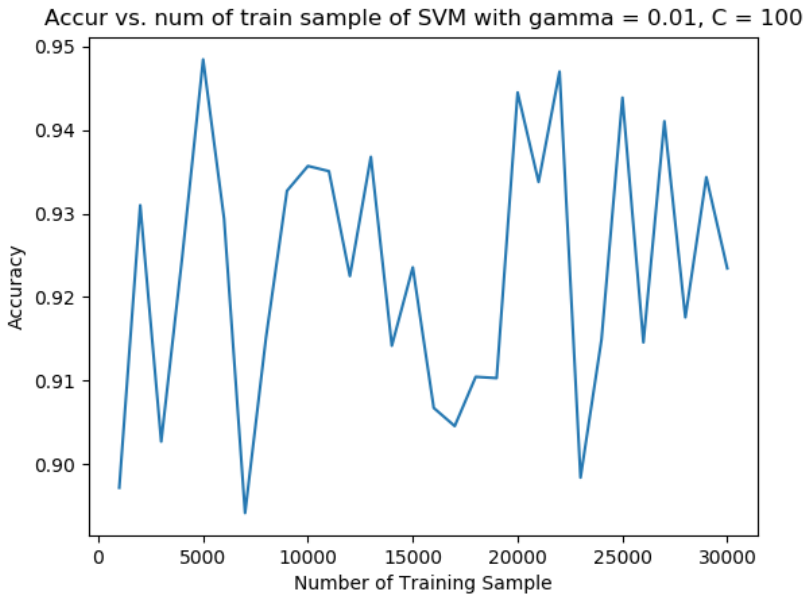
d. AMS

AMS is defined in (https://www.kaggle.com/c/higgs-boson/overview/evaluation) as the criteria to evaluate the result. It is a concept close to the signal to noise ratio. With C = 100 and $\gamma = 0.01$, the model is trained using the 30K training set. Using the best threshold

determined in c. and the trained model, we computed AMS on the 20K testing set. The AMS is 1.2034.

e. Accuracy vs. Number of Training Sample

With C = 100 and $\gamma = 0.01$, the model is trained using nK training set where n = 1, … , 30. The best threshold is determined by computing AMS on the same nK training set and following the similar step in c. Using the best threshold and the trained model, compute the accuracy on the 20K testing set. Repeat for n = 1, … , 30 and plot the result. The plot shows that no significant relation is shown between accuracy and the number of training samples.



The image of ROC with its AUC is saved in "./plots/SVM/SVM_accuracies.png"

f. Confusion Matrix

With C = 100 and $\gamma = 0.01$, the model is trained using the 30K training set. Using the best threshold determined in c. and the trained model, confusion matrix is computed on the 20K testing set.

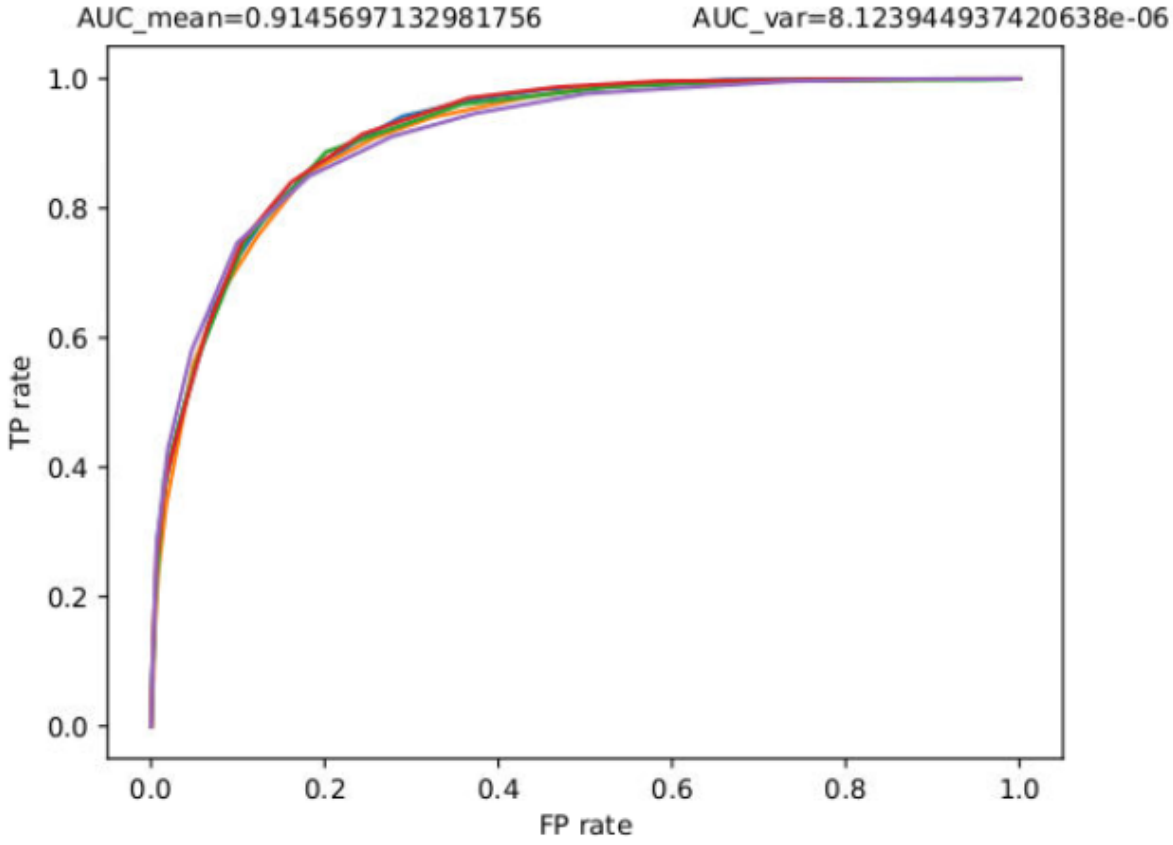
		Actual Value	
		Positive	Negative
Predicted Value	Positive	TP = 59.145	FP = 2385.9529
	Negative	FN = 20.9879	TN = 22304.6943

3. Adaptive Boosted Decision Tree

Tuning:

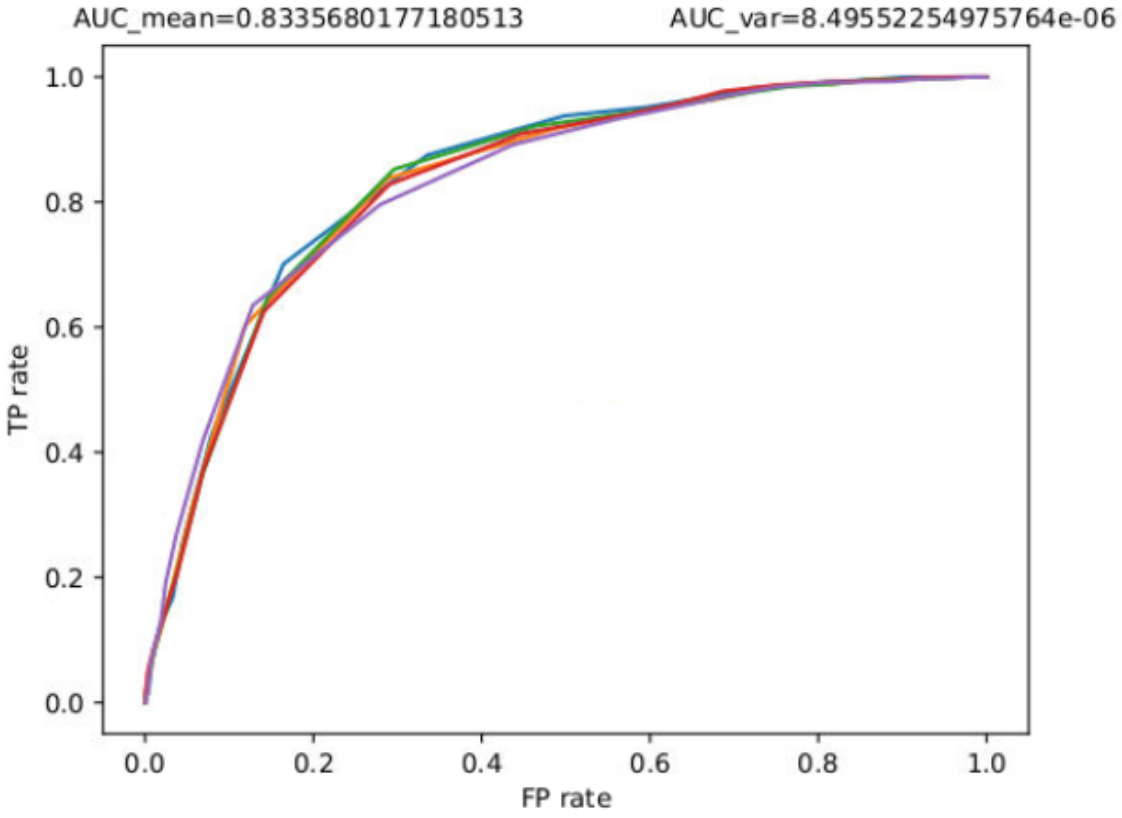
There are 90 combinations, and I just give the best case and the worst case. The variance for all the combinations of the hyperparameters are in the order of 10^(-5). Since the training are pretty robust, we just use the mean of the AUC as the criteria to choose the hyperparameters

Best combination: number of the tree = 500, number of the layers = 4, learning rate = 0.01 mean of AUC = 0.9146. The ROC curve for 5 times k-fold for the best combination is shown in the following plot.



.

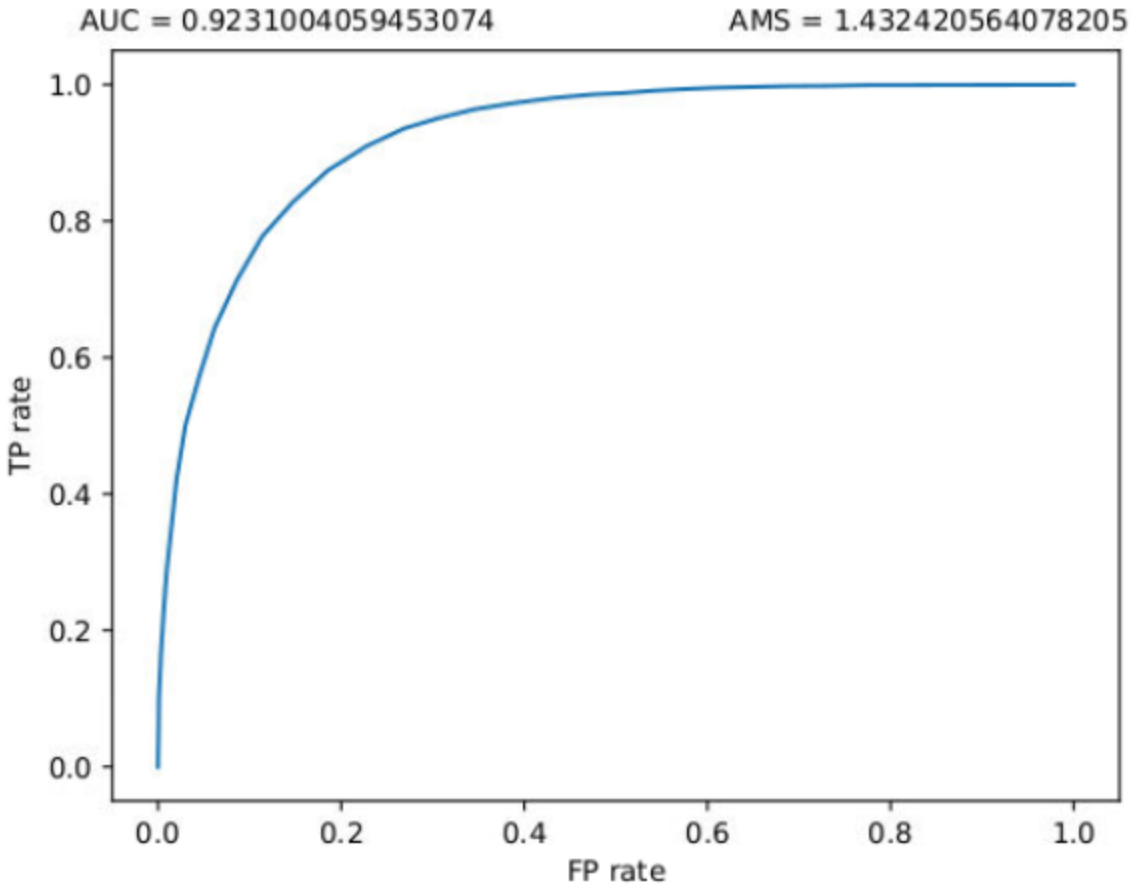
Worst case: number of the tree = 100, number of the layers = 6, learning rate = 1 mean of AUC = 0.8336. The ROC curve for 5 time k-fold for the worst combination is shown in the following plot.



The results for all the other combinations of the hyperparameters are in the folder plots/AdaBoostTree/, the mean and variance of the AUC are labeled on the top of the plots. In the file name, nest = number of the tree, nlay = number of the layers, lr = learning rate.

Implement in the testing set:

The classifier with the max mean of the AUC are chosen to implement on the testing set. The classifier is re-trained with all the 30000 training samples and applied on the testing set for the prediction. The ROC curve on the testing set is shown in the following plot plots(plots/AdaBoostTree/ROC_final.pdf).



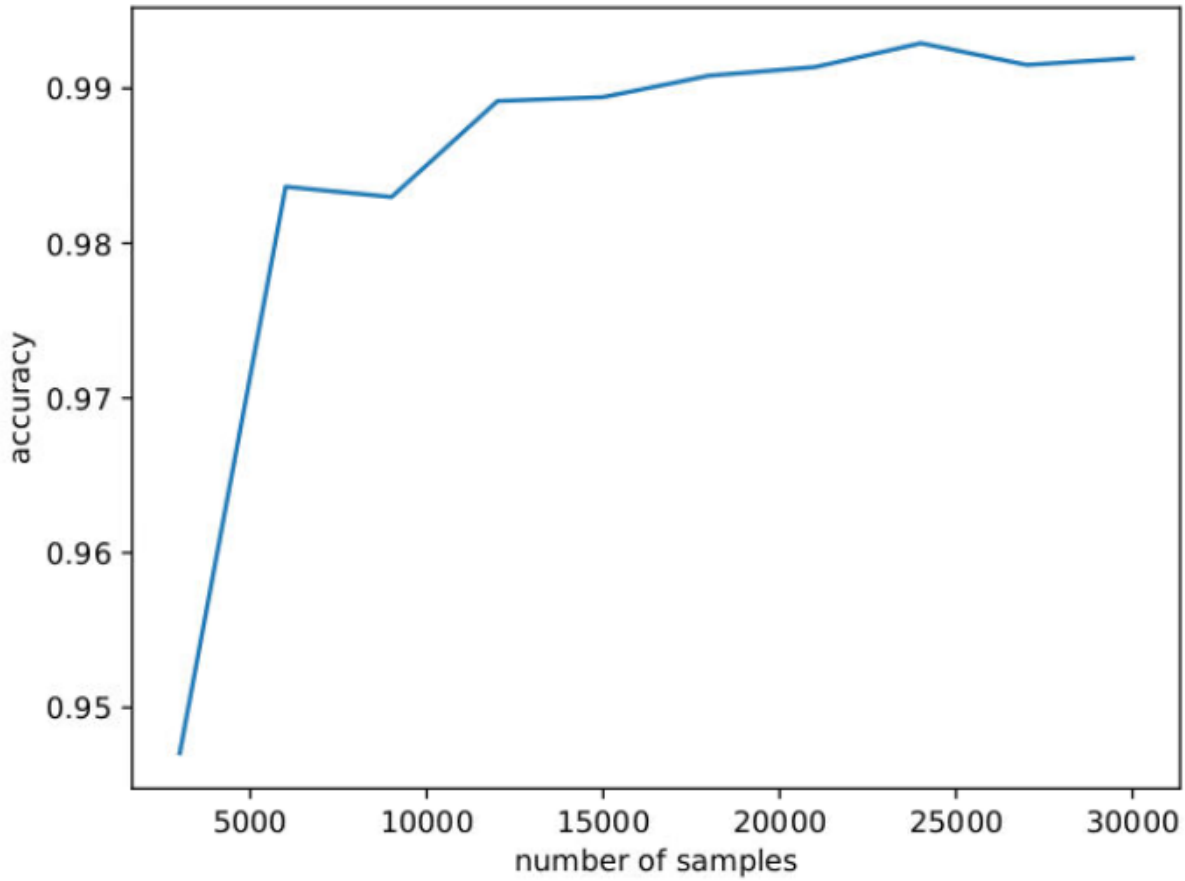
The AUC on the testing set is 0.9231. Since the classifier returns a score for each sample, a best threshold of the score is need. This is done by maximizing the AMS(*approximate median significance*) which is calculated with unnormalized TP and FP rates. The best threshold is -0.532 and the maximum of the AMS is 1.4324 . The following table shows the confusion matrix with the best threshold on the testing set.

		Actual Value	
		Positive	Negative
Predicted Value	Positive	TP = 17.6881	FP = 136.7000
	Negative	FN = 62.4448	TN = 24553.9372

Accuracy versus number of the training samples:

The number of the training samples are set to be 3000, 6000, 9000…..30000. The accuracy versus the number of the training sample is shown is the following plot (plots/AdaBoostTree/accVsnSample.pdf). The accuracy is high since the sample is unbalanced

and the total number of the background is about 100 times that for the signal. The accuracy increases with increasing the number of the training sample. And when the number of the training sample is greater than 12000, the curve is plain, which means the classifier is trained well.



.

Conclusion

Each of them shows excellent AUC and AMS on the testing set. First, AUCs of Neural Net, SVM, and Adaptive Boosted Decision Tree are 0.9233, 0.9144, and 0.923, respectively. Since the variance of AUCs were very low during 5-fold cross validation, we can conclude that Neural Net and Adaptive Boosted Decision Tree shows better AUC than SVM. Second, AMSs of Neural Net, SVM, and Adaptive Boosted Decision Tree are 1.2059, 1.2034, and 1.4324, respectively. Consequently, we can conclude that Adaptive Boosted Decision Tree shows better AMS than others. Third, Adaptive Boosted Decision Tree shows better accuracies depending on different numbers of training samples than others. All things considered, Adaptive Boosted Decision Tree is better than other models for this problem.

Project Repository

https://github.com/minxiyang/CS_578_Project