Sujeeth Shetty 03/07/2020

BUAN 6341 Applied Machine Learning

ASSIGNMENT 2

Implementation of SVM, Decision Trees & Boosting

Introduction:

The objective of this project is to implement Support Vector Machine, Decision Trees & Boosting to perform binary classification on two Datasets. This report details the various experiments conducted using the two datasets to understand the effect of the cross-validation and various hyperparameters. It also shows the effect of the use of various independent variables/features in selecting the best model.

About the Data:

The first data set is of SGEMM GPU kernel performance which consists of 14 features and 241600 records. This data set measures the running time of a matrix-matrix product A*B = C, where all matrices have size 2048 x 2048, using a parameterizable SGEMM GPU kernel with 261400 possible parameter combinations. Out of 14 features, the first 10 are ordinal and can only take up to 4 different powers of two values, and the 4 last variables are binary.

The second dataset is Bank Marketing UCI Dataset published by Banco de Portugal which consists of 20 features and 45307 records. This dataset is used to measures the success of Bank Telemarketing where the binary classification goal is to predict if the client will subscribe to a bank term deposit. There are few missing values, and all are coded with the “unknown” label. These missing values are treated as a possible class label. One of the reasons for selecting this dataset is because of the imbalance in class distribution. This would help in understanding the caveats when trying to leverage cross-validation, bias associated with classification output and risk associated with false-negative or false-positive predictions.

Project Outline:

Algorithm Implementation

The Support Vector Machine and Decision Tree is implemented using python package “Scikit-learn” and boosting is implemented using “H2O”. Scikit-learn is a very popular package because of its ease of use. It allows us to visualize our tree using the Graphviz library. On the other hand, the H2O package is memory efficient and data frames are much smaller in memory, in comparison with Pandas. One more reason to choose H20 is that the Random Forest algorithm is more efficient than any other packages.

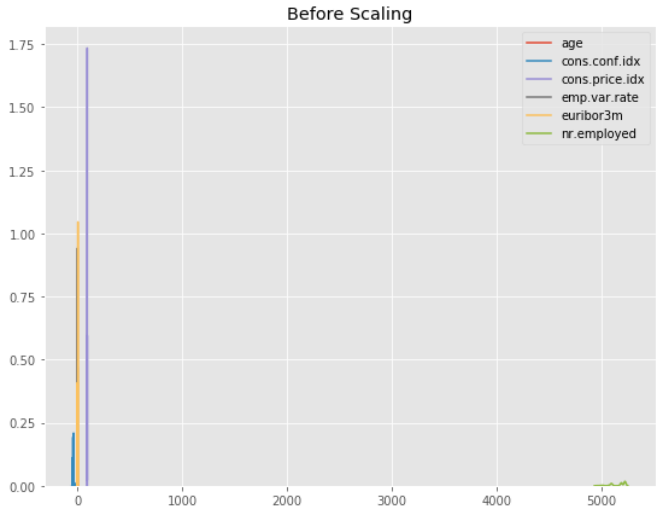
The learning algorithms are implemented for two different binary classification datasets. The algorithms are implemented with options to change the hyperparameters: kernel, penalty param, gamma, max tree depth, metric to split on variables.

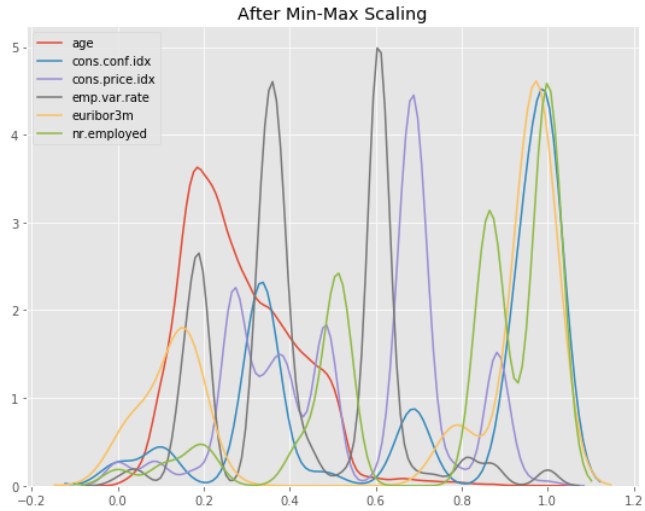
Data Preparation and Exploratory Data Analysis

For **SGEMM GPU Kernel Performance Dataset**, the runtime was converted into a classification problem by classifying all values above the mean runtime as 1 and value below as 0. By this classification, we are trying to predict when the GPU runtime is high. Scaling wasn't performed on the features since out of 14 features, 10 are ordinal and 4 are categorical. The result might not be greatly impacted when ordinal variables are not scaled.

The objective of **Bank Marketing UCI Dataset** is to correctly predict if a client will subscribe to a bank term deposited. The correlation plot indicated a high correlation between employment variation rate, Euribor 3-month rate and the number of employees(nr.employeed). To deal with multicollinearity, the variables nr.employeed & emp.var.rate are removed from the dataset.

The alogorithm performs much better when all the numeric features have similar scale. So, all these features standardized using Min-Max scaling.





To improve the model performance, one hot encoding was performed on all categorical variable and label encoding was performed on month & day_of_week.

The target variable 'y' has imbalanced in class distribution. The dataset has only 5091 positive instances and the rest 40216 are of negative class.

```
y
0    40216
1     5091
dtype: int64
```

Experimentation and Results

Three tasks were undertaken in which we ran the classification problem on SVM, Decision Tree, Gradient Boosting & Regression Tree. At least two experiments were conducted on each learning algorithms where we explored the effect of changing kernel, hyperparameter, performing cross-validation, and discuss the effectiveness of feature selection in predictive accuracy. These tasks & experiments were repeated both for SGEMM GPU Kernel Performance Dataset and Bank Marketing Dataset. Finally, the results of the experiments were discussed in the results section and further improvement opportunities were discussed.

Tasks & Experimentation:

1 Support Vector Machine

1.1 SGEMM GPU Kernel Performance Dataset

Initially, the data was split into Train, Test & Validation with a ratio of 64:20:16. A Linear classification was performed on the training dataset using SVM with default parameters C=1. Surprisingly the test & validation accuracy was around 93%.

	precision	recall	f1-score	support
0	0.94	0.97	0.95	114232
1	0.90	0.84	0.87	40392
accuracy			0.93	154624
macro avg	0.92	0.90	0.91	154624
weighted avg	0.93	0.93	0.93	154624

	precision	recall	f1-score	support
0	0.94	0.96	0.95	35677
1	0.89	0.84	0.87	12643
accuracy			0.93	48320
macro avg	0.92	0.90	0.91	48320
weighted avg	0.93	0.93	0.93	48320

Since cross-validation works well on a small dataset, a random subset of 50000 was taken and 10-fold cross-validation was performed on the same. The experiment was undertaken on three different kernels 'linear', 'rbf' & 'polynomial'. The results of all three models are depicted below. By looking at the accuracy, learning curve & AUC-ROC plot, the linear model performed better than the rest. The linear model showed an accuracy of 93% on test dataset & best bias-variance trade-off

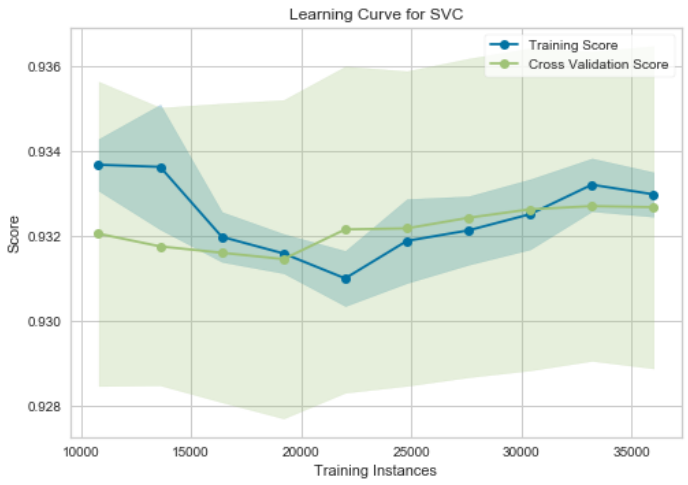


Figure 1 Learning Curve – Linear

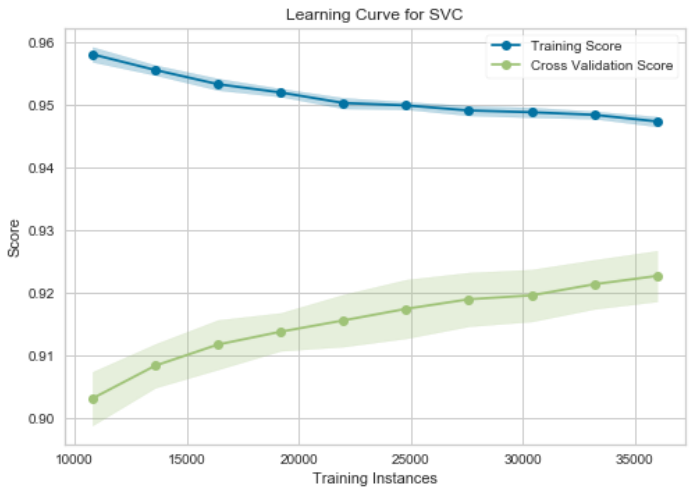
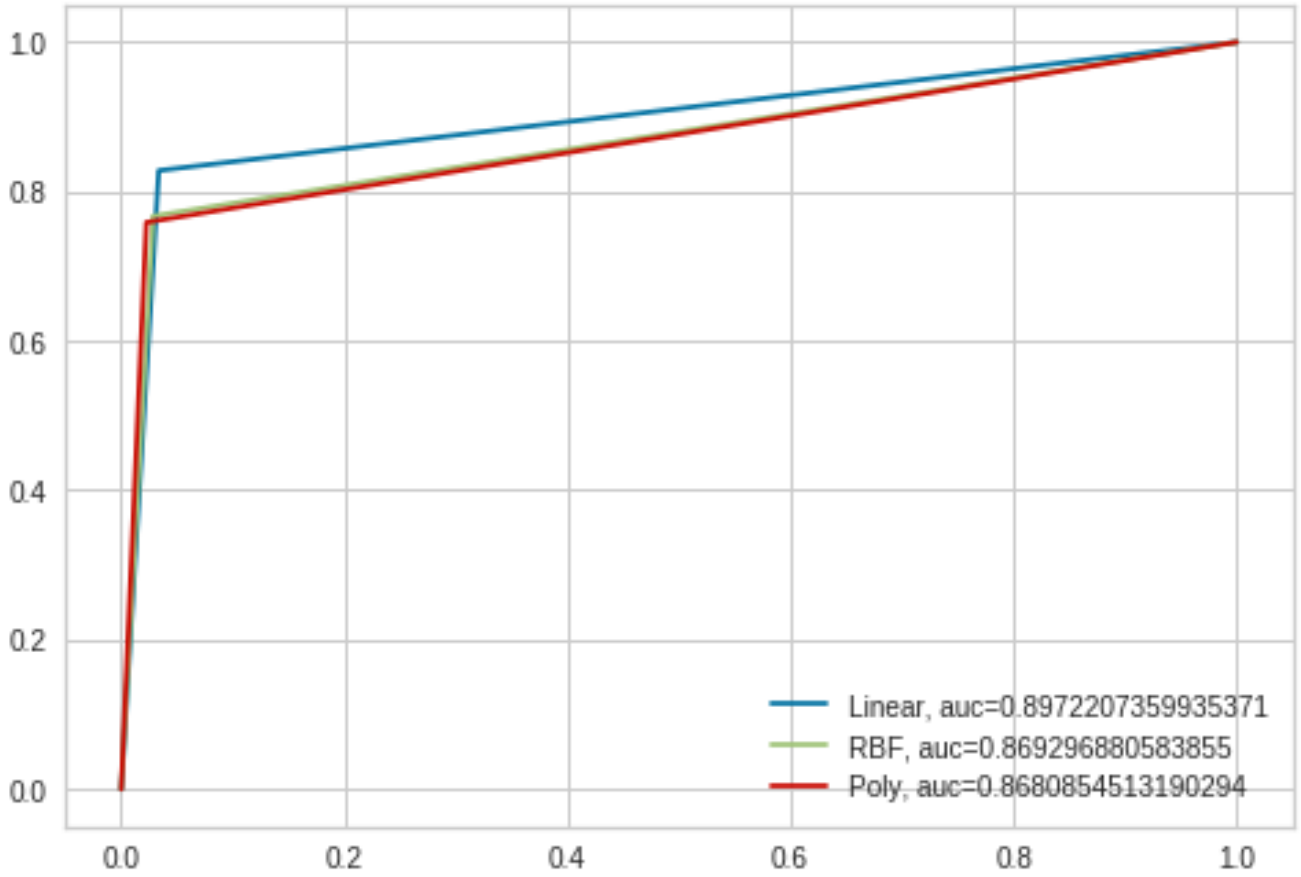


Figure 2 Learning Curve – RBF

Model	Linear	RBF	Polynomial
Test Accuracy	93	92	92
Train Accuracy	93.21	92.04	92.20

Grid search was performed on the best model which is Linear to find the best hyperparameters. The model was run again with C=0.01 & gamma=0.01 to achieve a better result. The optimized model gave a better result with 94% accuracy against test dataset and mean AUC of 0.9024

	precision	recall	f1-score	support
0	0.94	0.97	0.96	7371
1	0.92	0.83	0.87	2629
accuracy			0.94	10000
macro avg	0.93	0.90	0.92	10000
weighted avg	0.94	0.94	0.94	10000



1.2 Bank Marketing UCI Dataset

Ran the linear SVM without any hyperparameters on the entire train dataset which was split in the ratio 64:20:16. Later the model was predicated on a smaller Test and Validation dataset to get an accuracy of around 91%.

	precision	recall	f1-score	support
0	0.93	0.97	0.95	8045
1	0.64	0.41	0.50	1017
accuracy			0.91	9062
macro avg	0.78	0.69	0.73	9062
weighted avg	0.90	0.91	0.90	9062

Figure 3 Test Dataset

	precision	recall	f1-score	support
0	0.93	0.97	0.95	6452
1	0.65	0.43	0.52	797
accuracy			0.91	7249
macro avg	0.79	0.70	0.73	7249
weighted avg	0.90	0.91	0.90	7249

Figure 4 Validation Dataset

Unlike previous dataset, upon k-fold cross validation, all the kernel performed similarly with a test accuracy of 90%

Model	Linear	RBF	Polynomial
Test Accuracy	90	90	90
Train Accuracy	90.46	90.49	89.93

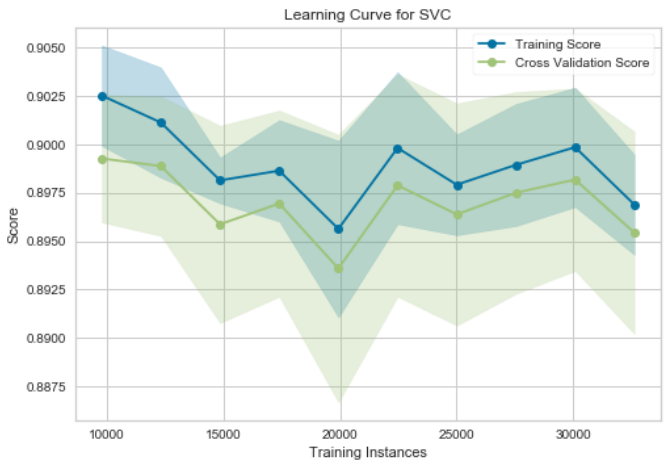


Figure 5 Linear Kernel

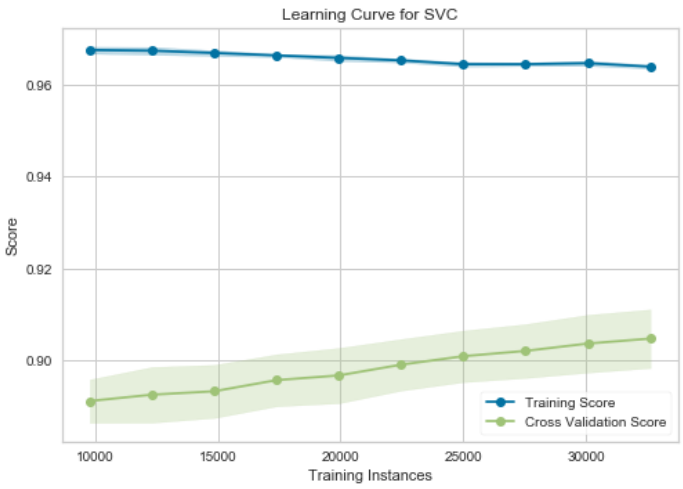
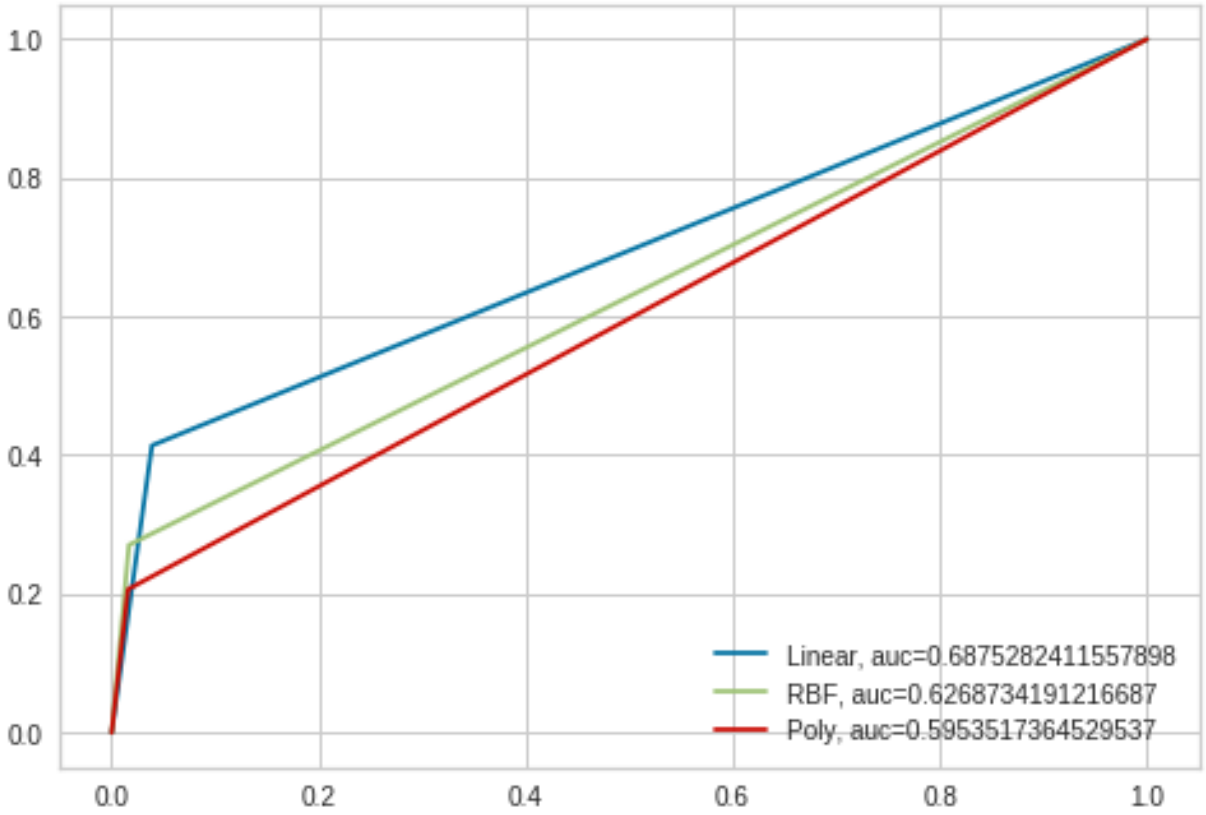


Figure 6 RBF Kernel



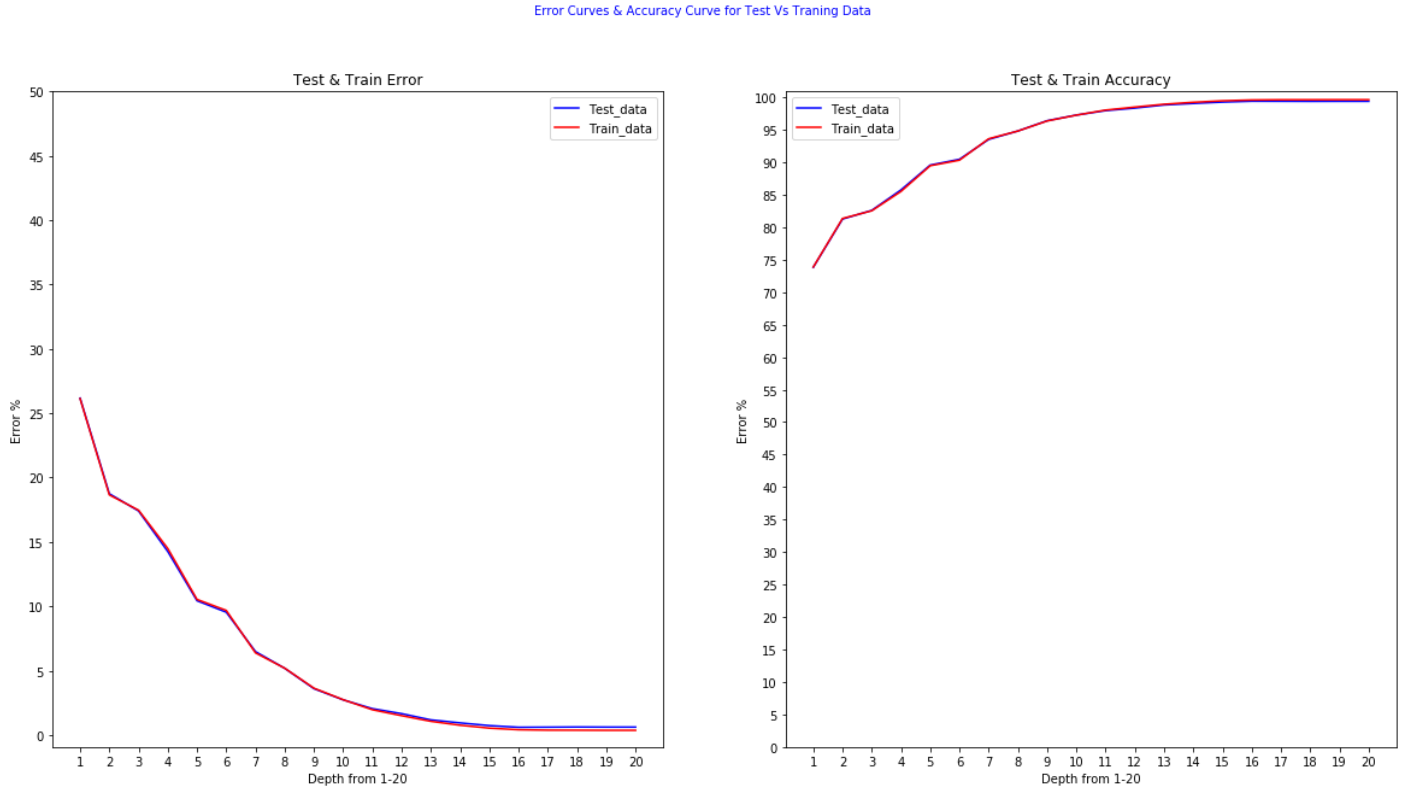
Tuning the parameter of linear kernel i.e. C=0.1 and gamma =0.01 gave slightly better result of 91% accuracy

	precision	recall	f1-score	support
0	0.93	0.97	0.95	8045
1	0.65	0.39	0.49	1017
accuracy			0.91	9062
macro avg	0.79	0.68	0.72	9062
weighted avg	0.90	0.91	0.90	9062

2 Decision Trees

2.1 SGEMM GPU Kernel Performance Dataset

Decision Tree was run for a range of max_depth(1:20) to find the ultimate pruned tree. The metric here used was Information Gain since it used Entropy measure as the impurity measure and splits the node such that it gives the most amount of information gain. The minimum steps required to attain perfect knowledge of a state(zero entropy) were 7 i.e. to predict 0 (low GPU runtime). By looking at the Accuracy & Error plot, the best-pruned tree was found at max_dept=6

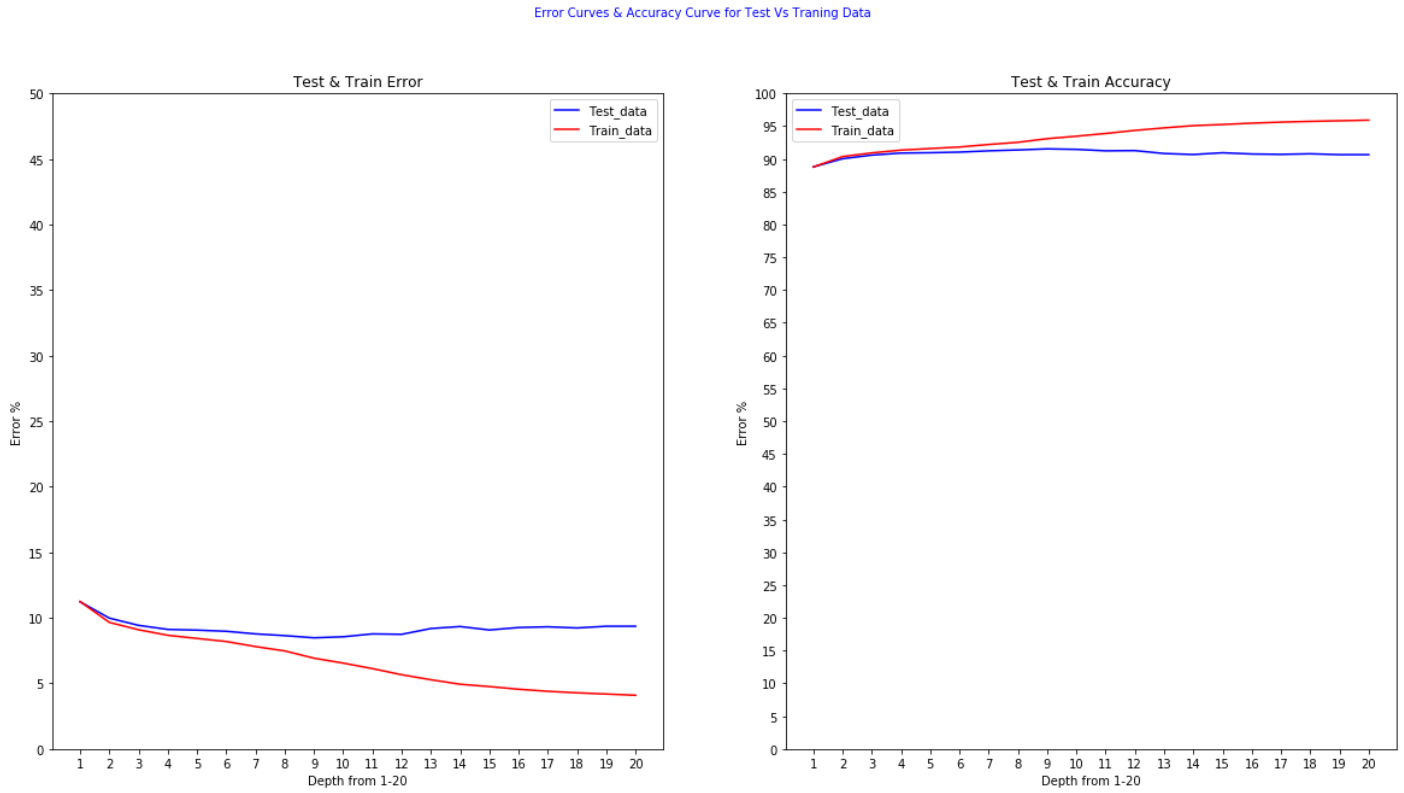


Then the max_dept was set to 6 and rerun the decision tree for two metrics. Here Gini Index was selected as another metric because it measures the divergences between the probability distributions of the target attributes value and splits a node such that it gives the least amount of impurity. The choice of splitting data didn't make much of a difference in the tree performance as the accuracy & error were similar.

Metric	Entropy %	Gini %
Test Accuracy	90.235	90.525
Validation Accuracy	90.20	90.21
Train Accuracy	90.39	90.39
Test Error	0.09765	0.09475
Validation Error	0.0979375	0.0978125
Train Error	0.0960	0.0960

2.2 Bank Marketing UCI Dataset

A similar experiment was performed on the Bank Marketing dataset and based on the accuracy & error plot, max_depth =9 was preferred.



After running the Decision Tree for Information Gain & Gini Index, the accuracy and error for test/validation/train were noted down.

Metric	Entropy %	Gini %
Test Accuracy	91.4	90.97
Validation Accuracy	91.58	91.44
Train Accuracy	93.04	93.41
Test Error	0.085	0.09
Validation Error	0.0841	0.0855
Train Error	0.069	0.065

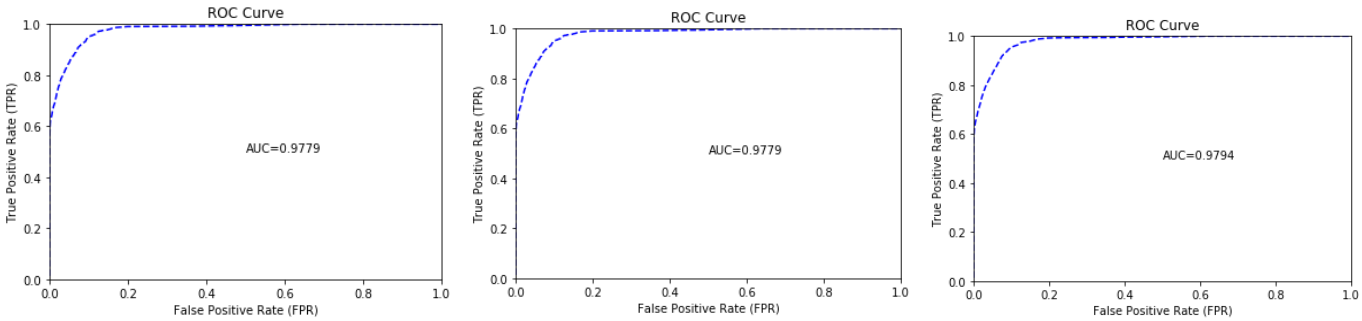
3 Boosting

3.1 SGEMM GPU Kernel Performance Dataset

Firstly, the classification problem was run on H2O Random Forest to set the benchmark for other ensemble models. The max_dept was set as 6 and it was run on 5-fold cross-validation. The experiment was repeated for H2O Gradient Boosting with the same parameters. It is quite evident that the Random Forest was the better model.

	Random Forest	Gradient Boosting	Stacked Ensemble
Test Accuracy	92.13	90.96	92.50
Validation Accuracy	92.44	91.32	92.50
Train Accuracy	92.43	91.34	92.38
Test Error	0.0878	0.0936	0.0777
Validation Error	0.078	0.0898	0.0777
Train Error	0.0819	0.0925	0.076

These models were later stacked to find the optimal combination of base learners. Unlike bagging and boosting, the goal in stacking is to ensemble strong, diverse sets of learners together. The new predictions were better and got an accuracy of 93.92%(an increase of 1.1%) with an error of 0.0618 on the training dataset.

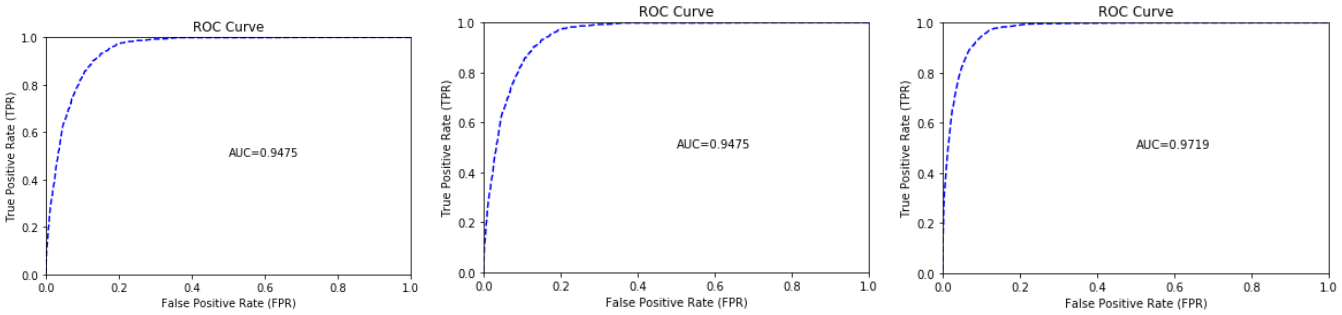


3.2 Bank Marketing UCI Dataset

A similar experiment was conducted on Bank Marketing Dataset and a comparison between Random Forest and Gradient Boosting is shown in the table. Here the max_dept was set to 9.

	Random Forest	Gradient Boosting	Stacked Ensemble
Test Accuracy	90.99	91.25	91.83
Validation Accuracy	91.25	91.69	91.83
Train Accuracy	92.21	93.89	93.91
Test Error	0.0989	0.0957	0.0921
Validation Error	0.1045	0.0934	0.0921
Train Error	0.0821	0.0653	0.0634

Stacked Ensembles of Random Forest and Gradient Boosting slightly increased(1.8%) the performance of model.



Results

Through experimentation, we found that input from multiple models can provide new and unique insight that can help Data Scientists to understand various properties of the data. We also found that hyperparameters tuning like C, gamma, max_depth is effective for better accuracy of the classification. Looking at the performance of the modes, for the **SGEMM GPU Kernel Dataset**, SVM gave a better accuracy of 94%. On the other hand, for **Bank Marketing UCI Dataset,** Stacked Ensemble was a better performing model with accuracy of 93.91%. This concludes that, a single learning cannot be used to learn datasets of various domains.

Also, we saw the importance of feature selection and engineering to improve the performance of the ML algorithm. The best predictors for **SGEMM GPU Kernel** dataset are MWG, NWG, MDIMC, NDIMC, VWM, VWN. The per-matrix 2D tiling at workgroup level, local workgroup size & local memory shape might reduce the GPU runtime. In case of **Bank Marketing**, duration of call had a significant impact on classifying subscription of term deposit by client. One important thing note here is that, this attribute highly affects the output target (e.g., if duration=0 then y="no"). Yet, the duration is not known before a call is performed. Also, after the end of the call y is obviously known. Thus, this input should only be included for benchmark purposes and in future it will be discarded while building realistic predictive model.

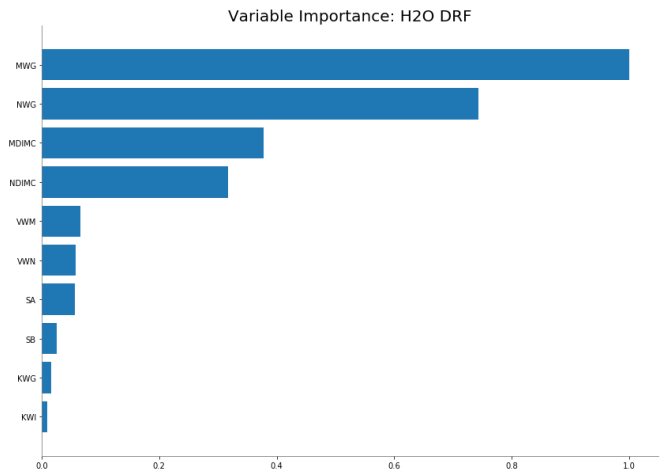


Figure 7 SGEMM GPU Kernel Performance

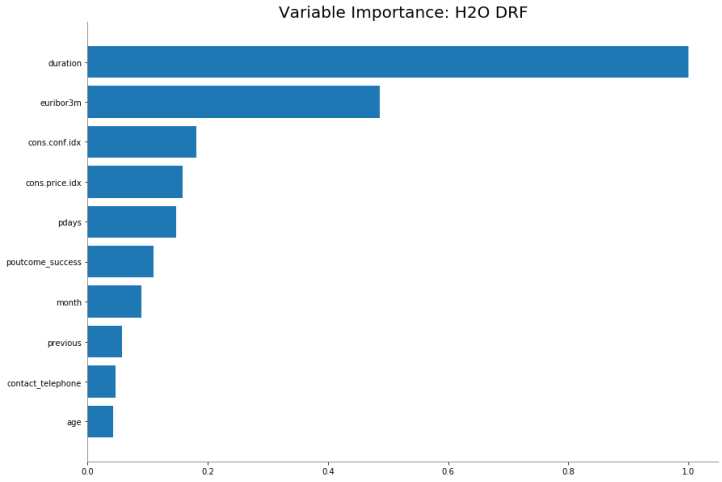


Figure 8 Bank Marketing UCI

Going forward we can further improve the performance of the model by dealing with the imbalanced dataset. For the Bank Marketing Dataset, using Accuracy to measure the goodness of the model which classifies all testing samples into "0" will have an excellent accuracy of 91%, but this model won't provide any valuable information. The alternative evaluation metrics like Precision & Recall should be used. Apart from this, oversampling can be used since the quantity of data is insufficient. .

Citation

[Moro et al., 2014] S. Moro, P. Cortez and P. Rita. A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems, In press, http://dx.doi.org/10.1016/j.dss.2014.03.001

Rafael Ballester-Ripoll, Enrique G. Paredes, Renato Pajarola. Sobol Tensor Trains for Global Sensitivity Analysis. In arXiv Computer Science / Numerical Analysis e-prints, 2017 (https://128.84.21.199/abs/1712.00233)

Cedric Nugteren and Valeriu Codreanu. CLTune: A Generic Auto-Tuner for OpenCL Kernels. In: MCSoC: 9th International Symposium on Embedded Multicore/Many-core Systems-on-Chip. IEEE, 2015 (http://ieeexplore.ieee.org/document/7328205/)