Implementation of Classifying Methods

Introduction

In this project, the objectives are to analyze capabilities of three classifying methods, including SVM. Decision Tree and Random Forest, on a specific dataset in terms of accuracy pf prediction and run time. Moreover, in each technique the effect of hyperparameters are illustrated in plots.

About the Dataset

The dataset consists of 18 features and 241600 records. The first14 features demonstrate specific parameters of a Graphic Processing Unit (GPU) recorded as integer numbers and the last 4 columns include run time of 4 specific program code in seconds on the unit with the mentioned specification. The objective is to classify the units in two distinct classes as high speed and low speed.

Data Preparation

In the preprocessing stage, the dataset is analyzed and prepared for the prediction purpose. Following is the list of actions preformed on dataset.

- A review of the data for missing values shows that we don't have any missing values in the dataset.
- Since the objective is to classify the units in two groups, the average of 4 last rows for each record of data is calculated and then, these columns are replaced with a single target column. The criterion of (average value=100) is used to classify the target column to classes of 0 and 1. Finally, we have 14 features and 1 target.

Algorithm Implementation

The classification algorithm is implemented classify the records as "low speed" and "high speed". To do so, three different machine learning techniques, namely, SVM, Decision Tree and Random Forest are utilized. The algorithm is implemented with options to change the hyperparameters of each method in order to find the proper values for each model.

Project Outline

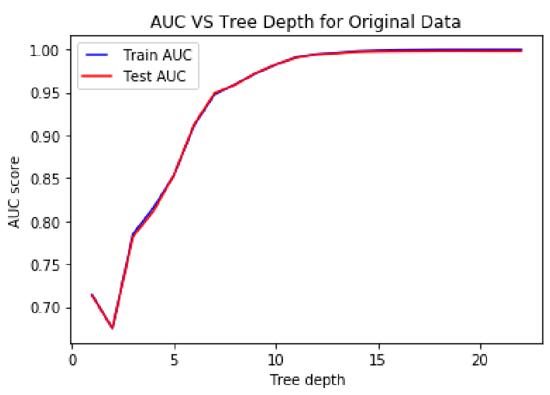
The Project is outlined in 3 parts:

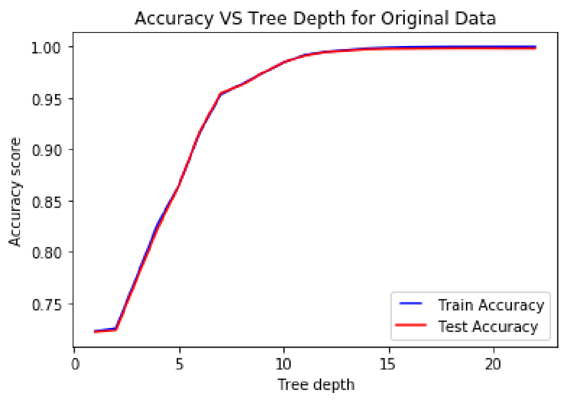
- Decision Tree
- Random Forest
- SVM

Part 1: Implementation of Decision Tree

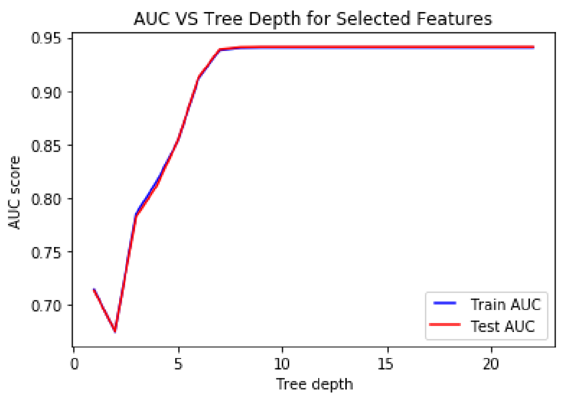
In the first step, implementation of Decision Tree as a classifier is analyzed on our dataset. The dataset is split into train and test set using fraction of 0.3.

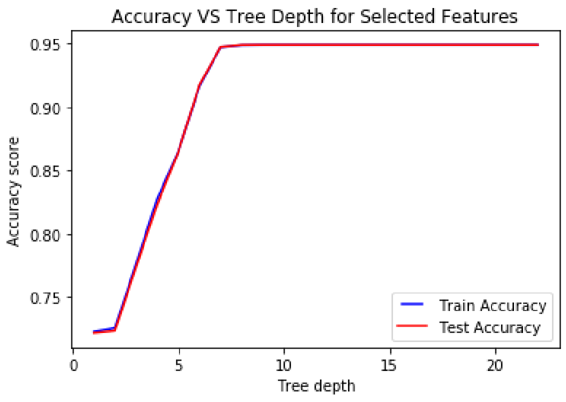
Experimentation 1: Here, the effect of Tree Depth is analyzed. For this experiment, the number of tree depth is changed from 2 to 20 with criterion parameter of “gini”. Then the model is trained and the prediction is done on test dataset. Following plots depict the capability of decision Tree with various Tree Depths by means of AUC score and Accuracy score.





Experimentation 2: In this experiment, the effect of Tree Depth is analyzed on the dataset after feature selection. The number records is so higher that the run time rises dramatically. Therefore, as a compromise in terms of accuracy and run time, feature selection is performed on the dataset and the first 5 most influential features are selected and the rest are dropped. For feature selection, Linear SVC model with C=0.000002 and penalty='l1' is used. Then the Decision Tree Classifier is applied on the dataset. Just the same as the last experiment, the number of Tree Depth value is changed from 2 to 20 with criterion of “gini”. Following plots depict the capability of Decision Tree with different Tree Depths by means of AUC score and Accuracy score.





Results:

The accuracy score of trained data for original and feature selected dataset is shown in the following table. As can be seen, the accuracy score on trained data is 1 and for test data is 0.948. Furthermore, accuracy score of the test data for original and Feature reduced data is 0.998 and 0.945, respectively.

	accuracy_score (y_train,y_pred_train)	accuracy_score (y_test,y_pred_test)
Original Dataset	1.0	0.9981926048565122
Feature Selected Dataset	0.9487523651844844	0.9488272626931568

In the following table, you can observe the confusion matrices for both datasets.

	Confusion matrix (y_pred_train, y_train)	Confusion matrix (y_pred_test, y_test)
Original Data	$\begin{bmatrix} 98863 & 0 \\ 0 & 70257 \end{bmatrix}$	$\begin{bmatrix} 42023 & 57 \\ 74 & 30326 \end{bmatrix}$
Feature Reduced Data	$\begin{bmatrix} 97826 & 7630 \\ 1037 & 62627 \end{bmatrix}$	$\begin{bmatrix} 41650 & 3262 \\ 447 & 27121 \end{bmatrix}$

Discussion:

After analyzing the results, the following points are observed:

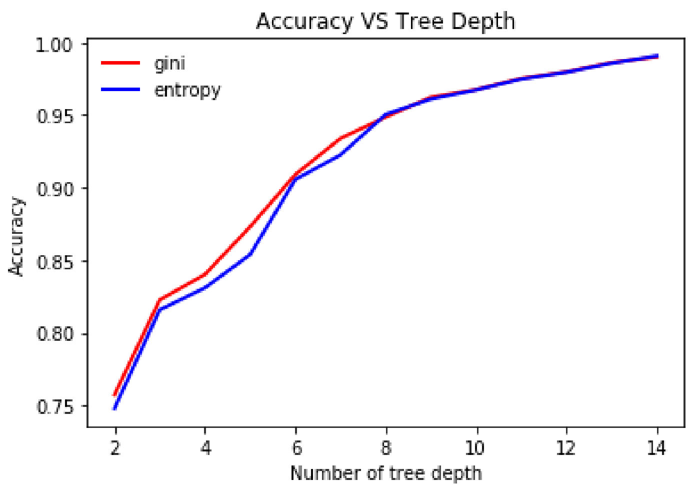
- The run time of Feature reduced dataset is less than half of the runtime for original dataset.
- AUC and Accuracy increase as the Depth of Tree increases.

- After a specific number of Depth (13 for original and 8 for feature reduced dataset), AUC and Accuracy don't change with increase of tree depth.
- In both experimentations, "gini" and "entropy" give almost the same results.
- Feature reduction reduces run train and test run time around 60% while decreasing the accuracy less than 5%. This notation can be used in large datasets.

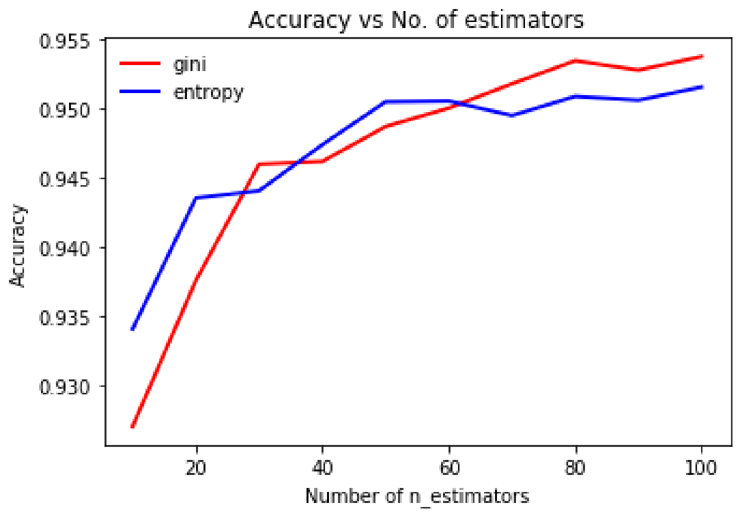
Part 2: Implementation of Random Forest

In this part, implementation of Random Forest classifier is analyzed on dataset. The dataset is split into train and test set using fraction of 0.3.

Experimentation 1: In this experimentation, the effect of Tree Depth in Random Forest is investigated. For this experiment, the number of tree depth is changed from 2 to 14 with criterion of "gini" and "entropy". The number of estimators is kept constant at 50. Then the model is trained and the prediction is done on test dataset. Following plots depict the capability of Random Forest with different Tree Depth by means of Accuracy score.



Experimentation 2: In this step, the effect of number of estimators in Random Forest is investigated. For this purpose, the number of estimators is changed from 10 to 100 (in steps of 10) with criterion of "gini" and "entropy" in classifier model. Here, number of Tree Depth is kept constant at 8. Then the model is trained and the prediction is done on test dataset. Following plots depict the capability of Random Forest with different number of estimators by means of Accuracy score.



Results:

Based on the experimentations, the following values of test Accuracy for gini and entropy in different tree depth and estimators are acquired.

Tree Depth	gini	entropy
2	0.757408940397351	0.7480684326710817
3	0.8227511037527594	0.8159216335540839
4	0.8402593818984547	0.8310430463576159
5	0.8730960264900662	0.8538907284768212
6	0.9090921633554084	0.905711920529801З
7	0.9339955849889625	0.9226131346578367
8	0.9487168874172185	0.9505104856512141
9	0.9623620309050772	0.9611341059602649
10	0.9676048565121412	0.967370309050772б
11	0.9752897350993377	0.9748896247240618
12	0.9798013245033113	0.9795667770419426
13	0.9862306843267108	0.9858305739514349
14	0.9901903973509933	0.9907974613686534

No. Estimators	gini	entropy
10	0.9270557395143488	0.9340921633554083
20	0.9375965783664459	0.9435568432671082
30	0.9459988962472407	0.9440811258278146
40	0.9461920529801324	0.9474061810154525
50	0.9487168874172185	0.9505104856512141
60	0.9500551876379691	0.9505656732891832
70	0.9518073951434879	0.9495171081677705
80	0.953476821192053	0.9508967991169978
90	0.9528007726269315	0.9506208609271524
100	0.9537803532008831	0.9515728476821192

Furthermore, the importance of features in prediction is presented in the following table. As it is obvious, the 5 most important features are the one with numbers 1,2,4,5 and 9.

Feature No.	Feature Importance (Implementation 1)	Feature Importance (Implementation 2)
1	0.32111101	0.369187950
2	0.24114682	0.244409875
3	0.01020494	0.006920159
4	0.13305912	0.130642166
5	0.14763489	0.123802571
6	0.01209841	0.009693779
7	0.01202733	0.008498002
8	0.00155137	0.000210438
9	0.0449622	0.044108681
10	0.03030076	0.032727374
11	0.01531964	0.010620216
12	0.00178149	0.0002.98464
13	0.02141715	0.015989273
14	0.00738487	0.002891049

Discussion:

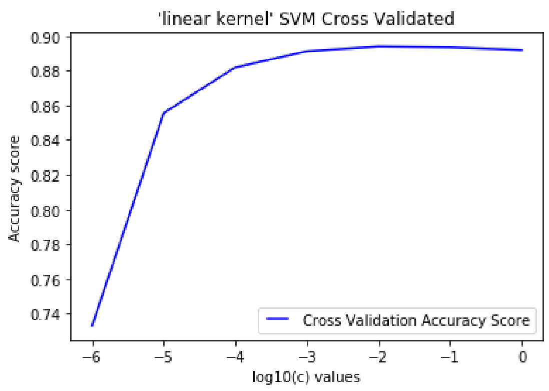
After analyzing the results, the following points are observed:

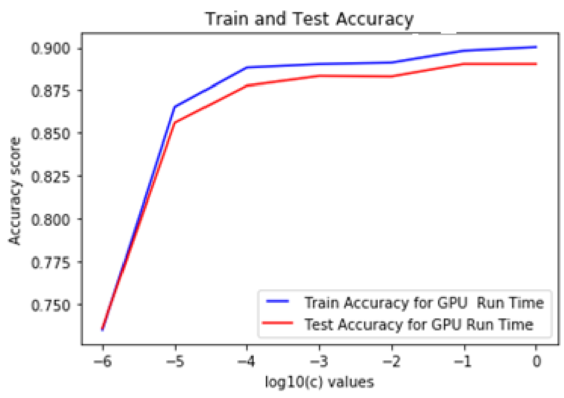
- The increase in number of estimators and tree depths improves accuracy of prediction subsequently.
- In experimentations, “gini” criterion gives slightly better accuracy results than “entropy”.
- Random Forest is faster than the previous model.

Part 3: Implementation of SVM

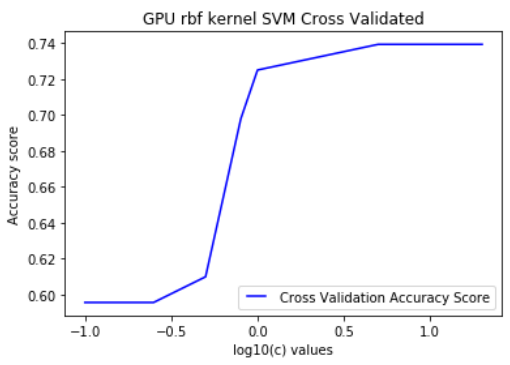
In the last step, SVM method is implemented as a classifier technique to analyze its ability on our classification problem. The dataset is split into train and test set with fraction of 0.3. To analyze the power of SVM classifier, hyperparameter tuning is performed in each experimentation and the results are illustrated in different plots. Since the run time of SVM on datasets with larger features increase drastically, feature selection could be applied on the dataset and as calculated in the last part, then only 5 most important features would be included in the dataset.

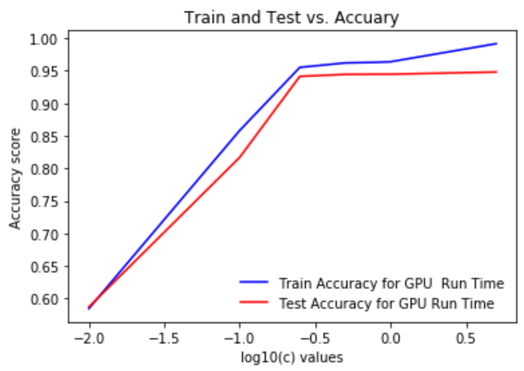
Experimentation 1: In the first experiment, SVM model with “linear” kernel and different C values of {10^-6,10^-5, 10^-4,10^-3,10^-2,10^-1,10^0} is used. The value of ϒ is set to ‘scale’. Also, scoring of 'accuracy' and k-Fold Cross-Validation with k=5 is used. Following plots show the results of prediction.





Experimentation 2: In this experiment, SVM model with “rbf” kernel and different C values of {0.1,0.25,0.5,0.8,1,5,10,20} is used. The value of ϒ is set to ‘scale’. Also, scoring of 'accuracy' and k-Fold Cross-Validation with k=5 is implemented. Following plots show the results of prediction.





Results:

Based on the experimentations, test and train Accuracy increases with the increase of parameter C. But after a specific amount of C, the Accuracy hardly changes.

Optimum hyperparameter for linear kernel SVM is log(C)=0 and for rfb kernel SVM is log(C)=0.5.

Discussion:

After analyzing the results, the following points are observed:

- Increase of C hyperparameter results in increase in train and test Accuracy.

- SVM is very slow for datasets with higher number of features. In this investigation, it took more than 5 hours for the code to run.
- Training and prediction algorithm of SVM method with “linear” kernel is faster than SVM with “rbf” kernel for this dataset.

Conclusion

In this project, three different classifiers were utilized on a large dataset and optimum haperparameters for each model were selected. Comparing the method, following results are inferred.

- Decision tree is much faster than SVM model and is more effective in this data set because the data values in this dataset are all discrete and specific numbers for each feature. For instance, “MWG” feature takes the values of 16, 32, 64 or 128.
- Random Forest is the fastest model with a higher prediction ability. In fact, random forest is one of the most effective methods of classification with in terms of accuracy and speed.
- SVM train time is several times more than that of decision tree or random forest. Therefore, it can be concluded that for higher data dimensions with larger datasets, SVM has the lowest speed among classifiers.
- Only a few kernels work for a specific dataset. Therefore, finding a proper kernel is a challenge.