Machine Learning: CS-7641

Assignment 1: Supervised Learning

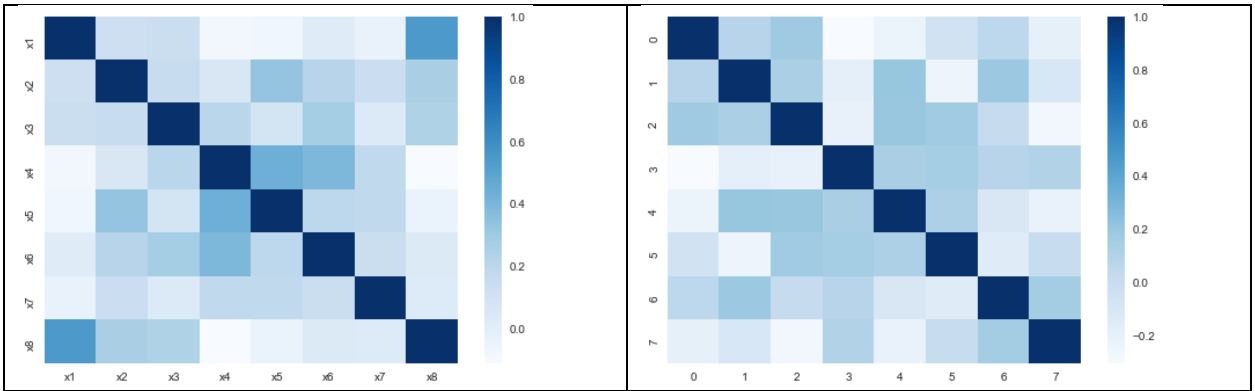
Arjun Bastola
abastola3@gatech.edu

Description of Classification Problems

1. **Dataset 1: Pima Indians Diabetes**
   This binary classification dataset describes the medical records for Pima Indians and whether or not each patient will have an onset of diabetes within five years. This database was downloaded from Kaggle. This dataset contains 8 features and 1 class variable (1: tested positive for diabetes, 0: tested negative for diabetes). There are total of 768 samples and features are not corelated. Our goal is to program 5 classifiers to predict tested positive or negative for diabetes.

2. **Dataset 2: Synthetic Dataset Constructed using sklearn.datasets.make_classification**
   This binary classification dataset contains 2000 samples with 7 different features. The features are not corelated.

You can see the heatmap of dataset 1 and 2 below. The heatmaps show that the features are not corelated in both datasets.



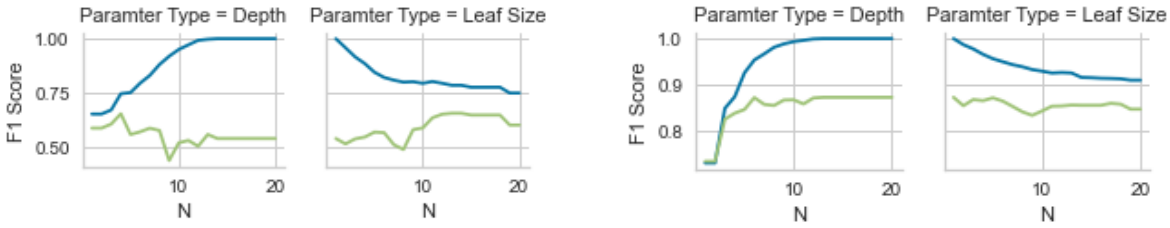
Heatmaps for Dataset 1 and Dataset 2

Classifiers

Decision Tree

Hyperparameters Exploration

For this project, Decision Tree will be hyper tuned by adjusting: ['max_depth', 'min_samples_leaf']. The following charts show how the accuracy is affected when the hyperparameter(s) are changed for both dataset 1 and dataset 2:



F1 Score for different ['max_depth', 'min_samples_leaf'].

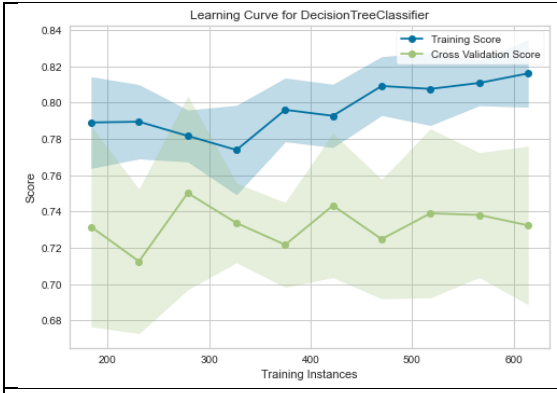
In both datasets, when the value of max_depth is lower, the models underfit. As the max_depth increases, the models overfit. When max_depth increases beyond 10, as the models fit exactly like training data resulting to overfitted models. It looks like a good max_depth for both datasets will be between 5 and 10 based on initial model exploration. We will now use GridSearchCV to find the most optimal max_depth. While we are at it, let's also find the optimal min_samples_leaf.

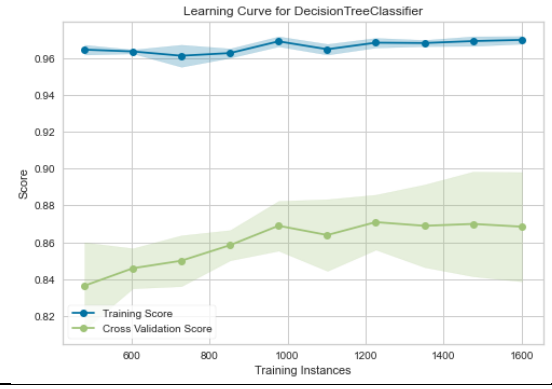
Hyper tuning

Dataset 1 Results:
The optimal value of max_depth was 6. The optimal value of min_samples_leaf was 17. Likewise, the accuracy of Decision Tree classifier was 0.7635918937805731 when the optimized hyperparameter(s) value(s) were used.

Dataset 2 Results:
The optimal value of max_depth was 10. The optimal value of min_samples_leaf was 3. Likewis e, the accuracy of Decision Tree classifier was 0.8764285714285714 when the optimized hyperp arameter(s) value(s) were used.

Looking at the learning curves of both datasets, we can see that our first dataset (with small sample) suffers from high variance (backed up by the spread of light green and light green area). As the training instances increase, there isn't much improvement in cross validation score. The high variance can be reduced using less features and by increasing the training samples. In case of second dataset, as training instances increase, the cross-validation score increases. The predictions for second dataset also have a high variance. As mentioned earlier, we can decrease the variance by removing some features (making our tree less complex) or by using performing ensemble learning like boosting.



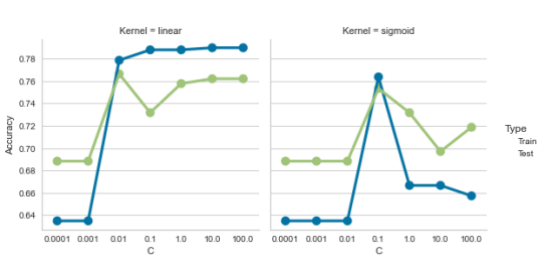


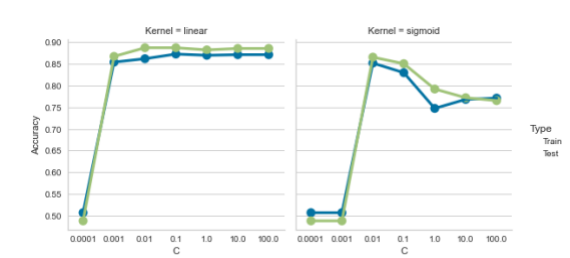
Learning Curves for Dataset 1 and Dataset 2

Support Vector Machine (SVM)

Hyperparameters Exploration

For this project, Support Vector Machine will be hyper tuned by adjusting: ['C', 'kernel']. The following charts show how the accuracy is affected when the hyperparameter(s) are changed for both dataset 1 and dataset 2:





Training and Testing Accuracy for different [‘C’, 'kernel'] for dataset 1 and 2.

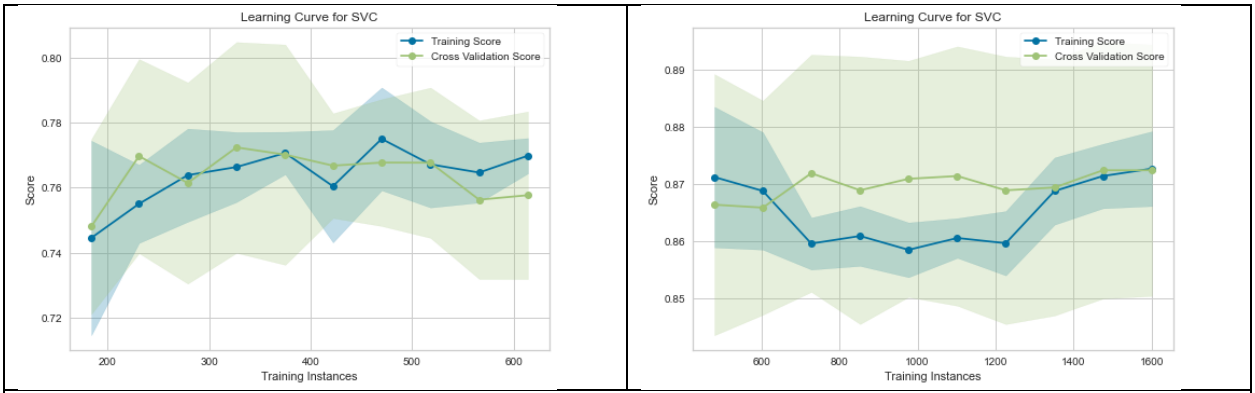
During our early model exploration, we see that “linear” kernel performs better than “sigmoid” kernel in both datasets. For “sigmoid” kernel, as value of “C” increases beyond 0.1, the models start underfitting. In dataset one, the difference between training accuracy and testing accuracy indicating that the dataset suffers from high variance (also deduced in decision tree section). It looks like, for both datasets, the optimal kernel will be “linear” and the optimal value of “C” will be between 0.1 to 100. We will now use GridSearchCV to find the most optimal kernel and value of C.

Hyper tuning

Dataset 1 Results:
The optimal value of C was 1.0. The optimal value of kernel was linear. Likewise, the accuracy of

Support Vector Machine classifier was 0.7820526133610246 when the optimized hyperparameter(s) value(s) were used.

Dataset 2 Results:
The optimal value of C was 100.0. The optimal value of kernel was linear. Likewise, the accurac y of Support Vector Machine classifier was 0.8635714285714287 when the optimized hyperpara meter(s) value(s) were used.



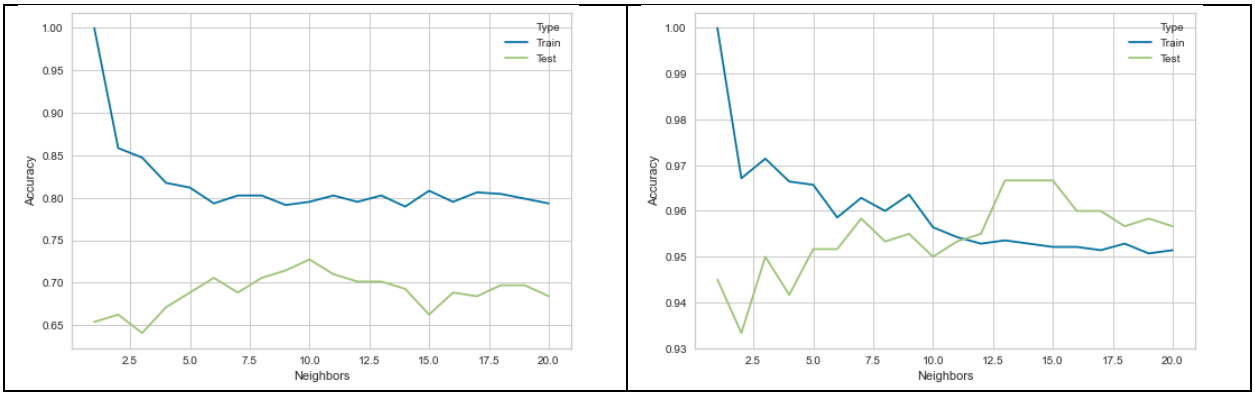
Learning Curves for Dataset 1 and Dataset 2

Looking at the learning curves, for both datasets, the training score and cross validation score converge as the number of training instances increase. Even with less regularization (increased value of C), the models do not improve indicating underfitting. Therefore, this model will benefit from increase training samples for both datasets.

K Nearest Neighbors (KNN)

Hyperparameters Exploration

For this project, K-Nearest Neighbors will be hyper tuned by adjusting: ['n_neighbors']. The follo wing charts show how the accuracy is affected when the hyperparameter(s) are changed for both dataset 1 and dataset 2:



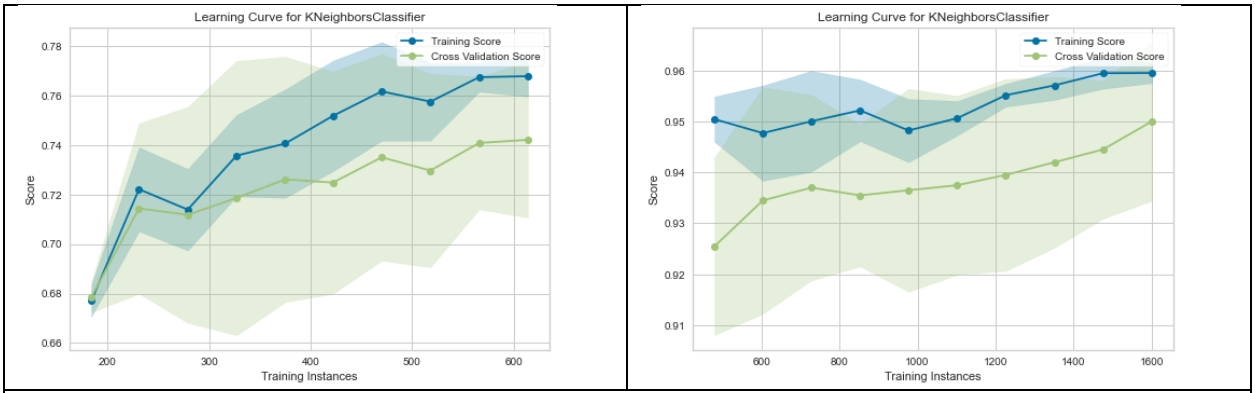
Training and Testing Accuracy for different [‘n_neighbors'] for dataset 1 and 2.

During our early model exploration, both datasets suffer from overfitting for k_neighbors less than 6 indicating overfitting. However, as the value of k increases, the testing and training accuracy start to converge for both datasets. Based on the accuracy graphs, for both datasets, the optimal value of k_neighbors will be around 10. We will now use GridSearchCV to find the most optimal value of k_neighbors.

Hyper tuning

Dataset 1 Results:
The optimal value of n_neighbors was 12. Likewise, the accuracy of K-Nearest Neighbors classif ier was 0.7690377293181032 when the optimized hyperparameter(s) value(s) were used.

Dataset 2 Results:
The optimal value of n_neighbors was 11. Likewise, the accuracy of K-Nearest Neighbors classif ier was 0.9464285714285714 when the optimized hyperparameter(s) value(s) were used.



Learning Curves for Dataset 1 and Dataset 2

Looking at the learning curves, for both datasets, both training score and cross validation score in crease as training instances increase indicating that the model will benefit from more data. We ca n also observe that the bias and variance both decrease as more training samples are introduced f urther solidifying the idea that the model will highly benefit from introduction of more data.

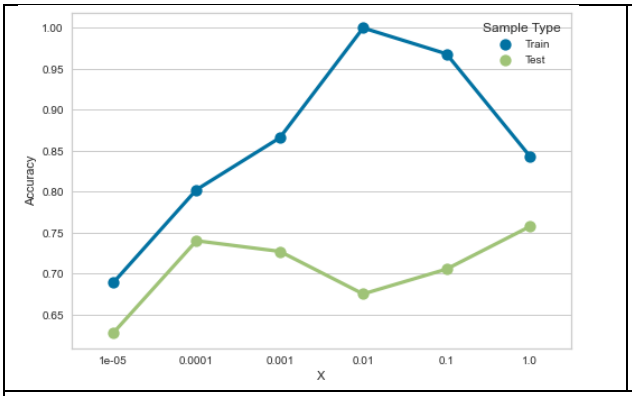
Neural Network

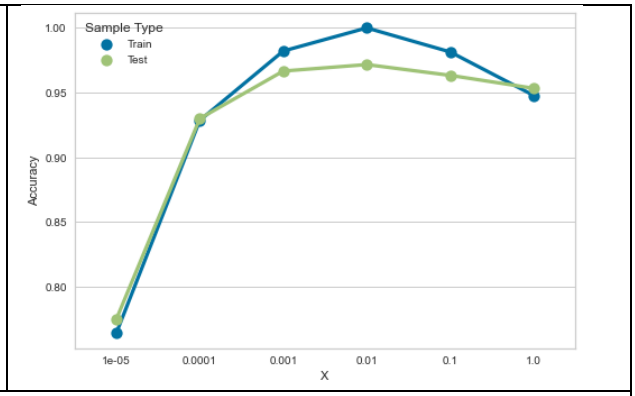
Hyperparameters Exploration

For this project, Neural Network will be hyper tuned by adjusting: ['hidden_layer_sizes', 'learning_rate_init']. We will be using the default activation function “ReLU” for our Neural

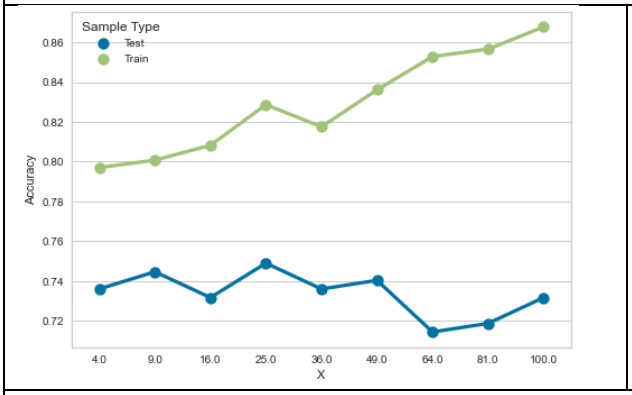
Network Learner. The following charts show how the accuracy is affected when the hyperparamter(s) are changed for both dataset 1 and dataset 2:

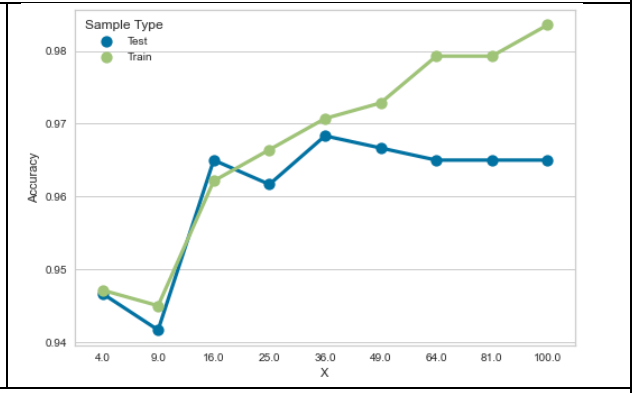
During our early model exploration, both datasets benefit when around learning rate is around 0.01. When the learning rate is set to low, the model underfit for both datasets. However, as learning rate increases, underfitting decreases and test accuracy increases up until 0.1. The test accuracy starts decreasing after learning rate goes beyond 0.1. Similarly, higher the number of hidden layers better the accuracy is for both datasets. We will now use GridSearchCV to find the most optimal learning rate and optimal hidden layers.





Training and Testing Accuracy for different [‘learning_rate_init’] for dataset 1 and 2.





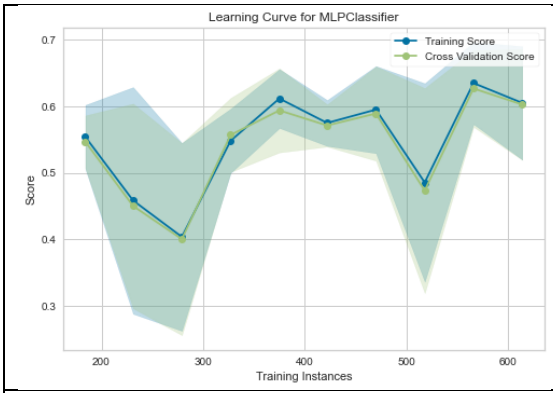
Training and Testing Accuracy for different [‘hidden_layer_sizes’] for dataset 1 and 2.

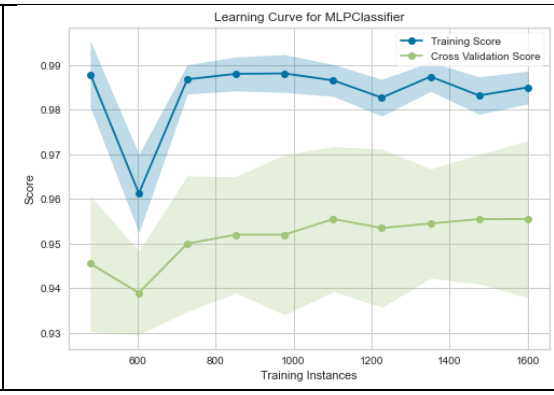
Hyper tuning

Dataset 1 Results:
The optimal value of hidden_layer_sizes was 4. The optimal value of learning_rate_init was 0.01 . Likewise, the accuracy of Neural Network classifier was 0.7840321453529001 when the optimi zed hyperparameter(s) value(s) were used.

Dataset 2 Results:
The optimal value of hidden_layer_sizes was 64. The optimal value of learning_rate_init was 0.0

1. Likewise, the accuracy of Neural Network classifier was 0.9635714285714286 when the opti mized hyperparameter(s) value(s) were used.





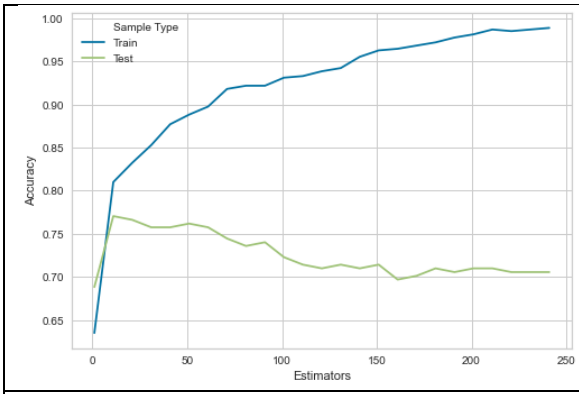
Learning Curves for Dataset 1 and Dataset 2

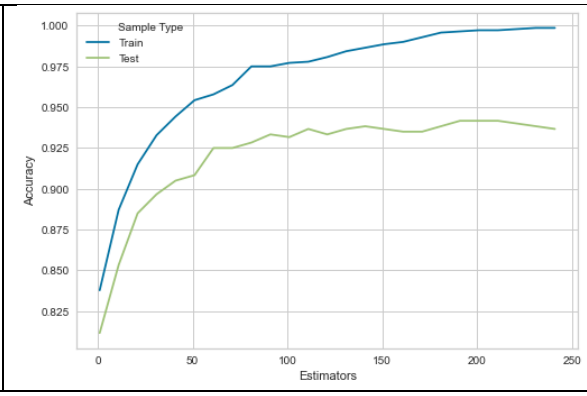
The learning curve of dataset 1 suggests that due to high variance of the data, adding more training instances will not necessarily increase the cross-validation score. However, adding more test instances does remove bias and variance in dataset 1. In dataset 2, the cross-validation score increases as more training instances are introduced indicating that the model will benefit from adding more training data.

Boosting

Hyperparameters Exploration

For our GradientBoostingClassifier models, Gradient Boosting will be hyper tuned by adjusting: ['n_estimators']. we will pre-prune max_depth to 3 and min_samples_leaf to 10. This was done to create quickly create high number of weak trees which will be bagged together to create a powerful GradientBoostingClassifier model. The following charts show how the accuracy is affected when the hyperparameter(s) are changed for both dataset 1 and dataset 2:





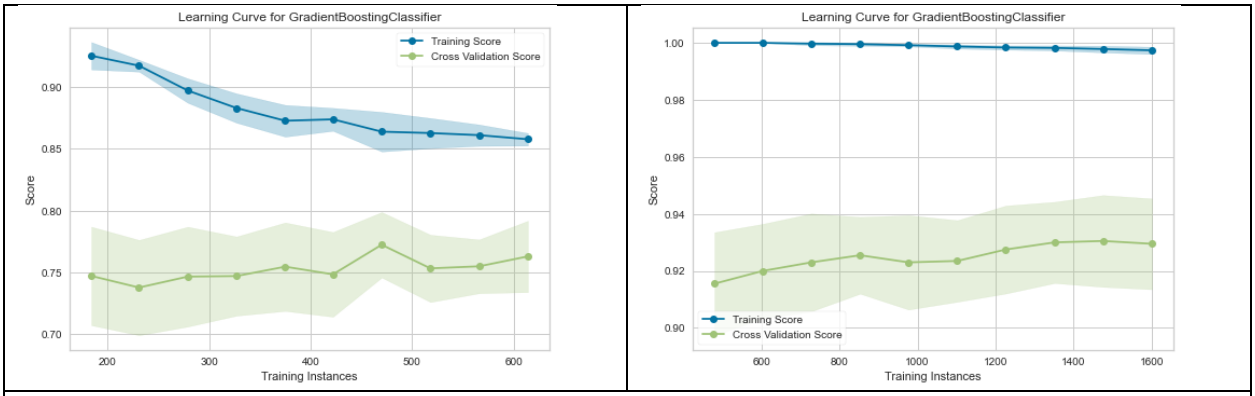
Training and Testing Accuracy for different [‘n_estimators'] for dataset 1 and 2.

During our early model exploration, we can observe that, for dataset 1, the accuracy of the model increases at first then decreases as number of estimators is increased. However, for dataset 2, the accuracy of the model increases as number of estimators is increased and then the model overfits for higher values of estimators. It looks like ideal value of "n_estimators" is between 10 and 50 for dataset 1 and around 200 for dataset 2. We will now use GridSearchCV to find the most optimal "n_estimators".

Hyper tuning

Dataset 1 Results:
The optimal value of n_estimators was 51. Likewise the accuracy of Gradient Boosting classifier was 0.7803284416491962 when the optimized hyperparameter(s) value(s) were used.

Dataset 2 Results:
The optimal value of n_estimators was 241. Likewise the accuracy of Gradient Boosting classifie r was 0.927142857142857 when the optimized hyperparameter(s) value(s) were used.



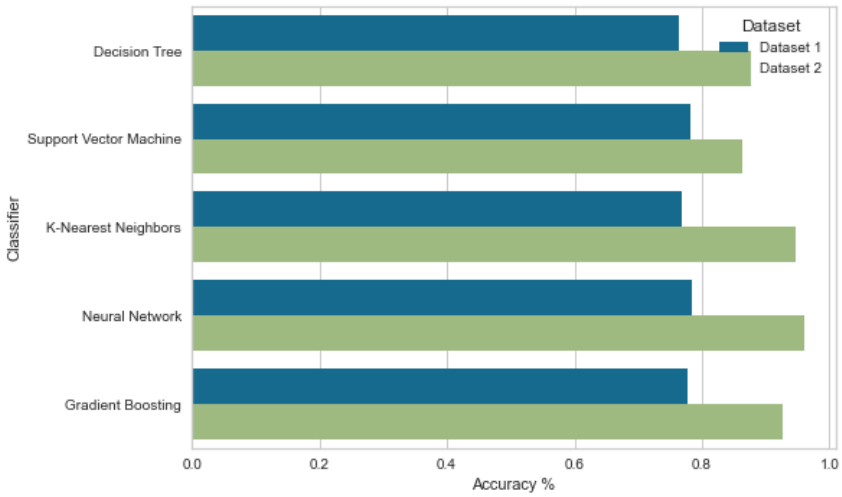
Learning Curves for Dataset 1 and Dataset 2

Looking at the learning curve, for both datasets, cross validation score increase as training instances increases indicating that the model will benefit from more data.

Classifiers Evaluation

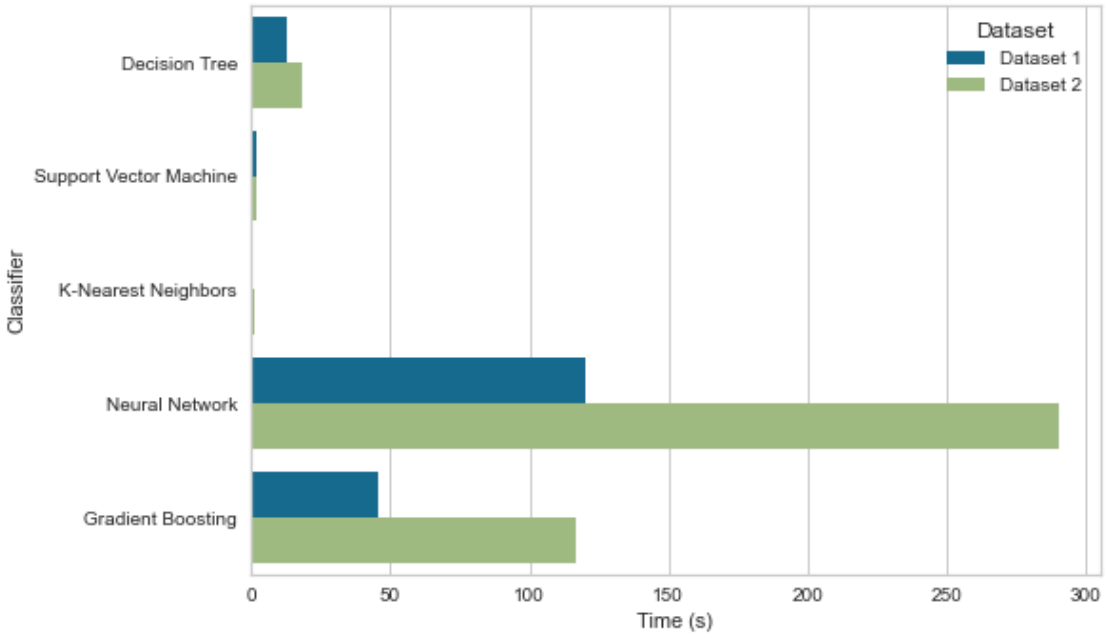
Accuracy (Best Score)

For both datasets, Neural Network was the most accurate (had the best score). K-Nearest Neighbors was the least accurate for dataset 1 whereas SVM was the least accurate for dataset 2.



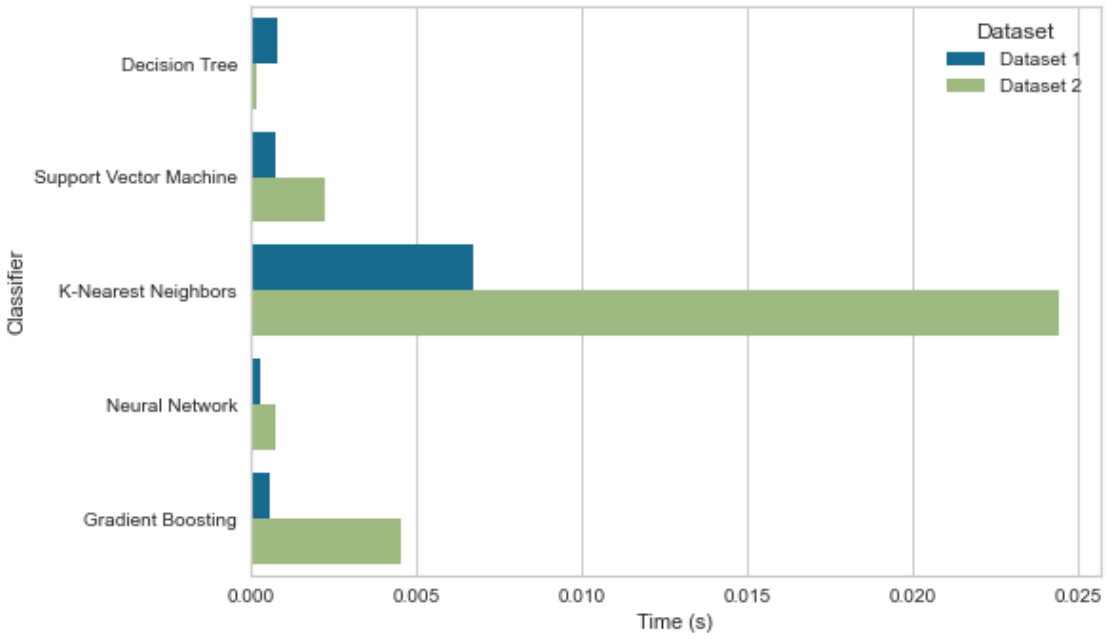
Training Time

For both datasets, Neural Network has the highest training time. K-Nearest Neighbors has the lowest training time for both datasets.



Testing Time

For both datasets, K-Nearest Neighbors had the highest testing time. Neural Network has the lowest testing time for dataset 1 whereas Decision Tree has the lowest testing time for dataset 2.



References

"1. Supervised Learning¶." *Scikit*, scikit-learn.org/stable/supervised_learning.html#supervised-learning.

"CSE6242OAN,O01,O3, Fall 2020 Data and Visual Analytics." *Data and Visual Analytics | CSE6242OAN,O01,O3 Fall 2020 | Georgia Tech*, poloclub.github.io/cse6242-2020fall-online/.

K, Ani Karenovna. "Part II. Evaluating a Predictive Model: Cross Validation and Bias and Variance Tradeoff." *Medium*, Medium, 9 June 2019, medium.com/@karenovna.ak/part-ii-evaluating-a-predictive-model-cross-validation-and-bias-and-variance-tradeoff-9874b836cd2e.

Kumar. "Pima-Indians-Diabetes.csv." *Kaggle*, 27 Feb. 2018, www.kaggle.com/kumargh/pimaindiansdiabetescsv.

"Machine Learning for Trading Course." *CS7646 Machine Learning for Trading*, lucylabs.gatech.edu/ml4t/.