Supervised Learning

Cameron Buster
cbuster6@gatech.edu

***Abstract*—In this paper we investigate five supervised learning algorithms applied to the UCI Machine Learning Repository (UCI-MLR) breast cancer and steel plate fault dataset. We begin by describing the shape of each dataset, feature vectors, and target variable. We then discuss each of the five shallow machine learning algorithms in depth as well as the hyperparameters that contribute heavily to each model's performance. We conclude by presenting each model's performance on the training and testing set, hyperparameter plots, and learning curves.**

1 CLASSIFICATION PROBLEMS

In Sec. 1, we will describe the breast cancer[1] and steel plate fault[2,3] datasets from the UCI Machine Learning Repository.

1.1 Breast Cancer Dataset

The UCI-MLR breast cancer dataset is a multivariate binary classification-oriented dataset with 286 instances of nine categorical attributes that describes whether the patient had a recurrence event or no recurrence event.

1.1.1 *Feature Vectors*

The nine categorical attributes that describe whether or not a recurrence event was experience include the patient's age (10-19, 20-29, 30-39, 40-49, 50-59, 60-69, 70-79, 80-89, 90-99), menopause status (less than 40 years of age, greater than 40 years of age, premenopausal), size of tumor (0-4, 5-9, 10-14, 15-19, 20-24, 25-29, 30-34, 35-39, 40-44, 45-49, 50-54, 55-59), number of axillary lymph nodes that contain metastatic breast cancer (0-2, 3-5, 6-8, 9-11, 12-14, 15-17, 18-20, 21-23, 24-26, 27-29, 30-32, 33-35, 36-39), whether the metastasized cancer has remained contained in the capsule of the lymph node (yes, no), degree of malignancy (1, 2, 3),

[1] UCI Machine Learning Repository: Breast Cancer Data Set
[2] UCI Machine Learning Repository: Steel Plates Faults Data Set (original)
[3] Faulty Steel Plates | Kaggle (with headers)

breast (left, right), breast quadrant (left-up, left-low, right-up, right-low, central), whether the patient has received radiation treatment (yes, no).

1.1.2 *Target Variable*

As mentioned above in Sec. 1.1, this dataset represents a multivariate binary classification problem. Specifically, we seek to predict whether there was or was not a recurrence event experienced by the patient after at least one year of total remission.

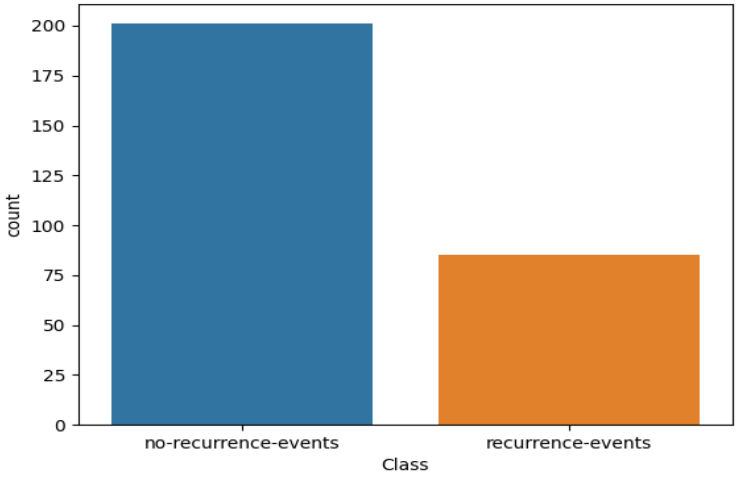


Figure 1 – Binary class target counts for UCI-MLR breast cancer dataset.

In Fig. 1, we see that 201 patients did not experience a recurrence event after at least one year of total remission. 86 patients did experience a recurrence event after at least one year of total remission.

1.2 Steel Plate Faults Dataset

The UCI-MLR steel plate faults dataset is a multivariate multiclass classification-oriented dataset with 1941 instances of 27 integer and real number valued attributes that describes the type of surface fault observed in manufactured steel plates.

1.2.1 *Feature Vectors*

The 27 categorical attributes that describe whether a surface fault was observed in a steel plate include X_Minimum, X_Maximum, Y_Minimum, Y_Maximum,,

Pixels_Areas, X_Perimeter, Y_Perimeter, Sum_of_Luminosity, Minimum_of_Luminosity, Maximum_of_Luminosity, Length_of_Conveyer, TypeOfSteel_A300, TypeOfSteel_A400, Steel_Plate_Thickness, Edges_Index, Empty_Index, Square_Index, Outside_X_Index, Edges_X_Index, Edges_Y_Index, Outside_Global_Index, LogsOfAreas, Log_X_Index, Log_Y_Index, Orientation_Index, Luminosity_Index, and SigmoidOfAreas. Unfortunately, there is no further available descriptors for these feature vectors as there are with the feature vectors described in Sec. 1.1.1. Only that all 27 are integer and real-valued.

1.2.2 *Target Variable*

As mentioned above in Sec. 1.2.1, this dataset represents a multivariate multiclass classification problem. Specifically, we seek to predict whether there was or was not one of seven possible surface faults observed in a steel plate – Pastry, Z_Scratch, K_Scatch, Stains, Dirtiness, Bumps, or Other_Faults. These targets are encoded 0, 1, 2, 3, 4, 5, and 6, respectively.

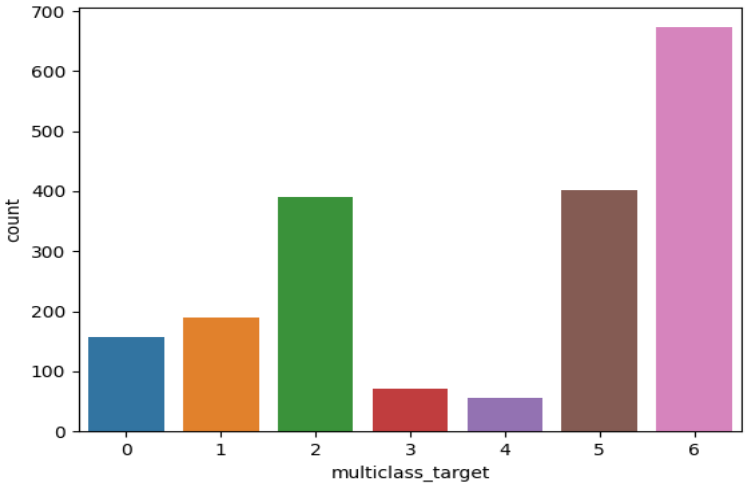


Figure 2 – Multiclass target counts for UCI-MLR breast cancer dataset.

In Fig. 2, we see of the 1941 steel plate surface faults, 158 were Pastry, 190 were Z_Scratch, 391 were K_Scatch, 72 were Stains, 55 were Dirtiness, 402 were Bumps, and 673 were Other Faults.

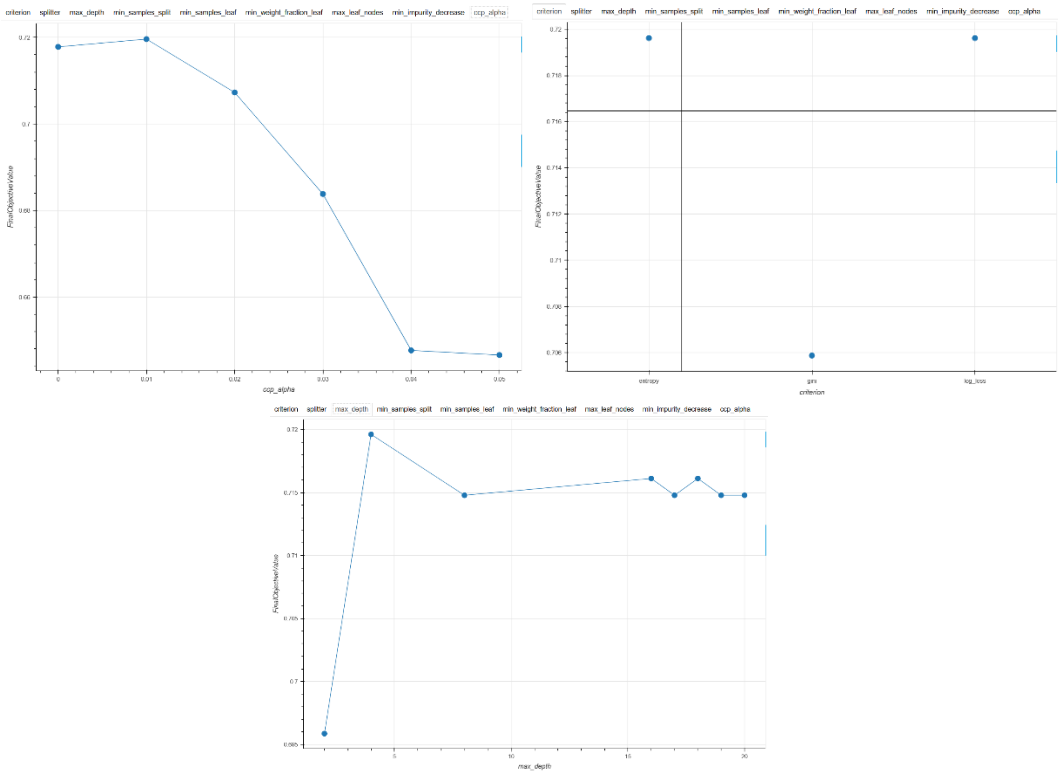
2 TRAINING VS. TESTING

In Sec. 2, we discussed each of the five machine learning algorithms used at a high level and introduced some of the more specific implementations of these algorithms with our sklearn==1.2.1 library. In Sec. 3, we will present learning curves that compare the training scores versus the cross-validation scores (k=5) for hyperparameters of interest for each of the five algorithms.

2.1 Decision Tree Learning Curves

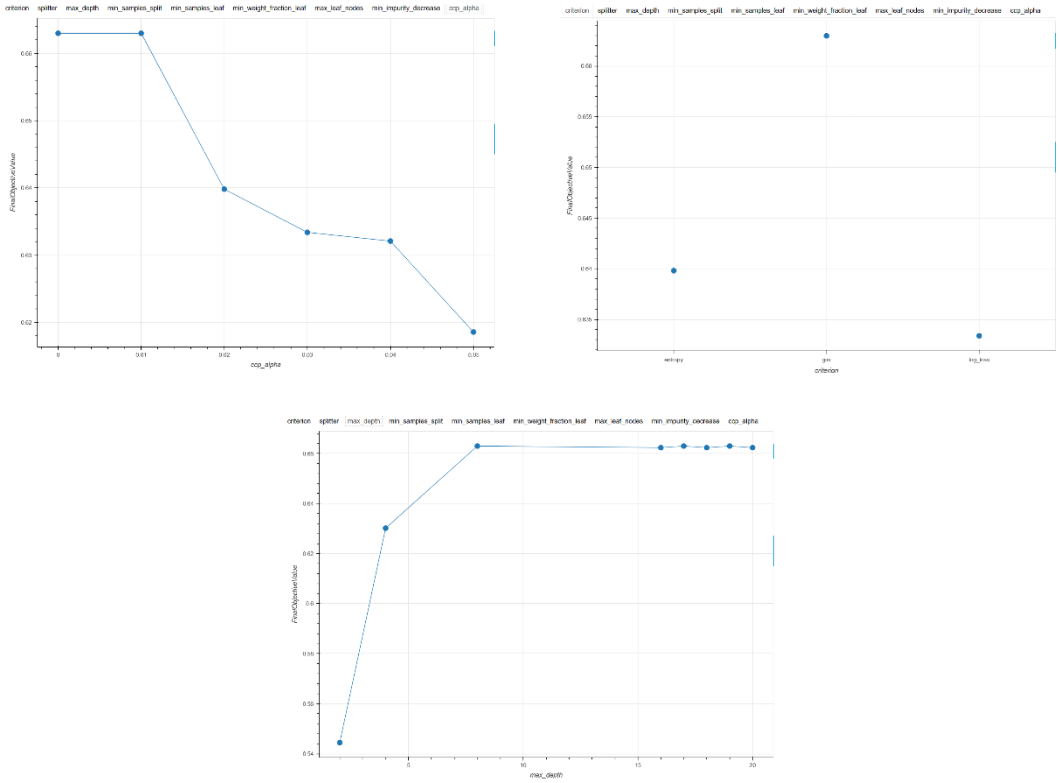
Although the hyperparameter grid was quite large, we are interested in three hyperparameters specifically – criterion (information gain), max_depth (tree depth), and ccp_alpha (pruning).

2.1.1 *Breast Cancer*



The ccp_alpha values uses were 0.0, 0.01, 0.02, 0.03, 0.04, and 0.05. The criterion values used were entropy and gini. The max depth values used were 2, 4, 8, 16, 17, 18, 19, and 20. The highest achieved ROCAUC score achieved on the test set was 0.72 with ccp_alpha = 0.01, criterion = entropy, and max depth = 4.

2.1.2 *Steel Plate Faults*

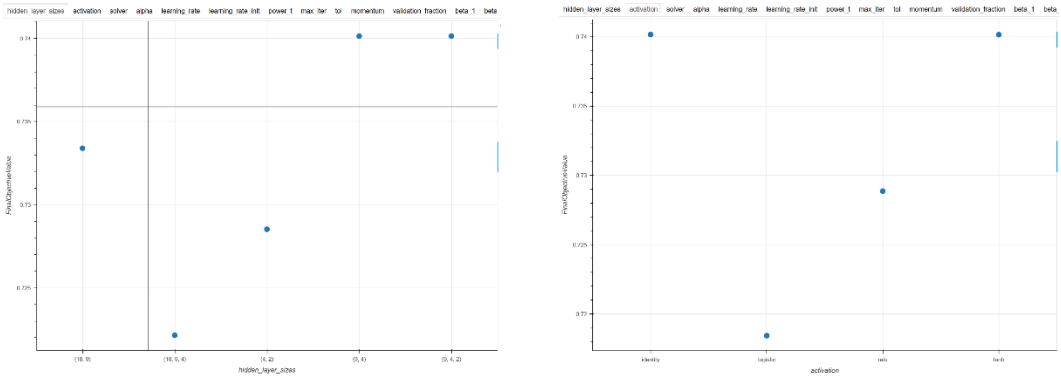


The same hyperparameter grid was used from 3.1.1. The highest achieved ROCAUC score achieved on the test set was 0.663 with ccp_alpha = 0.01, criterion = gini, and max depth = 8.

2.2 Neural Network Learning Curves

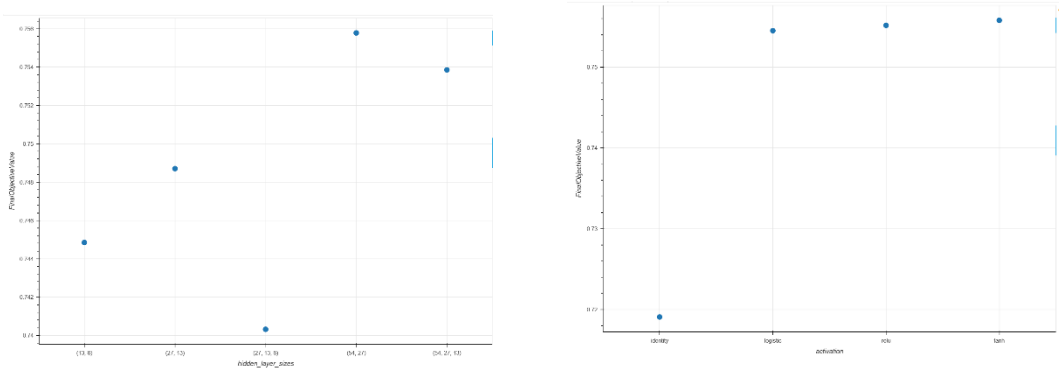
We are interested in examining different number of layers, different numbers of nodes in each layer, and different activation functions.

2.2.1 *Breast Cancer*



The hidden layer structures used are (18, 9, 4), (9, 4, 2), (18, 9), (9, 4), and (4, 2). The activation functions used were identity, logistic, relu, and tanh. The highest ROCAUC achieved on the test set was 0.74 on both (9, 4, 2) and (9, 4) with activation functions of identity and hyperbolic tangent.

2.2.2 *Steel Plate Faults*

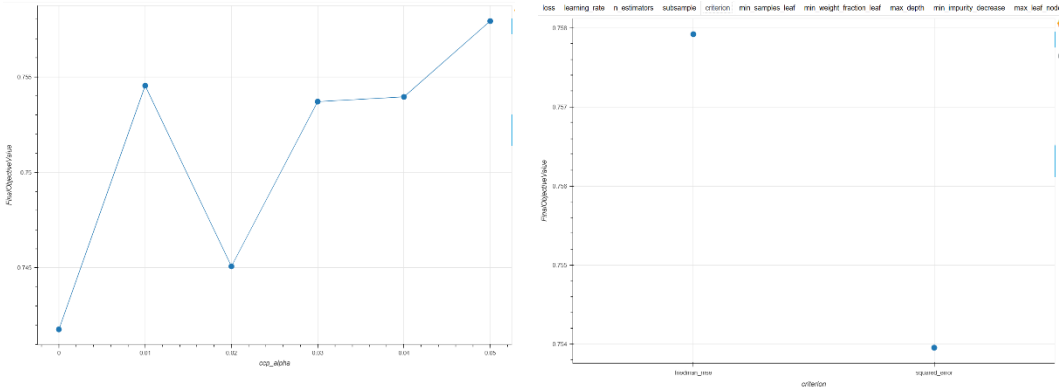


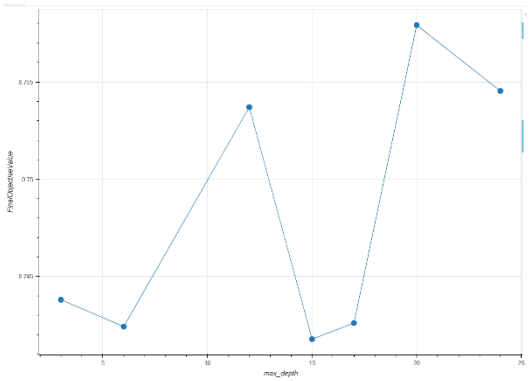
The hidden layer structures used are (54, 27, 13), (27, 13, 6), (54, 27), (27, 13), and (13, 6). The activation functions used were identity, logistic, relu, and tanh. The highest ROCAUC achieved on the test set was 0.756 on (54, 27) with an activation function hyperbolic tangent.

2.3 Boosting Learning Curves

For boosting, we use a gradient boosting classifier which is a boosted version of the decision tree used above. We compare the same hyperparameters as we did with the decision tree - criterion (information gain), max_depth (tree depth), and ccp_alpha (pruning).

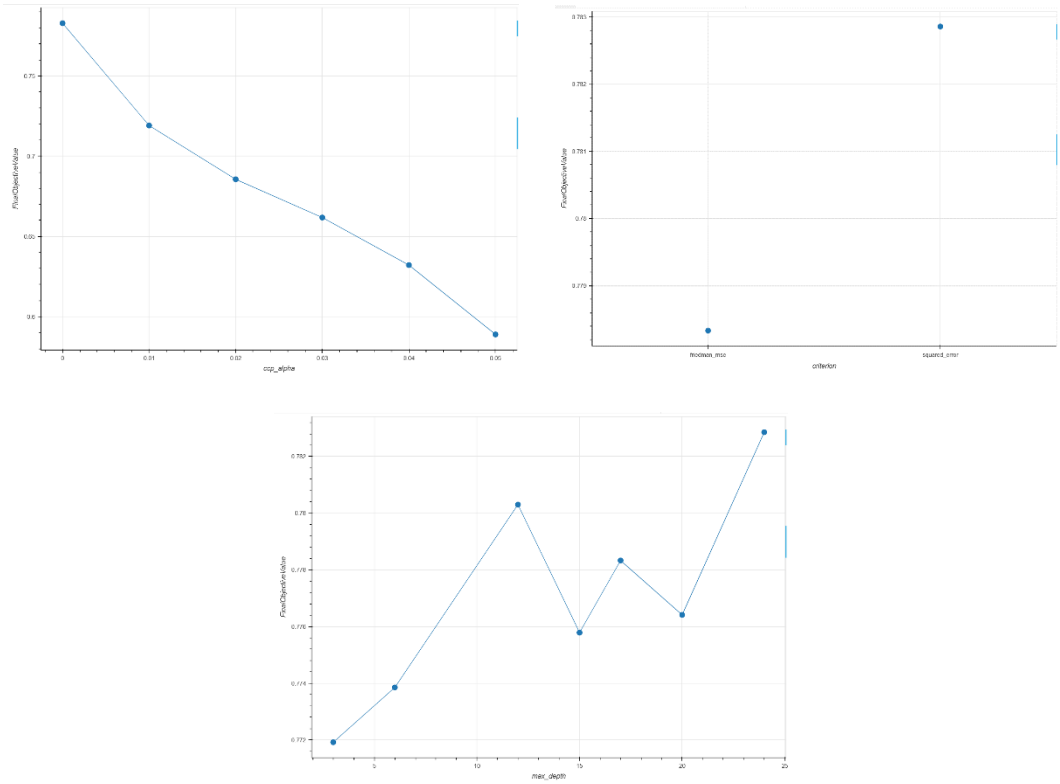
2.3.1 *Breast Cancer*





The same hyperparameter grid was used from Sec. 3.1 with the exception of the criterion – we use friedman_mse and squared_error.. The highest achieved ROCAUC score achieved on the test set was 0.758 with ccp_alpha = 0.05, criterion = friedman_mse, and max depth = 20.

2.3.2 *Steel Plate Faults*

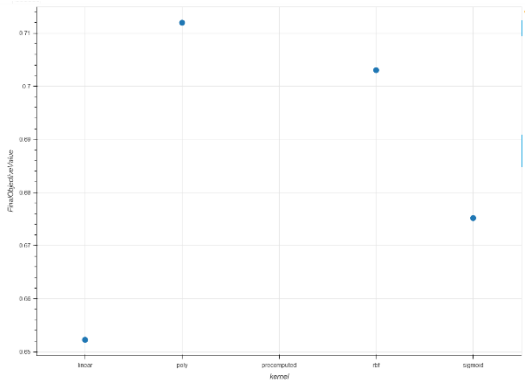


The same hyperparameter grid was used from Sec. 3.1 with the exception of the criterion – we use friedman_mse and squared_error.. The highest achieved ROCAUC score achieved on the test set was 0.784 with ccp_alpha = 0.0, criterion = squared_error, and max depth = 24.

2.4 Support Vector Machine Learning Curves

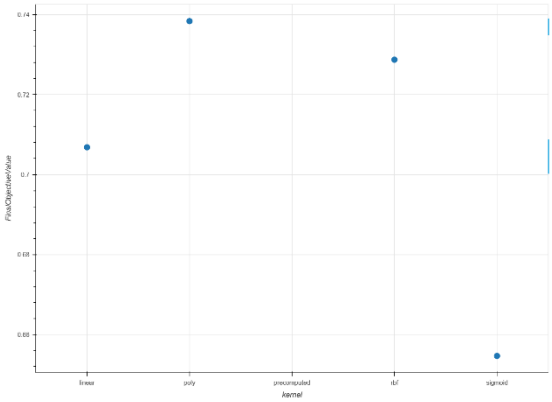
Our support vector machines hyperparameter grid was smaller and even though the quadratic fit equation can cause long training times, the datasets chosen along with the size of the hyperparameter grid made training times very reasonable. More on wall clock time below in Sec. 4. We are interested in kernel functions.

2.4.1 *Breast Cancer*



The kernel functions used are linear, poly, precomputed, rbf, and sigmoid. The highest achieved ROCAUC score on the test set was 0.712 with a kernel of poly.

2.4.2 *Steel Plate Faults*

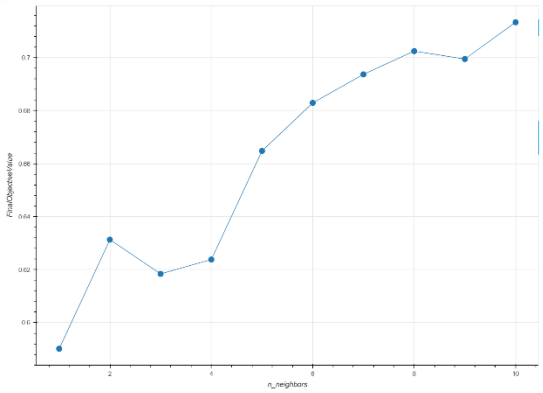


The kernel functions used are linear, poly, precomputed, rbf, and sigmoid. The highest achieved ROCAUC score on the test set was 0.738 with a kernel of poly.

2.5 K-Nearest Neighbor Learning Curves

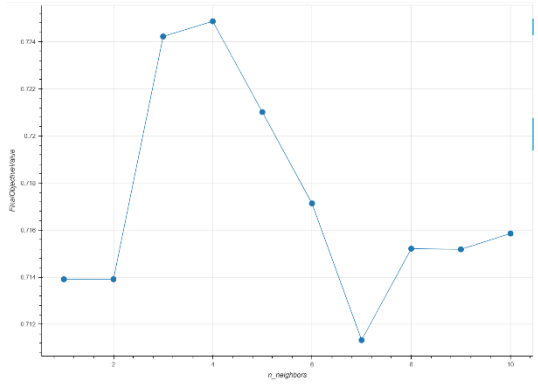
For KNN, the hyperparameter of choice is different values of k.

2.5.1 *Breast Cancer*



The choice of k ranged between 1 and 10. The highest achieved ROCAUC score on the test set was 0.714 with a k value of 10.

2.5.2 *Steel Plate Faults*



The choice of k ranged between 1 and 10. The highest achieved ROCAUC score on the test set was 0.725 with a k value of 4.

3 ANALYSIS OF RESULTS

In Sec. 3, we presented the hyperparameter learning curves for each model and each dataset as well as the mean ROCAUC score achieved with the optimal hyperparameters. In Sec. 4, we will present the results in their entirety.

3.1 Decision Tree Results

3.1.1 *Breast Cancer*

The ROCAUC training score was 0.778 whereas the test set score was 0.590 with a 5-fold cross validation score of 0.720.

3.1.2 *Steel Plate Faults*

The ROCAUC 5-fold cross validation score was 0.663.

3.2 Neural Network Results

3.2.1 *Breast Cancer*

The ROCAUC training score was 0.731 whereas the test set score was 0.686 with a 5-fold cross validation score of 0.740.

3.2.2 *Steel Plate Faults*

The ROCAUC 5-fold cross validation score was 0.756.

3.3 Boosting Results

3.3.1 *Breast Cancer*

The ROCAUC training score was 0.716 whereas the test set score was 0.626 with a 5-fold cross validation score of 0.758.

3.3.2 *Steel Plate Faults*

The ROCAUC 5-fold cross validation score was 0. 784.

3.4 Support Vector Machine Results

3.4.1 *Breast Cancer*

The ROCAUC training score was 0.923 whereas the test set score was 0.673 with a 5-fold cross validation score of 0.712.

3.4.2 *Steel Plate Faults*

The ROCAUC 5-fold cross validation score was 0.738.

3.5 K-nearest Neighbors Results

3.5.1 *Breast Cancer*

The ROCAUC training score was 0.812 whereas the test set score was 0.503 with a 5-fold cross validation score of 0.714.

3.5.2 *Steel Plate Faults*

The ROCAUC 5-fold cross validation score was 0.725.

4 CONCLUSION

In Sec. 4, we presented the scores on the training and test sets. In sec. 5, we will conclude with a discussion on the differences in scoring.

4.1 Why, why, why?

For each dataset and each of the five algorithms, 50,000 estimators were cross validated 5 times each for a total of 250,000 fits per model. That's 500,000 estimators with 2.5M fits in total. On an i7-13700k, this took 22121 seconds in total or 6.14 hours. For the breast cancer dataset, the decision tree took 7.26 minutes, the MLP took 19.26 minutes, the gradient boosted decision tree took 10.53 minutes, the SVM took 1.38 minutes, and the KNN took 2.01 minutes. For the steel plate faults dataset, the decision tree took 34.08 minutes, the MLP took 173.48 minutes, the gradient boosted decision tree took 95.46 minutes, the SVM took 11.48 minutes, and the KNN took 13.31 minutes.

Still, this represents only about 10% of the hyperparameter space. I suspect that this is why all these models are vastly overfit. One can tell that they are overfit by seeing that the training scores are quite a bit higher than the test set on all of them. Cross validation seeks to lower this aggregate error rate across the test set but to no avail, unfortunately. I believe the correct thing to do here would be to either run an exhaustive grid search over the tens of millions of hyperparameter combinations or to cut the size of the hyperparameter grids. This would lead to models that are better calibrated and that lead to less overfitting. Cross validation scores would hopefully be even closer to the performance that is observed on the

test set. If this is achieved, we can be sure that statically sound models have been trained.

5 REFERENCES

1. L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. Wadsworth, Belmont, CA, 1984.Joyner, D. A. (2017). Scaling Expert Feedback: Two Case Studies. In *Proceedings of the Fourth Annual ACM Conference on Learning at Scale*. Cambridge, Massachusetts.
2. "1.10. Decision Trees." *Scikit*, 1.10. Decision Trees — scikit-learn 1.2.1 documentation.
3. *sklearn.svm.SVC*. (n.d.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
4. *LIBSVM -- A Library for Support Vector Machines*. (n.d.). https://www.csie.ntu.edu.tw/%7Ecjlin/libsvm/