Supervised Learning Report

CS7641
Gregory DeVos (gdevos3)

***Abstract*—Comparing five supervised machine learning algorithms on two medical time-series datasets.**

I. DATASETS

A. *EEG Eye State Dataset*

The medical field provides many opportunities for addressing time-series classification problems using diagnostic equipment. One such equipment, the EEG Eye State Dataset (Roesler, 2013), is a collection of Electroencephalography (EEG) measurements used to study brain activity. The EEG activity was mapped to eye state using a recorded video afterward, open or closed.

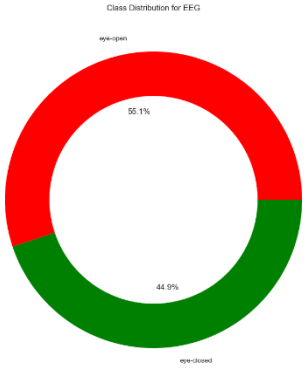


Figure 1: Class Distribution for EEG dataset

As shown in Figure 1, the 14,980 data points are balanced between open and closed.

B. *MIT-BIH Arrhythmia Dataset*

Physionet's MIT-BIH Arrhythmia Dataset (Moody GB, 2001) is another medical time-series classification problem (abbreviated to MIT-BIH or mitbih in charts). This dataset has two sampling channels resulting in 188 total features.

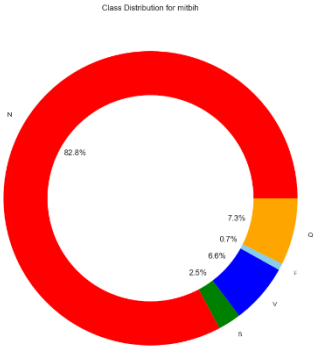


Figure 2: Class distribution for MIT-BIH dataset

As shown in Figure 2, this dataset is unbalanced with a wide distribution of classes, with some labels significantly under-represented. A macro F1 score metric gives more weight to the minority classes, which is discussed further in section 2.

C. *Why are they interesting?*

As prices in the healthcare industry continue to rise, there is a growing need to find improvements in testing. By automating test analyses using machine learning, results can instantaneously provide doctors with crucial information, thus reducing the time and financial pressures on healthcare professionals.

The two aforementioned datasets are part of the time-series classifications commonly gathered in hospitals using instrumentation. Specifically, I was interested in understanding how these models handle time-series data, high dimensionality, and class imbalance.

II. EXPERIMENT SETUP

The dataset required minimym pre-processing First, the dataset was separated into train/test subsets on a 70/30 split. Then, the standard scaler was computed on the training set and applied to both subsets.

During model training and visualization, cross-validation (CV) was used to characterize the model by gathering comprehensive data metrics. This avoided finding a lucky split, wherein a model performs either exceptionally well or poorly.

Additionally, cross-validation aids learning the features of imbalanced datasets, such as the MIT-BIH. Using a Stratified Shuffle Split preserved the percentage of samples for each class. Ten folds were used for all training and plots.

This experiment used the F1 score metric to compare the models. This metric was desirable because it accounted for how many instances were correctly classified as well as how many samples were missed. This balance between precision and recall was especially beneficial when comparing imbalanced datasets. Moreover, the F1 score used macro average mode to give more weight to the minor label in the imbalanced datasets.

III. Decision Trees

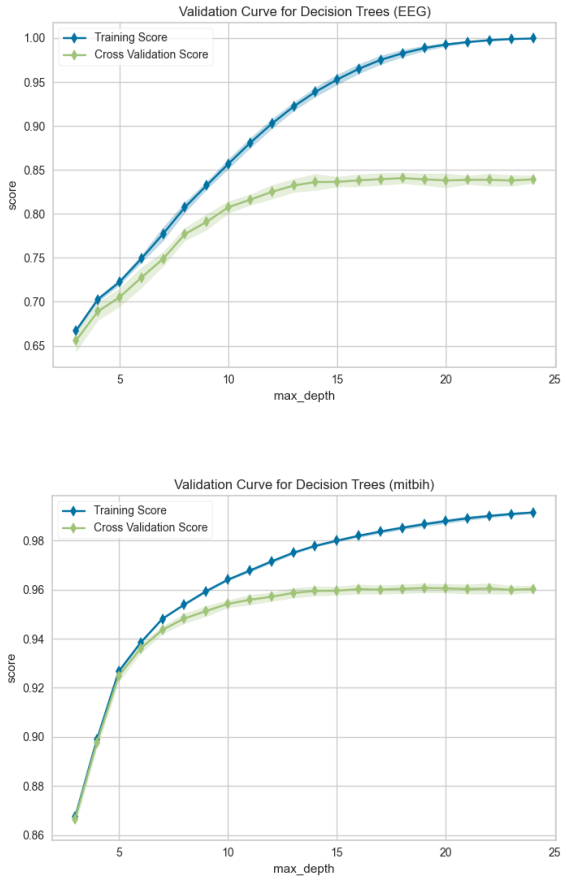


Figure 3: Validation Curves for max depth vs. F1 score (Top: EEG, Bottom:MIT-BIH)

As shown in Figure 3, there were several worthwhile findings found in the validation curves. First, the decision tree max depth was surprisingly resilient to overfitting. Even though the score was expected to drop with depth increase, the F1 score did not diminish from choosing a greater depth in either datasets.

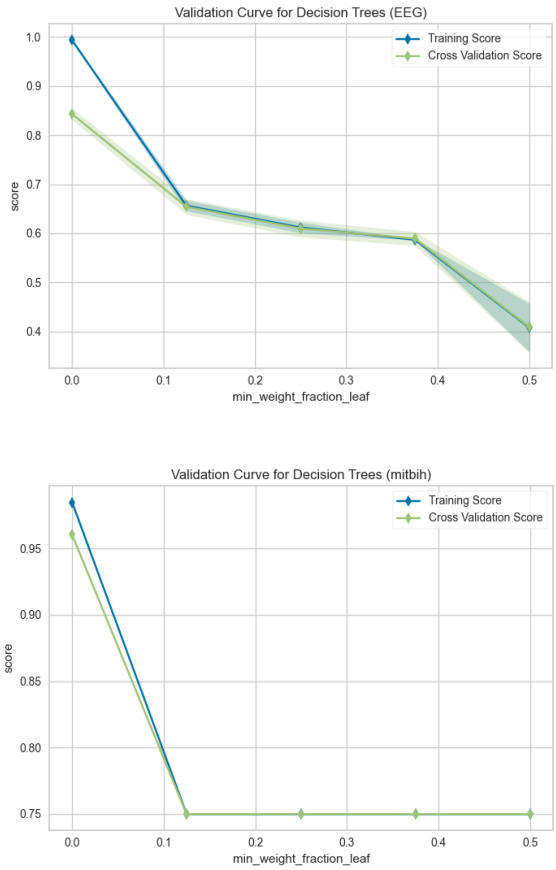


Figure 4: Validation Curves for min weight fraction leaf vs. F1 score (Top: EEG, Bottom:MIT-BIH)

As shown in Figure 4, the minimum weight fraction leaf is the overall weight required to be a leaf node. Although this parameter was included to reduce overfitting, the plots illustrate the opposite effect. MIT-BIH suffered more from this hyperparameter because crossing the threshold was more challenging for the minority classes.

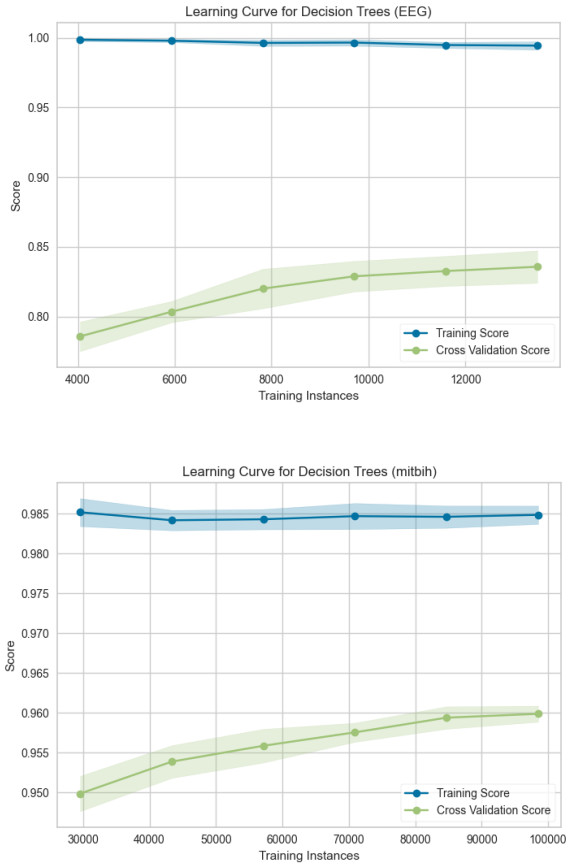


Figure 5: Learning curve for Decision Tree (Top: EEG, Bottom:MIT-BIH)

As shown in Figure 5, the decision trees benefitted from having more data—ideally, they would fully generalize by collecting more data. This is most notable for EEG, where an increase in data collection could help close the 0.17 difference between the training and CV scores. As shown in Figure 5, the standard deviation of the MIT-BIH model had a higher bias error than that of the EEG model (but both errors are still low). Both models also have a minimal variance error shown in the standard deviation of the CV score.

IV. NEURAL NETWORKS

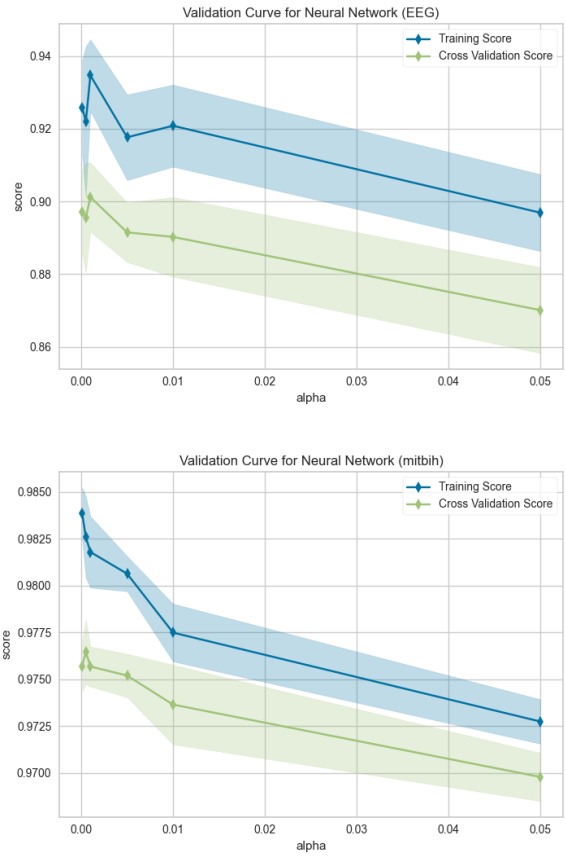


Figure 6: Validation Curve for Alpha used for L2 regularization vs. F1 score (Top: EEG, Bottom:MIT-BIH)

L2 regularization reduced model overfitting chances. In fact, higher alpha values in both data sets result in the models underfitting. A small alpha of 0.005 produces the best results, which is consistent with the results from the decision tree, where attempts to prevent overfitting result in weaker performance.

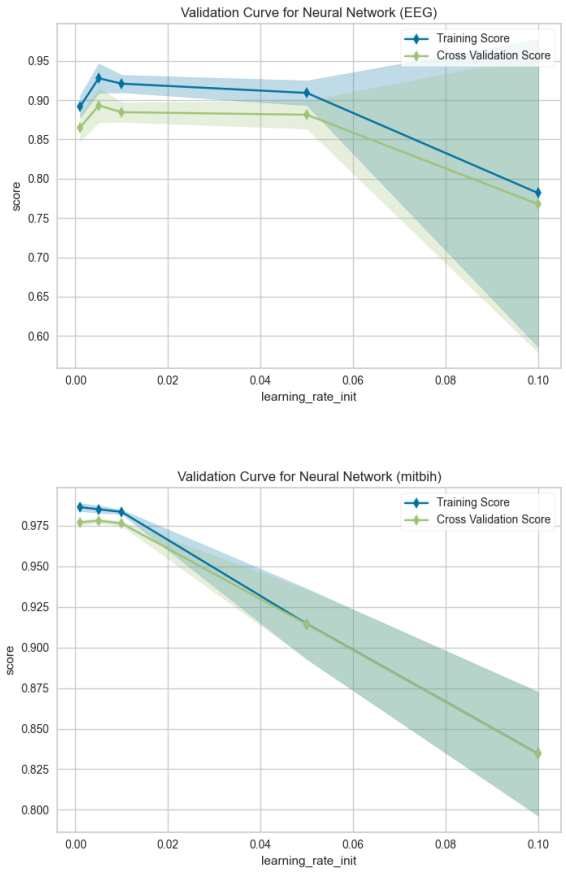
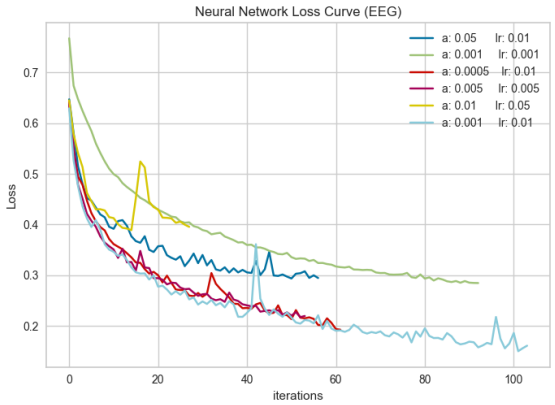


Figure 7: Validation Curve for learning rate vs. F1 score (Top: EEG, Bottom:MIT-BIH)

As shown in Figure 7, the learning rate also produced positive findings. As expected, a low learning rate produced the best results but took significantly more epochs to train. The standard deviation (variance) increases as the learning rate increases. A higher learning rate allows the weights in the neural network to change more between each epoch.



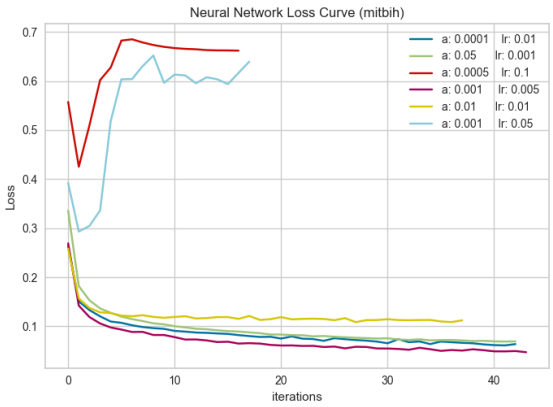


Figure 8: Neutral Network Loss Curve with a Patience of 15 (Top: EEG, Bottom:MIT-BIH)

The loss curve is useful for tracking the performance of a Neural Network. Figure 8 illustrates a sample of the loss curves used during training. Here, higher learning rates sporadically alter the weights, ultimately triggering an early stop. This graph also shows that when the learning rate is consistent, a low alpha value appears to have a smoother loss curve. This is intuitive, since alpha is the L2 regulation that keeps the weights low.

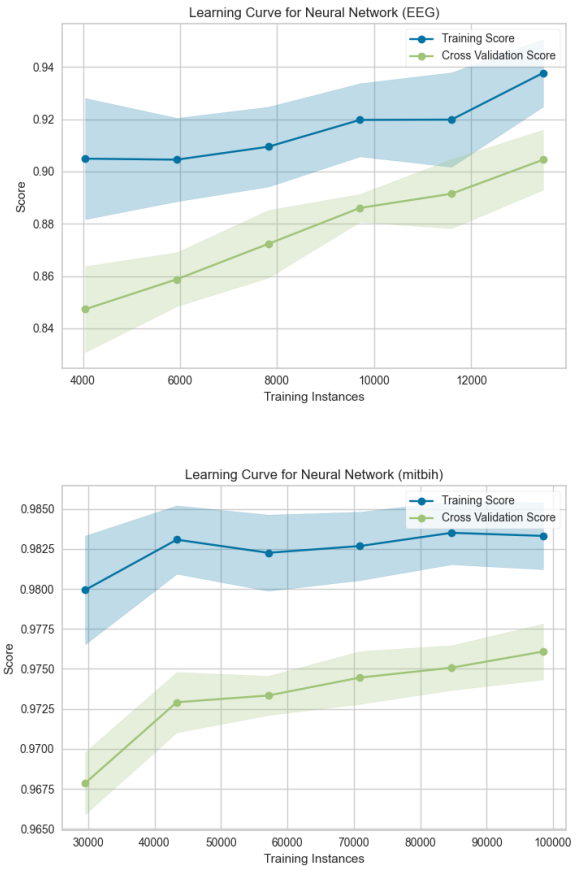


Figure 9: Learning curve for Neural Network (Top: EEG, Bottom:MIT-BIH)

The EEG neural network model would have benefited with more data with a meaningful 8% improvement. In addition, the standard deviation shows that this model also suffered error from both 0.04 bias and 0.03 variance.

Even with very low amounts of data, the MIT-BIH model performed very well. There was almost no bias or variance, with a respective 0.005 and 0.0035, and the imbalance dataset was handled better than that of any other model.

V. BOOSTING

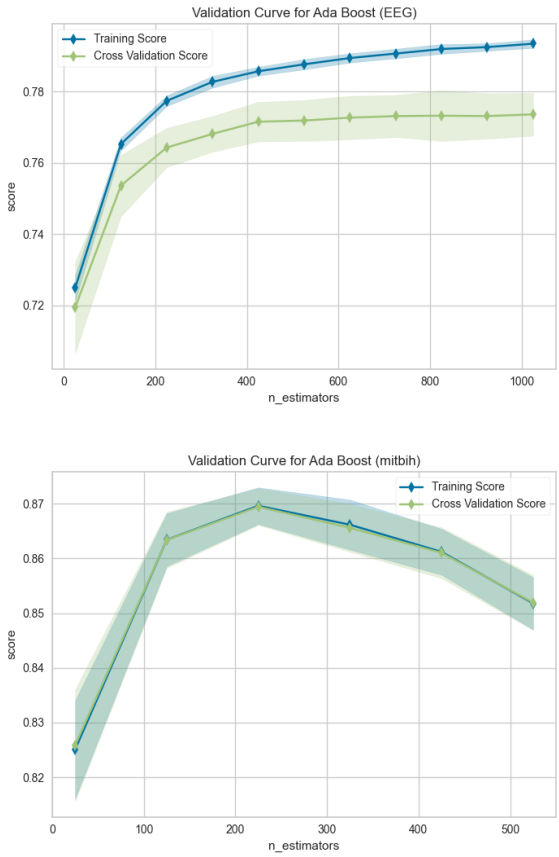
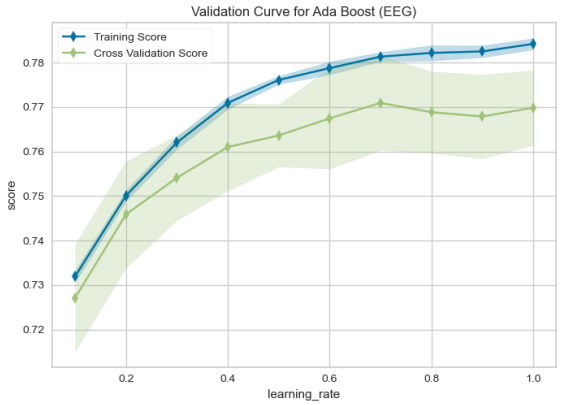


Figure 10: Validation Curve for number of estimators vs. F1 Score (Top: EEG, Bottom:MIT-BIH)

One of the benefits of Ada Boost is that they are resilient to overfitting (with the exception of pink noise). In Figure 10, the models are under-fit on the left side of the graph, 425 and 125 estimators, respectively. This benefit is observed in the EEG dataset but surprisingly did worse on the MIT-BIH with more estimators. Both the training and CV score dropped, so it was not overfitting the training set. It is possible that the learning rate needed to be lower for the higher number of estimators.



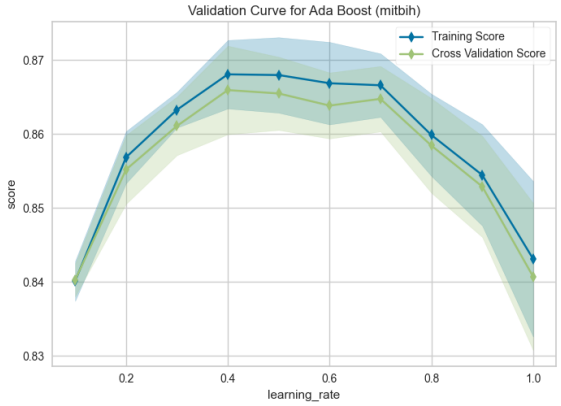
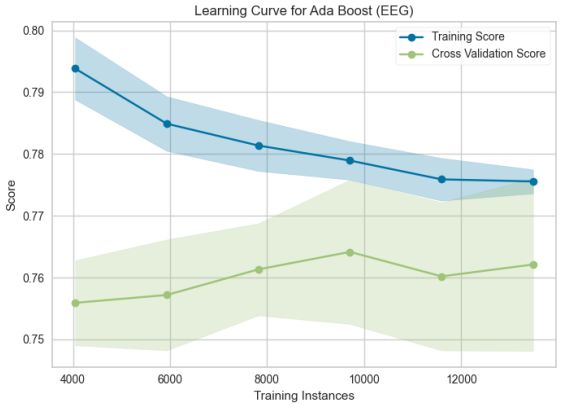


Figure 11: Validation Curve for learning rate vs. F1 score (Top: EEG, Bottom:MIT-BIH)

The learning rate controls how much each weak learner will contribute to the model. Generally, more weak learners are needed when the learning rate is low. Thus, the models are underfitting for learning rates less than about 0.4. Interestingly, only the MIT-BIH shows high variance as the learning rate increases. It is possible that weak learner fluctuates more because the dataset is imbalanced.



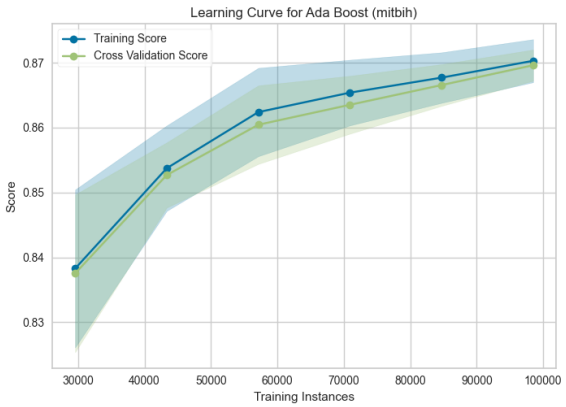
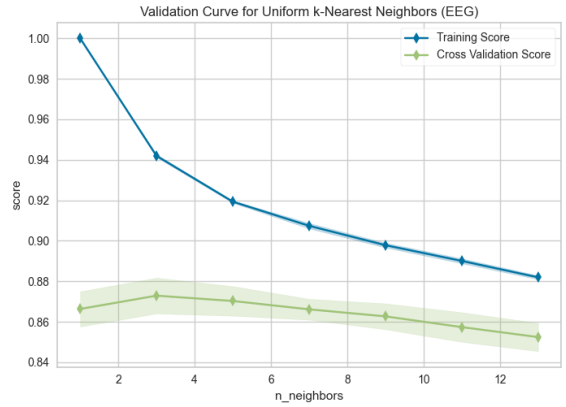


Figure 12: Learning Curve for Ada Boost (Top: EEG, Bottom:MIT-BIH)

The learning curve shows some fascinating results. The EEG model's F1 score leveled out as more instances were added. It appears the model would not have benefited from more training data. The F1 Score for the training and CV in the MIT-BIH dataset are tracked very closely with each other. It is surprising how closely they follow, and it is possible they are suffering from the same error. This is likely due to the considerable underrepresentation of some of the classes. The standard deviation of the CV scores in both models shows that model had a small variance error of about 0.02.

VI. K-NN



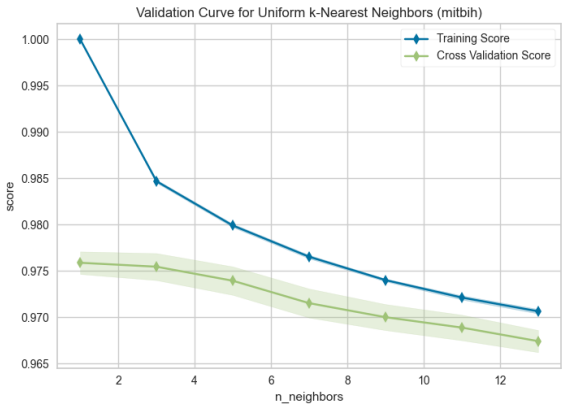
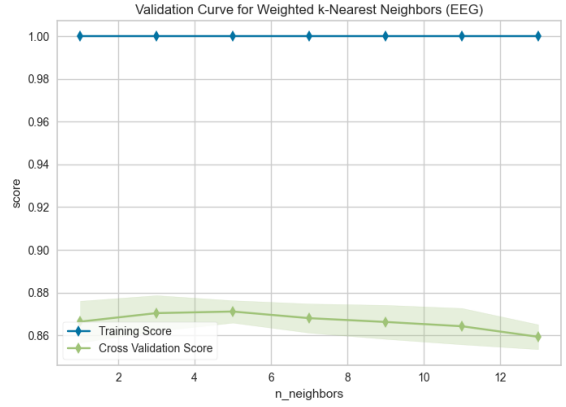


Figure 13: Validation Curve for number of neighbors vs. F1 Score (Uniform k-NN) (Top: EEG, Bottom:MIT-BIH)

The uniform k-NN models performed worse on the CV score (high variance error) when k (the number of neighbors) increased past three. An interesting observation is that as k increased, it is generalized to the training data. This is because a k with a value of of one is going to return the exact label.



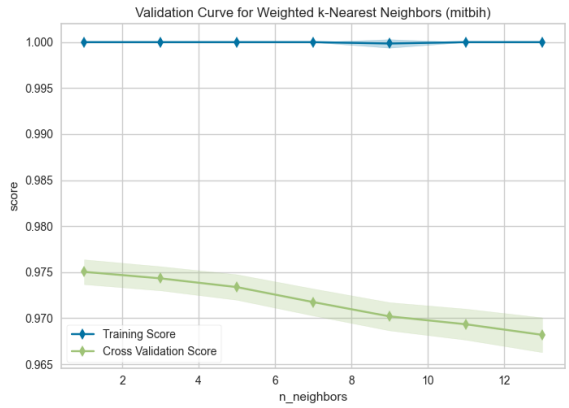
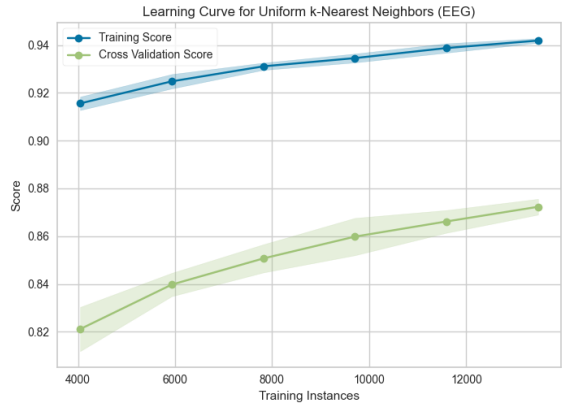


Figure 14 Validation Curve for number of estimators vs. F1 Score (Weighted K-NN) (Top: EEG, Bottom:MIT-BIH)

Similar to the uniform k-NN, the weighted k-NN models experienced higher variance error on the CV score when the number of neighbors increased past three. The training score did not drop as it did in the uniform since the datapoint matched a training point exactly.

The uniform versus weighted had little impact on the overall results. It also appears that the data had a few outliers. While more neighbors did result in a worsened generalization, it was expected to have a bigger impact. Using fewer neighbors also had the benefit of improved prediction times. Moving forward, a uniform k-NN will be utilized.



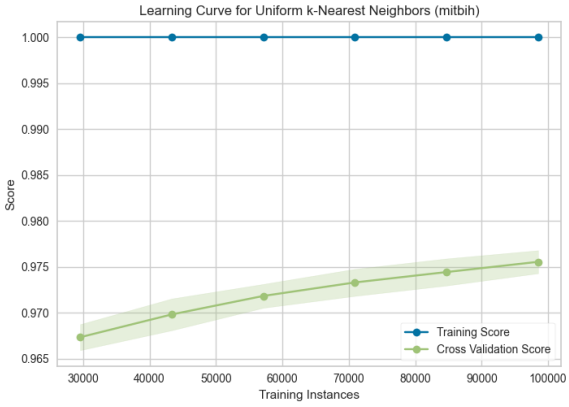
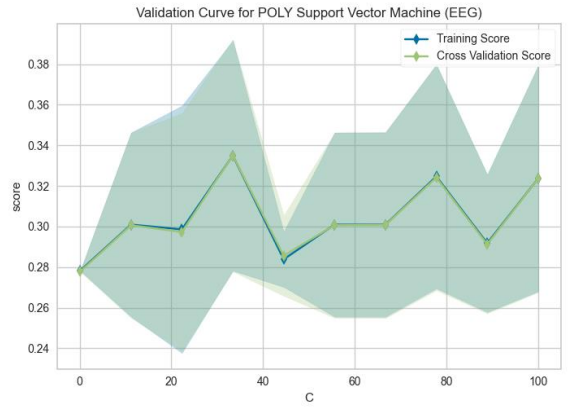


Figure 15: Learning Curve for uniform K-NN (Top: EEG, Bottom:MIT-BIH)

Both the k-NN models continued to see marginal benefit from more data since the CV score continued to increase. However, the MIT-BIH model generalized particularly well, even with low amounts of data. There was almost no bias or variance in either model. The MIT-BIH model also handled the imbalance dataset well.

VII. SUPPORT VECTOR MACHINES



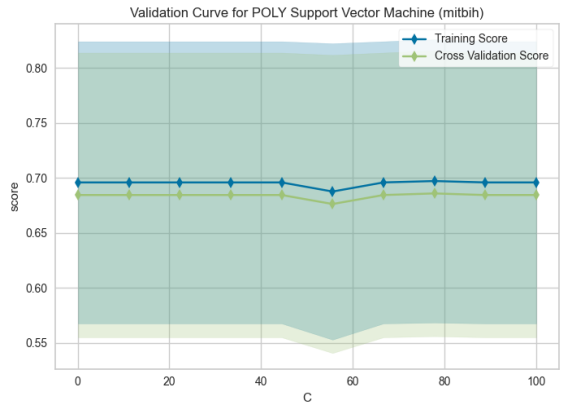
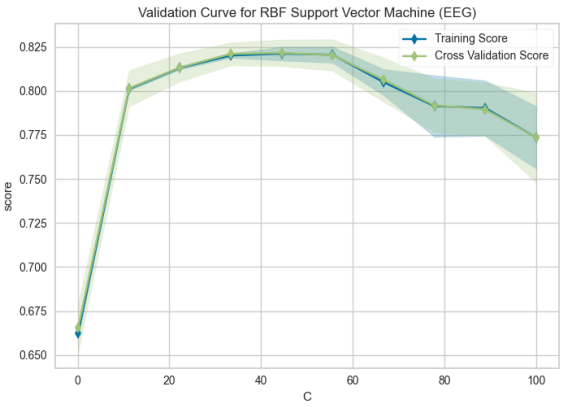


Figure 16: Validation Curve for C vs. F1 score using the POLY kernel (Top: EEG, Bottom:MIT-BIH)



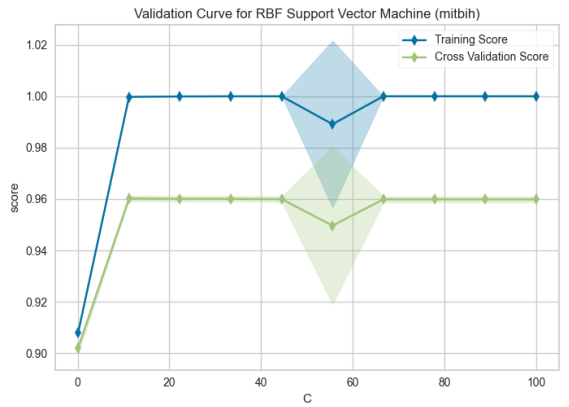
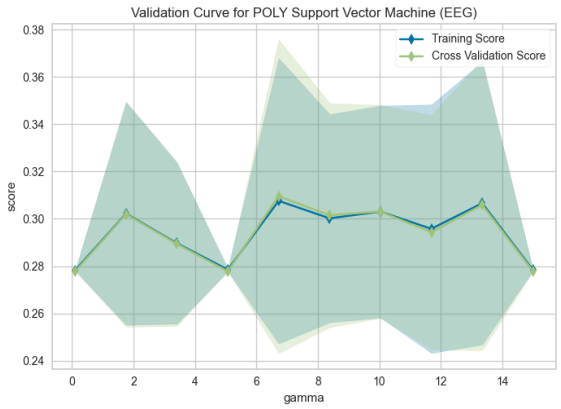


Figure 17: Validation Curve for C vs. F1 score using the RBF kernel (Top: EEG, Bottom:MIT-BIH)

Similar to alpha in neural networks, C is a regularization parameter. C controls the number of support vectors, wherein a smaller C will learn to more support vectors. Very high C values can increase the train time. The testing showed that different C values had an inconsistent effect. A possible explanation could be that the SVM struggled to converge in a reasonable amount of time due to the dimensionality. Testing with high

values of C was also not a good fit because of the train time. In fact, creating the validation curve for the RBF support vector machine on MIT-BIH took over 10 hours alone. It is hard to find meaningful analysis in these charts since the bias and variance are so sporadic. Thus, the best conclusion in this case is that the SVMs are not a good choice for this problem.



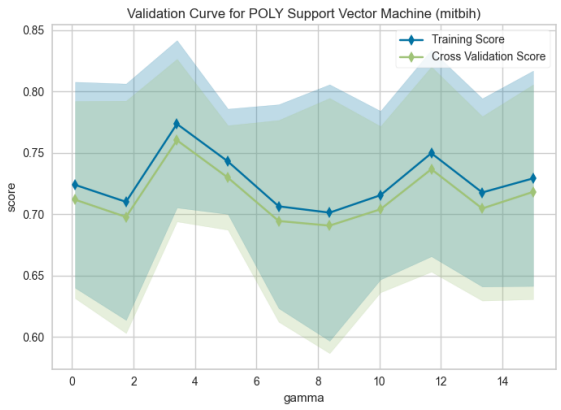
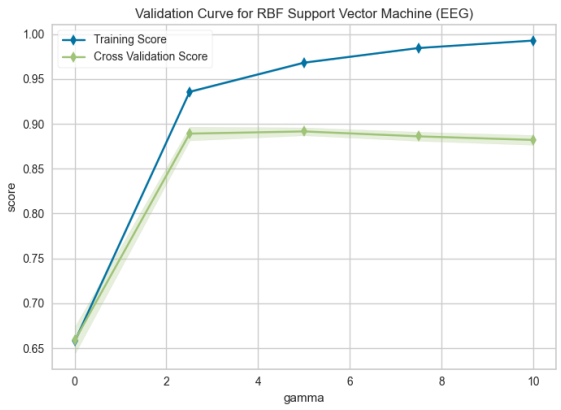


Figure 18: Validation Curve for gamma vs. F1 score using the POLY kernel (Top: EEG, Bottom:MIT-BIH)



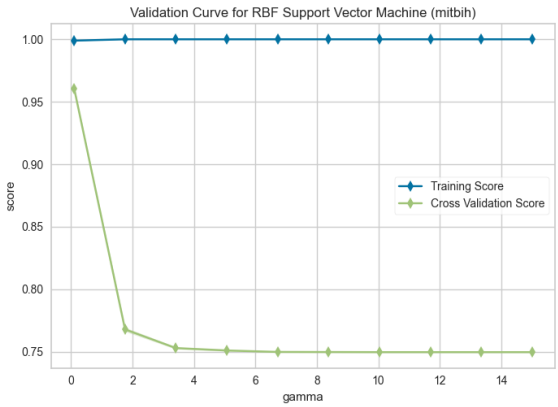
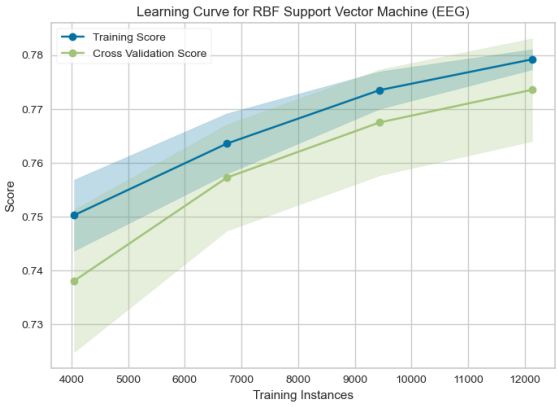


Figure 19: Validation Curve for gamma vs. F1 score using the RBF kernel (Top: EEG, Bottom:MIT-BIH)

The Poly kernel experienced significant levels of bias and variance error with gamma similar to C. This is either due to the wrong kernel for the data or that other hyperparameters need to be considered. Gamma on the RBF kernel created a typical learning curve with minimum variance. The RBF kernel will be used for the rest of the anaylsis.



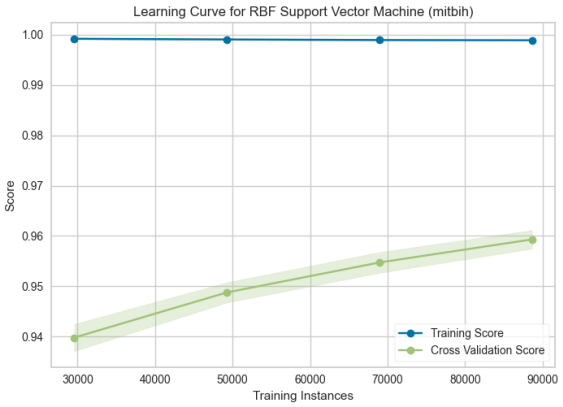


Figure 20: Learning Curve for RBF SVM vs. F1 score (Top: EEG, Bottom:MIT-BIH)

The F1 score for the training and CV in the EEG dataset are tracked closely with each other. The

steady increase shows it benefits from more data. Bias and variance error was also similar to that of Ada Boosting and the neural network.

The MIT-BIH also would benefit from more data. Interestingly, the training score was almost perfect. This discrepancy between the training and CV score could be explained by the imbalanced dataset. Since it the model has seen the data before, it can predict the minority class in the training set but not generalize to the test set. While still performing well overall, the last bit of performance would be gained by adding more minority class data or balancing the dataset.

VIII. RESULTS

Results are compared using two metrics, F1 score and run time. As discussed in section 2, the experiments used a macro F1 score to provide more weight to the minority classes. Run time will compare both the fit time and the predict time.

A. F1 Score Results

Model Type	F1 Score
Decision Trees	0.827
Neural Network	0.887
Uniform k-Nearest Neighbors	0.862
Ada Boost	0.758
RBF Support Vector Machine	0.767

Table 1: F1 macro scores for the EEG dataset

Model Type	F1 Score
Decision Trees	0.82
Neural Network	0.884
Uniform k-Nearest Neighbors	0.873
Ada Boost	0.5
RBF Support Vector Machine	0.854

Table 2: F1 macro scores for the MIT-BIH dataset

With the current popularity of neural networks, it is not a surprise that they performed the best on both datasets. What is particularly interesting is how well decision trees and k-NN performed—they were only a few hundredths behind the neural networks. The RBF did well in the MIT-BIH dataset but not the EEG dataset. This was surprising since the dimensionality of MIT-BIH was much higher (188 compared to 14). SVM can struggle to converge with high dimensionality, so this was unexpected. Ada boost performed the worst in both datasets. The validation curves and learning curve suggests this is the limit of the model.

Figure 21: Confusion Matrix for Neural Networks (Top: EEG, Bottom: MIT-BIH)

Since neural networks performed the best, Figure 19 shows the confusion matrix for both datasets. The error was distributed evenly between false positives and false negatives of the EEG dataset.

The MIT-BIH confusion matrix shows the model poorly performed on S labels, classifying a third of them as N. The model also classified a fourth of the F label incorrectly as N. It is expected that these two classes would perform the worse since they make up just 2.5% and 0.7% of the total population, respectively.

B. *Run Time*

Model Type	Train Time(s)	Predict Time(s)
Decision Trees	0.109	0.001
Neural Network	2.487	0.008
Uniform k-Nearest Neighbors	0.081	0.255
Ada Boost	5.548	0.305
RBF Support Vector Machine	4.189	3.474

Table 3: Train and predict times for EEG models

Model Type	Train Time(s)	Predict Time(s)
Decision Trees	21.799	0.025
Neural Network	25.501	0.076
Uniform k-Nearest Neighbors	0.023	72.858
Ada Boost	587.786	10.897
RBF Support Vector Machine	577.372	175.771

Table 4: Train and predict times for MIT-BIH model in seconds

The run time followed the expected run time complexity. The decision tree only builds one tree, so it is both fast to train and predict. In Ada Boost, each weak learner is quick to train but repeating it in the 500 times requires to avoid underfitting takes significant time. While not the fastest, predicting with Ada Boost was still reasonably fast. k-NN is very fast to train because it merely loads the data. The cost occurs when predicting labels. The RBF Support Vector Machine was the slowest to train and predict, requiring a tremendous number of iterations to converge due to the dimensionality. The neural network was one of the faster models to train and predict while also having the highest F1 score gaining their speed up from batching the data.

The difference in train time vs. test time provides an excellent example of eager learners vs. lazy learners. The decision tree, neural network, Ada Boost, and support vector machine being eager learners took significantly more time to train than to predict. In contrast, the k-NN took substantially more time to predict than to train.

IX. Code Repository

All source code can be found at:

https://github.com/gdevos010/ml_supervised_learning

X. References

Moody GB, M. R. (2001). The impact of the MIT-BIH Arrhythmia Database. *IEEE Eng in Med and Biology*, 45-50.

Roesler, O. (2013). *EEG Eye State*. Retrieved from UCI Machine Learning Repository.: https://archive.ics.uci.edu/ml/datasets/EEG+Eye+State#