CS7641 Machine Learning (Fall 2020)

Assignment 1 – Supervised Learning

Sergiy Palguyev (gtid: spalguyev3)

I. ABSTRACT

The following analysis explores five Machine learning techniques and how tuning their individual hyper-parameters may affect training, prediction, and learning rate. The analysis explores two datasets, both associated with Classification and containing multiple attributes to train the Machine Learning algorithms on. The algorithms used to evaluate both datasets include Decision Trees, Boosting, K-Nearest Neighbors, Neural Nets and Support Vector Machines.

II. INTRODUCTION

Two data sets are chosen to be used for this analysis from the UCI database. Both datasets will be used for each Machine Learning algorithm and will attempt to learn across the same range of hyper-parameter tuning and optimization.

A. Banknote Authentication Dataset

The Banknote Authentication dataset [1] consists of four continuous values representing image attributes used for evaluation of an authentication procedure for banknotes. The classification is binary, classifying each banknote as authentic, or not. This dataset is very interesting as it must have high accuracy and precision in determining authenticity. In banking, honoring inauthentic banknotes can cause severe consequences, especially for those moving large sums of money. Furthermore, this classification is attempting to define a very simple binary True/False classification from a relatively small set of parameters. If successful, such a Machine Learning model can be very powerful for the banking world.

B. Letter Recognition Dataset

The Letter Recognition dataset [2] consists of sixteen attributes which define the image representation of a written letter. The goal of this dataset is a multi-class classification of the letter of the English alphabet, or twenty-six letters. This dataset is very interesting, because of its implication across multiple fields of study. For example, medical records are notoriously poor in digitization as most are hand-written and difficult to transcribe to digital records. Obtaining a good classification Machine Learning algorithm may help aid in a host of Natural Language Processing problems, including the one described.

C. Comparing the two datasets

It is interesting that the Letter dataset differs from Banknote dataset as the resulting class contains 26 classifiers versus a binary classifier. Additionally, all attributes of the Letter dataset are integers, versus continues values of the Banknote dataset. Both are classification problems but both require widely different approaches and evaluation, the difference between which is demonstrated and analyzed in the following Sections.

D. Varying Training, Testing, and Validation Sets

Training data is a subset of the dataset which is used to train the model. It is the data that the model uses to learn from. As such, there must be significant number of examples in order to train the model for high predictive accuracy.

Alternatively, Testing data is a different subset of data which the training model never sees, and cannot overlap with the training data. This data is used to test the model, providing correct answers to compare against the model predictions.

Finally, Validation data is yet another subset of data used to determine model performance. Cross-Validated data sets is a form of validation data which do not require a chunk of the original dataset, instead, the training data is k-folded over many times in order to better utilize training data for model tuning.

The ratio of Train, Test, and (Cross-)Validation sets which work best will change from model to model, and dataset to dataset. As a general rule-of-thumb, models with few hyper-parameters will be easy to validate and tune, requiring a small validation set.

III. MEASURING PERFORMANCE

Prior to executing any algorithm, it is vital to specify how an algorithm will be judged, evaluated, and scored for performance. Accuracy of predicted values seems an obvious choice, but to determine how well an algorithm may perform over unseen data, unavailable to the training model, other performance metrics are required.

A. Confusion Matrix

In simple terms, the confusion Matrix provides an understanding of how often the trained model will classify a class “A” as a class “B”, and vice versa. Each row of a Confusion Matrix is the class being evaluated, with each column being the predicted class by the model. The reason

the Confusion Matric was chosen over, for example, Receiver Operating Characteristic (ROC) curve and its corresponding Area Under the Curve (AUC), is due to the nature of our two datasets. The ROC/AUC metrics are suited best for binary classification; thus although the Banknote dataset would be a good candidate, the Letter dataset would not. The Confusion Matrix metric, on the other hand, is more than capable of evaluating both binary and multi-class datasets. [5]

B. Precision Score

Precision Score metric is simply a more concise metric of the Confusion Matrix. This score is aimed at evaluating how many True Positives are detected out of the entirety of all positives (True Positive and False Positive). Also represented as PS = (TP)/(TP+FP).

C. Recall Score

Similarly to the Precision Score, the Recall Score is a subset of the Confusion Matrix. This score determines how good the model can determine True Positives against classes which were mislabeled by the model as negative, but were actually positive. Also represented as RS = (TP)/(TP+FN).

D. F1 Score

Finally, the F1 Score seeks to combine the Precision Score and the Recall Score into one evaluative metric, or harmonic mean of the two metrics. This metric prefers classifiers which have similar values for both Precision and Recall scores.

E. Reasoning

Depending on the dataset, one score may be more preferred to another. For example, for the Banknote dataset it may be argued that a high Precision Score and a low Recall score would be desired in order to guarantee only authentic banknotes to be honored, even if a few authentic ones get rejected. On the other hand, it may be beneficial to have a more balanced Precision-Recall score for Letter Recognition since handwriting varies incredibly, and a highly precise model may tend to over-fit easily.

IV. DECISION TREES

A Decision Tree is a Machine Learning technique used for both classification and regression problems. The tree is build upside-down, starting with one node at the top, splitting the tree further into nodes, branches and leaves downwards. The nodes are the features of an evaluated dataset. The branches are the decisions on how to split the decisions from the previous node. Finally, each leaf is a possible outcome of the decision chain.

A. Best Hyper-parameter

First, the best hyper-parameters are calculated by prebuild sklearn algorithm RandomizedSearchCV. The output of this function will be used as values that may be locked while others are varied, to see effects on model accuracy, as well as to determine a set of "optimized" parameters for the rest of the analysis to compare to.

Banknote Authentication Decision Tree RandomizedSearchCV calculated BEST parameters are 10 Leaves, 10 Tree Depth with 0.96 score at 50% Train and 50%

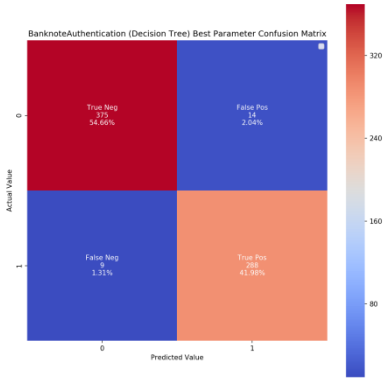

Figure 1: Banknote Authentication

The Confusion Matrix with these parameters shown in figure 1, determined that the model is predicting with 98.8% Train Accuracy, 96.6% Test Accuracy, 94.3% CV Accuracy, 95.4% Precision Score, 97.0% Recall Score and 96.2% F1 Score.

Letter Recognition Decision Tree RandomizedSearchCV calculated BEST parameters are 11 Leaves, 11 Tree Depth with 0.72 score at 66% Train and 34% Test

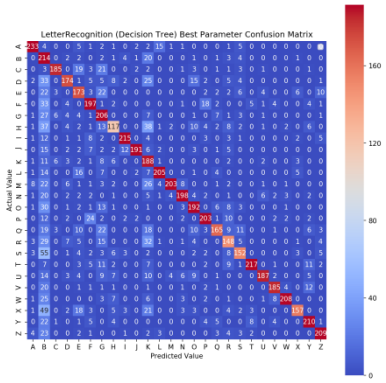

Figure 2: Letter Recognition

The Confusion Matrix with these parameters shown in figure 2, determined that the model is predicting with 75.7% Train Accuracy, 72.5% Test Accuracy, 71.0% CV Accuracy,

78.6% Precision Score, 72.5% Recall Score and 74.2% F1 Score.

As demonstrated, the Decision Tree classifier is suited much more for the binary dataset Banknote Authorization, providing higher scores across the board versus the Letter Recognition dataset. The following section demonstrate how varying certain hyper-parameters may affect the classification model, and shed light on why the Decision Tree classifier may not be the best at classifying the Letter Recognition dataset. Furthermore the number of miss-classified number of classes in the Confusion Matrix for the Letter Recognition data set is also indicative of poor performance.

B. Varying Training vs. Test size

The following figures display how adjusting Training and Testing data between 10%-90% of the original dataset in increments of 10% is displayed in the figures below.

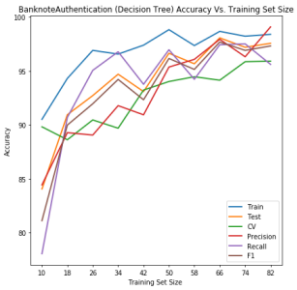

Figure 3: Banknote Authentication

Figure 4: Letter Recognition

From Figure 3 Banknote dataset, it appears as though a 66% Train/34% Test would be an optimal choice. Alternatively, it is evident that the sklearn algorithm picked a 50%/50% split from Section A above. Observing the plot in figure 1, the standout feature which highlights the 50/50 split is the increase in Training and Cross-Validation accuracy at the 50/50 split versus the 66/34 split. This is a valid reason for an algorithm to choose the 50/50 split as the optimal one, all things considered, since cross-validation acts as a measure of the model's performance to work with unseen data.

From Figure 4 Letter Recognition dataset and in tandem with the sklearn algorithm form Section A, it is evident that the 64/34 split is the best ratio for the Letter Recognition dataset. This ratio carries the highest accuracy for Precision and Recall Scores while also perfectly matching the Precision with the Test scores. This is a very interesting finding which also points to the reason the 66/34 split is much preferred.

C. Varying Tree Depth

The following figures display how adjusting Tree Depth data between 2 to 11 in increments of 1 is displayed in the figures below. The number of terminal nodes quickly increases with depth of a tree. As such, it becomes more and more difficult to understand the decision rules of a tree with more depth.

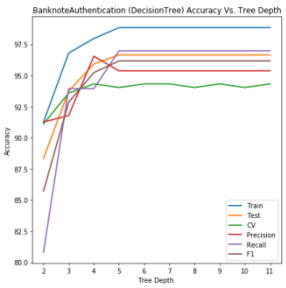

Figure 5: Banknote Authentication

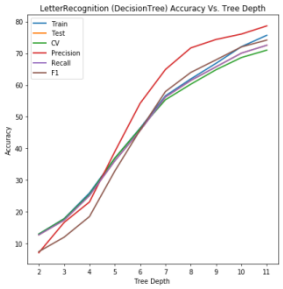

Figure 6: Letter Recognition

Figure 5 shows a clear case of overfitting on the Banknote dataset for depth values above 5. However, it is apparent that Cross-Validation accuracy continues to fluctuate which may explain the reason why sklearn algorithm from Section A picked a value of 10 as the optimal Depth. However, this should not have been the case as there are only 5 attributes on the Banknote dataset, and it is often the case that the depth closely matches the number of attributes for well-fitting datasets.

Figure 6 shows a somewhat logarithmic curve to the Letter Recognition dataset learning as the tree-depth increases. This is a logical observation as the dataset has 26 classes to predict amongst. Thus, as deeper the tree gets, the more capable it is in predicting the correct values, but the more complicated it gets as well. Thus we see that accuracy increases in all scores, but with diminishing returns as the complexity increases significantly as well. This is evident from the sklearn evaluation as well as at the optimal value of hyper-parameters, the accuracy scores on evaluative metrics are all approximately75%.

This contrast between the Banknote and Letter dataset points to the difference in classifiers. One is binary, while the other has 26 classes to predict. This corresponds to the depth of tree required for good accuracy, with Banknote quickly overfitting above depth of 5, while Letter displaying a logarithmic curve.

D. Varying Tree Leaves

The following figures display how adjusting maximum leaf nodes between 10 to 19 in increments of 1 affects accuracy. Limiting the maximum number of leaf nodes is a way of pruning the Decision Tree so it does not grow too large.

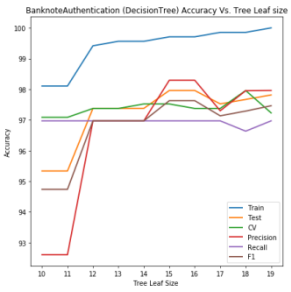

Figure 7: Banknote Authentication

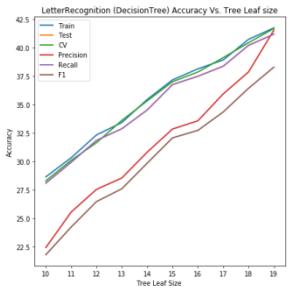

Figure 8: Letter Recognition

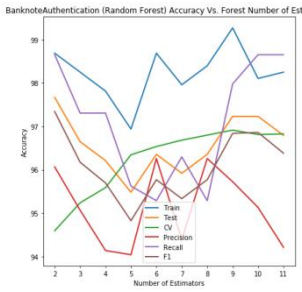

Figure 9: Banknote Authentication

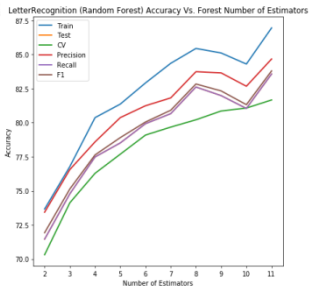

Figure 10: Letter Recognition

Figure 7 Banknote dataset displays very interesting behavior with maximum leaf nodes variations. Low scores on Precision, Train, Test and Precision at size 10 demonstrate a form of clipping that is initiated due to limiting maximum leaf nodes. Effectively, the tree requires more nodes than is allowed. However, as maximum leaf nodes increase and optimum value is reached with overfitting quickly taking over. From Figure 5, it is evident that a value of 15 is optimal. However, sklearn optimization from Section A picked a value of leaf value 10. This is an interesting contrast to what Figure 5 displays and points to a possible reason for the sklearn algorithm to prefer the Recall accuracy metric. However, using sklearn optimized value would not be a good decision for this dataset. With banknote authentication, the client would want high Precision with low Recall, so as to avoid processing false positives – a maximum leaf node value of 15 would be a better choice.

Figure 8, shows a similar trend as in previous hyper-parameter evaluations. The Letter Recognition dataset, containing multiple classes requires very large trees. A linear trend for all accuracy metrics is observed in Figure 6, increasing with leaf complexity.

E. Varying Tree Forest

Decision Tree Random Forests are collections of Decision Trees trained on the same dataset. The purpose of these Random Forests is to mitigate the effect of overfitting often observed in single Decision Tree classifiers. In order to test this, the number of trees in the forest is varied between 2 and 11 trees, in increments of 1. All trees use the optimal values generated by the sklearn.RandomSearchCV algorithm.

Figure 9 displays high variance amongst the accuracy metrics used to evaluate the Random Forest classifier. However, a tree value of 9 presents the best accuracy in Training and CV values. However, the Recall value is higher than Precision at this number of trees. Considering this, the tree value of 8 presents a better outcome, with slightly lower accuracy values but a better performance of Precision versus Recall required by the Banknote authentication dataset.

Figure 10, again displays the complexity required to classify the Letter dataset. With more trees, better accuracy metrics are observed across all scorers. The maximum number of tested trees has the best performance, showing that a larger value may present even better accuracy if tested.

V. BOOSTING

Boosting is a technique of applying certain weights to the results of other classifiers. The output of, for example, a decision tree is weighted heavier towards the miss-classified data points such that the next classification iteration of the boosting algorithm will give greater attention to those value points. This technique results in better learning outcomes from the values which a lone classifier may get wrong.

A. Best Hyper-parameter

BanknoteAuthentication AdaBoost calculated BEST parameters are 100 Estimator, 0.001 Learning Rate with 0.99 score at 74% Train and 26% Test

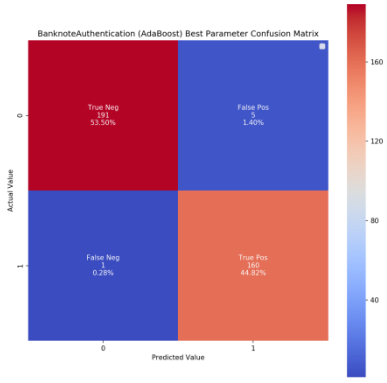

Figure 11: Banknote Authentication

The Confusion Matrix with these parameters shown in figure 11, determined that the model is predicting with 100.0% Train Accuracy, 98.3% Test Accuracy, 98.3% CV Accuracy, 97.0% Precision Score, 99.4% Recall Score and 98.2% F1 Score

LetterRecognition AdaBoost calculated BEST parameters are 90 Estimator, 0.00021544346900318845 Learning Rate with 0.85 score at 82% Train and 18% Test

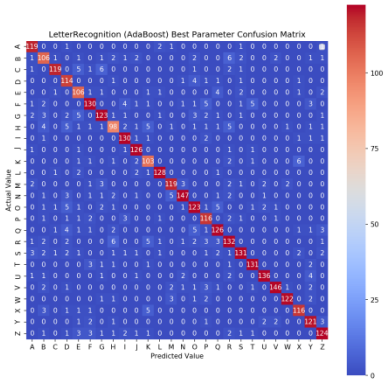

Figure 12: Letter Recognition

The Confusion Matrix with these parameters shown in figure 12, determined that the model is predicting with 100.0% Train Accuracy, 88.7% Test Accuracy, 85.2% CV Accuracy, 88.8% Precision Score, 88.7% Recall Score and 88.7% F1 Score

Observing the values generated by RandomSearchCV it is evident that boosting can generate slightly better accuracy for Banknote dataset, but provide much better accuracy for Letter Recognition. The following section vary the hyper-parameters and analyze how doing so increases accuracy of the AdaBoost classifier. Additionally, the number of miss-classifications for both dataset is very low, as displayed by the Confusion Matrix.

B. Varying Training vs. Test size

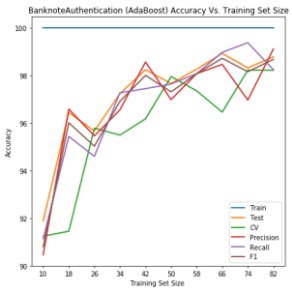

Figure 13: Banknote Authentication

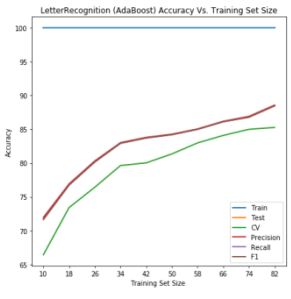

Figure 14: Letter Recognition

Figure 13, Banknote dataset displays an increase in accuracy with an increase in Train/Test ratio up to approximately 66/34%. The sklearn optimized parameter chose 74/26 ratio, showing a clear preference for the Recall metric as all other accuracy metrics are lower at that ratio versus the 66/34 ratio. Alternatively, that 82/18 ratio may be considered preferential for its high Precision, a desired trait for Banknote authentication.

Figure 14, Letter Recognition dataset corresponds with sklearn optimal values of 82/18 split showing the highest accuracy for all metrics. Interestingly, Precision, Recall, F1, Test and Train all track one of top of the other in accuracy. This is a fascinating effect, possible showing a lack of overfitting where the accuracy of training is exactly the same as the accuracy for training and predicting correct values.

C. Varying Tree Depth

Boosting uses the same Decision Tree classifier as in Section IV thus varying the same hyper-parameters is a valid analysis of model behavior.

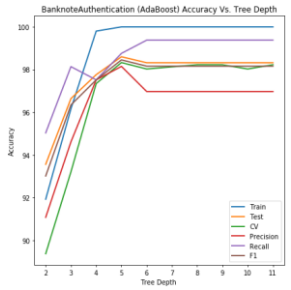

Figure 15: Banknote Authentication

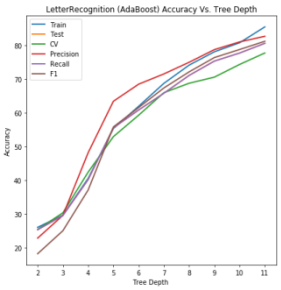

Figure 16: Letter Recognition

Figure 15 displays a clear case of overfitting for the Banknote dataset about a tree depth of 5. The same conclusion is drawn in Decision Tree analysis Section IV.C.

Figure 16, similarly, displays a linear increase in accuracy of learning Letter Recognition dataset with growth of tree depth. This again points to the complexity required by the Letter Recognition dataset, containing 26 classifiers to be evaluated.

D. Varying Tree Leaves

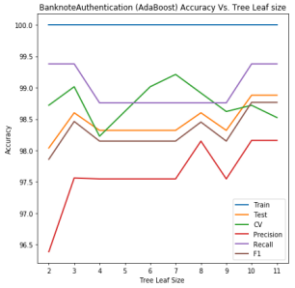

Figure 17: Banknote Authentication

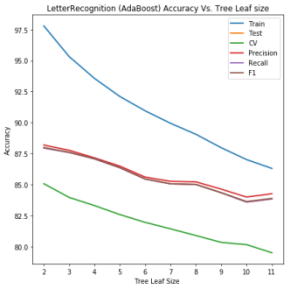

Figure 18: Letter Recognition

Figure 17 analysis shows that varying maximum number of tree leaves for Banknote dataset increases

performance somewhat with an optimal CV at depth of 7. This behavior is suggesting that varying tree depth for a Boosting algorithm is unlikely to provide any useful accuracy increase to improve the learning model.

Figure 18, is a very interesting behavior of the Letter Recognition dataset. With increase in depth the accuracy decreases for all metrics. Considering the complexity of the dataset, the most likely cause of this behavior is that the boosting feature is increasing noise of the dataset, rather than the true classes required to be learned by the model.

E. Varying Number of Estimators

AdaBost also depends on the number of Decision Trees it is evaluating. The following figures explore the number of trees hyper-parameter of values between 10 and 100 in increments of 10. Increasing the number of trees, implies more weak classifiers to combine at the end, and more variations in the decision boundaries of these classifiers.

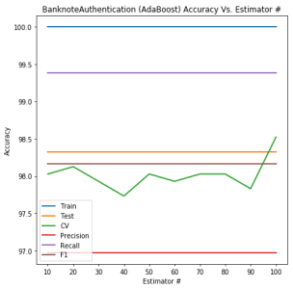

Figure 19: Banknote Authentication

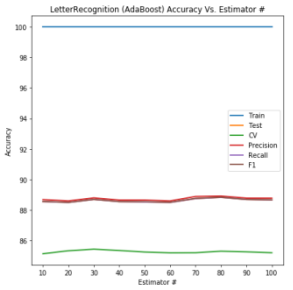

Figure 20: Letter Recognition

Both Figure 19 and Figure 20 display a poor indication that increasing the number of trees does anything for increasing accuracy. Only the Banknote dataset shows a slight increase in CV accuracy about 100 trees. What this behavior shows I that for both datasets, the concert for overfitting (which multiple trees avoids) is not of much concern as it has little effect on accuracy.

F. Varying Learning Rate

Learning rate determines how much each iterative model contributes to the existing model. The figures below display an analysis of learning rates across a logarithmic scale from 0.0001 to 0.1 in 10 increments spread evenly.

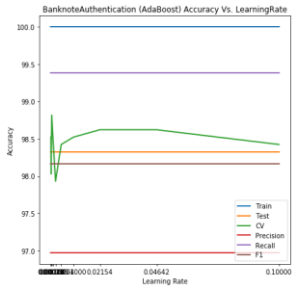

Figure 21: Banknote Authentication

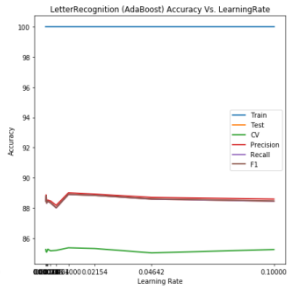

Figure 22: Letter Recognition

Both Figure 21 and Figure 22 display a clear preference for low learning rate values. What this implies is that both datasets favor slow learning rate, translating into smaller variations of the weighted data points and therefore fewer differences between the weak classifier decision boundaries.

VI. K-NEAREST NEIGHBORS

K-Nearest Neighbors machine learning algorithm is what is termed a “lazy learner” where all the data from the dataset is used at the time of prediction, not at the time of training. The concept behind KNN being that neighbors close-by will be classified similarly, thus determining a value’s classification is a matter of looking at its closest neighbors.

A. Best Hyper-parameter

BanknoteAuthentication KNN calculated BEST parameters are minkowski Metric, 5 Neighbors with 1.0 score at 82% Train and 18% Test

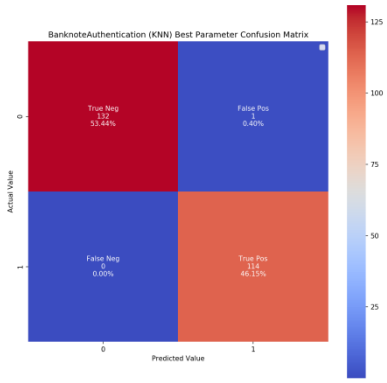

Figure 23: Banknote Authentication

The Confusion Matrix with these parameters shown in figure 23, determined that the model is predicting with 99.9% Train Accuracy, 99.6% Test Accuracy, 99.9% CV Accuracy, 99.1% Precision Score, 100.0% Recall Score and 99.6% F1 Score

LetterRecognition KNN calculated BEST parameters are euclidean Metric, 1 Neighbors with 0.94 score at 82% Train and 18% Test

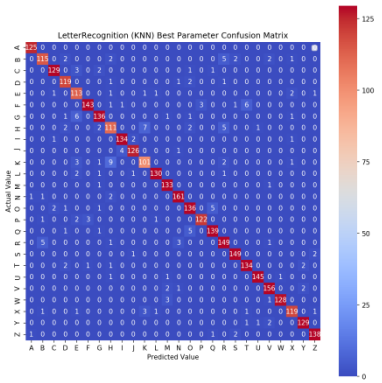

Figure 24: Letter Recognition

The Confusion Matrix with these parameters shown in figure 24, determined that the model is predicting with 100.0% Train Accuracy, 95.0% Test Accuracy, 94.0% CV Accuracy, 95.0% Precision Score, 95.0% Recall Score and 95.0% F1 Score

Observing sklearn. RandomizedSearchCV values for KNN learner, it is evident that KNN is a great candidate for both datasets. Reaching an accuracy score for all metrics above 95%, with Banknote dataset metrics performing about 99% accuracy. Additionally, the number of miss-classifications for both dataset is near zero, as displayed by the Confusion Matrix.

B. Varying Training vs. Test size

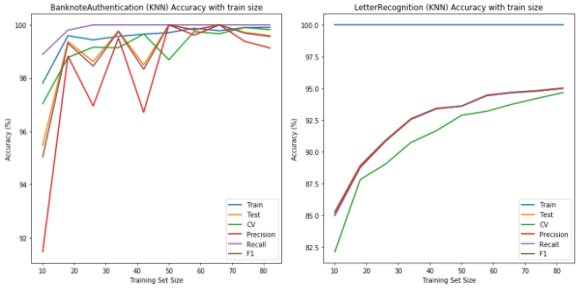

Figure 25: Banknote Authentication

Figure 26: Letter Recognition

Figure 25 and Figure 26 both coincide with the sklearn values of train/test ratios of 82/18. Initially, Figure 23 appears to contain a good Train/Test ratio of 66/34, but the CV accuracy increases towards the 82/18 ratio while F1 and Precision fall only slightly. In regards to the Banknote dataset, the 66/34 ratio may, in fact, be preferable due to the higher Precision, a metric more important for authentication, but with a worse performing CV. This indicated that the precision may be better, but the real-world performance may, in fact be worse. As such the 82/18 ratio is preferred for Banknote authentication.

Figure 24, as previously, shows a very interesting trend of all metric values overlaying on one another, except the CV value, showing a resistance to noise and overfitting.

C. Varying Number of Neighbors

The number of neighbors hyper-parameter is used to classify a value based on the distance to a specific number of neighbors surrounding it. The number of neighbors is varied between 1 and 10 with increment of 1.

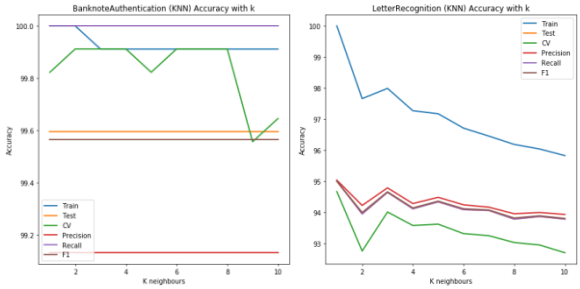

Figure 27: Banknote Authentication

Figure 28: Letter Recognition

Figure 27 and Figure 28 both show very interesting trends of preferring very few neighbors to use for classification. Figure 27 shows an optimal value of 2 and Figure 28, an optimal value of 3. Both of these observations differ from the evaluation of sklearn optimal parameters but nevertheless, point to the same conclusion. With increased neighbor considerations, the accuracy scores decrease for both datasets. More specifically, the CV accuracy decreases for Banknote dataset, implying the model becomes worse at predicting real-world values while retaining good true-to-predicted value accuracy. While for the Letter Recognition dataset, all test metrics become worse with more neighbors.

D. Varying Metrics

The different metrics define the distance measure used to calculate different neighbor proximity.

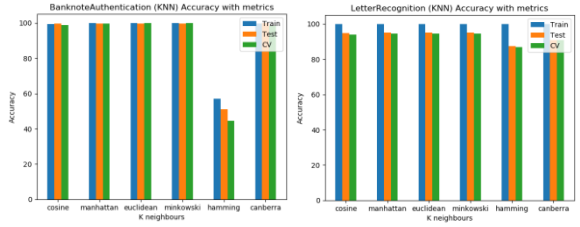

Figure 29: Banknote Authentication

Figure 30: Letter Recognition

In both Figure 29 and Figure 30, all classification metrics perform similarly well except for the “hamming” distance metric. This is expected since the hamming metric is much more suitable for Regression datasets rather than classification datasets.

VII. NEURAL NETS

Neural Nets machine learning technique aim to mimic the functional workings of brain neurons. More specifically a single "neuron" has many inputs, weighted differently, which have a certain threshold that will trigger a positive signal on the output. Neural Nets can then build layers of neurons and connections which together produce an output.

A. Best Hyper-parameter

BanknoteAuthentication NeuralNet calculated BEST parameters are (126, 40) Layers, 0.001 Alpha with 1.0 score at 26% Train and 74% Test

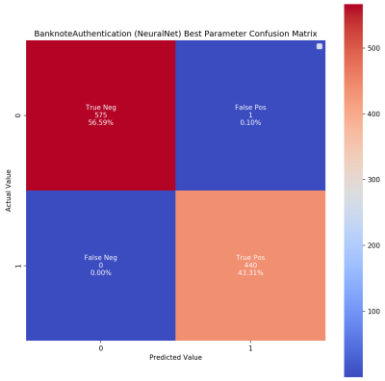

Figure 31: Banknote Authentication

The Confusion Matrix with these parameters shown in figure 31, determined that the model is predicting with 100.0% Train Accuracy, 99.9% Test Accuracy, 100.0% CV Accuracy, 99.8% Precision Score, 100.0% Recall Score and 99.9% F1 Score

LetterRecognition NeuralNet calculated BEST parameters are (150, 51) Layers, 0.1 Alpha with 0.95 score at 82% Train and 18% Test

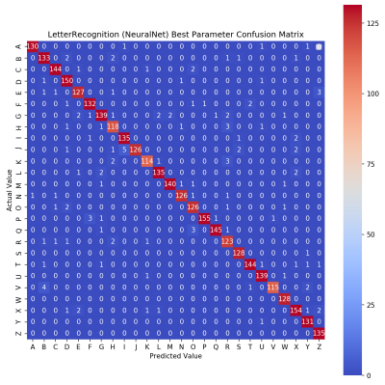

Figure 32: Letter Recognition

The Confusion Matrix with these parameters shown in figure 32, determined that the model is predicting with 99.3% Train Accuracy, 96.4% Test Accuracy, 95.3% CV Accuracy, 96.5% Precision Score, 96.4% Recall Score and 96.4% F1 Score

The sklearn RandomizedSearchCV values for both datasets show very impressive accuracy values well above 95%, with Banknote reaching above 99% in all metrics. Additionally, the number of miss-classifications for both dataset is near zero, as displayed by the Confusion Matrix.

B. Varying Training vs. Test size

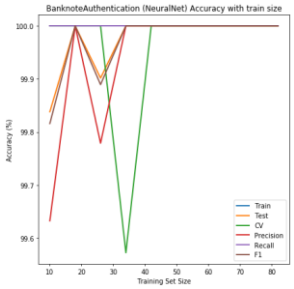

Figure 33: Banknote Authentication

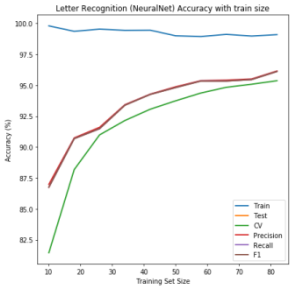

Figure 34: Letter Recognition

Figure 33, Banknote dataset display a very interesting behavior. As seen, all accuracy metrics reach 100% peak with a very small training set. This is in line with what sklear.RandomizedSearchCV values present with a Train to Test ratio of 26/74. Figure 33 displays very sharp drops and swings in accuracy metrics in higher ratios, most likely pointing to overfitting caused by large numbers of hidden neural networks, the effects of which are analyzed in the following sections.

Figure 34, displays a much different result with Letter Recognition dataset. The best Train/Test ratio at 82/18 is in line with Figure 34, showing all accuracy metrics improving with larger Test/Train ratio. It is interesting to notice a consistent decrease in Training accuracy. The most likely culprit for this is the complexity of the dataset, whereby the number of neurons and neural layers lower training accuracy.

C. Varying Number of Layers

Neural Nets can be made up of many layers of neurons. The following figures display how adjusting neural net layers data between 2 and 11 in increments of 1 affects accuracy and is displayed in the figures below.

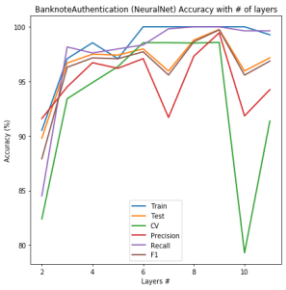

Figure 35: Banknote Authentication

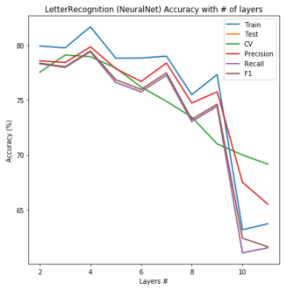

Figure 36: Letter Recognition

Figure 35, Banknote dataset displays a clear case of overfitting as accuracy metrics begin to decrease after 6 layers, and fall sharply after 9 layers of neural nets.

Figure 36, shows a similar behavior of overfitting of the Letter Recognition dataset, with any Neural Net more than 4 layers consistently decreasing in every accuracy metric. The most interesting being 10 layers of neural nets which drop training and F1 accuracy to below 65%.

D. Varying Number of Neurons

Neural Nets can be made up of many numbers of neurons at each layer. The following figures display how adjusting neuron numbers between 10 and 100 in increments of 10 affects accuracy and is displayed in the figures below.

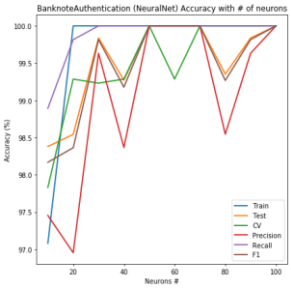

Figure 37: Banknote Authentication

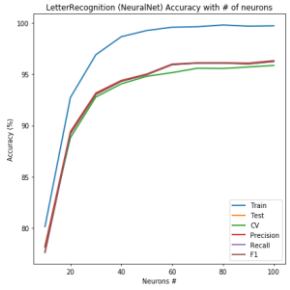

Figure 38: Letter Recognition

Figure 37, similarly points to a case of overfitting with increase in number or neurons. The best value approximately at 50 neurons, after which the CV, Precision, Recall and F1 value begin dropping with increased number of neurons. Similar to Section C above, increasing neural layers or neuron numbers too much shows a detrimental effect on the classifier as the model becomes too complex and miss-classifications begin to occur.

Figure 38, Letter Recognition displays a logarithmically increasing accuracy with an increase in neuron numbers. Beyond, approximately 60 neurons, the returns on increasing neuron count diminished with accuracy continuing to rise but in very miniscule values compared to the increase in Neural Net complexity.

E. Varying Alpha Value

The Alpha value is the learning rate of the Neural Net learner. This value determines how the weights of the neuron inputs get modified with each training data. Low alpha values reduce the weight modification induced with each iteration, whereas high alpha values change the weights more radically. The alpha values are varied between 0.0001 and 0.1 on a logarithmic, evenly spaced scale in 10 increments.

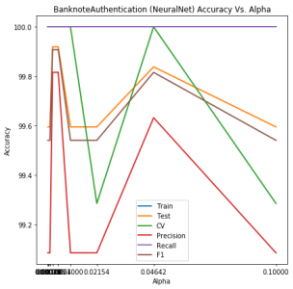

Figure 39: Banknote Authentication

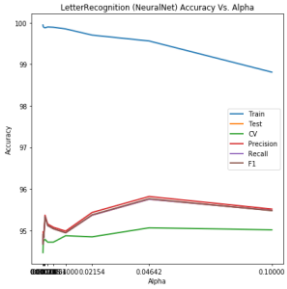

Figure 40: Letter Recognition

Figure 39, Banknote dataset shows a clear preference for small alpha values. Consistent with optimized value from Section A of 0.001, the low value implies a slow learning rate, putting very little emphasis on changing weights with each iteration. In a broader context, between alpha values of 0.001 and 0.1, the accuracy differences of all metrics vary only between 99.1-100%. In some application these differences may be considered insignificant.

Figure 40, Letter dataset, interestingly shows an opposite preference from that of the Banknote dataset. The gamma values for this data are leaning towards larger values of 0.04. This is indicative of the learning algorithm preferring the change weights more radically with each iteration. Note, although the values are small, the preferred alpha for the Letter dataset is an orders of magnitude (10x) larger than that of the Banknote dataset.

VIII. SUPPORT VECTOR MACHINES

The purpose of a Support Vector Machine learning technique is to determine a hyperplane which best separates the various classes of the dataset. The goal of tuning the SVM is to find the maximum distance between the points of opposing classes which best separates them. The largest distance allows for room for real-world data variations and noise to still be properly classified.

A. Best Hyper-parameter

BanknoteAuthentication SVM calculated BEST parameters are ‘rbf’ Kernel, 3.593813663804626 C value, 0.21544346900318834 Gamma with 1.0 score at 10% Train and 90% Test

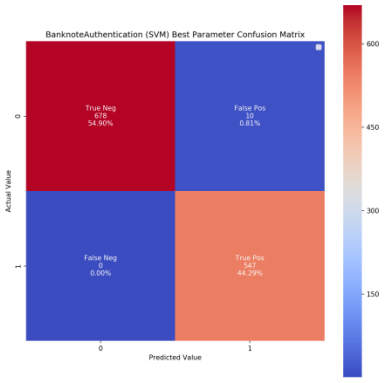

Figure 41: Banknote Authentication

The Confusion Matrix with these parameters shown in figure 41, determined that the model is predicting with 100.0% Train Accuracy, 99.2% Test Accuracy, 100.0% CV Accuracy, 98.2% Precision Score, 100.0% Recall Score and 99.1% F1 Score

LetterRecognition SVM calculated BEST parameters are rbf Kernel, 3.593813663804626 C value, 0.3593813663804626 Gamma with 0.96 score at 82% Train and 18% Test

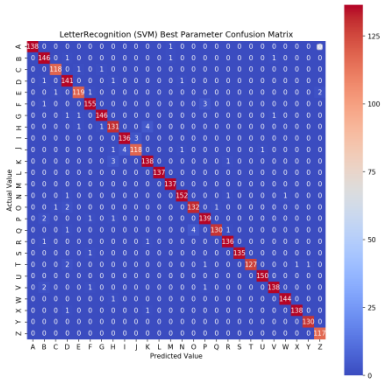

Figure 42: Letter Recognition

The Confusion Matrix with these parameters shown in figure 42, determined that the model is predicting with 100.0% Train Accuracy, 98.0% Test Accuracy, 96.4% CV Accuracy, 98.0% Precision Score, 98.0% Recall Score and 98.0% \F-1 Score

The sklearn RandomizedSearchCV values for both datasets show very impressive accuracy values well above 98%, with Banknote reaching above 99% in all metrics. Additionally, the number of miss-classifications for both dataset is near zero, as displayed by the Confusion Matrix.

B. Varying Training vs. Test size

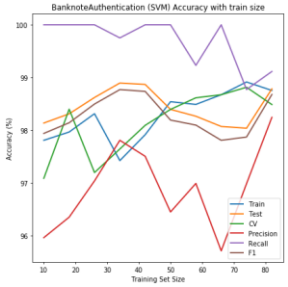

Figure 43: Banknote Authentication

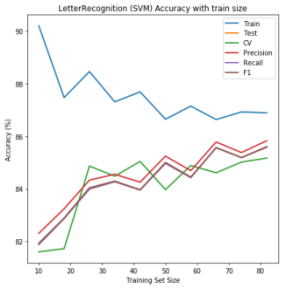

Figure 44: Letter Recognition

Figure 43, Banknote dataset, shows better performance with large Test/Train ratios observed by the increase in Precision, Test and SCV accuracy at the 82/18 ratio. This is in stark contrast to the output of sklearn.RandomizedCV function which attained best values at the 10/90 ratio. It is not clear from this analysis why the sklearn optimized value algorithm would choose this ratio, but one indicator consistently falling with increase in training is Recall accuracy. It is a valid assumption that sklear values represent those at which the least number of False Positives are observed.

Figure 44, similarly, shows best performance with a 82/18 Train/Test ratio. This is corroborated by the RandomizedCVSearch values achieving best performance at the same ratio. Interestingly, training accuracy falls with larger training set, most likely caused by the complexity of the dataset and attempting to classify among 26 classes.

C. Varying Kernels

Kernels are an important part of SVM learning. Depending on the dataset, a different kernel may be better suited at finding best hyperplanes than others. The goal of the kernel is a method by which data is transformed into higher dimensions, allowing hyperplanes to be determined for better classification.

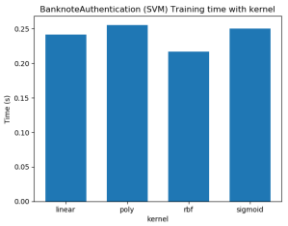

Figure 45: Banknote Authentication

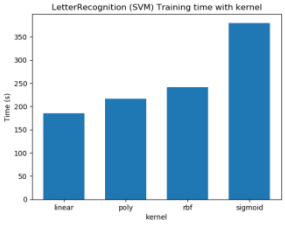

Figure 46: Letter Recognition

Figure 45, shows the training time of each kernel over the Banknote dataset. The ‘rbf’ kernel, the most popular to use and the one with best accuracy according to RandomizedSearchCV is also the one requiring least amount to train. This kernel will be used for the rest of the hyper-parameter evaluations.

Figure 46, similarly shows a good training time for the rbf kernel. However, attention should be brought to the Y-scale of the two dataset's training time. While the Banknote dataset training time is in fractions of a second, the Letter dataset is in hundreds of seconds. This is indicative of the complexity of the dataset and the time required to train and tune the hyper-parameters for each learner.

D. Varying C Value

For SVM classification, the C value determines the allowed error rate of the hyperplane classification. For high values of C, the SVM learner is not allowed to misclassify data points. For low values of C, the SVM learner is allowed some misclassifications, but this allows the algorithm to define a better hyperplane. In other words, a large C value provides low bias and high variance while a small C value provides high bias and low variance of the data. For this evaluation, the C value are varied between 0.1 and 10 on a logarithmic scale with equal spacing.

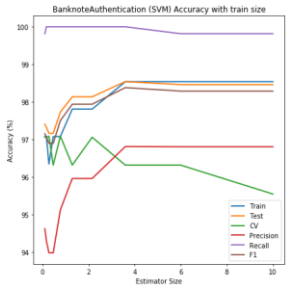

Figure 47: Banknote Authentication

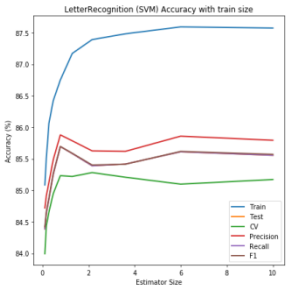

Figure 48: Letter Recognition

Figure 47, Banknote dataset attain the best accuracy scores approximately at a C value of 4. This is verified by the RandomizedSerach CV also favoring a value of 3.59. This value of C retains good CV accuracy while achieving best accuracies in other metrics. Continue to increase the C value beyond this point lowers the accuracy metrics. This is an indication of overfitting as the hyperplanes begin to over fit the data.

Similarly, Figure 48 displays a graph where the best C value is reached at approximate value of 2. This is evident by the drop-off in CV accuracy as the C value increases. Similarly, this decrease in accuracy for higher values of C is a clear sign of overfitting on hyperplanes to the data.

E. Varying Gamma

The gamma value, used only in the 'rbf' kernel is used to vary the separation of points in higher dimensions of the kernel. A small gamma provides small separation in higher dimensions, while a large gamma provides a large separation. In other words, a small gamma will provide low bias and high variance, while a large gamma will provide a high bias and low variance of data. For this evaluation, the gamma values are varied between 0.1 and 1.0 on a logarithmic scale, equally spaced

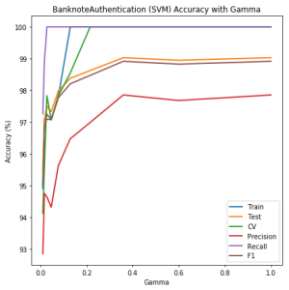

Figure 49: Banknote Authentication

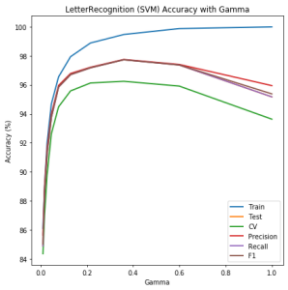

Figure 50: Letter Recognition

Figure 49, Banknote authentication displays a clear trend upwards with a best value approximately at 0.4. Above the gamma value of 0.4, overfitting takes over and Test, Precision and F1 accuracy metrics begin to lower.

Figure 50, Letter Recognition displays a similar trend as figure 49. Above a gamma value of 0.4, overfitting takes over with all accuracy metrics lowering except for Training accuracy. This implies the learner loses accuracy on un-seen data as it is overfitting to the training set.

IX. TIME ANALYSIS

Each dataset and its corresponding learner behaves optimally for different learning parameters. The following set of figures show how learning time varies with each dataset and learning technique applied throughout this analysis. The index value of the x-axis is a normalization of hyper-parameters and Train/Test ratios to values 1 to 10.

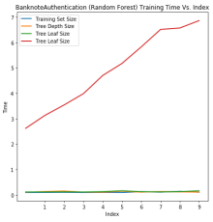

Figure 51: Banknote Decision Trees

Figure 52: Banknote AdaBoost

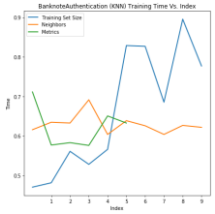

Figure 53: Banknote KNN

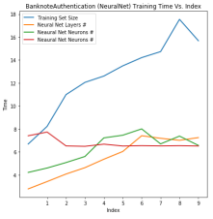

Figure 54: Banknote Neaural Net

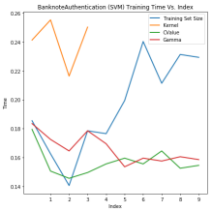

Figure 55: Banknote SVM

For Banknote dataset, as evident in Figures 53 and 55 KNN and SVM training were the quickest. This is due to the

binary nature of the classification. SVM does not require complex dimensionality to classify the data and KNN is itself an algorithm that is a lazy learner, leaving evaluation to the prediction phase. Of the slowest learners, Neural Nets, Figure 54, training phase took the longest amount of time as building multiple neurons and layers, as well as adjusting all the weights is a time extensive process.

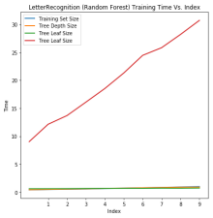

Figure 56: Letter Decision Trees

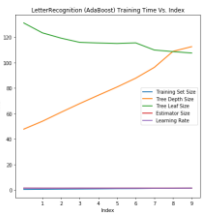

Figure 57Letter AdaBoost

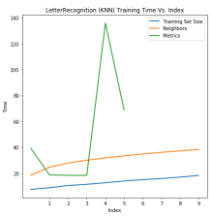

Figure 58: Letter KNN

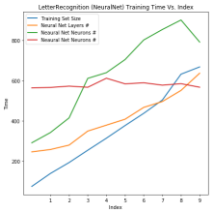

Figure 59: Letter Neural Nets

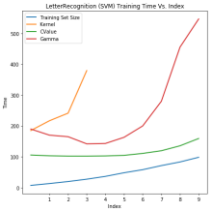

Figure 60: Letter SVM

For Letter dataset, it is evident that all algorithms struggled with the classification of 26 classes. The worst performing learner was Neural Nets considering the cost of numbers of neurons, layers, and weight scaling, this is not surprising. KNN, a lazy learner also had some delays in learning with different distancing metrics and decision Trees took the longest evaluating leaf size. All the delays can be attributed to the complexity of the dataset under evaluation.

X. CONCLUSION

The two datasets evaluated in this analysis were picked due to their interesting characteristics. One with binary classification and continuous valued attributes, while the other is a multi-class dataset with integer attributes. These characteristics translated directly into performance with learning techniques analyzed.

The Banknote dataset performed very well with the simplest classifier, Decision Trees, attaining Test accuracy scores over 96%. However, the Letter dataset struggled with a Test accuracy of only 72%. More complex learners such as Neural Net and SVM were able to bring up the accuracy of Letter Recognition to ~96%.

The analysis of hyper-parameters was the most fascinating to compare as the simple Banknote dataset would often over fit high values of most hyper-parameters, while the Letter dataset would balance between proving too complex of a classification and overfitting to 26 classes while tuning the hyper-parameters.

In conclusion, comparing manual variation of hyper-parameters to an optimized function such as sklearn.RandomizedSearchCV showed that tuning requires careful attention. Complex datasets often would result in values from sklearn which did not translate correctly to those present by analyzing each parameter separately. Going forward, it is critical to identify, characterize, and fine-tune the best applicable learner and its hyper-parameters, carefully fitting the data there is being trained on. From this analysis, it is clear that all algorithms were capable of achieving accuracy metrics well above chance (50%) with SVM learning attaining the highest scores for both datasets. Overall, the analysis run time took over 24 hours to execute all code.

XI. REFERENCES

[1] Banknote Authentication Dataset – Retrieved from https://archive.ics.uci.edu/ml/datasets/banknote+authentication

[2] Letter Recognition Dataset – Retrieved from https://archive.ics.uci.edu/ml/datasets/Letter+Recognition

[3] Source Code https://github.gatech.edu/spalguyev3/Fall2020CS7641

[4] https://www2.unil.ch/biomapper/Download/Lobo-GloEcoBioGeo-2007.pdf

[5] https://www.geeksforgeeks.org/confusion-matrix-machine-learning/