**Supervised Learning**

**Abstract**

This assignment explores various supervised learning algorithms by comparing and contrasting their properties while training on two different datasets. The algorithms will first be analyzed according to different hyperparameters in order to choose the best performing settings for each dataset, then the algorithms will be aggregated together to compare the performance on training/testing a similar dataset. There will be two datasets used for this assignment as described as follows:

* **MNIST Handwritten Digits:** This dataset is one of the most well known in data science. It consists of 60K training and 10K test examples and was created by LeCun et al. The supervised learning task associated with the data is: given an image of a handwritten digit, determine which digit the image represents. For this assignment, the data was split into columns representing the individual pixel values and put into csv files for easier computation. The distribution of digits is approximately even, providing the algorithms with an even dataset to classify digits, but the high number of features means that overfitting is a concern, hence the focus on graphing multiple hyperparameters with respect to the validation/training performance.
* **Wine Quality:** This dataset was obtained from the public repository maintained by University of California Irvine. The data consist of 6497 measurements of the physicochemical properties of both red and white Portuguese “Vinho Verde” wine. Each wine sample has an associated quality rating, which was determined by taking the median of at least 3 evaluations made by wine experts. The dataset presents two important distinctions from MNIST: the larger presence of noise in the labeling and the smaller sample size. Since the evaluations were made by taking the median of human opinion, there exist some variation on the true quality of a wine sample, so the relationship between wine characteristics and rating may not be 1 to 1. The smaller sample size will also contribute to significant differences in performance compared to MNIST, since there will be less data to make decisions on.

**Methods**

All algorithms were trained on two datasets. The sets were split into a training set and validation set; each training run used 6-fold validation to run the training. 6-fold validation is when the training set is split into 6 different sections, and for 6 different iterations the algorithm is trained on 5 of the 6 sets, with the 6th set used for validation. Each algorithm is tuned using two hyperparameters and trained on both of the aforementioned datasets. The final analysis is conducted with the algorithm tuned with the best performing hyperparameter and compared according to training set size.

**Decision Tree**

A Decision Tree utilizes a tree of decisions to make a final classification on a data point. The goal of each decision is to decrease the number of possible outcome classifications a given input could have; the tree is run on the entire dataset and generated by choosing features that decrease the overall entropy of the dataset.

**Hyperparameters**

The decision tree is limited by the depth of the tree and the number of leaves it may have.

**MNIST**

The above charts are of the Decision Tree trained on the MNIST dataset. According to the left chart, the max number of leaves seems to only matter up to a certain point, specifically around the 400 mark. An increase in the number of leaves continues to improve the training performance but does not improve the validation performance. A likely explanation for this trend could be the higher number of leaves cause overfitting on the data.

Similar performance can be observed on the right graph. As the tree depth grows larger, the decision tree performance on the validation set reaches a plateau at around 0.85 accuracy. Again, the cause of the increase in training performance but stagnating validation performance could be due to overfitting.

**Wine Quality**

The above charts are the Decision Trees trained on the wine quality dataset. Both charts show similar learning curves: the training and validation accuracy are similar at the beginning, but quickly diverge as the hyperparameter increases. Since the training accuracy approaches 1 for max depth and even reaches 1 for max leaves, but the validation accuracy remains relatively constant, a likely explanation for these two graphs is overfitting. With the small amount of data but high number of hyperparameters, the decision tree begins to learn the noise of the dataset and fails to generalize across the entire function space. The small dataset size also contributed far more to overfitting in the wine dataset, since unlike MNIST dataset, the validation curves do not reach high accuracy at all.

**Final Hyperparameter Choice**

Max Leaves: 400

Max Depth: 10

**KNN**

KNN, or K-Nearest Neighbors, uses a distance metric among a cloud of points to classify a given data point. Each sample in a dataset is considered a point in N-dimensional space, where N is the number of features a sample has. Distance is then calculated to find the nearest K points and classified according to the points found.

**Hyperparameters**

The algorithm is trained by varying K nearest neighbors and the distance metric

**MNIST**

For the left graph, as K is adjusted, both the validation accuracy and the training accuracy decrease. The training accuracy begins at 1, which is expected since the first nearest neighbor among the training set is always the sample itself. As more K is calculated, the training error increases, since the more samples that are taken into account, the more likely the sample itself will be misclassified into another class.

**Wine Quality**

Above is wine quality dataset

**SVM**

SVM, or Support Vector Machine, draws hyperplanes on a dataset and divides the data.

**Hyperparameters**

The algorithm is trained with hyperparameters as follows

**MNIST**

Above is MNIST dataset

**Wine Quality**

Above is wine quality dataset

**Boosting**

Boosting is an iterative variant on Decision Trees in which a weak classifier is fit on a dataset, and then subsequent copies of the classifier are iteratively fit with each iteration focusing more on the difficult classification problems. Weights are used to focus on the difficult, or wrongly classified, samples, with higher weights being placed on incorrect classification versus lower weights being placed on correct classification. This assignment used the Adaboost.SAMME.R variation of the algorithm, which supports multiclass classification.

**Hyperparameters**

Adaboost is adjusted by the learning rate and maximum number of estimators.

**MNIST**

The curves for validation and training closely follow each other for both hyperparameters. The number of estimators seems to show significant improvement for the first 30, but as more estimators are added, the accuracy does not improve at all. Overfitting is unlikely to be present in this case, since the training accuracy closely followed validation accuracy. We can conclude that the algorithm can generalize this dataset well, since the training accuracy is almost the same as the validation accuracy.

**Wine Quality**

We see again that the curves for the Adaboost hyperparameters closely follow each other, albeit at a much lower accuracy. For this dataset, the number of estimators does not have a significant impact at all on the accuracy of the predictions. The only noticeable effect is a slightly higher accuracy rate for 1 estimator, but for all other number of estimators the accuracy hovers around 0.39. Strangely, the learning rate hyperparameter for this dataset improves as the learning rate increases.

**Final Hyperparameter Choice**

Number of Estimators: 50

Learning Rate: 1

**Neural Network**

Neural Networks, in this case MLP, or multilayer perceptron, uses a network of functions that takes in inputs and changes weights iteratively.

**Hyperparameters**

The network is trained with hyperparameters: Alpha Value and Number of Hidden Layers

**MNIST**

Above is MNIST dataset

**Wine Quality**

Above is wine quality dataset

**Analysis**

Below is a plot of the aggregate performance on a test dataset.

**Aggregate Learning Curve Graph**