Supervised Learning

Rohan Ramakrishnan
Georgia Institute of Technology
rohanrk@gatech.edu

***Abstract*—In this analysis, several supervised learning algorithms were applied to two different datasets. The results and performance of each algorithm are compared and examined across both datasets. Each dataset was split into training and test sets with the split being 70% training and 30% test. The algorithms used include decision trees, boosting, k-nearest neighbors, support vector machines, and neural networks. The datasets were tested using the implementations provided by Weka. All training set analyses were run using 10-fold cross validation.**

I. Datasets

The first dataset is a breast cancer dataset comprising 286 instances. These instances were split into 199 training instances and 87 testing instances. Each instance is defined by 9 attributes, both nominal and linear, excluding the class label. Attributes represent features relevant to breast cancer diagnosis such as age, menopause. There are 2 classes, which correspond to whether the cancer is recurrent or not.

The second dataset, optdigits, consists of normalized bitmaps of handwritten digits. There are a total of 5620 instances, each divided into 10 distinct classes representing every decimal digit. Each instance contains 64 attributes, each of which corresponds to the number of pixels in non-overlapping 4x4 partitions of 32x32 bitmaps. The dataset contains a total of 5620 instances which are divided into 3928 training instances and 1692 test instances.

Both datasets in this paper are publicly available in the UCI Machine Learning Repository. The motivation for selecting these two in particular is to compare the performance of supervised learning algorithms on datasets of different complexities. Both datasets differ in size, number of attributes, and number of labels. The breast cancer dataset is a binary classification problem with 286 instances, each defined by 9 attributes. By contrast, optdigits is a multi-class dataset containing 5620 instances defined by 64 attributes. This presents a more complex classification problem and allows for the comparison of supervised learning algorithms. Furthermore, the number of instances in each class are approximately uniform in optdigits. However, in the breast cancer dataset, most of the instances belong to the negative class. This difference offers some insight into how each algorithm will perform against datasets with different distributions.

II. Decision Trees

A. Introduction

Decision trees are supervised learning models that map instances, attributes, and attribute values to a corresponding tree structure. Each non-leaf node represents a particular attribute, each branch represents a possible value of its parent node, and each leaf node represents a class label for an instance. For the purposes of this project, Weka's J48 classifier implementing the C4.5 decision tree model was used.

The main factors that determine the accuracy of a decision tree are the algorithm used and the pruning methods used to avoid overfitting. C4.5 is an improved version of the ID3 algorithm that uses information gain to pick the best attribute as the root tree and recursively builds a decision tree from there. The J48 tree utilizes two pruning methods to decrease the chance of overfitting to the training set. The first is known as subtree replacement, which means certain nodes in the decision tree may be replaced with a leaf node, thereby reducing the number of tests along a certain path. The second type of pruning used is known as subtree raising. Certain subtrees may be moved upwards closer to the root of the tree, replacing parent subtrees. Both these methods were used when running the J48 algorithm on these datasets.

The first step involved tuning the hyperparameters of the algorithm. There were two hyper parameters that were tuned. The first is the confidence factor of the tree. This value is used for pruning the decision tree. The smaller the value, the more leaves are pruned from the tree resulting in smaller decision trees. The second value is the minimum number of instances per leaf. This value guarantees that at each split, at least two branches will have the minimum number of instances. This value is useful to split the data in a way that maximizes information gain. Figure 1 and Figure 2 illustrate the change in accuracy with respect to different confidence factors given a constant minimum number of objects value of 2.

B. Results

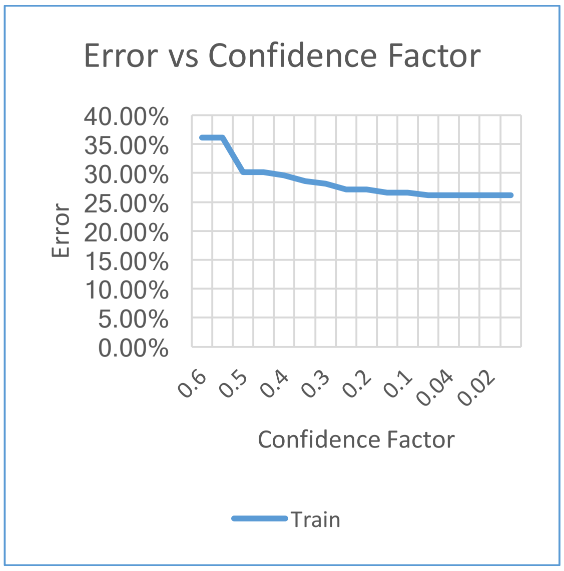

Figure 1: Breast Cancer - Percent error with respect to confidence factor

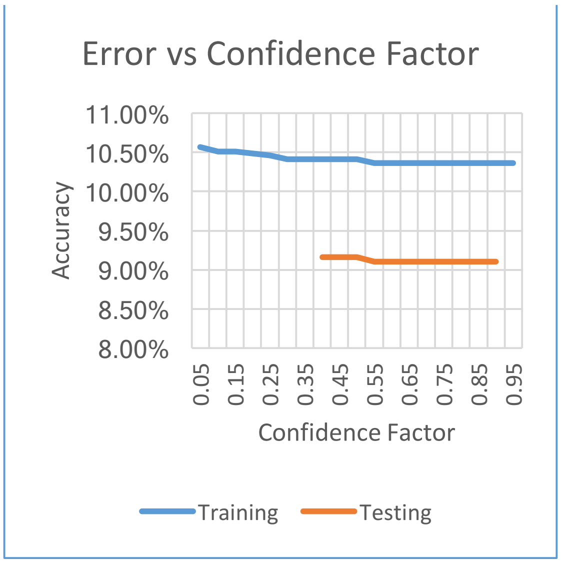

Figure 2: Optdigits - Percent error with respect to confidence factor

Figure 1 and figure 2 show the percent error of the J48 tree given different confidence factor values and constant minimum number of objects per leaf node. For the breast cancer dataset, the optimal confidence factor is a low number. The nine attributes must be pruned significantly in order to accurately classify positive and negative instances.

Optdigits required a much higher confidence factor at around 0.75 in order for J48 to provide optimal results. This is due to the nature of the attributes in the dataset. In the breast cancer dataset, there were relatively few attributes and each attribute had a smaller number of possible values. In optdigits, each instance has 64 attributes that can take on 10 possible values. Therefore, the decision tree is going to have several branches and a low confidence factor can lead to pruning important branches that are needed to classify a given instance. Another key difference is that optdigits is a multiclass dataset, so each attribute is considered more important to split the instances into several different classes as opposed to positive and negative classes.

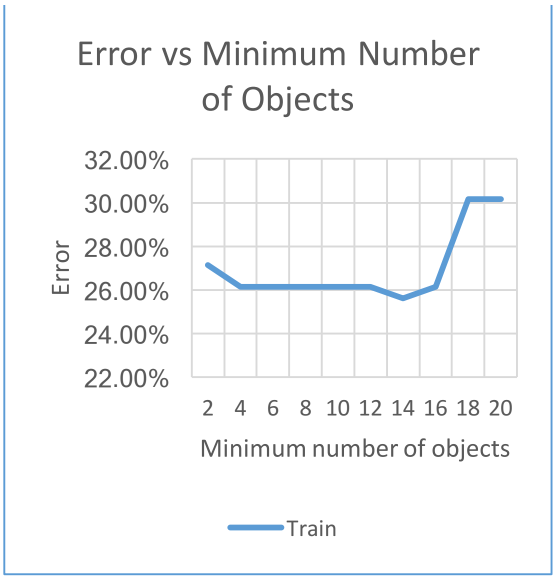

Figure 3: Breast Cancer - Error with respect to minimum number of objects parameter

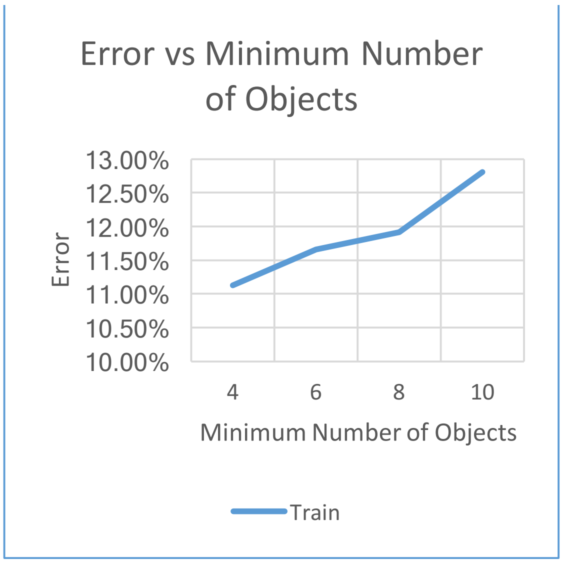

Figure 4: Optdigits - Error with respect to the minimum number of objects parameter

Given our ideal confidence factor, figure 3 and figure 4 show the accuracy of J48 across different values of the minimum number of objects per leaf node. The optimal value is around 14 for the breast cancer dataset and two for the optdigit dataset. This parameter forces our decision tree to prefer attributes that will provide the most information gain by forcing an even split of instances along the tree. If we could only isolate one instance in a node that would not give us much information, but by attempting to split a unuform number of instances across each branch, we maximize the information gain and thus, the optimal parameter for optdigits is a low number.

Normally having a smaller number of instances per leaf is better for a decision tree, but in the case of our first dataset, a higher value is more useful in splitting the decision tree. This result illustrates a feature of our dataset. Since there are so many more negative instances than positive instances, a decision tree with higher minimum instances per leaf will help identify properties of a positive instance and help avoid misclassification by separating those instances from a larger group of negative instances.

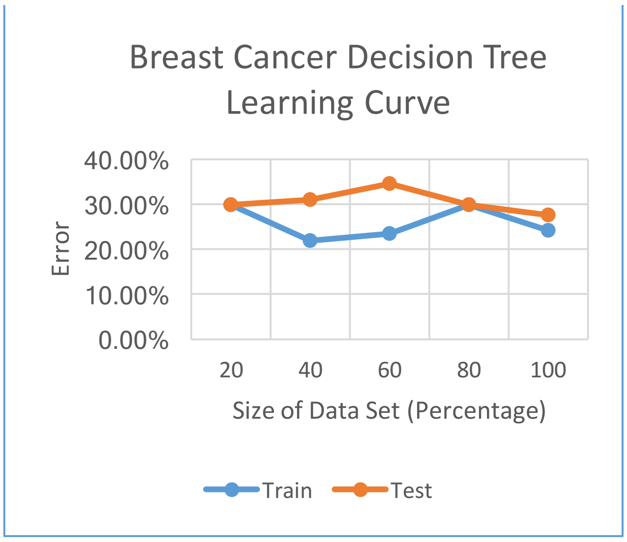

Figure 5: Learning curve for Weka's J48 decision tree on breast cancer dataset

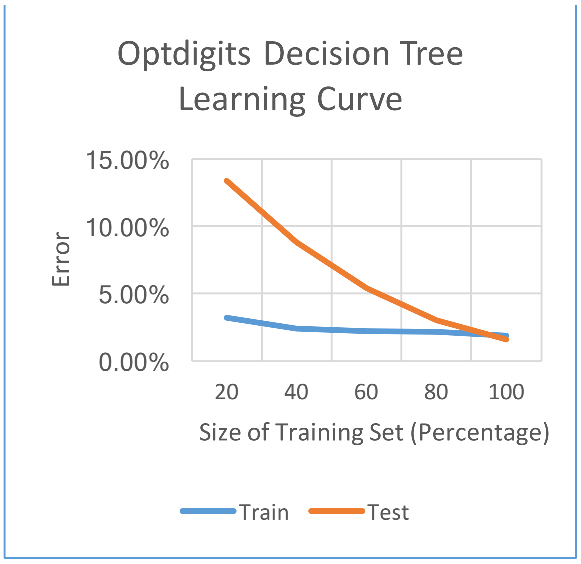

Figure 6: Learning curve for Weka's J48 decision tree on optdigits dataset

Figure 5 and figure 6 show the learning curves of the decision tree algorithm, the function of training and testing accuracies with respect to different training set sizes. The algorithm was run with optimal parameters that were found from the previous figures. The gap between the curves is small in for the breast cancer dataset. However the gap is much larger for optdigits at smaller training set sizes. This showcases the importance of generalization for the optdigits dataset and the necessity of several examples to learn the patterns in the attributes to output a reasonable model for the classification problem.

An interesting observation is that the error varies around between 40% and 80% in the breast cancer dataset. This is the result of variance in randomly sampling the training dataset. It also shows that the number of examples are less important than the specific examples chosen since at 20% the error was lower as the training size increased till 80%. So the examples chosen at 20% were more effective in classifying the data than at higher percentages.

Table 1: Breast Cancer Testing Results

Breast Cancer		
Confidence	***minNumObj***	***Error***
0.05	14	27.59%
0.05	12	27.59%

Figure 7: Breast Cancer - Decision tree results on test set with optimal parameters

Table 2: Optdigits Testing Results

Optdigits		
Confidence	***minNumObj***	***Error***
0.75	2	9.10%
0.7	2	9.10%

Figure 8: Optdigits - Decision tree results on test set with optimal parameters

C. *Analysis*

Decision trees performed quite well on both datasets. The breast cancer dataset was a great dataset for decision trees as it has mostly discrete attributes which help reduce the number of branches needed.

It is also important to note that decision trees ran very quickly across both datasets, with the longest time to build the model being little over three seconds. This is especially important considering the size of optdigits.

III. Boosting

A. *Introduction*

Before we can begin a discussion on boosting, we must first define the term weak learner. A weak learner is defined as any classification algorithm that will perform better than random chance given any distribution of examples. Boosting is an iterative probabilistic method that improves the performance of any weak learner. This method works by assigning higher priority or weight to examples that the given learner labels incorrectly in the previous iteration. This allows the learner to improve its performance over each iteration. For this discussion, Weka's AdaBoosM1 algorithm was used with a J48 decision tree selected as the weak learner to apply to the selected datasets.

B. *Results*

Figures show the error with respect to the number of iterations. When applying the algorithm to the optdigits dataset, the confidence factor was set to 0.25. However, for the breast cancer dataset, the confidence factor was also varied because the dataset was so small that boosting past a certain point would prune the tree and end with just one root node as the decision tree.

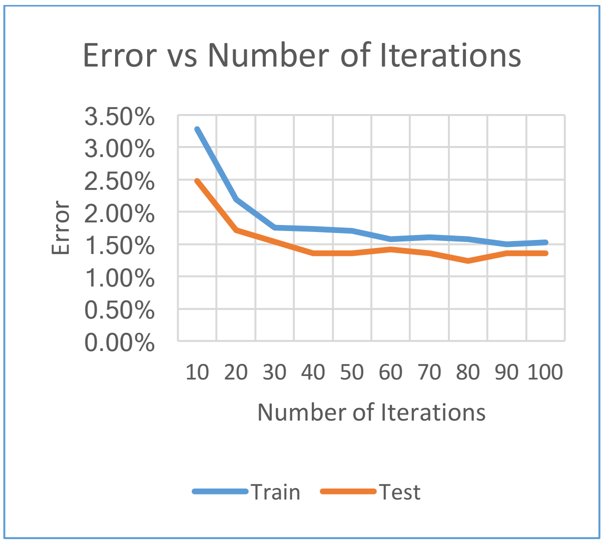

Figure 9: Optdigits - Error compared to the number of iterations performed when applying AdaBoostM1 to the dataset

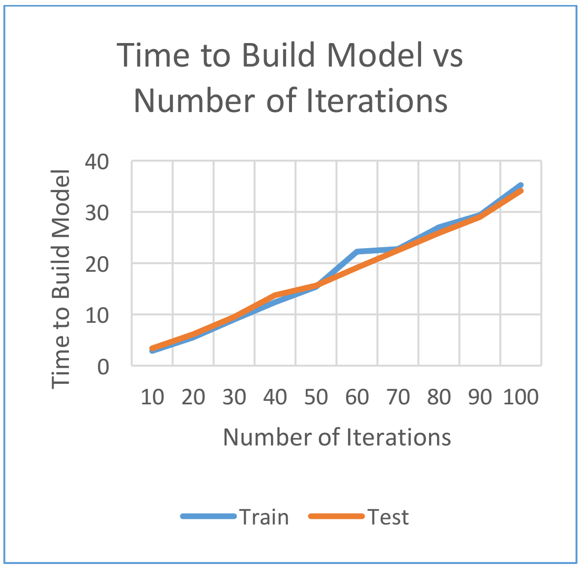

Figure 10: Optdigits - Time taken to build model with respect to the number of iterations

Figure 9 shows the performance of boosting as the number of iterations increase. The error decreases as the number of iterations increase as the weak learner constantly improves after each iteration by building a new decision tree to account for its mistakes from the previous iteration. This model appears avoid overfitting as well as both 10-fold cross validation and testing error are similar. The error is exceptionally low compared the error from running a decision tree algorithm to this dataset without boosting.

Figure 10 shows the time taken to build the model with respect to the number of iterations. This is negligible in the breast cancer dataset but in optdigits, the time increases up to a high of about 35 seconds. The time taken increases almost linearly as more iterations are performed, which is expected in almost any iterative method.

The breast cancer results vary much more than optdigits'. For the purposes of this dataset, the confidence factor of the decision tree was also varied. As the number of iterations increased, the error would decrease up to a certain point. However, the number of iterations performed on the data plateaued at a certain point. At this point, the previous errors were classified using a decision tree with only one vertex. Due to this, error increased if the algorithm hit this point. So the confidence factor was varied in order to compare performances across different trees and to find how many iterations would be performed before error increases. The table below summarizes results on the test data.

Table 3: Boosting Results on Breast Cancer Test Set

Number of Iterations	**Confidence Factor**	**Percent Error**
10	0.25	3.45%
20	0.25	2.30%
30	0.25	3.45%
15	0.5	2.30%
20	0.5	3.45%
25	0.5	3.45%
30	0.5	3.45%
35	0.5	3.45%

Figure 11: Shows the number of iterations increasing error for small datasets

This result occurs due to the small size of the breast cancer dataset. The optdigit dataset does not encounter this result since it has significantly more attribute with more possible values. So the number of iterations required to reach a similar result is extremely high. The breast cancer dataset only has nine attributes and most of those have fewer than 10 possible values so the decision tree required

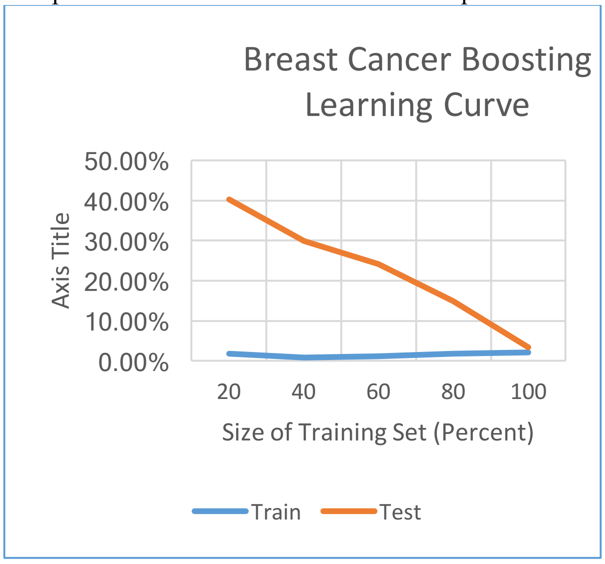

Figure 12: Learning curve for Weka's AdaBoostM1 algorithm on breast cancer dataset

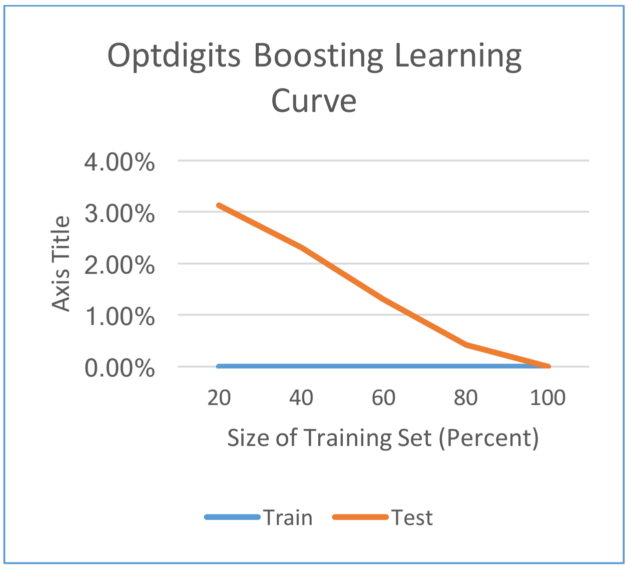

Figure 13: Learning curve for Weka's AdaboostM1 algorithm on optdigits dataset

C. *Analysis*

Both results do show that boosting greatly improves the performance of the weak learner. The decision tree's performance is greatly enhanced using the AdaBoostM1 algorithm. Error for both datasets decrease over 8% on average which is excellent and shows the effectiveness of boosting algorithms in improving the performance of weak learners.

However, the learning curve for the breast cancer dataset has high variance between training and test curves. This indicates high levels of overfitting and bias in the model generated by the weak learner even after boosting is applied. The model can tell that the majority of instances are in the negative class, however it struggles to distinguish between the many negative instances and the fewer positive instances. This illustrates the detriment of overfitting in real world machine learning applications as no one would want close to 40% error when diagnosing recurrent breast cancer.

Another trend to note is optdigit's learning curve. The training error is 0% no matter what the size of the training set is. Despite this, the testing curve's error slowly converges to 0% error as well implying that even though this model is overfitting to the training data, it still generalizes well enough to unseen instances to be considered an effective hypothesis.

IV. K-Nearest Neighbors

A. *Introduction*

The nearest neighbor algorithm is a model that allows us to classify unknown input data with respect to the current model. For any given data point, we calculate the k closest points to that point and then either use voting (that is, assign the label that occurs most frequently among the neighbors) for classification or average out the output values of those neighbors for regression and assign the result to our input. In order to do this, we need some distance function. Weka uses the Euclidean distance as the distance function in its implementation of KNN. For this section, Weka's IBK algorithm was utilized to show the performance as the number of neighbors considered and type distance weighting were varied.

B. *Results*

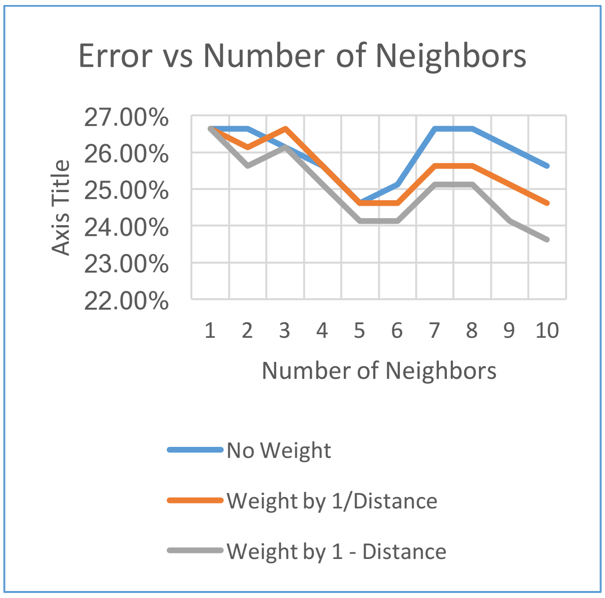

Figure 14: Breast Cancer - Error with respect to the number of neighbors selected for all distance weighting

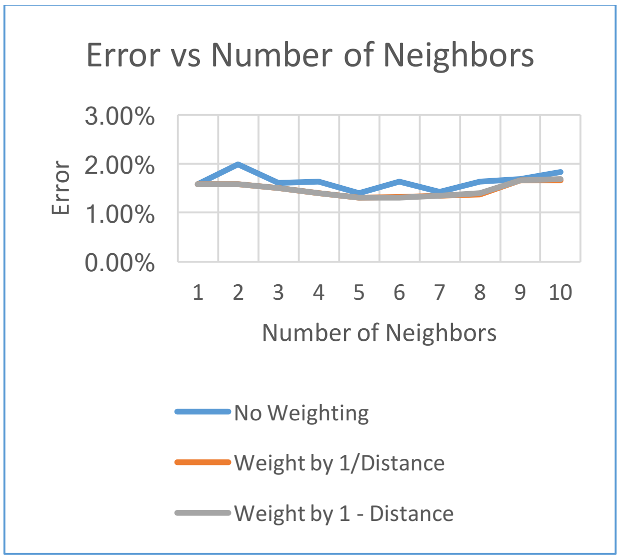

Figure 15: Optdigits - Error with respect to the number of neighbors selected for all distance weighting

Figures 11 and 12 show the cross validation error of the KNN algorithm as the number of neighbors considered increases. All three types of distance weighting were used so the results of increasing the weight of closer neighbors can be observed. By increasing the weight of neighbors that are closer to a given data point, the error decreases slightly which is expected as the foundational assumption of KNN Is that neighbors that are closer together are more likely to be similar than neighbors that are further apart. This error is more easily observed with the breast cancer dataset, which shows a greater variance of errors across the different weights.

Table 4: Breast Cancer - Results of KNN on testing set with tuned hyperparameters

KNN	Weighting	Percent Error	Mean Absolute Error
5	No Weighting	26.44%	0.3511
6	No Weighting	26.44%	0.348
5	by 1/Distance	26.44%	0.3358
6	by 1/Distance	26.44%	0.3327
5	by 1 - Distance	26.44%	0.3481
6	by 1 - Distance	26.44%	0.3454

Figure 16: Breast Cancer - Hyperparameters chosen based on highest performance on cross validation. Top two hyperparameters per weighting method were selected

Table 5: Optdigits - Results of KNN on testing set with tuned hyperparameters

KNN	Weighting	Percent Error	Mean Absolute Error
5	No Weighting	1.30%	0.0043
7	No Weighting	1.42%	0.0051
5	by 1/Distance	1.30%	0.0041
6	by 1/Distance	1.06%	0.0045
5	by 1 - Distance	1.30%	0.0043
6	by 1 - Distance	1.06%	0.0047

Figure 17: Optdigits - Hyperparameters chosen based on highest performance on cross validation. Top two hyperparameters per weighting method were selected

Tables 2 and 3 show the testing results of KNN tuning the number of neighbors based on the best performance with cross validation. The results highlight a key difference between both datasets. The breast cancer dataset has a constant testing error for all the values of K and all types of weighting. This indicates that data points that are close to a particular instance, as calculated by the Euclidean distance algorithm, are not necessarily more similar to that instance since no matter how we weight the points, the error remains the same.

This observation also indicates that an instance's neighbors are not a good metric for classifying the unseen instances of this dataset. As a binary dataset, there. This is consistent with real life for there are several patients with similar symptoms who may be diagnosed differently. Logically, just because some patients have similar symptoms does not imply that every one of them has cancer. This is also a potential flaw with the dataset for diagnosing cancer with only nine attributes is a difficult task, even for humans.

By contrast, optdigits' testing error decreases as weighting is applied which suggests that data points with similar attributes will most likely belong to the same class. Not only that, but data points that are closer to a given instance are likely part of the same class as shown by the decrease in error once distance weighting is applied.

Additionally, KNN's performance on optdigits' is noticeably better than its performance on the breast cancer dataset since the underlying assumption of KNN (neighbors are likely part of the same class) holds true for the data in optdigits (bitmaps belonging to the same digit are likely to look similar) but it does not necessarily hold true for the data in the breast cancer dataset.

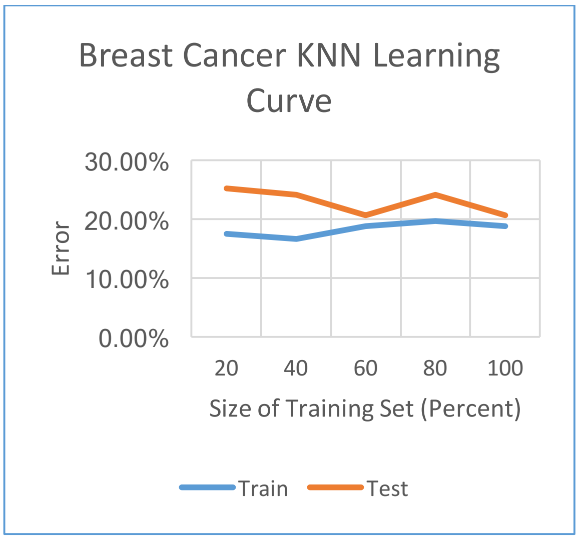

Figure 18: Learning curve for Weka's IBK algorithm applied to breast cancer dataset

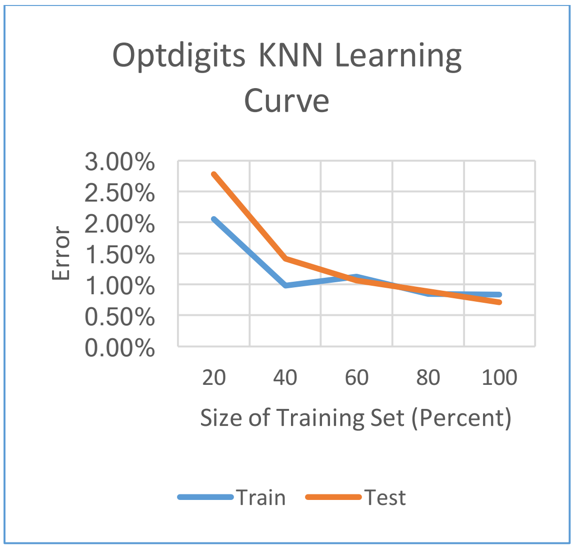

Figure 19: Learning curve for Weka's IBK algorithm applied to optdigit dataset

C. Analysis

Figures 17 and 18 show the learning curves for KNN on both datasets. The breast cancer learning curve is further prove of the ineffectiveness of KNN on that particular set of data. The error is relatively constant no matter how the training set increases which shows that the number of possible neighbors does not improve performance. Optdigit's learning curve, on the other hand, deccreases steadily as more possible neighbors become available.

One can see that KNN has high potential for multiclass datasets with a high number of attributes as it will be able to separate different instances more effectively. KNN's runtime also increases as size of the training set increases as shown in figure 19

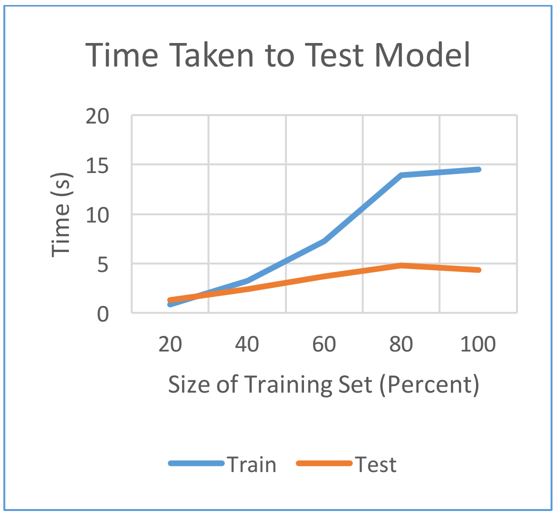

Figure 20: Optdigits - Time taken to test KNN model with respect to training size

This trend is expected. As the number of possible neighbors increases, the algorithm has more points to compare using the distance function which results in a higher runtime.

V. Support Vector Machines

A. Introuction

Support Vector Machine is an algorithm that generates a hyperplane to separate instances into classes. In order to do this, SVMs often apply a function, known as a kernel, to transform the training data into a higher number of dimensions so that instances can be separated. For this section, two different kernel functions were used. Weka's SMO algorithm (sequential minimal optimization) was used with both a Polynomial Kernel and Radial Basis Function. Although SVMs are linear binary classification problems, there are methods to utilize them for multiclass problems. Weka's SMO classifier uses pairwise classification, aka, one-against-one approach, to solve multiclass problems.

B. Results

1) Polynomial Kernel

The polynomial kernel function is defined as according to equation 1 [1].

Equation 1: Polynomial Kernel Function

$$k(\vec{x},\vec{z}) = (\vec{z}^T\vec{x} + c)^n$$

For the polynomial kernel function, the exponent was varied in order to test which value would produce the best hyperplane that is capable of generalizing both datasets. Cross validation results are shown in table 5 and table 6.

Table 6: Results of Polynomial Kernel on Breast Cancer

Exponent	CV Percent Error	CV Mean Absolute Error	Testing Percent Error	Testing Mean Absolute Error
0.125	31.16%	0.3116	35.63%	0.3563
0.25	30.15%	0.3015	35.63%	0.3563
0.5	28.64%	0.2864	39.08%	0.3908
1	25.63%	0.2563	31.03%	0.3103
2	42.71%	0.4271	34.48%	0.3448
4	34.17%	0.3417	33.33%	0.3333
8	38.19%	0.3819	57.47%	0.5747
16	54.77%	0.5477	59.77%	0.5977

Figure 21: : Performance of polynomial kernel function on breast cancer dataset with varying exponents

Table 7: Results of Polynomial Kernel on Optdigits

Exponent	CV Percent Error	CV Mean Absolute Error	Testing Percent Error	Testing Mean Absolute Error	Time to Build Model (s)
0.125	4.71%	0.2726	4.55%	0.2726	3.62
0.25	4.05%	0.2725	3.61%	0.2724	2.2
0.5	3.28%	0.2722	3.01%	0.2722	1.49
1	1.91%	0.2718	1.30%	0.2718	0.4
2	1.50%	0.2719	0.71%	0.2718	0.89
4	1.20%	0.272	0.95%	0.272	1.28
8	1.93%	0.2722	1.42%	0.2721	1.25
16	6.49%	0.2726	3.84%	0.2723	1.51

Figure 22: Performance of polynomial kernel function on optdigits dataset with varying exponents

Once again, the performance on the optdigits dataset outshines the performance on the breast cancer dataset. The optdigits works better for SVMs as the notion of similarity between each instance can be defined by a number or a vector (since we can easily subtract the values in a bitmap and compare the difference between them). So each handwritten digit will have similar values in the bitmaps and so SVMs can use this property to separate instances once they have been mapped to a higher dimension. The breast cancer dataset, on the other hand, does not have this property which makes it much harder to separate data points even though it is a binary classification dataset.

2) Radial Basis Function

The Radial Basis Function is as follows

Equation 2: Radial Basis Function

$$g(\vec{X}) = \sum_{n=1}^{N} \vec{W}_n^T f(||\vec{X} - \vec{X_n}||)$$

So we are summing the Euclidean distance of some set of points $\{\vec{X_N}\}$.

Table 8: Results of RBF Kernel on Breast Cancer Dataset

Gamma	CV Percent Error	CV Mean Absolute Error	Testing Percent Error	Testing Mean Absolute Error
0.01	29.65%	0.2965	29.89%	0.2989
0.02	29.65%	0.2965	28.74%	0.2874
0.05	24.12%	0.2412	29.89%	0.2989
0.1	24.12%	0.2412	26.44%	0.2644
0.25	25.13%	0.2513	27.59%	0.2759
0.5	28.14%	0.2814	27.59%	0.2759
1	30.65%	0.3065	28.74%	0.2874
1.25	30.65%	0.3065	28.74%	0.2874

Figure 23: Performance of RBF kernel function on breast cancer dataset with varying exponents

Table 9: Results of RBF Kernel on Optdigits Dataset

Gamma	CV Percent Error	CV Mean Absolute Error	Testing Percent Error	Testing Mean Absolute Error
0.01	3.31%	0.1602	3.31%	0.1602
0.02	2.47%	0.1602	2.48%	0.1602
0.05	1.73%	0.1601	1.54%	0.1601
0.1	1.25%	0.1601	1.12%	0.1601
0.25	0.92%	0.1601	0.83%	0.16
0.5	1.02%	0.1601	0.77%	0.1601
1	1.55%	0.1602	1.36%	0.1602
1.25	2.27%	0.1603	1.77%	0.1603

Figure 24: Performance of RBF kernel function on optdigist dataset with varying exponents

The gamma parameter defines how much influence a single training example has in defining similar instances [2]. Lower values mean each example has higher influence. Both datasets perform poorly for the default value of gamme but they have different optimal gammas showing that an example in optdigits has more influence in defining similar attributes than the breast cancer dataset since it has a lower optimal gamma. This is intuitive, as an example of a handwritten digit is a good predictor of future examples, while one cancer patient may not be the best predictor of future cancer patients.

The results here follow the same trend as before as both datasets peak at a certain gamma value before falling off. However overall, the RBF kernel function does better at separating the data than the polynomial kernel function.

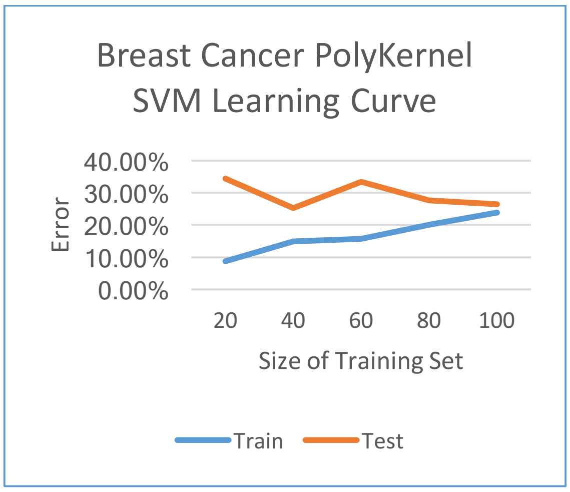

Figure 25: Learning curve of polynomial kernel on breast cancer dataset

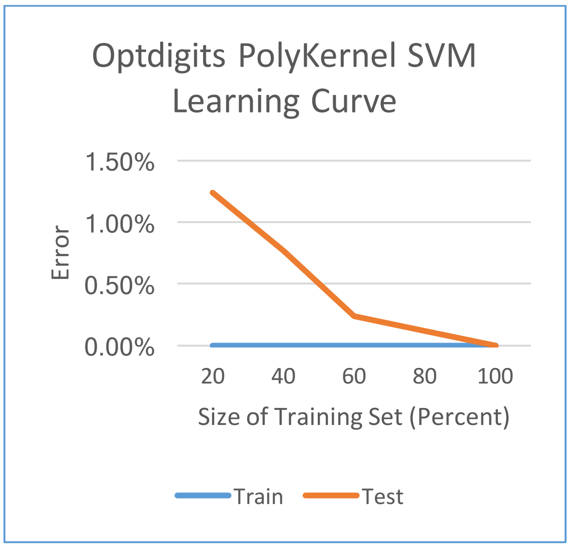

Figure 26: Learning curve for Polynomial Kernel on Optdigits

C. *Analysis*

The learning curves of SVMs is comparable to the boosting learning curve, though the variance is slightly lower for the breast cancer dataset. The optdigits learning curve once again overfits the model but as more training samples are added, the model does generalize better.

The breast cancer dataset's learning curve shows the ineffectiveness of SVMs on the dataset as the error increases with respect to the number of training examples given.

VI. Neural Networks

A. *Introduction*

In this section, the multilayer perception classifier was applied to both datasets. The input and output layer perceptron are connected to the input and output vectors. Weka's MultilayerPerceptron function is used to generate the models and the learning rate and number of epochs or training time were tuned to find an accurate model.

B. *Results*

Learning rate is known as the rate of exploration vs exploitation. Meaning that a higher learning rate will cause our neural net to explore outside of its biases more. A lower learning rate will cause a neural net to stick to its biases and explore less. This is an important hyperparameter as it indicates how much influence training instances have on the neural net. Depending on the learning rate, a neural net may be quick to change its model given a particular example.

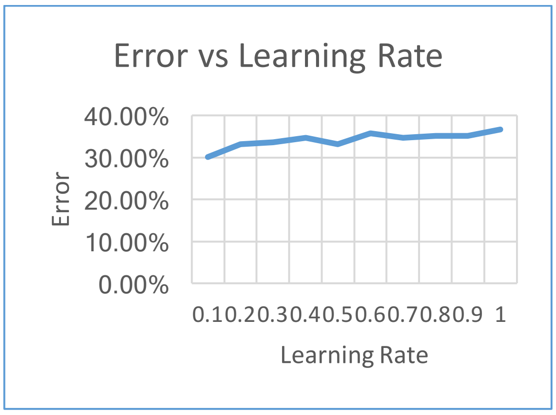

Figure 27: Breast cancer - Error with respect to learning rate of a neural net applied to dataset

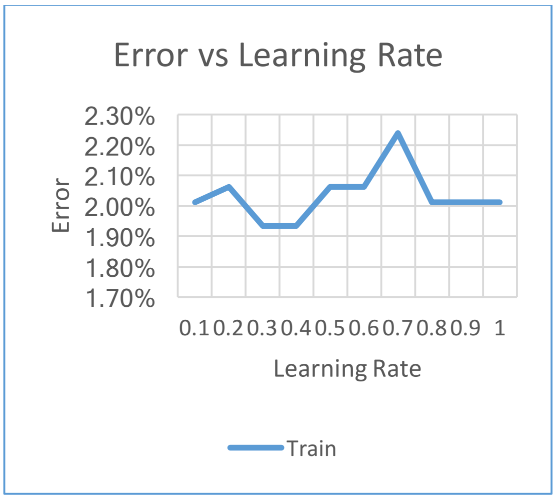

Figure 28: Optdigits - Error with respect to learning rate of a neural net applied to dataset

Figures 24 and 25 show the error when neural nets of various learning rates and a constant epoch of 500 are applied to each dataset. The breast cancer dataset requires that the neural net hold fast to its biases and not allow conflicting data to change its model easily. This is a good sign as any model trying to identify cancer should not be quick to change.

The optdigits results are more varied but the optimal learning rate is around 0.3 or 0.4. This is still relatively low but does allow for some change as some characters do appear similar when written down.

The number of epochs or training time, is the number of times the dataset is sent forward and backward through the neural net. If the number of epochs is too small, the neural net's model will underfit the training data. If it is too high, the model will overfit the training data. Figures show how the error relates to the number of epochs given a constant learning rate.

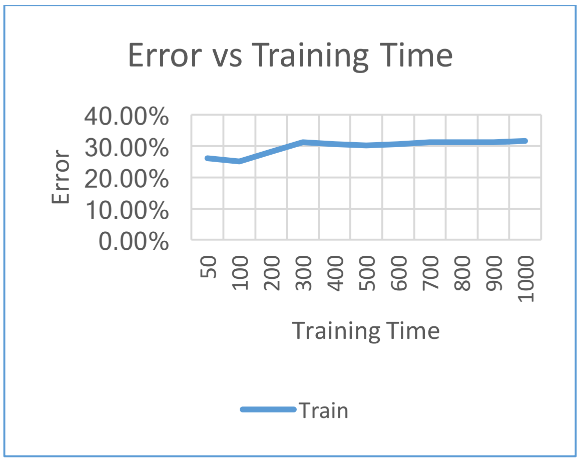

Figure 29: Breast cancer - Error rate with respect to number of epoches

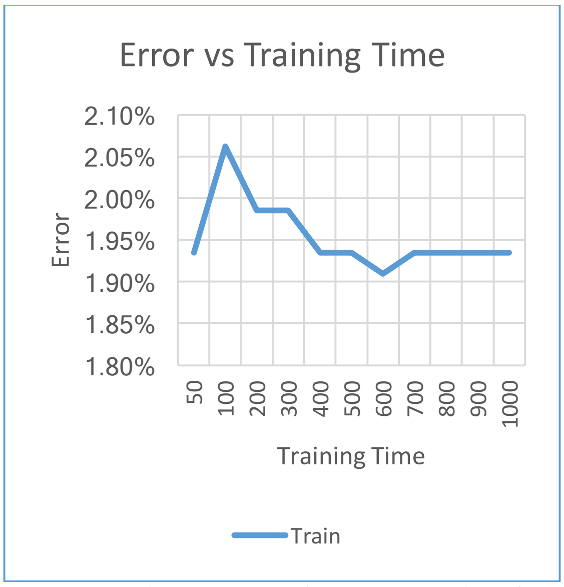

Figure 30: Optdigits - Error with respect to number of epochs

The breast cancer dataset also requires a fewer number of epoches or it loses performance and easily overfits to the training data that is given according to the cross validation data shown. Optdigits' results are significantly more varied but hits the peak training time at 600.

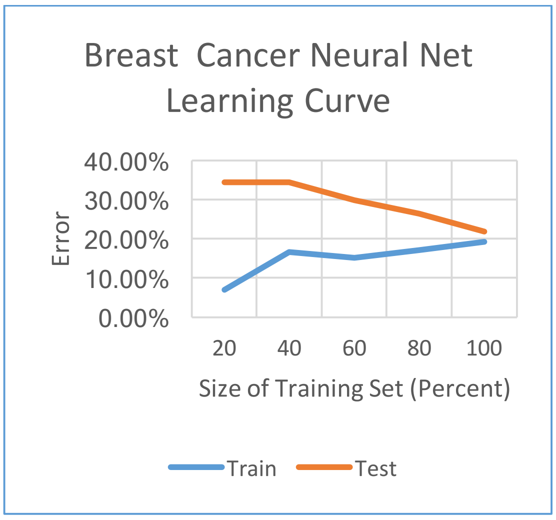

Figure 31: Learning curve of neural net on breast cancer dataset

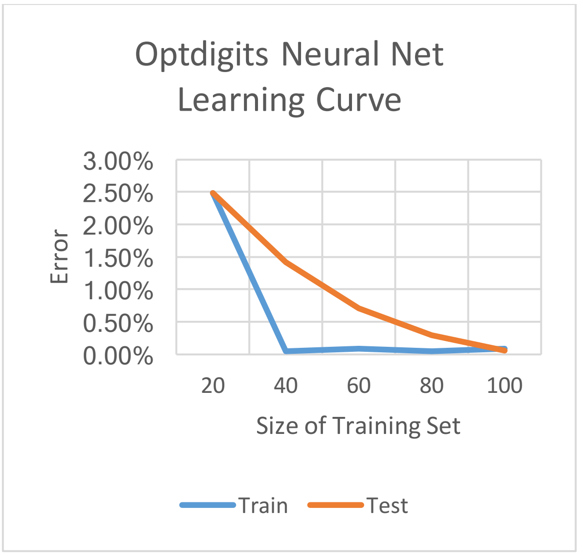

Figure 32: Learning curve of neural net on optdigit dataset

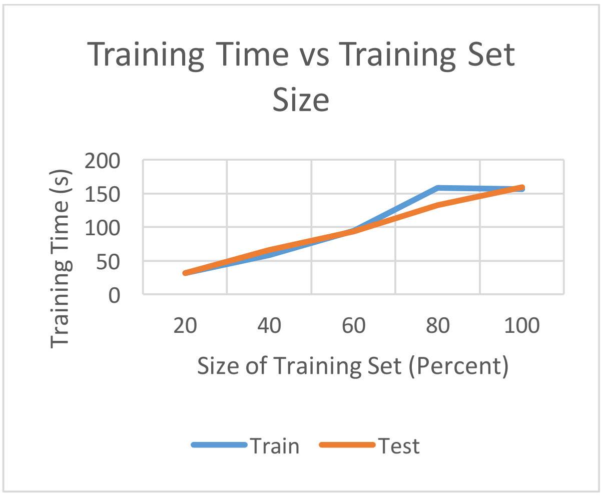

Figure 33: Training time with respect to training size of optdigits

C. *Analysis*

Figure 30 shows us the computational power required to run a neural net. As the size of the training set increases, the time required to train the neural net increases dramatically. Optdigits is a dataset consisting of little over 5,000 instances, but larger datasets will see much longer training time.

The learning curves of the neural net have high variance for both datasets (and contrary to the figure the error never hits 0). An interesting result is that the breast cancer learning curve actually increases as the number of training examples increase. This may be due to overfitting but it may also be due to new biases introduced by new examples causing the neural net to misclassify certain examples.

References

[1] Crowley, J. L. (n.d.). Kernel Functions and Support Vector Machines | http://www-prima.imag.fr/jlc/Courses/2015/ENSI2.SIRR/ENSI2.SIRR.S5.pdf

[2] https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html

[3] E. Alpaydin, C. Kaynak (1998). UCI Machine Learning Repository [https://archive.ics.uci.edu/ml/datasets/optical+recognition+of+handwritten+digits]. Bogazici University, 80815 Istanbul Turkey

[4] Matjaz Zwitter & Milan Soklic (1988). UCI Machine Learning Repository [https://archive.ics.uci.edu/ml/datasets/breast+cancer]. University Medical Center. Ljubljana, Yugoslavia

[5] “Intro to Machine Learning Course | Udacity.” Course | Udacity.Udacity, n.d. Web 09 Feb