CS-7641 - Project 1: Supervised Learning

Stephen Shepherd | sshepherd35@gatech.edu | 02/10/2022

Overview of the Problems and Datasets

In this paper I will apply various supervised learning approaches to two classification problems. The first problem is a binary classification problem of learning to identify individuals that have been diagnosed with heart disease and delineate them from those that have not. The second problem is a multi-class problem of learning to correctly identify the handwritten digits 0-9.

The Heart[1] Disease classification problem is interesting for machine learning applications because the learner is presented with a number of general telephone-surveyed general demographic and behavioral attributes that may be loosely correlated with the outcome but none of which is sufficient to accurately predict the outcome on it's own. It also presents a number of data quality and noise challenges: the individual may misrepresent or decline reporting some attributes, and in some cases may actually have heart disease that is currently undiagnosed or misdiagnosed. The dataset is relatively large and contains attributes that are a mix of continuous and categorical in nature.

Unlike the heart disease problem, the Digits[2] classification problem is multi-class (10) and the features represent pixel intensity of images of handwritten digits. Accordingly, the information in this dataset is less likely to be misrepresented or missing, with noise and variance more likely attributed to image resolution and different handwriting styles. The dataset is smaller and only contains continuous attributes. This provides a nice contrast to the heart disease data. Even though the data is arguably cleaner and more direct, it presents the learner with the challenge of differentiating between multiple classes and generalizing across handwriting styles.

Preprocessing, Validation and Libraries

Each dataset was split randomly into a 80% training set and a 20% validation set. The same pre-processing steps were taken for each dataset with the exception of under-sampling the negative class of the Heart disease training set to achieve an even mixture (the validation set remained imalanced), resulting in a subset of 48k training observations down from the original 202k. This was done to prevent the learners from simply predicting the 90% majority class for every inference. Undersampling proved to be a safe strategy as the Heart data showed low variance on the learning curves, indicating there was not a need for more training samples. Otherwise, any categorical variables were one-hot encoded and all

[1] https://www.kaggle.com/alexteboul/heart-disease-health-indicators-dataset, 01/23/2022
[2] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_digits.html, 01/23/2022

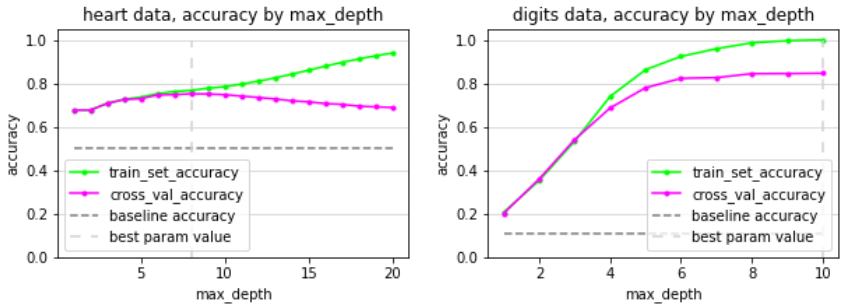
variables were standard-scaled; scaling and encoding were fit only to the training set but used to transform both the training and the validation set. Any cross-validation was performed within the training set and used five folds. Scikit-Learn version 1.0.2 was used to implement each algorithm[3].

Summary of Results From Tuned Learners

Dataset	Classes	Train Set Highest Class Prevalence	Test Set Highest Class Prevalence	Train Set Records	Test Set Records	Learner	Train Set Accuracy	Cross Validation Accuracy	Test Set Accuracy	Test Set Recall	Train Seconds	Predict Seconds
Heart Disease	2	50%	91%	38,158	50,736	Decision Tree	77%	75%	72%	81%	0.057	0.007
						Neural Network	77%	77%	73%	84%	0.739	0.006
						Boosted Trees	77%	77%	75%	80%	6.223	0.957
						Support Vector Machine	77%	77%	74%	82%	134.855	35.198
						K-Nearest Neighbors	76%	76%	72%	81%	0.007	74.794
Digits	10	11%	12%	1,437	360	Decision Tree	100%	90%	91%		0.065	0.010
						Neural Network	100%	97%	97%		0.141	0.001
						Boosted Trees	100%	96%	94%	N/A	1.312	0.024
						Support Vector Machine	100%	99%	99%		0.045	0.010
						K-Nearest Neighbors	100%	98%	97%		0.001	0.041

Decision Trees

I applied a Decision Tree learner to both problems as a starting point. This learner used the CART algorithm with Entropy used as the splitting criterion. Examining training set and cross-validation accuracy by tree depth reveals a tendency to overfit starting at depth 10 for the heart classification and depth 5 for the digits classification (note that an unpruned tree grew to a depth of 41 for the large heart dataset and only 10 for the smaller digits dataset). Interestingly, cross-validation accuracy diminished at higher depths for the heart classification but remained high for the digits classification. Post-pruning using Minimal Cost-Complexity Pruning yielded similar cross-validation accuracy on the heart disease problem with the complexity parameter α set to a small value (0.0021), but did not improve cross-validation accuracy on the digits problem: any post-pruning appears to weaken the learner's inductive power for the digits dataset.

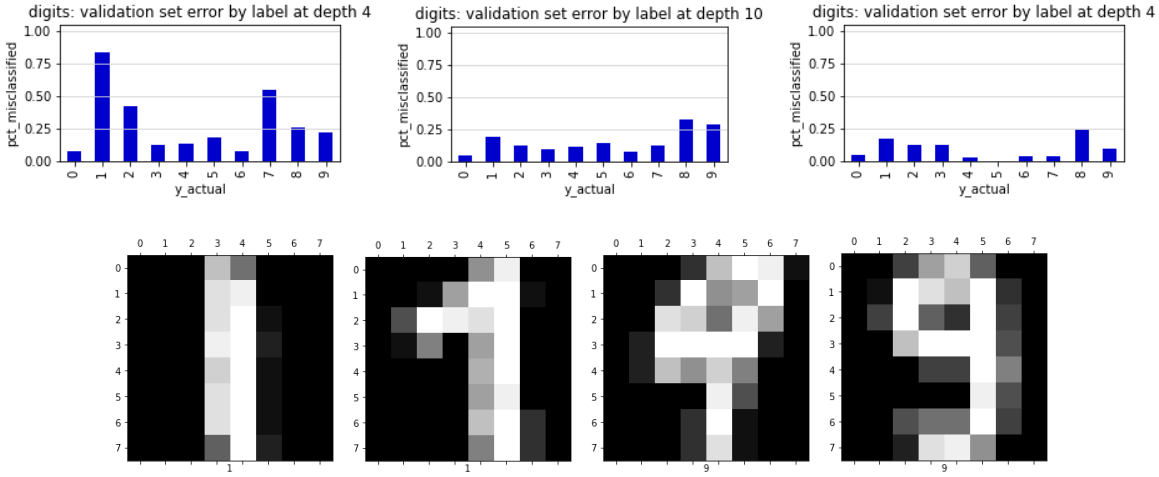


Why does an overfit tree yield the best cross-validation accuracy (~84%) for the digits classification, but poorer cross-validation accuracy for the heart disease dataset? It is related to the digits

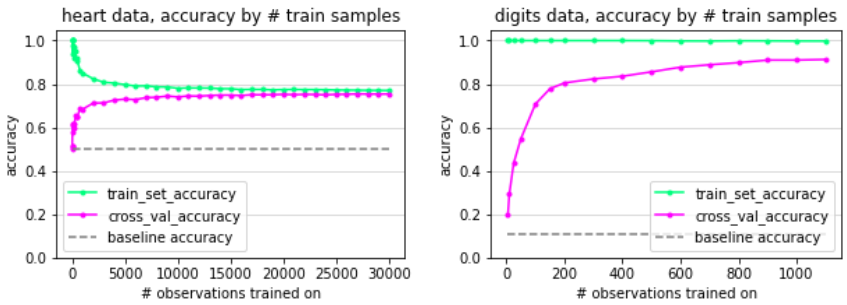
[3] https://scikit-learn.org/stable/, 2022-02-07

problem being a more complex, multi-class classification problem. The digits learner is able to generalize well for some digits, but overfits for others like 8, 9, and 1. We see a stable generalization of some digits and an unstable generalization of others as the tree grows. Comparing accuracy by class for different depths, it is clear that larger depth helps accuracy for most digits, but error for digits 8 and 9 increases. So, the simpler digits help stabilize cross validation accuracy as depth grows but a few more complex digits keep it from converging up towards the training set accuracy.

Single Learner *One vs. One Trained Learners*



Given this information, I tried reducing this single multi-class problem into multiple binary classification problems by using a One vs. One training approach and this was able to increase the cross validation accuracy to 90% by allowing separate trees to focus on recognizing each digit vs. the others. Additionally, best performance was seen at a lower depth of 4 vs. 10. The digits 1 and 8 still posed a challenge – I'm hoping that Boosted Trees can reduce this.



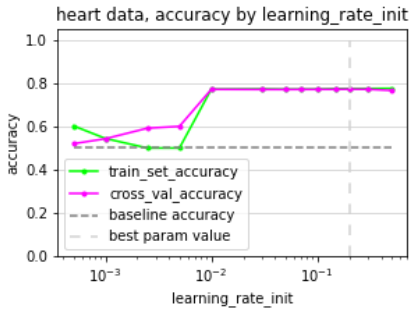
Training curves using the best performing max depth parameters for cross-validation accuracy (8 for the heart learner and 4 for the digits learner) reveal low variance for the heart learner and medium variance for the digits learner. The digits learner shows low bias with near perfect accuracy on the training set. Given the medium variance and the difficulty it encountered in classifying some digits, I

would argue that this learner needs more training observations related to nuanced representations of complex digits (a 1 with a flag, for example) to be able to reduce the variance and improve accuracy.

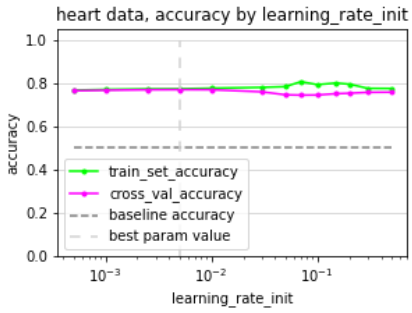
The heart disease dataset, being survey based, is bound to contain some noise and data quality issues and the tree may be fitting noise rather than a true function for the heart disease outcome. There also may be negative samples that actually have heart disease that is not diagnosed yet. Accordingly, limiting the tree to a max_depth of 8, the training and cross-validation accuracy converge to a decent but imperfect accuracy of ~75%. When tested on the original imbalanced data (and removing samples that were in the training set), the accuracy decreases to ~72%. This is because heart disease labels are less prevalent (~10% vs. ~50%) in the imbalanced holdout data and the learner produces many false positives, although recall within the positive class is high (81%). I will discuss this later on.

Neural Networks

I then applied a traditional feed-forward Neural Network learner that uses Backpropagation. Neural Networks inherently provide support for multi-class classification with the flexible output layer. I began with a Sigmoid activation function for the hidden units which proved effective on the Digits dataset but yielded nearly identical train and cross validation accuracies of ~77% on the Heart data regardless of the number of hidden units, hidden layers, the learning rate, regularization, tolerance or momentum used. Below is an analysis of a 30x30x30 network alongside learning rate. 77% accuracy beat the Decision Tree's 75%, but I expected it to vary more as a function of network structure and suspected underfitting.



Sigmoid/Logistic, 30x30x30

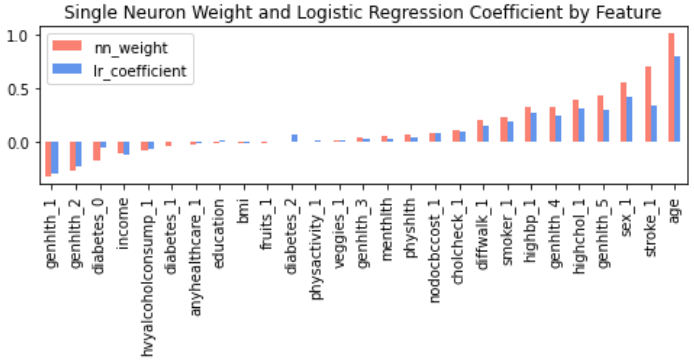


ReLU, 30x30x30

Fearing the stochastic gradient descent was getting stuck in local optima, I researched and found that Sigmoid activation functions can suffer from the "Vanishing Gradient Problem[4]" where the Sigmoid function's small derivative causes it to yield similar output for large changes in input, posing a challenge for Gradient Descent in deeper networks where many small weighted values are multiplied together and diminish the variability in the error that the gradient descent attempts to detect. Given that a network consisting of a single sigmoid unit was able to produce 77% accuracy for the Heart data in just 14

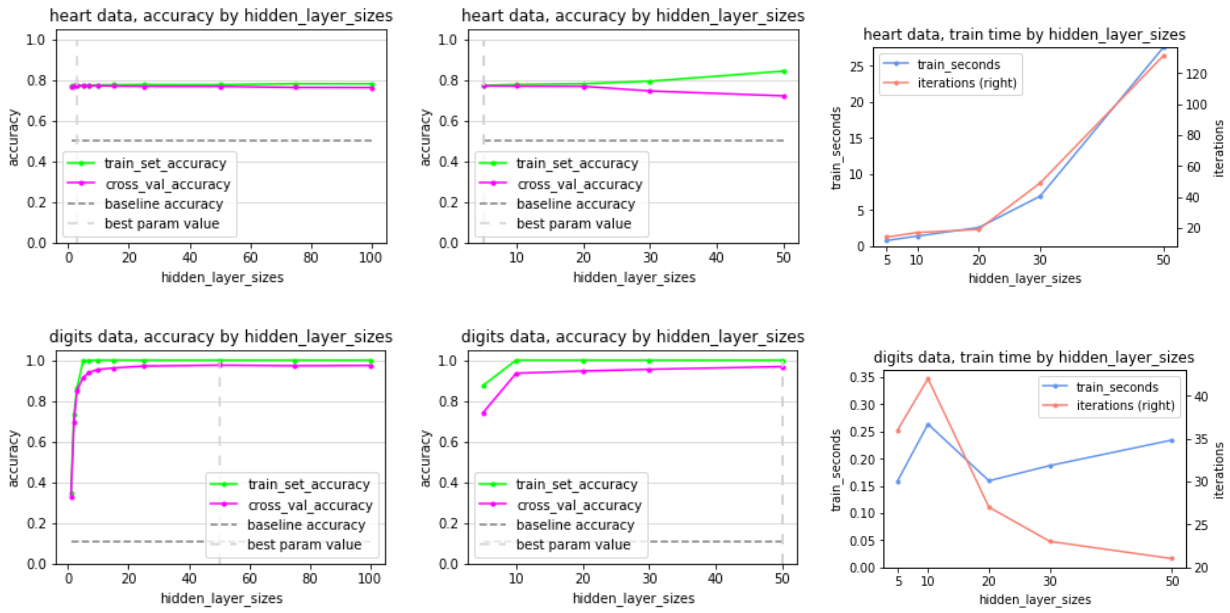
[4] Hochreiter, Sepp. 1998. "The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions." *International Journal of Uncertainty Fuzziness and Knowledge-Based Systems*: 6(2):107-116. https://www.researchgate.net/publication/220355039.

stochastic epochs, the learner seems to identify the key patterns in the simpler Heart data early on, leaving little further fluctuation for SGD to utilize in more complex networks. The single Sigmoid unit was a powerful learner for the Heart problem: the resulting input weights (with a positive output weight) are almost identical to the coefficients produced by a Logistic Regression Learner, which also uses a Sigmoid transformation and performs similarly.



I then tried the ReLU activation function to investigate whether that would diminish underfitting on the Heart problem. Indeed, it yielded more variance with deeper network structures, but ironically it was not able to beat the Sigmoid's cross validation accuracy of 77% due to *overfitting* with increasing network complexity. This suggests that the Heart data itself is simply not offering regions of the feature space with low entropy, which I'll explore in the next section (Boosting).

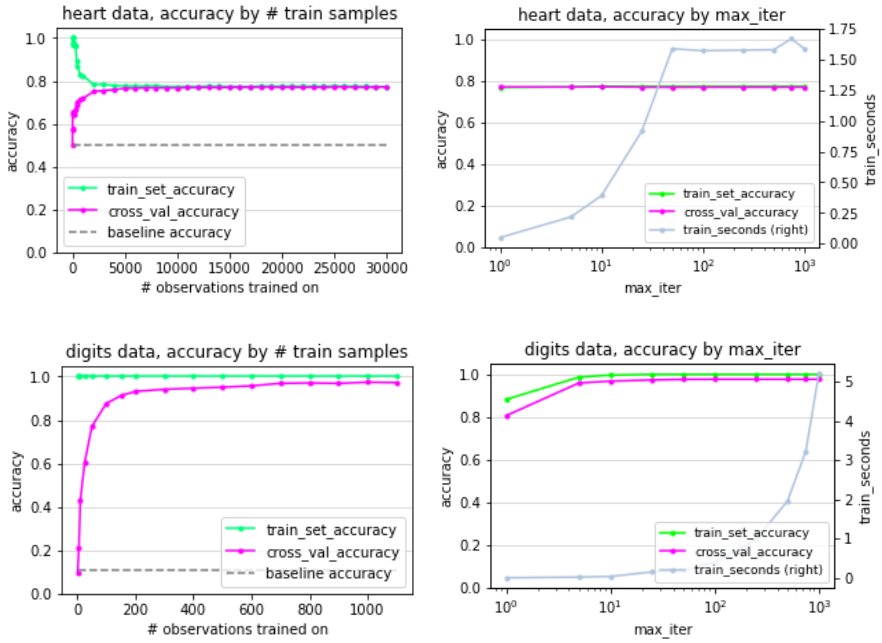
1 hidden layer *3 hidden layers*



The Neural Network performed extremely well on the Digits classification with ~97% cross-validation accuracy (compared to the Decision Tree's 90%) with just one layer and 25-50 units; it appears Neural Networks' capability of capturing many non-linearities and feature interactions is a good

fit for classifying the more complex digits that the Decision Tree struggled with. The cross-validation accuracy converges up to the perfect training accuracy for the Digits dataset with a slightly complex network, whereas overfitting occurs on the binary class Heart dataset alongside an exploding increase in training times and epochs for more complex structure. Interestingly, training times and epochs showed a more random trend alongside network complexity suggesting a more complex network identifies useful trends with less effort for the smaller, multi-class Digits dataset.

On the learning curves, we see low variance converge early on, imperfect accuracy and plateauing training times for the Heart data alongside training samples and # of epochs. For the Digits data, we see diminishing variance, near-perfect accuracy and increasing training times alongside both training samples and # of epochs.

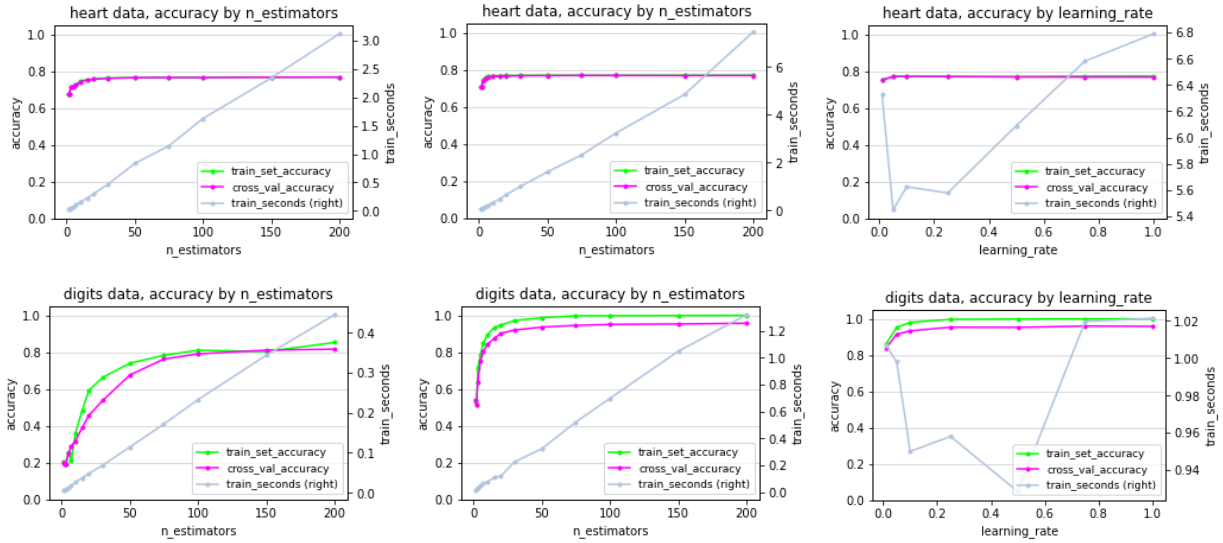


Decision Trees with Boosting

I explored boosted decision trees next, using the Ada boosting process with scikit-learn's SAMME implementation of the algorithm starting with a learning rate of ½ applied to $\boldsymbol{\alpha}$ per the class lectures. This algorithm supports multi-class classification without having to rely on a One vs. Rest or One vs. One training approach that I explored in the Decision Tree section[5]. I tried base weak learner trees with maximum depths of 1, 2 and 3. Results for the Heart data were similar regardless of this depth, but depth of 1 seemend to underfit the Digits data. This is intuitive as the Digits problem is multi-class and requires detection of deeper nested patterns, so depth helps. Recall from the Decision Trees section

[5] Hastie, Trevor, and Rosset, Saharon, and Zhu, Ji, and Zou, Hui. 2006. "Multi-class AdaBoost." *Statistics and its Interface:* 2(3). https://www.researchgate.net/publication/228947999.

that a stump does produce train accuracy > .5 on the Heart classification, but train accuracy of > .5 is only seen at depths >= 3 on the Digits classification (although the Digits classification is a multi-class problem). A grid search process across various base learner depths, learning rates and number of boosted estimators revealed that base learners with depth of 3 produced optimal cross validation accuracies although the difference between 3 and 2 was only materially beneficial for the Digits classification (96% at 3 vs. 93% at 2), and doubled the training time for both problems. The optimal number of boosted trees was 200 for both problems, but learning rates did vary with 0.1 for Heart and 0.75 for Digits. With a lower learning rate, the noisy Heart data required more subtle changes to the emphasis placed on certain training examples as it iterated. This is in contrast to the cleaner Digits data where a higher learning rate speedily identified and targeted the more difficult examples.

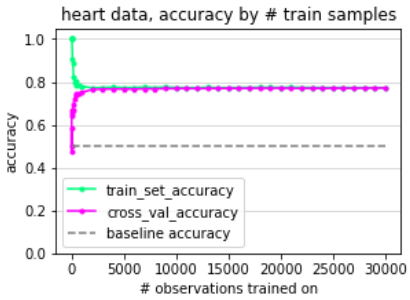


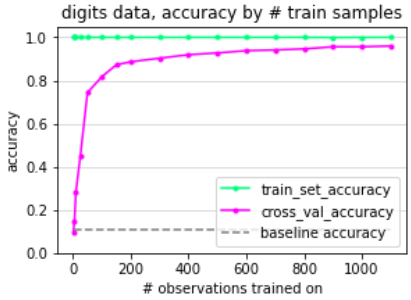
base estimator: tree max depth 1 *base estimator: tree max depth 3*

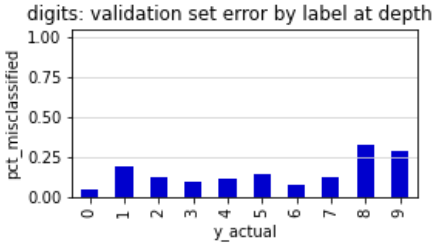
Given that a deep single tree was able to overfit the Heart dataset, I found it interesting that deep Boosting (and Neural Networks) only improved the Heart cross validation accuracy over the Decision Tree's by 2 percentage points (77% vs. 75%). A deeper look at the Heart data reveals that, even for the most potent combinations of the most powerful predictors, the number of samples with low entropy within those combinations is small and, within the low entropy subset, the entropy varies between Train and Validation sets. In fact, there are 141 combinations of feature values in the training set that map to both classes. So the data does not provide large subsets of the current feature space with low entropy on which to delineate the classes that generalize well to new data, and sometimes there are collisions of multiple classes represented by the exact same feature values. This is intuitive as the Heart dataset, being based on phone survey responses and containing the possibility of undiagnosed heart disease, is bound to be noisy.

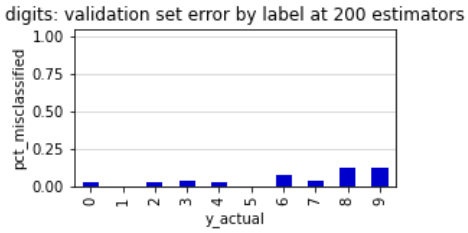
Feature			Train Set		Test Set	
Stroke_1	Sex_1	GenHlth_5	% of Observations	% Has Heart Disease	% of Observations	% Has Heart Disease
No	No	No	38%	56%	50%	12%
		Yes	4%	87%	3%	40%
	Yes	No	40%	74%	37%	24%
		Yes	4%	87%	2%	46%
Yes	No	No	5%	82%	4%	28%
		Yes	2%	92%	1%	57%
	Yes	No	6%	88%	3%	45%
		Yes	1%	95%	1%	67%

This helps explain the early, low variance convergence at 77% accuracy with the Heart dataset in the Learning Curve for both Boosted Trees and the Neural Network: the classes are mixed in most of the generalizable subsets of the feature space. Given this information, I would explore adding more features to the Heart dataset, or perform feature selection, and conclude that training on a sample of 5k to 10k observations would be sufficient for the current feature space. By contrast, boosting for the Digits problem yields great cross validation accuracy that converges up towards the perfect training accuracy, with good generalization for all digits at large number of estimators. Boosting improved upon the decision tree by focusing the boosted trees on the harder sample distributions corresponding to the more complex digits that the single tree struggled with.









Decision Tree *Boosted Trees*

Support Vector Machines

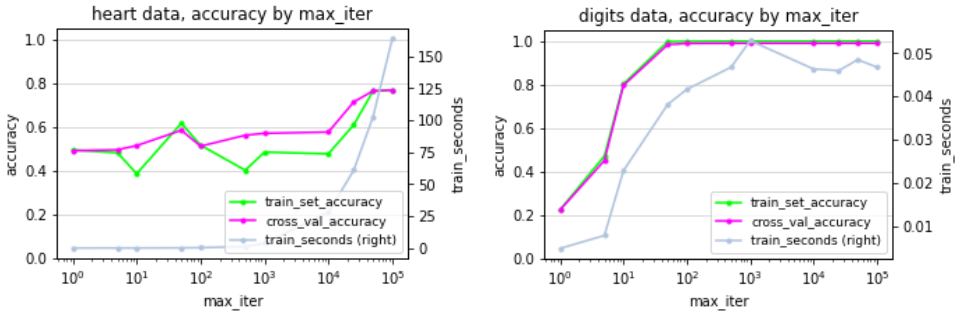
I then examined Support Vector Machines, starting by comparing results from different kernels in a grid search process alongside regularization and tolerance. Scikit-Learn's SVC implementation handles multi-class classification with a built-in One vs. One training approach[6]. Results were similar but Linear

[6] https://scikit-learn.org/stable/modules/svm.html#classification, 2022-02-05

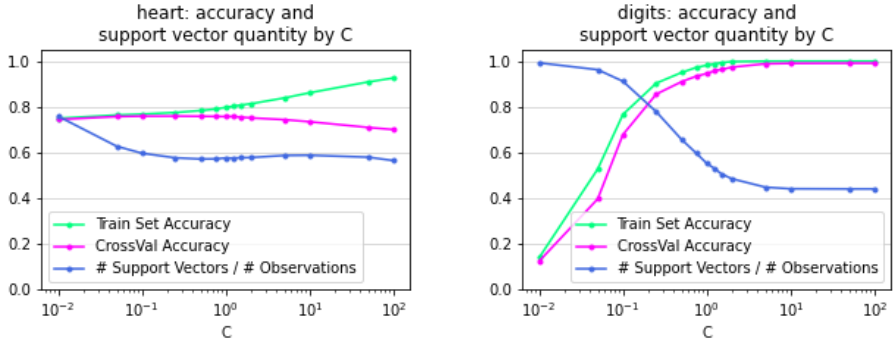
with more regularization (C=1) proved best for the Heart problem whereas Polynomial with less regularization (C=10) proved best for the Digits problem.

	Cross-Validation Accuracy by Kernel			
Dataset	Linear	Sigmoid	Polynomial	Radial Basis
Heart	76.5%	76.2%	74.8%	75.7%
Digits	98.3%	95.1%	99.0%	98.1%

Handwriting is naturally curvy and handwriting styles are diverse, so it's intuitive that a Polynomial kernel would produce a better decision boundary than a linear one. The Sigmoid function is also curvy, but the output approaches a constant as the input value becomes more extreme, thus restraining the flexibility of the curve. The Radial Basis function is able to draw flexible ovular shapes around observations. Based on this, it seems like a curvy function can fit the Digits data well if it is decently flexible, otherwise a linear function can almost attain the performance of the polynomial. The quadratic optimization took many more iterations to arrive at decent accuracy than it did for the Digits data. This points back to the theme of how noisy the Heart data is.



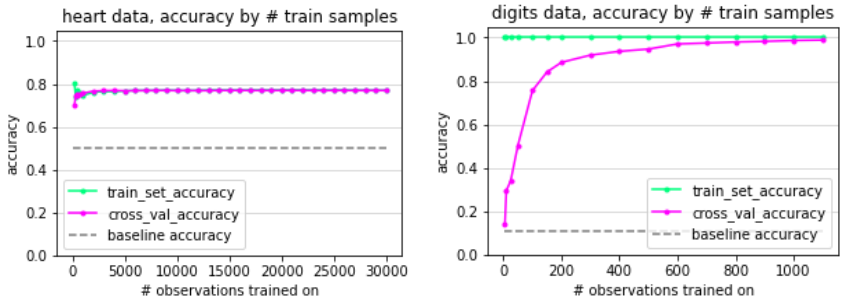
SVM was more sensitive to regularization strength when using a non-linear Kernel. Examining the radial basis kernel for Heart, it utilized a majority ~57% of the large Heart data samples as the support vectors, vs. a minority ~43% of the smaller Digits data samples. This difference is substantial considering the Heart data only has 2 classes to learn vs. the 10 that Digits presents.



The cleaner Digits dataset uses less support vectors and higher misclassification penalty (C), meaning a thin margin of separation defined by the points closer to the boundaries. Even though it has less classes,

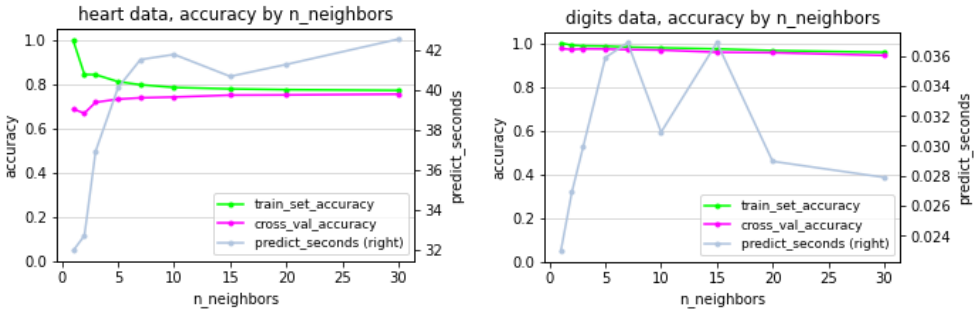
the Heart problem required more vectors and lower penalty, meaning a wider, more tolerant margin of separation relying on a majority of the samples as support vectors to fit the noisy data and produce small gains in accuracy. This difference is just another on the list of things that show how the noise in the Heart dataset makes things difficult for machine learning.

The learning curves here look familiar - quick convergence to an imperfect accuracy for the Heart data with < 20% of the total observations, and gradual convergence to a near perfect accuracy with the Digits data. SVM shows the lowest variance at the end of the learning curve of any of the learners for Digits. This is likely due to SVM's advantage of finding a hyperplane in the middle of a wide margin that allows it to generalize well.



K-Nearest Neighbors

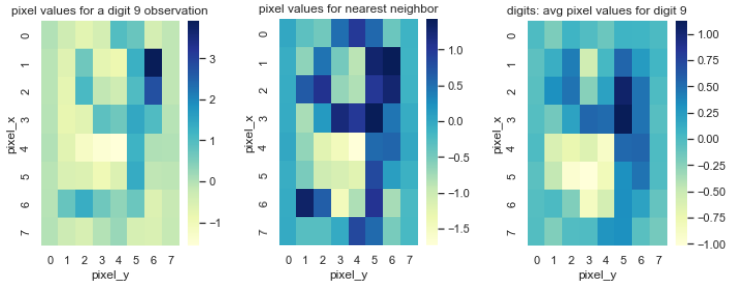
Finally I ran a simple K-Nearest Neighbors algorithm. It wasn't able to improve the accuracy for either problem and took up to 42 seconds to predict on the validation set for the Heart data, but yielded some interesting insights. Using weighted euclidean distance instead of uniform produced similar cross-validation accuracies, but produced more variance for Heart: the predictions were accordingly biased by weight towards a few close-by single instances in the training set.



Accuracy by # of neighbors, using uniform distance

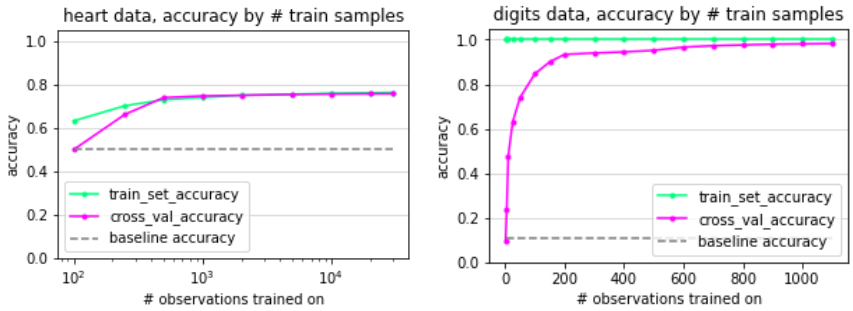
For the Digits data, a KNN learner provided cross-validation accuracy of 97% using just the single 1 nearest neighbor: accuracy decreased with # of neighbors used. This is quite interesting and is the opposite of the case for the Heart problem. Finding a nearest single neighbor, the model is able to hone in on the most similar example and reduce the confusion from pixels that don't generalize well for

that digit across other examples. This is demonstrated below: the single nearest neighbor appears very similar to the example observation across more pixels than the average for the digit 9. This single neighbor approach is explored in the SVM Tutorial reading where it's noted that it has a high VC dimension but can perform well empirically "provided no two points of opposite class lie right on top of each other."[7]



The Heart dataset, by contrast, contains more noise and overfits at smaller quantities of neighbors: the smaller the pool, the more likely that those instances are not generalizable examples. The within-class standard deviation for each Heart (standardized) feature is close to 1, whereas the same value for Digits is ~0.63, so there is more variation across the attributes within the same class in the Heart dataset and thus more confusion to overfit by using a small pool of neighbors.

A grid search process settled on 100 neighbors using uniform manhattan distance for the Heart classification and 1 neighbor using uniform manhattan distance for the Digits classification. The learning curves for these KNN models look similar to the ones for the previous approaches except that there is no overfitting on the Heart data to start due to starting at 100 random observations for the 100 neighbors. The single-nearest-neighbor Digits learner converges up to near-perfect accuracy once it's library of samples to draw neighbors from reaches ~200, while the 100-nearest-neighbor Heart learner converges to ~75% accuracy doing the best it can with the noisy data it's given. Although quick to "train," this algorithm takes 5x longer to retrieve predictions for the Digits classification compared to the other learners, and 11x longer for the Heart classification, due to the computation of pairwise distances.



[7] Burges, Christopher. 1998. "A Tutorial on Support Vector Machines for Pattern Recognition." *Data Mining and Knowledge Discovery*. 2, 121-167.

Results on the Imbalanced Heart Dataset

The learners perform worse when the models trained on the balanced Heart train set are applied to the validation set representing the real-world imbalance mixture (~10% disease). Although the accuracy is only a few percentage points lower (~73% vs. ~76%), the errors are distributed differently: false positives account for over 90% of the error whereas errors were split more evenly with the balanced dataset. With an influx of true negative samples, the quantity of false positives nearly doubles. I was satisfied to see the Recall (% of true positives that are predicted positive) remain high at ~80% – at least the learners were successful at identifying most of the positive class.

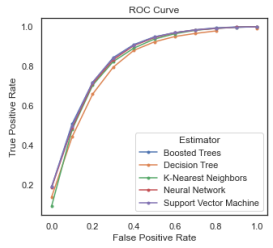
Train Set (balanced)

	Pred -	Pred +
True -	38%	13%
True +	10%	40%

Test Set (imbalanced)

	Pred -	Pred +
True -	68%	23%
True +	2%	8%

Confusion Matrices for the Boosted Trees Learner on Heart Data



Tuning the classification threshold to produce higher recall comes at the expense of more false positives, as the ROC curve shows. Some false positives may actually be instances where the person actually has heart disease that is not yet diagnosed. If we were able to obtain the true prevalence of undiagnosed heart disease for the population, we could adjust the classification probability threshold for the chosen learner to produce a false positive rate that matches that prevalence. This could be a final tuning step to address this significant issue in the dataset. Alternatively, we could focus training to optimize recall by weighting false negatives greater than false positives in the error calculation.

Conclusion

My favorite candidate for the Digits classification is the Support Vector Machine as it produced the highest validation set accuracy (99%) and was surprisingly inexpensive to train on this smaller dataset. It also showed the lowest variance at the end of the learning curve vs. the other algorithms, indicating that it had sufficient training observations. I was happy to see SVM's inherent guards against overfitting produce near perfect validation set accuracy. My favorite candidate for the Heart Disease classification is Boosted Trees as it was relatively efficient and yielded the highest accuracy (75%) on the imbalanced (eg. real-world) validation set. The Heart classification was a challenge: no algorithm was able to exceed 77% cross-validation accuracy even on the balanced set. The consistent high false positive rate across learners for the test set was concerning, but these predictions could be interpreted as a warning to individuals at risk of developing heart disease. I am hoping that feature selection or dimensionality reduction and other improvements can help the performance on the Heart classification.