Salary prediction and handwritten digit recognition with supervised learning

Peter Lucia

September 22, 2019

1 Abstract

In this work, I implement and analyze various supervised learning models for the task of salary prediction and handwritten digit recognition. Today, income inequality remains a growing concern, especially among some of the most disenfranchised populations. It is an interesting problem because with improved approaches we can more accurately forecast an individual's financial future and understand whether supervised learning approaches not originally considered by Rohavi et. al in [8] are more effective, paying special attention to the tradeoffs of each. It will also be interesting to see how the various supervised learning methods compare to one another for the task of handwritten digit recognition. Despite the ubiquity of technology, handwriting has persisted has a practical, personal way of recording and communicating information. [14] While neural network methods are increasingly effective for visual identification tasks such as character recognition, in this work, I also aim to understand whether other supervised learners offer similar accuracies but with shorter training times. [15] By designing each decision tree, neural network, K-Nearest Neighbor, Boosting, and SVM classifier to solve these problems, I aim to constrast and analyze each learning algorithm's individual characteristics and effectiveness.

2 Background

The salary prediction dataset comes from a census bureau database and consists of features (e.g. age, education, occupation, marital-status) and a label of whether or not the person's salary is greater than $50,000/year. [7] Comprised of the handwriting of 43 different people, the handwritten digit dataset is provided in normalized bitmap form, and consists of flattened 32x32 input matrices for every written digit where each element in the flattened array is an integer between 0 and 16.[6, 9] Additionally, all the supervised learning models are implemented and tuned using the scikit-learn machine learning framework in the Python programming language.[2, 3] During processing, the data is split between training and testing. Eighty percent of the data is reserved for training while the remaining twenty percent is reserved for testing. I apply one-hot encoding to some categorical variables in the dataset for salary prediction so that text-based groupings (e.g. married, divorced, single) are instead represented as an index in a lookup table. When a prediction is made, the model provides an index that can then easily be replaced with the original text-based grouping. It was not necessary to apply one-hot encoding to the the handwritten digit dataset as the entire dataframe is numeric. Additionally, a 3-fold cross validation splitting strategy is used for all learners.

3 Results

3.1 Decision Tree

As we traverse through the decision tree, we wish to reduce the impurity of the data by gaining more information at each split. With regards to the salary class dataset, we should incrementally arrive at the appropriate leaf node that contains the salary range for the one-hot encoded input vector. The difference between each parent and child attribute in our decision tree should then ideally provide a gain of information such that the entropy is reduced between the two. [1] The entropy and gini index

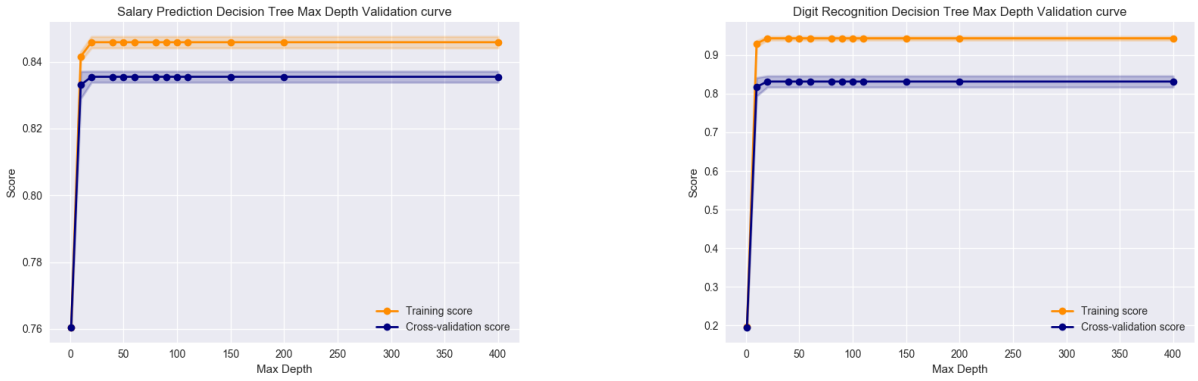
$$Entropy(S) = \sum_{i=1}^{c} -p_i log_2 p_i$$

$$Gini(S) = 1 - \sum_{j=1}^{k} p_j^2$$

measure the impurity of the data for an attribute S. [1, 10]

From these two equations it becomes clear that computing gini index is computationally less expensive than computing entropy because it does not contain any logarithmic functions. [12] Since there was no accuracy improvement with using entropy over the gini index, the gini index was selected to reduce the computational load.

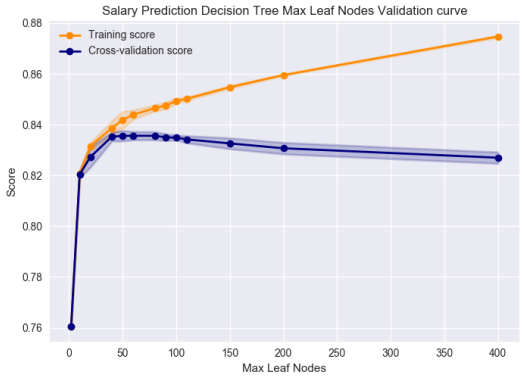
Initially after training and testing, it became apparent that the training accuracy for the decision tree was 100% while the test accuracy converged to less than 70%. This was a clear indication that the decision tree was overfitting the training data and not generalizing very well to data it had not seen before. Pruning is an effective way to reduce overfitting by improving its resilience to noise and any outliers that exist in the input space. There exists two methods: pre-pruning and post-pruning. Post-pruning is arguably a more effective method to avoid overfitting because you know the size and shape of the decision tree before starting the pruning. However, the post-pruning capabililty is not yet available in the scikit framework at the time the experiment was conducted. As a result, the decision tree is pre-pruned by being limited in the max number of leaf nodes and depth it can grow to during training. While it was not possible in this experiment, in future experiements it would be valuable to compare post-pruning techniques to pre-pruning in order to further reduce overfitting. Figures 1 and 2 show the validation curves used to determine the optimal values for maximum leaf nodes and max depth.



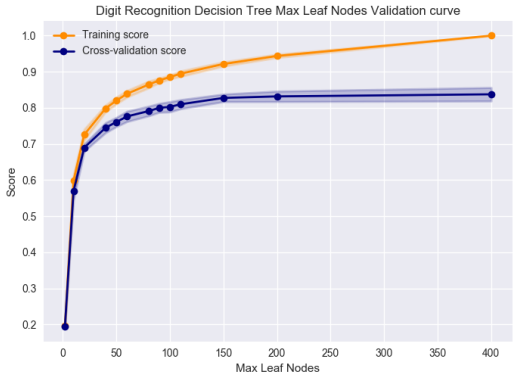
(a) Salary Prediction (b) Digit Recognition

Figure 1: Decision Tree Pruning with Max Depth Validation Curves

As is evident from the pre-pruning results against the salary dataset in Figures 1a and 2a, the cross validation and training scores stop increasing when the max depth is between 10 and 100 and the cross validation score begins to drop when the max leaf nodes passes 75. Therefore, I pre-pruned the salary prediction decision tree by setting the max depth to 50 and max leaf nodes to 75.



(a) Salary Prediction



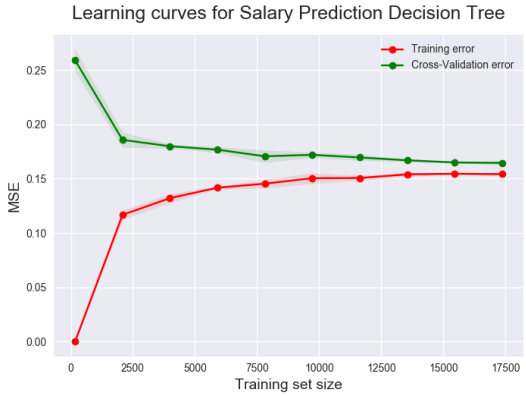
(b) Digit Recocognition

Figure 2: Decision Tree Pruning with Max Leaf Nodes

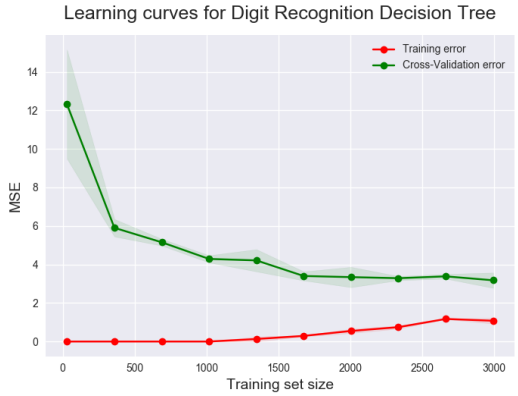
Figures 1b and 2b show the scoring as the pre-pruning parameters are varied for the decision tree against the digit dataset. The accuracy plateaus at 0.8 when the max depth is between 0 and 50, and sees little variation thereafter; so 25 is selected as the max depth. While varying the max leaf nodes, the cross-validation score converges to just over 0.8 when the max leaf nodes parameter passes through 200; therefore, 200 was selected as the max number of leaf nodes.

Because the decision tree is acting as a binary classifier to determine whether the predicted salary of the individual is greater than \$50,000 per year or not, the two classes (1/0) are given equal weight. Additionally, there is no limit to the number of features to be considered at each split and better accuracy was obtained when the best split is always selected instead of the best *random* split.

A learning curve is useful to understand the bias and variance of a supervised learning model. [11] If we look at the gap between the training and cross validation error in Figure 3a, we can see that the gap is small between the two curves and the overall MSE is low, indicating that the decision tree has low bias and low variance. Had the training and validation errors converged and then began to diverge such that the training error continued to decrease but the validation error started increasing, it would be an indication that the decision tree had begun overfitting the training data. Also, if the validation error was much greater than training and was still converging, that would be an indication that more training is needed so the model can generalize better to the validation set.



(a) Salary Prediction



(b) Digit Recognition

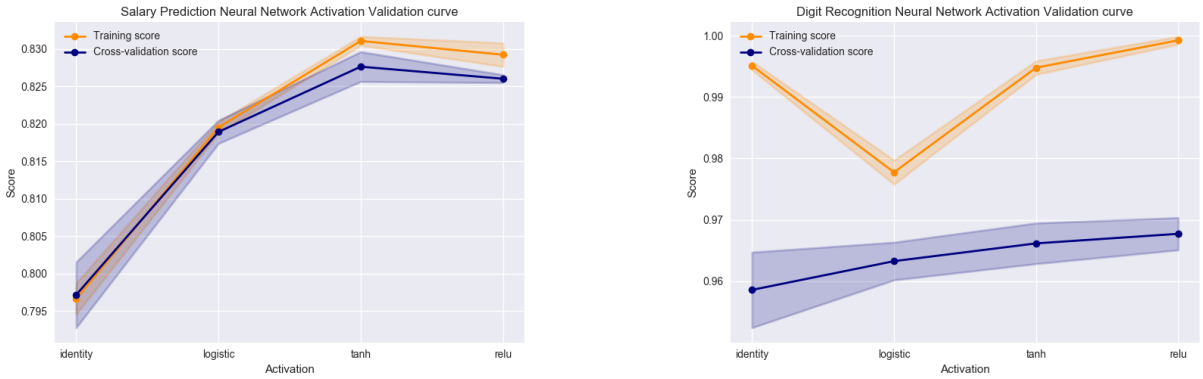
Figure 3: Decision Tree Learning Curves

Additionally, observe the learning curves of the decision tree tuned to both datasets in Figure 3. The decision tree tuned to the digit dataset shows more bias than the tree tuned to the salary dataset because both training and testing MSE curves show larger error. It also has a higher variance than the tree tuned to the salary dataset because both curves stop converging and leave a wider gap between them before training completes.

3.2 Neural Network

Inspired by biological systems, multilayer artificial neural networks trained with the backpropogation algorithm can express nonlinear decision surfaces in a two dimensional space with performance comparable to decision trees. [1]. In this experiment, the neural network's accuracy was within the top three performers against both datasets. The high score in Figure 18b against the digit dataset is to be expected because neural networks have demonstrated success in the literature with vision classification tasks much like the handwriting digit classification task of this experiment. Additionally, neural networks are more robust despite the presence of noise commonly seen in images and the perceptron training rule guarantees the ability to correctly classify examples that are linearly separable when a small learning rate is used. [1]

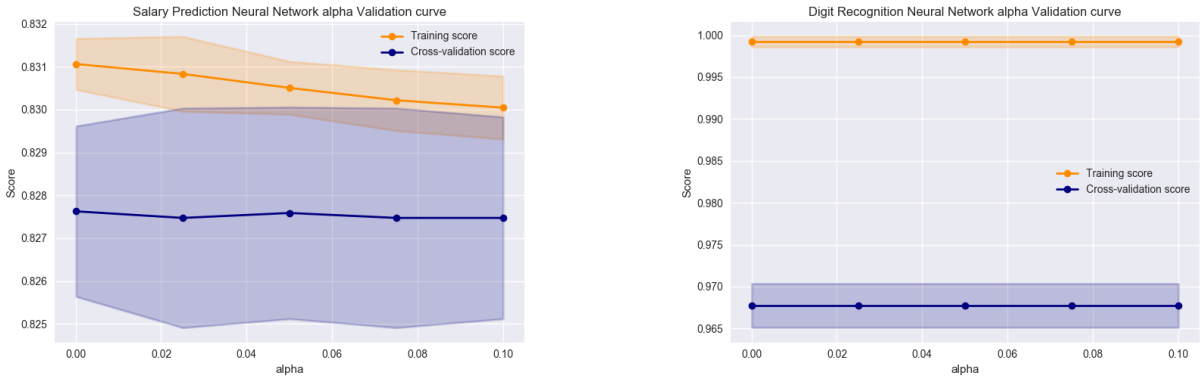
Figures 4a and 4b show the scores for various activation functions. The tanh function showed the best performance against the salary dataset and the relu function showed the best performance against the digit recognition dataset.



(a) Salary Prediction (b) Digit Recognition

Figure 4: Neural Network Activation Function Comparison

Figures 5a and 5b show the training and cross validation curves as alpha varies between 0.001 and 0.1. Because the scores for all values of alpha show so little movement within the rolling standard deviation in each test, the default value of 0.001 is chosen for both datasets.

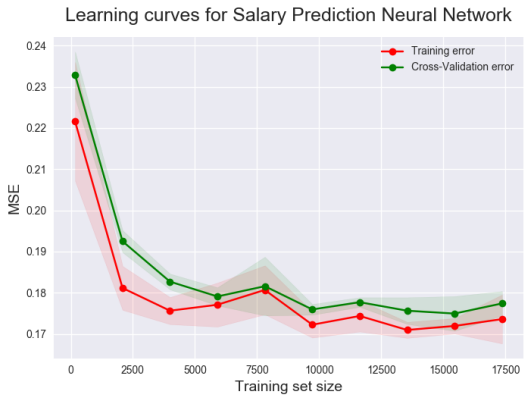


(a) Salary Prediction (b) Digit Recognition

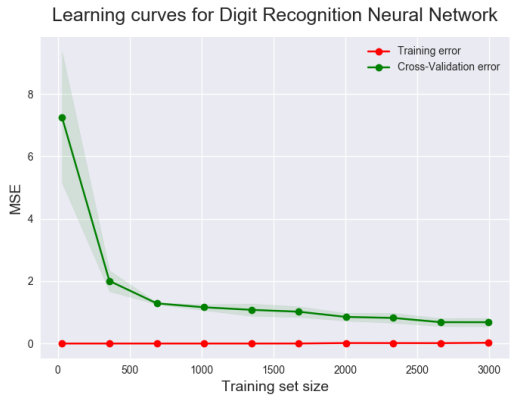
Figure 5: Neural Network Alpha Validation Curves

Figure 6 shows the learning curves of the neural network against both datsets. The neural network had a lower MSE against the salary dataset, and for that dataset the gap between the training and cross-validation MSE is also quite narrow, indicating low variance. Against the digit dataset, it had a higher variance because the gap between the two curves is wider. It also had higher bias in this case because the MSE is higher. One explanation

for the higher bias and higher variance of the neural network against the digit dataset is that is had fewer training examples from which to learn.



(a) Salary Prediction



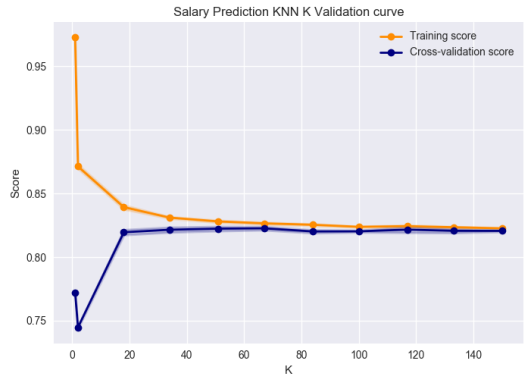
(b) Digit Recognition

Figure 6: Neural Network Learning Curves

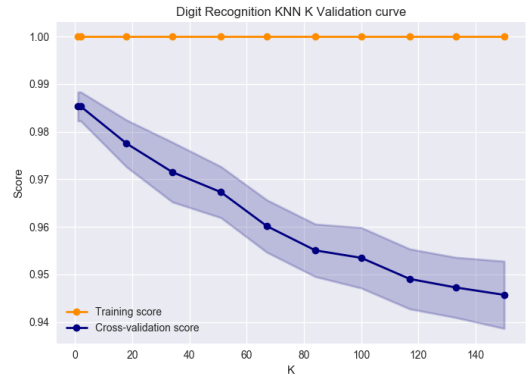
3.3 K-Nearest Neighbor

With regards to accuracy, the KNN model was the top performer against both datasets. This can be attributed to a variety of reasons. One of the advantages of instance-based approaches like K-Nearest Neighbor (KNN) over linear based methods like SVM is that we are more easily able to represent a complicated hypothesis space that is different for each region of data. Using the distance function, we can assign greater weights to points that are closer to the query point when nearby neighbors are sparse. KNN is effective when provided a sufficiently large training set and is robust to noisy training data. [1] As is shown in Figure 17, the KNN classifier had the second lowest training time but the best accuracy against both datasets. While neural networks have been proven to be effective for visual classification tasks, these results from Figures 17 and 18 suggest that when properly tuned, the KNN classifier can be more effective and require less than a tenth of the training time.

Figure 7 shows the validation curve against changing values of K. The highest accuracies for the salary prediction and digit recognition datasets were obtained with K = 50 and K = 2, respectively.



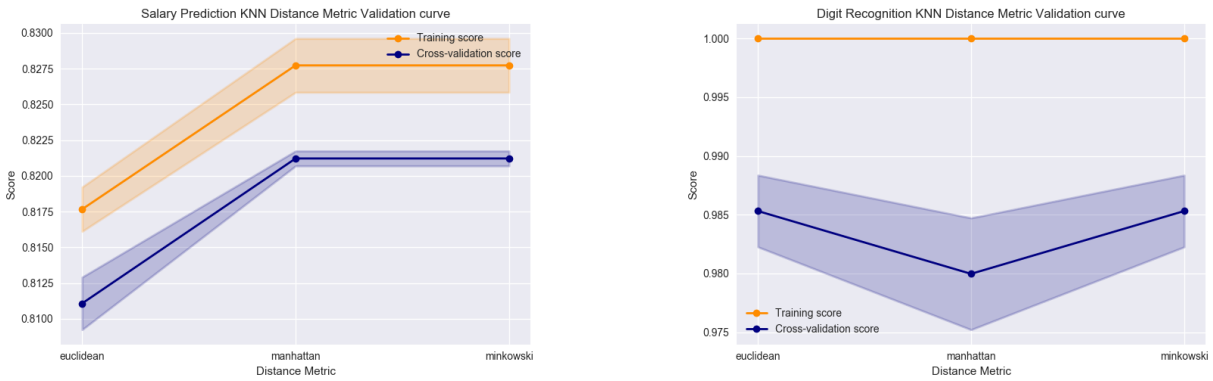
(a) Salary Prediction



(b) Digit Recognition

Figure 7: KNN Validation Curves with changing K

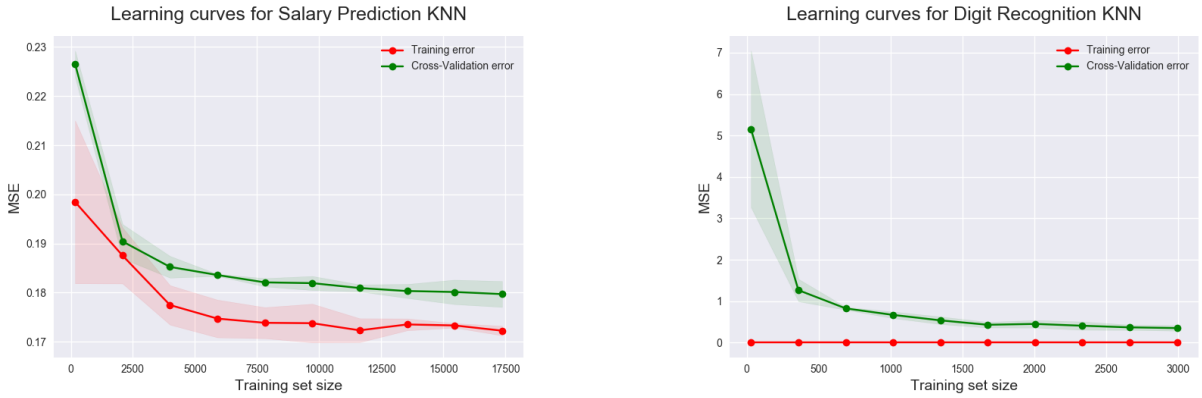
Figures 8a and 8b show the validation curves against three different distance metrics available. There was a slight difference between each of the distance metrics. The manhattan and minkowski distance metrics acheived 1% better accuracy against the salary dataset while the euclidean and minkowski metrics acheived 0.5% better accuracy against the digit dataset.



(a) Salary Prediction (b) Digit Recognition

Figure 8: KNN Validation Curves with changing distance metric

Applying the best performing values for K and the distance metric, Figure 9 shows the training and cross validaiton error of the KNN classifier against both datasets for an increasing training set size. For the salary dataset, the KNN's accuracy is better than the neural network but falls just short of the decision tree. While KNN is robust to noise, it requires all features to calculate the distance between instances. Therefore, KNN's performance agains the salary dataset in this experiment may potentially be improved by reducing the feature space to only the most critical attributes. KNN algorithm's test accuracy of 98%, shown in Figure 16, is the highest among all supervised learners tested against the digit dataset and the low bias and low variance is evident in Figure 9b.



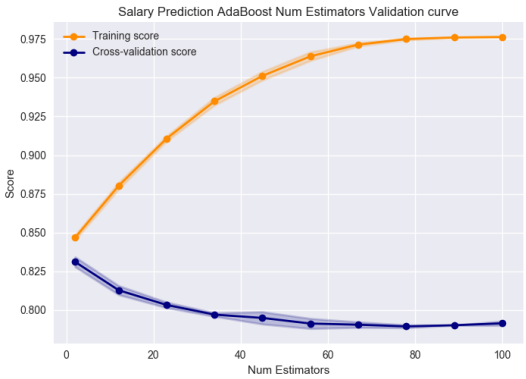
(a) Salary Prediction (b) Digit Recognition

Figure 9: KNN Learning Curves

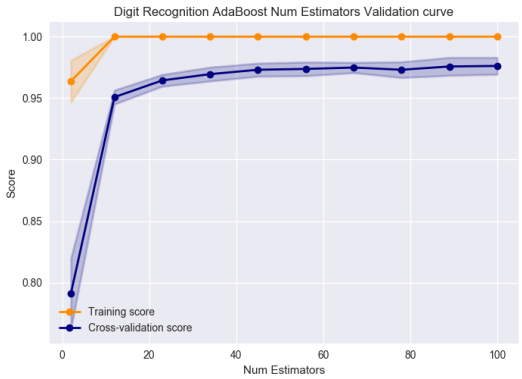
3.4 Boosting

Figure 10 shows the Adaboost validation curves for both datasets as the number of decision tree weak learners increases. Interestingly, the AdaBoost classifier performed worse in the cross-validation test set against the salary prediction dataset as more weak learners were added, while its accuracy increased against the training set. Conversely, its average accuracy against the digit cross-validation set converged to approximately 97% as more weak learners were added. Contrary to expectation, many weak decision tree learners performed worse than the single decision tree classifier did against the salary dataset. The AdaBoost classifier is especially susceptible to noise and will also fail to perform if there is not enough data. [13] The salary dataset is seven times larger than the digit dataset, and the AdaBoost algorithm performed well against the smaller dataset, making size unlikely to be a contributing factor in the poorer performance. However, the noise in the salary dataset may be the contributing factor. For example, it contains four features that have less than 1% importance: "workclass, sex, native-country, marital-status, and race". In the context of the classifier, these features may be contributing more as noise than as entropy reducing attributes in the decision tree weak learner. This also suggests that the dataset itself may not

contain the information needed to to produce an accurate target function.



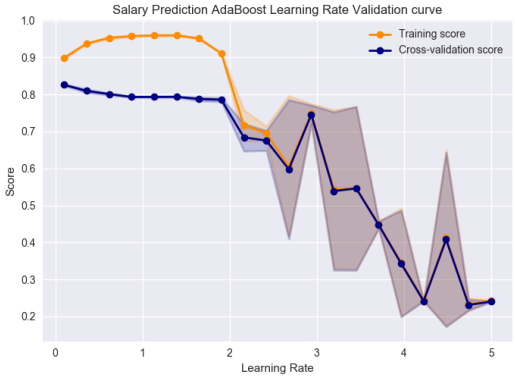
(a) Salary Prediction



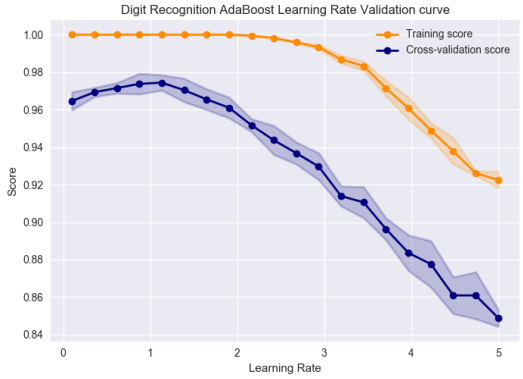
(b) Digit Recognition

Figure 10: AdaBoost Validation Curves with increasing number of weak learners

Figures 11a and 11b show that the optimal learning rate used to shrink the contribution of each weak learner was 1.13 for both datasets.



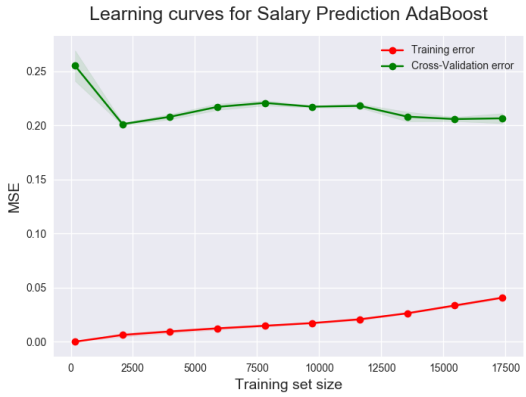
(a) Salary Prediction



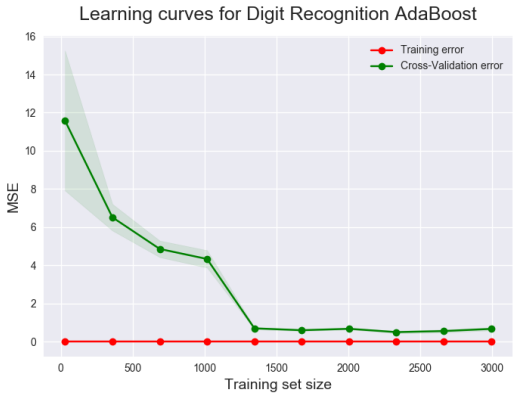
(b) Digit Recognition

Figure 11: AdaBoost Validation Curves with changing learning rate

The AdaBoost learning curves are shown in Figure 12. Note the superior performance of the AdaBoost algorithm against the digit dataset and the poorer perforamance against the salary dataset.



(a) Salary Prediction

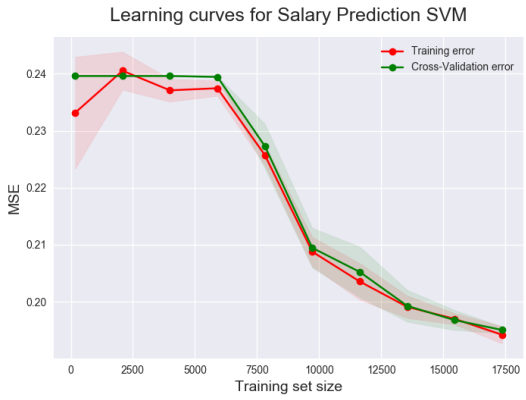


(b) Digit Recognition

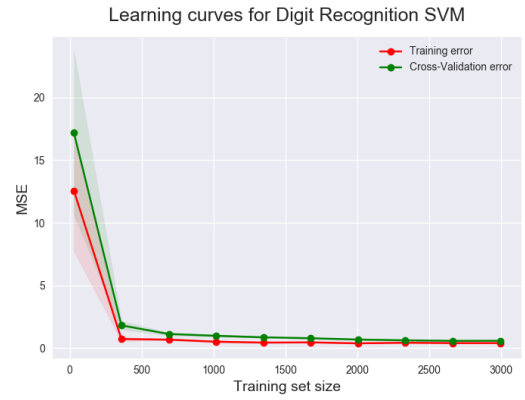
Figure 12: AdaBoost Learning Curves

3.5 Support Vector Machine

Figure 13 shows the learning curves for the support vector machine classifier against increasing training set size. The training and cross validation errors show low variance against both datasets and surprisingly low bias against the digit dataset. The learning curve against the salary dataset shows low variance as both training and cross-validation curves converge to an error under 0.20 MSE. It becomes clear from Figure 13a that the SVM required a larger training size to drop further than the inital plateau of 0.24 MSE. When the training and cross-valdiation errors do not converge, it is usually a sign that more training examples are needed as long as the cross-validation error is still decreasing. [11] In this experiment, the convergence of the two curves happened early and the MSE continued to drop further as more training samples were added. The 96% test accuracy in Figure 16 of SVM against the digit dataset is a result of the hyper-parameter tuning, as its initial accuracy was quite poor.



(a) Salary Prediction



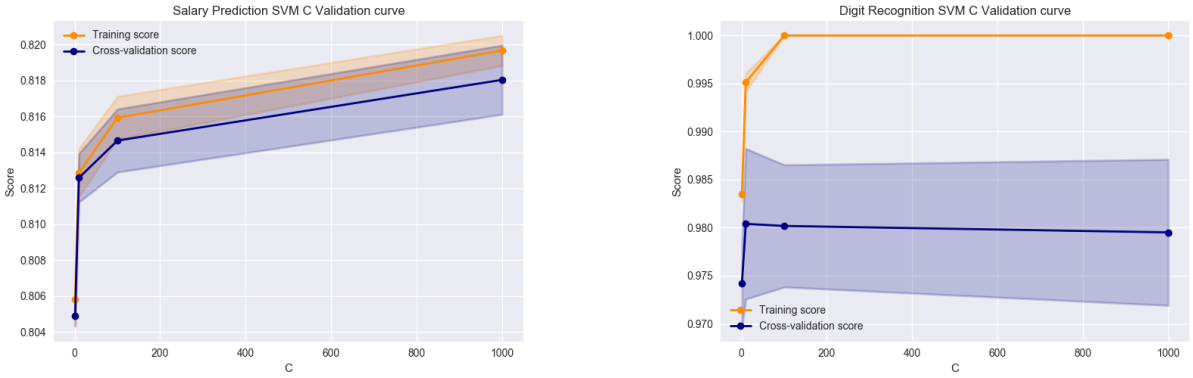
(b) Digit Recognition

Figure 13: SVM Learning Curves

The validation curves in Figure 14 show the score of the SVM classifier for changing values of C. The sklearn user guide mentions that the fit time for the C-Support Vector classifier will scale quadratically with the number of samples and therefore may be impractical beyond tens of thousands of samples.[3] As is shown in Figure 16 the SVM classifier took the second longest amount of time to train, second only to the neural network, but had the third best accuracy against the digit dataset. The lower test accuracy of 80% from Figure 16 against the salary dataset may be attributed to not having enough training samples, and the possibility that the dataset just doesn't have the information in it to generalize well for any learner. Therefore, while the accuracy of the SVM classifier was on par with the other learners, its high training time against the datasets, especially the salary dataset, is obtrusive. Therefore, the SVM classifier would not be an optimal choice for the chosen tasks.

classifier	training time (s)	train query (s)	test query (s)	train accuracy	test accuracy
Digit Recognition AdaBoost	0.825	0.132	0.043	1.0	0.97
Digit Recognition Decision Tree	0.012	0.002	0.001	0.93	0.82
Digit Recognition KNN	0.013	2.095	0.54	1.0	0.98
Digit Recognition Neural Network	5.924	0.013	0.004	1.0	0.97
Digit Recognition SVM	0.499	0.706	0.176	0.99	0.96
Salary Prediction AdaBoost	1.845	0.202	0.056	0.93	0.8
Salary Prediction Decision Tree	0.063	0.007	0.003	0.84	0.83
Salary Prediction KNN	0.324	5.447	1.326	0.83	0.83
Salary Prediction Neural Network	19.15	0.08	0.022	0.83	0.82
Salary Prediction SVM	16.551	7.486	1.859	0.81	0.8

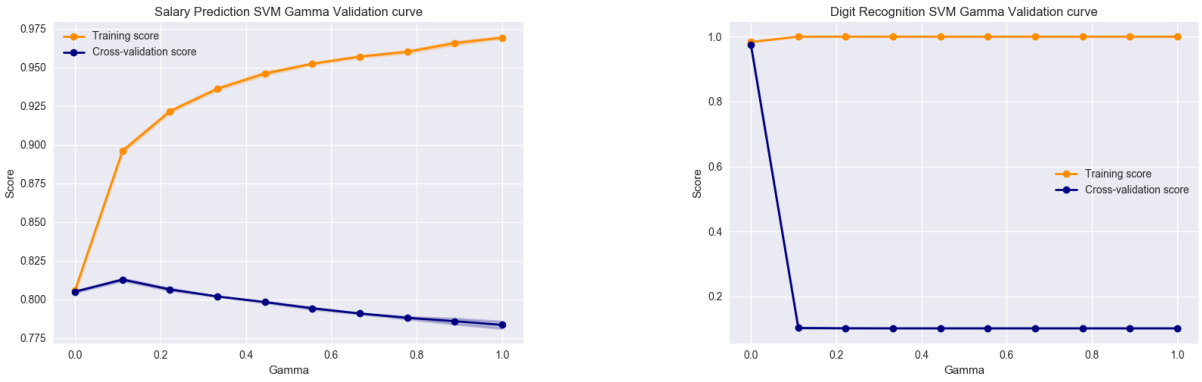
Figure 16: Performance comparison of each classifier



(a) Salary Prediction (b) Digit Recognition

Figure 14: SVM Validation Curves with increasing values for C

For the SVM learner, the best value for gamma can be determined from Figures 15a and 15b. A gamma value of 0.0001 showed the highest accuracy during both cross-validation and training for both datasets.



(a) Salary Prediction (b) Digit Recognition

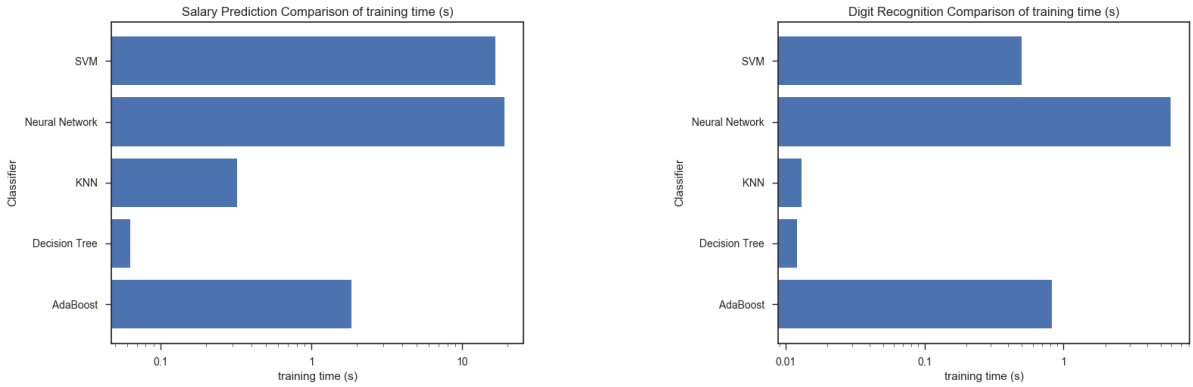
Figure 15: SVM Validation Curves with changing Gamma

3.6 Performance Comparison

Figure 16 shows a summary table of all the classifiers. All test accuracies and timings are measured against the test set, which is a reserved 20% partition of each dataset that no classifier had previously been exposed to.

Considering just the training times in Figure 17, the decision tree performed best against both datasets, while the support vector machine and neural network took the longest time to train against the salary and digit datasets,

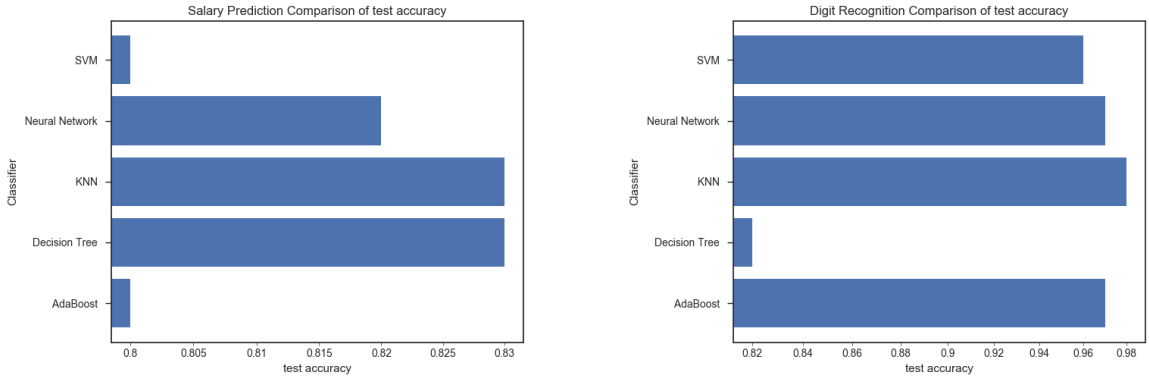
respectively.



(a) Salary Prediction (b) Digit Recognition

Figure 17: Training Times for each classifier

Considering just the accuracies against the test set in Figure 18, the KNN classifier performed best against both datasets, only matching the accuracy of the decision tree against the salary dataset, while the AdaBoost and SVM algorithms performed quite poorly against the salary prediction dataset and the decision tree had the worst performance against the digit recognition dataset.



(a) Salary Prediction (b) Digit Recognition

Figure 18: Test Accuracy for each classifier

Figure 18a and 18b show similar performances of the decision tree against both datasets, however Figure 18b shows that the decision tree failed to match the higher accuracies of the other learners. Some of the features of the salary dataset were weighted very low in imporance by the decision tree. However, the decision tree's ability to use a subset of instances when forming its hypothesis makes it more resilient to irrelevant features when generalizing to unseen instances. [1] However, the features of the digit dataset are quite different from the salary dataset, as two digits that are the same are more likely to vary in their spacial representations across attributes. The information in the flattened image data is not bounded in a way that makes it splittable by features in the way the decision tree requires. This likely gave the decision tree trouble against the handwriting recongition task, since the decision tree places more important features at the top of the tree, while the digit dataset feature importances are likely to change between each image and handwriting technique. While the decision tree had the highest relative accuracy against the salary dataset among the other learners, other learnes such as KNN and the neural network scored 83% and 82%, respectively.

We might expect a binary classification task to be easier for the learners, but this was surprisingly not the case. We would also expect the AdaBoost classifier, which consists of decision tree weak learners, to surpass the performance of a single decision tree. Interestingly, this was not the case and is likely because the adaboost classifier

needs additional tuning. Now, the top performing salary prediction classifier was only 3% more accurate than the worst performer. The collective low accuracy of all learners suggests that the salary dataset may not contain the right features or enough information that is needed to accurately predict salary range for an individual. In fact, the Kohavi et. al had to design a hybrid model based on a Decision Tree and Naive Bayes classifier just to reach an accuracy of 85.90%. [7, 8]

3.7 Conclusion

While neural networks have received increased focus in the literature recently with regards to visual recognition tasks, other supervised learning methods such as KNN and Boosting can demonstrate similar or better accuracies and spend less time training. The advantage found with these other learners was isolated to the visual task. As for the task of predicting the salary range of an individual based on the questionnaire of the salary dataset, I found various supervised learning methods to be no more effective than the hybrid decision tree proposed by Kohavi et. al [8]. While the training times of each supervised learning method relative to one another did not fluctuate between the two datasets, the accuracies did. For example, the decision tree performed quite poorly in the visual handwriting recognition task but tied for the highest accuracy with KNN in prediction of salary outcome. The KNN learner had the second shortest training times of all learners for both tasks, the best accuracy for both tasks among all learners, but had the second worst test query time agains the digit dataset and the second worst query time (only behind SVM) against the salary dataset. This long query time is in agreement with what we expect from the a lazy learner like KNN.

References

[1] Mitchell, Tom M. "Machine learning." (1997).

[2] Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." Journal of machine learning research 12.Oct (2011): 2825-2830.

[3] Buitinck, Lars, et al. "API design for machine learning software: experiences from the scikit-learn project." arXiv preprint arXiv:1309.0238 (2013).

[4] Bergstra, James, and Yoshua Bengio. "Random search for hyper-parameter optimization." Journal of Machine Learning Research 13.Feb (2012): 281-305.

[5] Brown, Michael Scott, Michael J. Pelosi, and Henry Dirska. "Dynamic-radius species-conserving genetic algorithm for the financial forecasting of Dow Jones index stocks." International Workshop on Machine Learning and Data Mining in Pattern Recognition. Springer, Berlin, Heidelberg, 2013.

[6] Garris, Michael D., James L. Blue, and Gerald T. Candela. "NIST form-based handprint recognition system." Technical Report NISTIR 5469 and CD-ROM, National Institute of Standards and Technology. 1994.

[7] https://archive.ics.uci.edu/ml/datasets/Census+Income

[8] Kohavi, Ron. "Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid." Kdd. Vol. 96. 1996.

[9] https://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits

[10] Wasserman, Larry. All of statistics: a concise course in statistic inference. Springer Science & Business Media, 2013.

[11] https://www.dataquest.io/blog/learning-curves-machine-learning/

[12] https://towardsdatascience.com/gini-index-vs-information-entropy-7a7e4fed3fcb

[13] Schapire, Robert E. "The boosting approach to machine learning: An overview." Nonlinear estimation and classification. Springer, New York, NY, 2003. 149-171.

[14] Plamondon, Réjean, and Sargur N. Srihari. "Online and off-line handwriting recognition: a comprehensive survey." IEEE Transactions on pattern analysis and machine intelligence 22.1 (2000): 63-84.

[15] LeCun, Yann, et al. "Learning algorithms for classification: A comparison on handwritten digit recognition." Neural networks: the statistical mechanics perspective 261 (1995): 276.