Aaron Luo
A13782379
COGS 118A
Tu

Final Project

1. Abstract

This experiment will test four different learning algorithms across three different datasets of varying complexity in an attempt to make an objective assessment over the effectiveness of each one.

2. Introduction

Given the large variety of supervised learning algorithms that have been developed over the last one or two decades, it can be difficult to keep track of the pros and cons of each method as well as determine which options would be the most efficient for what kinds of datasets. For example, lazy learning algorithms such as k-Nearest Neighbors classification can be advantageous in that they are generally easy to implement, but can also be very slow especially with high complexity datasets, and may not necessarily be appropriate. Support Vector Machines are also rather easy to implement and are guaranteed to to be optimized given that convex optimization always finds the global minimum, but they are also mostly used for small and clean datasets and typically don't handle larger and noiser datasets all that effectively as they can be fairly vulnerable to overfitting depending on the kernel used. Decision Tree classification is somewhat unique in that it handles qualitative/categorical data well in addition to being amongst the easiest algorithms to interpret, but, somewhat similarly to the previous learning algorithms, is prone to overfitting and doesn't handle high complexity or even large datasets all that well, decreasing in accuracy as depth or size increases. Random Forest classification, an algorithm that is more or less designed to compensate for the problems of the Decision Tree classifier, is essentially a combination of many decision trees and is actually completely immune to overfitting and is highly accurate across most dataset complexity and sizes, but does take exponentially more processing time as dataset complexity increases in addition to a interestingly opposite problem to decision trees in that its can be difficult to interpret given how many moving parts are present. Neural Networks are designed to overcome many of the problems typical of other Machine Learning algorithms such as poor scaling with dataset size and are, in theory, meant to outperform every other Machine Learning algorithm, but they still have their own set of caveats. Single Layered Neural Networks like the Perceptron algorithm have very fast computational speeds but are only really effective when handling low complexity datasets by virtue of being single layered; more attributes begets less accuracy. Multi-Layered Perceptron algorithms can somewhat compensate for this complexity issue, but still hold the same set of problems that plague Neural Networks in general in that it performs poorly on small datasets and is internally complex, making it difficult to understand the computational process behind each algorithm. This paper will attempt to physically demonstrate the characteristics of some of the learning algorithms discussed here as well as make practical comparisons between them from their performance on a select few datasets.

3. Methods

K-Nearest Neighbors: Used ten values of k neighbors from the numerical set K=[1, 10, 15, 20, 25, 30, 35, 40, 45, 50]. Only Euclidean distance was tested. This algorithm was selected for its simplicity as well as to test its performance on larger and higher complexity datasets in comparison to other algorithms.

Support Vector Machine: Used a radial basis function kernel over five penalty parameters from the numerical set C=[0.00001, 0.0001, 0.001, 0.01, 0.1]. Linear kernel was tested and then dropped in favor of an RBF kernel as it returned lower accuracies, probably due to the difficulty of linearly separating the datasets used in this experiment. This algorithm was also chosen for its simplicity/ease of implementation as well as to test its performance on large datasets that it traditionally is not used on.
Random Forest: Used 1024 decision tree estimators over eight minimum sample split values from the numerical set n=[1, 2, 4, 6, 8, 12, 16, 20]. The Gini impurity criterion was used over the Entropy criterion to remove logarithmic functions and cut down on computational power requirements, which were already rather high given the size and complexity of the datasets used. This algorithm was chosen for its exceptionally high performance to find something as close to maximum accuracy as is theoretically possible for any given dataset.
Perceptron: Used five different maximum passes over the training data from the numerical set n=[1, 5, 10, 20, 30]. This algorithm was chosen to establish a baseline performance of a neural net algorithm as well as it to test its effectiveness on high complexity datasets that it is not typically used on.
Multilayer Perceptron: Used six different hidden layer sizes from the numerical set n=[1, 2, 4, 8, 32, 128] with an adaptive learning rate and a maximum number of iterations of 500, the latter of which was set to allow the algorithm adequate iterations to converge. This algorithm was chosen to expand on the baseline performance demonstrated by the Perceptron algorithm and allow for further experimentation on neural net performance with higher dimensional datasets.

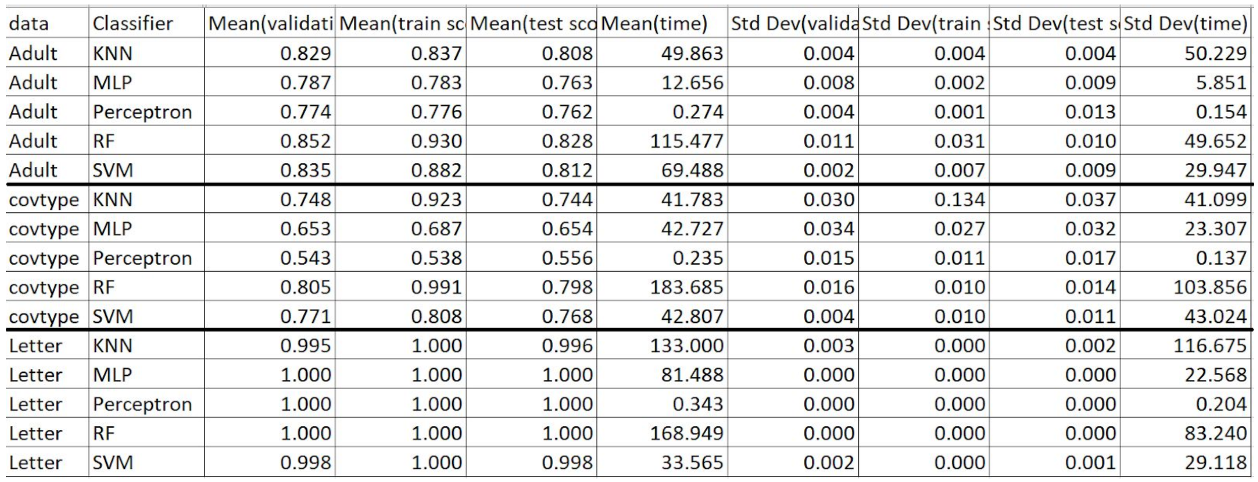
3.1 Datasets

ADULT: This dataset is heavily categorical, allowing for meaningful testing on algorithms that typically perform well on categorical variables such as Random Forest and algorithms that don't such as KNN. After one hot encoding the dataset expanded from 14 to 104 features, further allowing for meaningful testing on algorithms that handle high complexity datasets well and those that don't. Data points are binarily classified based on annual income, where an annual income of <=50K was classified as 1 and an annual income of >50K was classified as 0. Dataset was manually cleaned and one hot encoded using an algorithm so that any data points with missing data were removed and categorical features were properly measured.
COVTYPE: This dataset is somewhat categorical but significantly less so than the ADULT dataset, with 10 quantitative data types and 2 qualitative data types. One hot encoding expands dataset complexity from 12 features to 54 features. This dataset is significantly larger than any of the other datasets used in this paper, allowing for meaningful testing on algorithms such as neural nets that theoretically handle large datasets well and those that don't such as SVM. Data points are binarily classified based on seven different forest cover type, where wilderness areas with cover type 2 are classified as 1 and everything else is classified as 0, making for a somewhat unbalanced problem. Dataset was used precleaned and one hot encoded.
LETTER: This dataset is comparably small to the other datasets and does not use any qualitative data types. This allows for meaningful testing on algorithms that favor low size/low complexity datasets such as KNN and Perceptron. Data points are binarily classified based on 26 different letters, where every letter recognized as the letter 'O' is categorized as 1 and everything else is categorized as 0. This makes for a heavily unbalanced problem, but is necessary as letters are not necessarily structurally similar to each other and thus cannot be categorized as such. Dataset did not need to be cleaned or one hot encoded.

4. Experiment

ADULT, COVTYPE, and LETTER datasets were handled separately but were processed by similar algorithms. The ADULT dataset was first cleaned of all data points with missing data, followed by a removal of all unwanted punctuation, after which all categorical data types were one hot encoded. All datasets then underwent binary classification, as described in the 'Datasets' section above. The category chosen to be classified as 1 was selected based on the fact that it was the largest category except in the LETTER dataset, the category of which was selected based on its resemblance to other letters. Following this, each dataset was processed by every learning algorithm discussed in this paper (KNN, SVM, Random Forest, Perceptron, and MLP) with three different training and test data splits (20/80, 50/50, 80/20). ADULT and COVTYPE datasets were normalized when passed through KNN and SVM as these algorithms are particularly sensitive to feature bias, but not for the LETTER dataset it contained little to no such bias. Hyperparameters were tuned before data fitting using five fold grid search cross validation. Training and validation accuracy was visualized on a heatmap with every data split, and the relationship between each hyperparameter and its corresponding validation accuracy was plotted after every 80/20 split. Cross validation and training time were clocked to track the efficiency of each algorithm and data split. Test data was scored last with each learning algorithm and data split using the best corresponding hyperparameter.

5. Results



data	Classifier	Mean(validati	Mean(train sc	Mean(test sco	Mean(time)	Std Dev(valida	Std Dev(train	Std Dev(test s	Std Dev(time)
Adult	KNN	0.829	0.837	0.808	49.863	0.004	0.004	0.004	50.229
Adult	MLP	0.787	0.783	0.763	12.656	0.008	0.002	0.009	5.851
Adult	Perceptron	0.774	0.776	0.762	0.274	0.004	0.001	0.013	0.154
Adult	RF	0.852	0.930	0.828	115.477	0.011	0.031	0.010	49.652
Adult	SVM	0.835	0.882	0.812	69.488	0.002	0.007	0.009	29.947
covtype	KNN	0.748	0.923	0.744	41.783	0.030	0.134	0.037	41.099
covtype	MLP	0.653	0.687	0.654	42.727	0.034	0.027	0.032	23.307
covtype	Perceptron	0.543	0.538	0.556	0.235	0.015	0.011	0.017	0.137
covtype	RF	0.805	0.991	0.798	183.685	0.016	0.010	0.014	103.856
covtype	SVM	0.771	0.808	0.768	42.807	0.004	0.010	0.011	43.024
Letter	KNN	0.995	1.000	0.996	133.000	0.003	0.000	0.002	116.675
Letter	MLP	1.000	1.000	1.000	81.488	0.000	0.000	0.000	22.568
Letter	Perceptron	1.000	1.000	1.000	0.343	0.000	0.000	0.000	0.204
Letter	RF	1.000	1.000	1.000	168.949	0.000	0.000	0.000	83.240
Letter	SVM	0.998	1.000	0.998	33.565	0.002	0.000	0.001	29.118

As seen the figure above, Random Forest had overall the highest accuracy across all three datasets, but overall also took the most amount of processing time (cross validation and training time). Perceptron and MLP classification performed the worst when categorizing the ADULT and COVTYPE datasets, but were able to reach perfect accuracy along with Random Forest when categorizing the LETTER dataset. Perceptron was also the fastest algorithm by a significant margin in spite of its poor accuracy in two of the larger/more complex datasets. KNN and SVM both consistently outputted relatively high accuracy rates but at wildly inconsistent processing speeds; both are around the middle when it comes to processing time but are both on the higher end of timewise standard deviation, suggesting that their efficiency relies to an extent on how the dataset is split. This is corroborated by the raw data, which shows that both KNN and SVM in addition to Random Forest experience a rather dramatic increase in processing time as training data size increases. Obviously the other learning algorithms also show an increase in processing time as training size increases, but not to the same extent.

6. Conclusion

The results of this experiment more or less supports the ideas discussed in the introduction of this paper. KNN performed rather consistently well across all 3 datasets but did take a very long time to process the data especially in the larger datasets, as was already predicted. What was not predicted was KNN taking the most amount of processing time on average on the least complex dataset LETTER. Why this is the case is rather unclear, but is something that could be tested in the future. Overall, KNN seems to be an algorithm that one could easily apply to any kind of dataset, as long as time efficiency is not a concern. SVM, like KNN, also performed rather consistently in terms of test accuracy score across all three datasets, albeit slightly better than KNN. SVM also performed more towards expectations in terms of processing time, taking the longest in the larger and especially more complex datasets while taking the least amount of time in the smallest and least complex dataset. It can also be established, based on the results, that SVM is a superior option to KNN when the dataset is smaller and simpler but worse when the dataset is larger and more complex, a relationship that was also predicted in the introduction of this paper.

Random Forest performed just about as expected as well, having consistently one of the highest if not the highest test accuracy scores across all three datasets while also topping the board in terms of processing time. Interestingly enough, seems to take more time processing the lower complexity dataset LETTER than it takes to process the higher complexity dataset ADULT, an unintuitive relationship also seen earlier with KNN. This would perhaps require further testing, but it may have something to do with cross validating an algorithm that, in theory, doesn't need to be cross validated as it is immune to overfitting.

Perceptron and MLP also performed as predicted, with low accuracies in high complexity datasets and high accuracies in low complexity datasets. What may have went against expectations is the neural net's poorer performance in a larger dataset like ADULT or COVTYPE than in a smaller dataset like LETTER, but since LETTER is already rather big with 20,000 data points, this size difference may not have mattered as much as the dataset complexity. Assuming this, it can be established that Perceptron is superior to MLP and all other learning algorithms at lower dimensionality due to how fast it can process data. It is more difficult to say if MLP is ever superior to Perceptron at higher dimensions due to both returning about low test accuracy scores in the high complexity ADULT and COVTYPE datasets; however, since MLP did return a score that was significantly better than that of Perceptron in the largest COVTYPE dataset, despite both of them being rather low, it could eventually be established that MLP performs better at higher dataset sizes than Perceptron does. More testing supporting this would have to take place, however, to actually confirm this kind of relationship.

7. References

Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.
Caruana, R., & Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. ICML '06, 161–168.
Padmanabha, Akshay & Williams, Christopher. K-nearest Neighbors. *Brilliant.org*. Retrieved from https://brilliant.org/wiki/k-nearest-neighbors/
Bambrick, Noel. Support Vector Machines: A Simple Explanation. KDNuggets.com. Retrieved from https://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation.html
Donges, Niklas. "Pros and Cons of Neural Networks – Towards Data Science." *Towards Data Science*, 17 Apr. 2018, Retrieved from https://towardsdatascience.com/hype-disadvantages-of-neural-networks-6af04904ba5b.
Hamel, Gregory. "Advantages & Disadvantages of Decision Trees." *Techwalla*, Retrieved from www.techwalla.com/articles/advantages-disadvantages-of-decision-trees.
"MLP." Ada Boost - NickGillianWiki, Retrieved from www.nickgillian.com/wiki/pmwiki.php/GRT/MLP.
Blake, C., & Merz, C. (1998). UCI repository of machine learning databases. Retrieved from http://archive.ics.uci.edu/ml/index.php