CS 7641 Assignment 1: Supervised Learning

Ali Alrasheed
College of Computing
OMSCS

***Abstract*—This paper discusses the performance of five different machine learning algorithms using two different datasets. In particular, Decision Tree, Ada-Boosting, K-Nearest Neighbors, Artificial Neural Networks, and Vector Support Machine will be trained and tested on both Heart Disease and Letter Recognition datasets.**

***Index Terms*—Machine learning, ANN, SVM, KNN, DT, Boosting, Cross-validation, Letter Recognition, Heart Disease.**

I. Introduction

Machine learning is an area under Artificial Intelligence (AI) that focuses on developing statistical models and algorithms to make classifications and regression predictions. Machine learning consists of three branches: Supervised, Simi-Supervised, and Unsupervised learning. However, this paper will only focus on applying and analyzing supervised learning algorithms on two datasets: Heart Disease [1], and Letter Recognition [2]. In particular, Decision Tree, Ada-Boosting, K-Nearest Neighbors, Artificial Neural Networks, and Vector Support Machine will be discussed in the next sections.

II. Datasets

A. Heart disease prediction

Heart attack is one of the most serious health conditions. It happens when the flow going to the heart is blocked. The blockage is usually caused by a buildup of substances such as fat and cholesterol.

1) Why is the dataset interesting?: There are several aspects that make this particular dataset interesting. For example, it is a small dataset with only 300 samples, and this can show which algorithm can perform well with small samples. In addition, the attributes are both categorical and continuous which may present a challenge in prepossessing the dataset before it is passed to the algorithms. Furthermore, the dataset contains some outliers which makes it harder to achieve high accuracy in the predictions.

2) Exploring the dataset: This dataset contains 14 refined features that will be used to predict whether or not the patient has a heart disease which can prevent unexpected heart attacks. The dataset is provided by UCI machine Learning Repository [1]. The attributes in this dataset are: age, sex, Chest pain type (cp),resting blood pressure (trtbps), cholesterol level (chol), fasting blood sugar (fbs), resting electrocardiographic reslut (restecg), maximum hear rate achieved (thalach), exercise induced angina (exang), ST depression induced by exercise relative to rest (oldpeak), peak's slope of exercise ST segment (slp), number of major vessels (caa), and thal (normal,fixed defect, or reversable defect)

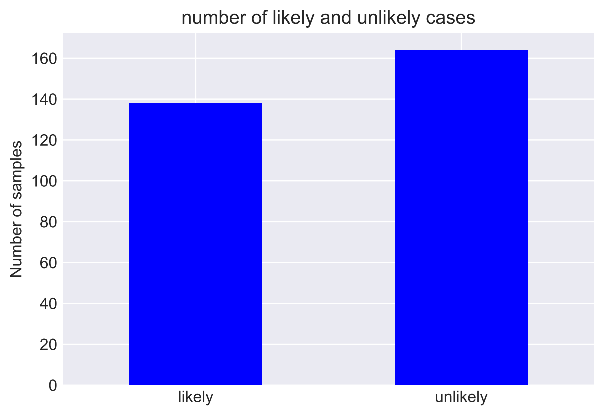

Fig. 1. Classes distributions [heart disease]

There are about 300 samples with binary labels (likely or unlikely). Figure 1 shows the classes distributions of the heart disease dataset. As is shown in the figure, the dataset is relatively balanced with 54% of the samples are labeled as unlikely (output = 1) compared to 46% likely (output = 0) that a heart disease present.

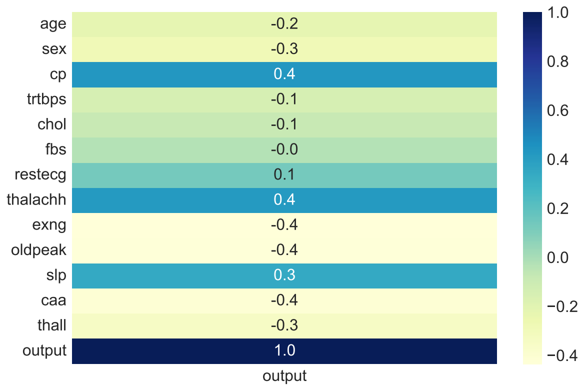

Fig. 2. correlations of attributes and presence of heart disease

Furthermore, figure 2 shows the correlation between the attributes and the presence of heart disease (output). A positive correlation means that increasing the attribute value increases the risk of heart disease. For instance, the higher the maximum heart rate achieved (thalach), the higher the probability that heart disease is present. On the contrary, negative correlations mean that decreasing the attribute values decrease the likelihood of heart disease being present. An obvious example of a negative correlation for this dataset is age. The younger the person, the less likely to have heart disease.

In addition, there are a couple of interesting observations that could be seen from figure 2. For example, looking solely at the cholesterol level does not necessarily guarantee the presence of heart disease as it has a relatively low correlation. In addition, surprisingly, age has a correlation of -0.2 which is one of the lowest in magnitudes compared to the other attributes in the dataset.

3) Preparing the dataset: preprocessing the dataset is an essential step for obtaining a good classifier. For this particular dataset, two steps were done. First, the categorical attributes were converted to one-hot encoding which is a common way to deal with categorical data in machine learning. The second step was to scale the continuous attributes using RobustScaler from sklearn which is more robust to outliers. This scaler removes the median and scales the data using the IQR quantile (the range between 25^{th} and 75^{th} quantiles). It must be noted that these preprocessing steps were able to significantly improve the performance of some algorithms. Table I shows the performance of the algorithms with and without applying the above-mentioned preprocessing steps. It is expected that impact on both Decision Tree and Ada-Boosting algorithms (based on DT) would be minimal as it is shown in table I. This can be explained since the splits of tree are not affected by the scale of that attributes. However, it is interesting to see that the performance of ANN has drastically increased when the preprocessing steps are applied. This shows the importance of scaling the attributes in algorithms similar to ANN since otherwise the attributes with higher ranges will be biased with higher impacts on the activation of the neurons. A similar argument could be also made with KNN since higher range attributes will greatly affect the calculation of the distance between neighbors, and hence the KNN will be more sensitive to the changes in attributes with higher ranges.

TABLE I
EFFECT OF PREPROCESSING ON THE PERFOMANCE

Algorithm Name	**with reprocessing**		**without reprocessing**	
	Avg CV	***Train[ms]***	***Avg CV***	**Train[ms]**
DT	71 ± 5	2	70 ± 3	2
Ada-Boosting	81 ± 7	67	80 ± 5	68
SVM	82 ± 5	4	80 ± 3	8
KNN	80 ± 5	2	65 ± 5	2
ANN	83 ± 3	601	46 ± 1	13

B. Letters Recognition

The Letter Recognition dataset was obtained from UCI Machine Learning Repository [2]. The aim of this dataset is to recognize black-and-white pixels of the English capital alphabet. The dataset consists of 20000 characters that were drawn with 20 different fonts. To utilize machine learning algorithms, each letter sample was used to extract 16 statistical attributes of moments, mean, edges, etc. The attributes were then scaled and discretized to integer values between 0 and 15.

1) Why is the dataset interesting?: Although letter recognition is commonly done using Optical Character Recognition(OCR) or Convolution Neural Networks, it would be interesting to break each letter into features and use common machine learning algorithms to predict the letters.

It is also interesting from the machine learning perspective since the Letter Recognition dataset has multiple differences from the Heart Disease dataset. Unlike the Heart Disease dataset, the Letter Recognition dataset is a multi-class dataset (26 classes), so it would be interesting to see which machine learning algorithm struggles with multi-class classification. In addition, this dataset is relatively large containing around 20000 samples compared to only 300 samples in the Heart Disease dataset. Thus, it would be interesting to see if there is a particular machine learning algorithm that performs well with large datasets or vice versa.

2) Exploring the dataset: Figure 3 shows the number of samples per class. It can be seen from the figure that dataset is relatively balanced except for I, and Z, and only slightly unbalanced for X and L.

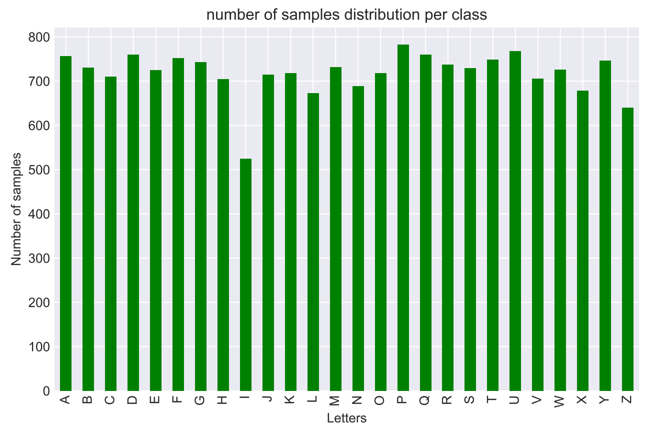

Fig. 3. Classes distributions [Letter Recognition]

In addition, figure 4 shows the correlation between the attributes and the letter classification decision. It can be seen from the figure that some attributes have a high impact on the classification decision such as ybar, xedgey, and x2ybar, while some attributes have very low impacts such as height and ybox. This may suggest that the number of attributes in this dataset could be reduced.

3) Preparing the dataset: The attributes in this data were discretized values from 0-15. One-hot encoding could be used. However, it would drastically increase the number of inputs to the machine learning algorithm. Hence, the attributes were kept unchanged. As the heart disease dataset, the samples were scaled using the RobustScale from Sklearn to reduce the effect of outliers. In addition, duplicate samples were removed (1332 duplicate samples).

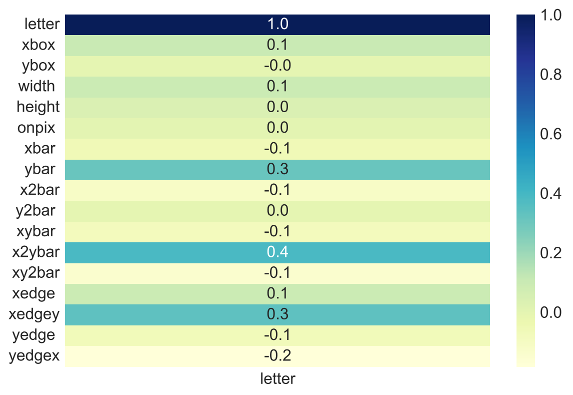

Fig. 4. Correlation between the attributes and letter classification

III. Methodology and Results

The dataset will be used to train common machine learning algorithms such as Decision tree, Boosting, SVM, Artificial Neural network, and KNN. The baseline hyperparameters were found based on a Grid Search approach using a mean score of five-fold cross-validation. To clarify, whenever cross-validation is mentioned in this paper, it should be assumed that five different models were trained on four unique combinations of the five-folds, and were tested on the remaining fold (untouched). The average of these 5 testing scores is used as the cross-validation score. In addition, the average cross-validation was also used to evaluate single hyperparameters experiments that will be shown in the next subsections.

Furthermore, to evaluate each algorithm, a learning curve and at least two single hyperparameters experiments will be presented. The accuracy and time complexity will be used as the primary matrices in this report. It should be noted that accuracy is defined as the number of samples that were correctly classified divided by the total number of samples.

A. Decision Tree (DT)

Decision Tree was implemented using Sklearn module in python. The sklearn implementation is based on a Classification and Regression Tree (CART) algorithm. In addition, the Sklearn implementation of the Decision tree classifier does not implicitly have pruning. However, it is mentioned in their documentation that the maximum depth and number of leaves could be used as pre-pruning and ccp alpha as post-pruning.

1) Heart Disease Dataset: Figure 5 shows the learning curve for decision tree. It is clear that more data is improving the average cross-validation accuracy. However, what is also interesting is that the training accuracy decreased with more samples. This could be because the Decisions Trees is considered a simple algorithm to fit a relatively large number of samples. In addition, it seems that there are some complex samples in the dataset which caused the Decision tree to be confused and under-fit. This could also explain the high variance scores in the cross-validation (green transparent area).

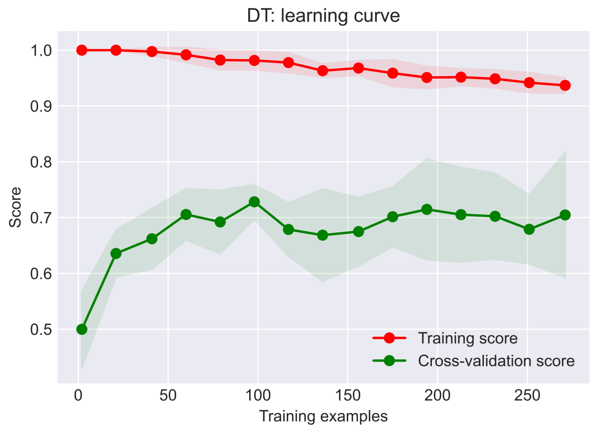

Fig. 5. DT learning curve [heart disease]

Figure 6 shows the accuracy of the decision tree in both training and average cross-validating scores (on testing splits) using different feature splitter methods (Gini and entropy). The max-leaf size is used as a pruning method as well as to avoid over-fitting the training data. It is clear from figure 6 that overfitting is present with leaf size exceeding around 7 as the training accuracy increases rapidly while the average cross-validation accuracy remands almost the same. In addition, the Gini criterion seems to be more susceptible to overfitting than the entropy. This is because Gini is generally better for selecting the best splits than entropy. However, this is a double sword weapon as selecting better splits based on the training data could lead to overfitting.

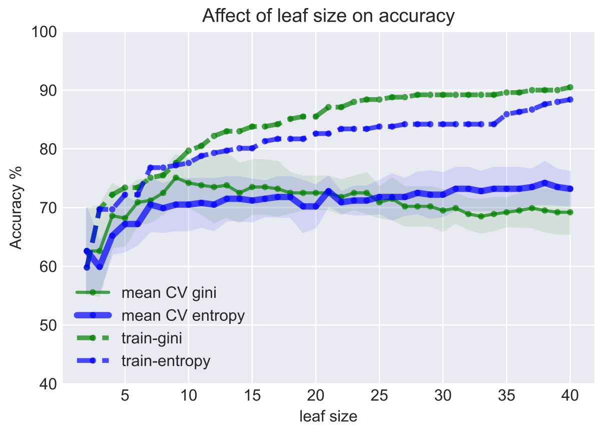

Fig. 6. Effect of leaf size [heart disease]

In addition, Decision Tree implementation in Sklearn uses a Minimal cost-complexity pruning algorithm which removes the weakest links in the tree. The number of pruned nodes can be controlled using the effective alpha parameters. The greater the value, the more nodes will be pruned. Figure 7 shows the tree depth with different value of effective alpha (ccp alpha). As can be seen from the figure, a smaller value of alpha produces a deeper tree. However, deeper trees are more prone to overfitting and have higher time complexity. On the contrary, a higher value of alpha produces a more shallow tree but also increases the impurities of the tree which could produce a less accurate tree. Therefore, choosing an appropriate value of alpha is very crucial for getting a decision tree that can obtain both good accuracy and reasonable time complexity.

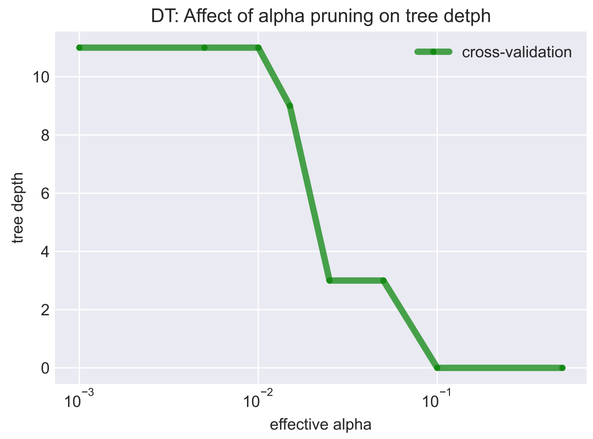

Fig. 7. Effect of Alpha pruning [heart disease]

2) Letter Recognition Dataset: Figure 10 shows the learning curve of the Decision Tree when using the Letter Recognition dataset. It is clear that the Decision Tree was able to utilize new samples to improve its average cross-validation accuracy. Since the Letter Recognition is more complex than the Heart Disease dataset (Binary vs 26 classes), the Decision Tree needs relatively large samples to produce a good performance. For example, with less than 1000 samples, the accuracy of the DT would be less than 60%. Furthermore, there is a gap between the training accuracy and average Cross-validation which indicates some variance.

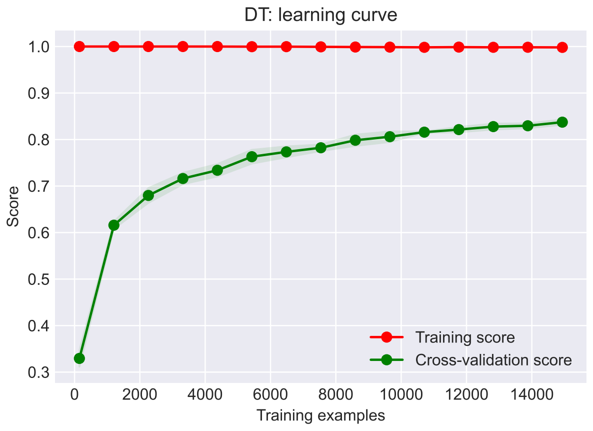

Fig. 8. DT Learning Curve for Letter Recognition

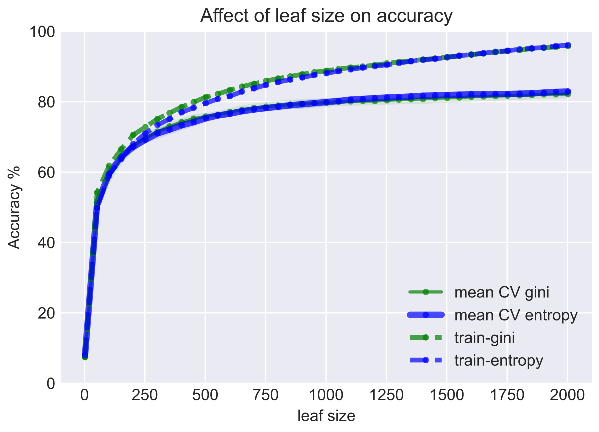

Fig. 9. Effect of leaf size [Letter Recognition]

Figure 9 shows accuracy using different combinations of leaf size and splitter functions (Gini and entropy). As stated earlier, the max-leaf size is used as a pruning method as well as to avoid over-fitting the training data. moreover, overfitting occurs when leaf size exceeds 1000 since at this point only the training accuracies increases while the average CV is almost unchanged. It is also clear from figure 6 that the performance of the DT greatly depends on the leaf size of the tree. Pruning the tree too much (low leaf size) is likely to produce bad performance. It should be also noted that since the Letter Recognition dataset is more complex than the Heart Disease dataset, it requires multiples of 10 more leaves to produce a reasonable result.

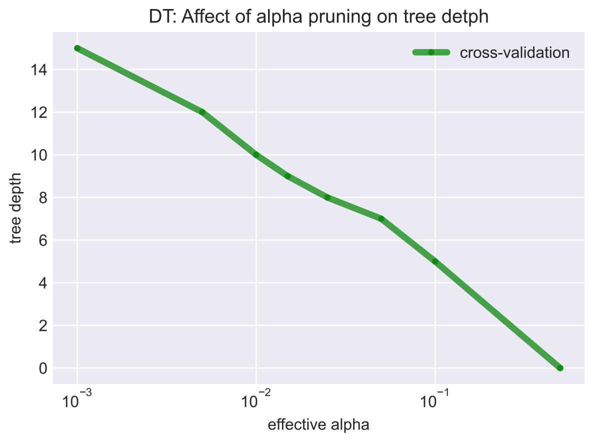

Fig. 10. Effect of Alpha pruning [Letter Recognition]

As stated previously, alpha pruning is the minimal cost-complexity algorithm that removes the weak links in the tree. The number of pruned nodes can be controlled using the effective alpha parameters. The greater the value, the more nodes will be pruned. Figure 7 shows the tree depth with different value of effective alpha (ccp alpha). As can be seen from the figure, a smaller value of alpha produces a deeper tree. However, deeper trees are more prone to overfitting and have higher time complexity. On the contrary, a higher value of alpha produces shallower trees but also increases the impurities of the tree which could lower the tree accuracy. Therefore, the alpha parameter allows the trade-off between the complexity of the tree (depth and leaves) and performance.

B. Boosting

Booting is a common type of ensemble algorithms that uses weak learners to produce one stronger learner. This paper will utilize Ada-boost algorithm implementation from Sklearn. Decision Tree is used as the base estimator for the Ada-boos ensemble.

1) Heart Disease Dataset: Figure 11 shows the learning curve of the Ada-boost classifier. It is clear that the more samples, the higher the average cross-validation accuracy which may suggest that more samples may help the model generalize better. In addition, there are two interesting observations that could be seen. The first is that the Ada-boost was able to learn faster (with less data) than one single decision tree. The second observation is that Ada-boosting was able to obtain good training accuracy at a higher number of samples compared to Decision Tree. This suggests that the Ada-boosting can model more complex data compared to a single decision tree.

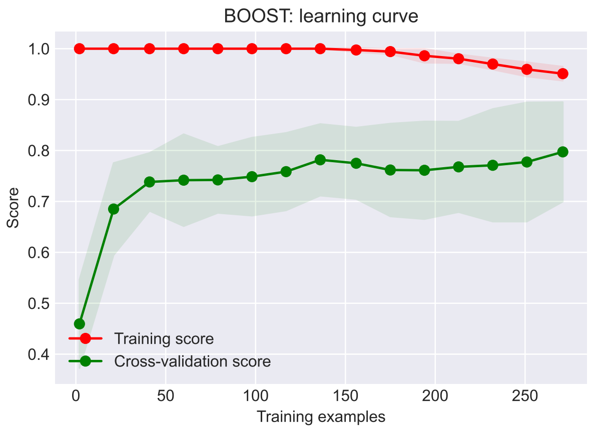

Fig. 11. Ada-Boosting learning curve [heart disease]

Furthermore, figure 12 shows the effect of the number of estimators of the Ada-boosting on the accuracy of the model. It is clear higher number of estimator lead to overfitting and high variance in accuracy on the testing splits. This can be observed since the training accuracy increases while the average cross-validation accuracy slowly decreases. The optimal number of estimators for this dataset seems to be around 10. Thus, an important observation is that increasing the number of estimators does not always result in a better classifier.

Additionally, figure 13 shows that selecting an appropriate learning rate for Ada-boosting is crucial since the learning rate is used to calculate the weight of each estimator. Thus, a too low or too high value of the learning rate can lead to very poor performance.

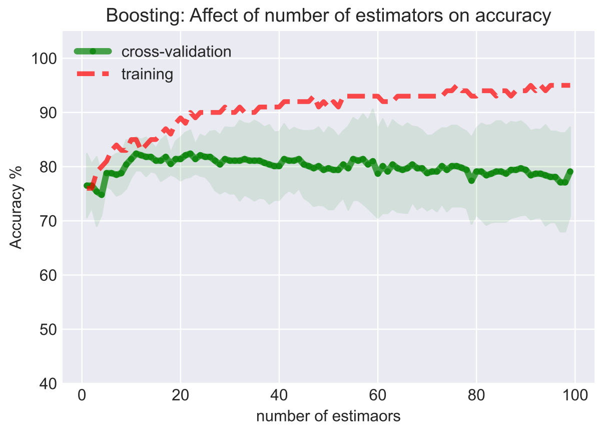

Fig. 12. Affect of number of estimators [heart disease]

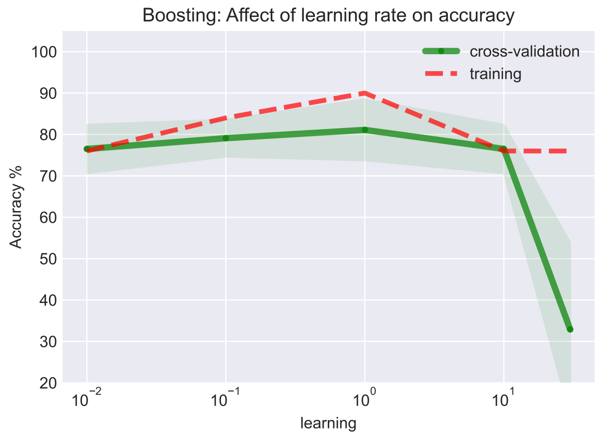

Fig. 13. Effect of the learning rate [heart disease]

2) Letter Recognition Dataset: Figure 14 shows the learning curve of boosting using the Letter Recognition dataset.

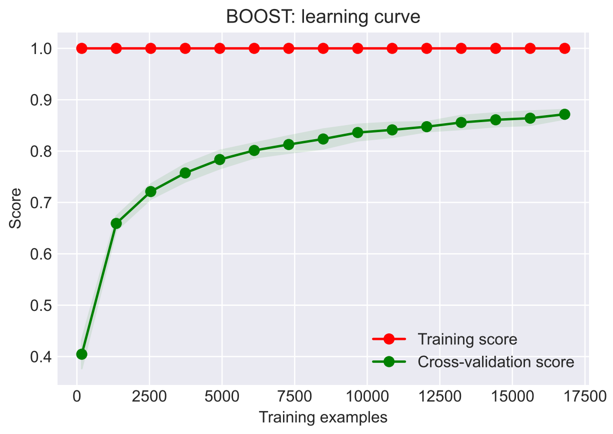

Fig. 14. Ada-Boosting learning curve [Letter Recognition]

It is clear from figure 14 that the Ada-Boosting model benefited from training on more samples. In addition, the Ada-boosting model was able to obtain better performance than one single decision tree. However, there is still a gap between the training and average cross-validation accuracy which indicates some variance.

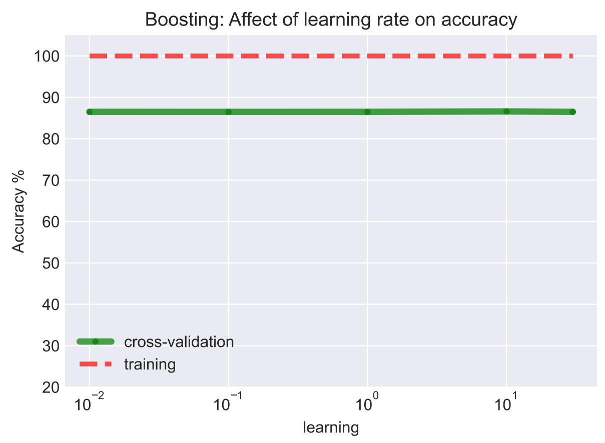

Fig. 15. Effect of the learning rate [Letter Recognition]

Furthermore, figure 15 shows the effect of the learning rate on the performance of the Ada-boosting model. Unlike in the Heart Disease dataset, it is interesting to see that the learning rate does not affect performance. This could be because the instability that is caused by the learning rate is mitigated by the number of learners and the huge number of samples. Unlike in the Heart Disease dataset, it is surprising that the performance of the Ada-boosting model is also not affected much by the number of estimators as it is shown in figure 16. The cross-validating accuracy oscillates slightly with low variance.

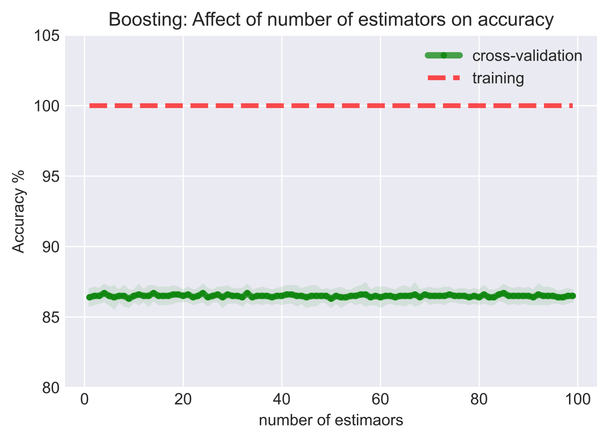

Fig. 16. Effect of number of estimators [Letter Recognition]

C. K-nearest Neighbor

KNN is a non-parametric algorithm that classifies samples based on their nearest K Neighbor (voting). It either uses uniform weights for all neighbors or distance-based weightings such as Euclidean, and Manhattan distance. The KNN algorithm used in this report is implemented using the Sklearn module in python, and distance-based weighting was used.

1) Heart Disease Dataset: Figure 17 illustrates the learning curve of KNN when trained on the Heart Disease dataset.

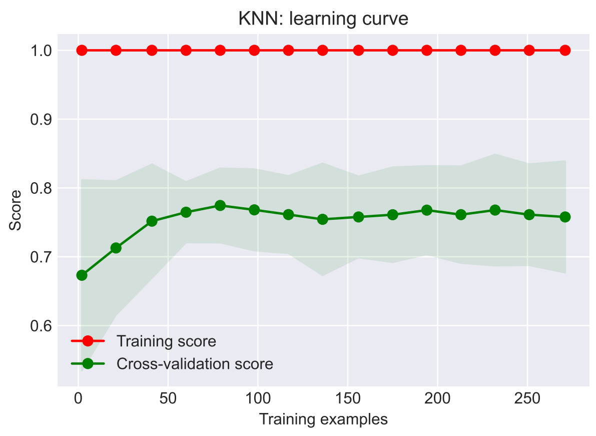

Fig. 17. KNN learning rate [heart disease]

Although the model benefits from more samples at the start, It is not clear from the figure that more samples will likely increase the accuracy of the testing splits. This might be because the dataset contains some outliers that may confuse the classifier.

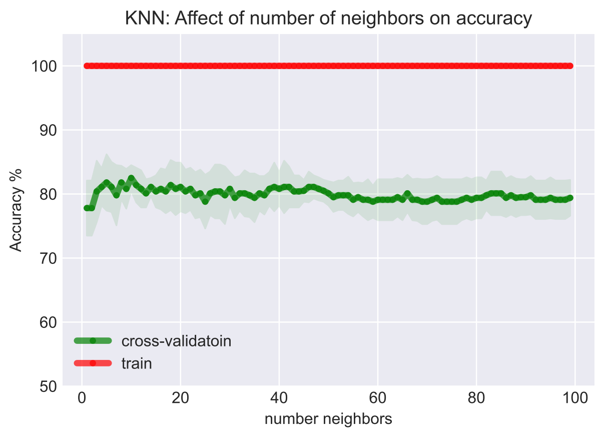

Fig. 18. Effect of K [heart disease]

Figure 18 shows the effect of the number of neighbors K that is used to classify samples. The figure shows that training accuracy is always 100% regardless of K. This suggests that the KNN is complex enough to be able to perfectly model the training data. However, increasing the number of K does not greatly affect the average cross-validation accuracy. This might suggest that there are some outliers that are very hard to classify correctly even with high number of voting (neighbors).

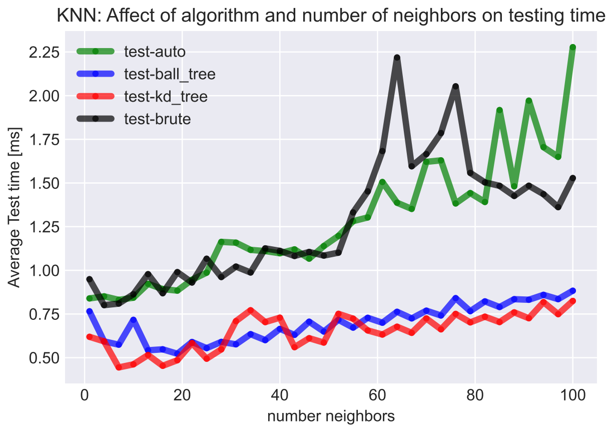

Fig. 19. Time complexity of search algorithms [heart disease]

Figure 19 shows the effect of using different algorithms to compute the nearest neighbors on the average testing time (inference). It can be seen from the figure that the effect of using more efficient algorithms becomes apparent as the number of neighbors increases. More efficient algorithms such as KD tree and Ball tree take significantly less time to find the neighbors compared to brute force or automatic search methods method from Sklearn.

2) Letter Recognition Dataset: Figure 20 illustrates the learning curve of KNN when trained on the Letter Recognition dataset. Unlike the Heart Disease dataset, adding more samples to the Letter Recognition dataset didn't confuse the KNN model. On the contrary, more samples clearly improved the average cross-validation accuracy of the model. This could mean that letter recognition dataset has fewer outliers per class than the Heart Disease dataset.

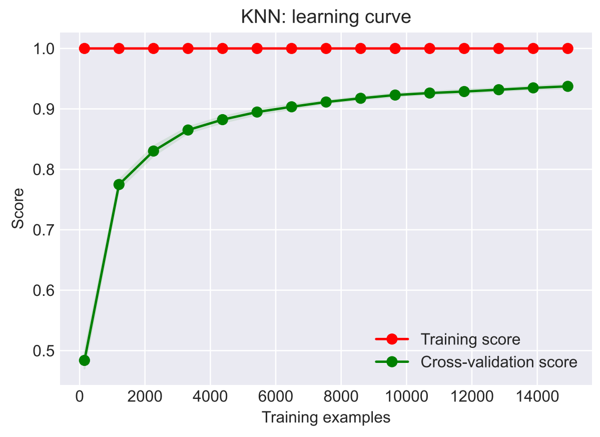

Fig. 20. KNN learning curve [Letter Recognition]

Figure 18 shows the effect of the number of neighbors K that is used to classify samples. The figure shows that training accuracy is always 100% regardless of K. This suggests that the KNN is complex enough to be able to perfectly model the training data. It can be also noticed that, unlike the Heart Disease dataset, increasing the number of neighbors resulted in worse performance in the average cross-validation. Since there are 26 classes, having more votes could introduce many neighbors that do not belong to the class of query, and hence reduce the accuracy.

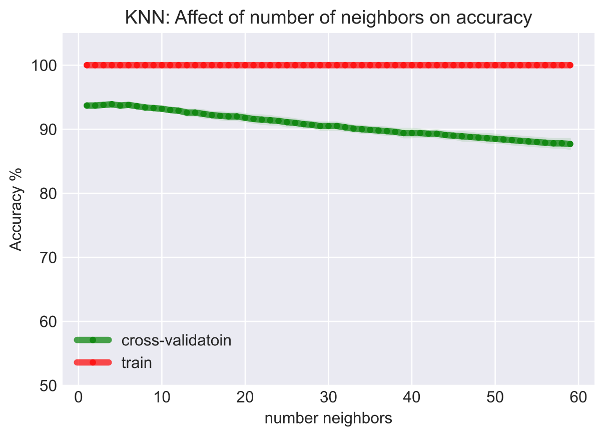

Fig. 21. Effect of K [Letter Recognition]

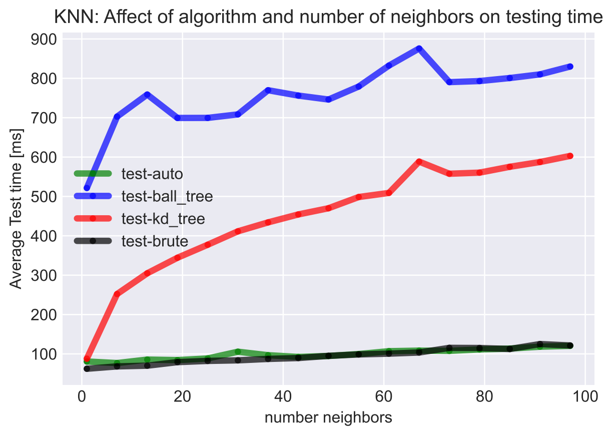

Fig. 22. Time complexity of search algorithms [Letter Recognition]

Figure 22 shows the performance of each algorithm for searching for nearest neighbors. It is interesting to see that the Ball tree and KD tree algorithms are not always the best/fastest choice for finding the nearest neighbors. In particular, they are cases where it makes sense to use brute force search instead. Several factors could affect the choice of the search algorithms used in KNN such as the data structure, number of neighbors (K), number of samples, and number of query points. The slow inference of the tree-based search algorithms could be caused by the intrinsic dimensionality of the dataset. Tree-based search algorithms perform very well when the data is sparse with a smaller intrinsic dimensionality which may not be the case in the letter dataset. In addition, unlike brute force search, the performance of tree-based algorithms also greatly deteriorates by increasing the number of nearest neighbors (K).

D. Support Vector Machine

Support Vector Machine works by separating the samples by a hyperplane. It utilizes Kernel functions to create more complicated hyperplanes such as polynomial, sigmoid, and Radial Basis Function (RBF). The hyperplane is calculated such that it maximizes the distance between each class which is usually referred to as the margin. SVM has a couple of important parameters that need to be carefully chosen to get a good performing model such as the kernel function, kernel coefficient (gamma), regularization coefficient C, etc. Sklearn implementation of SVM was used in this paper which is based on libsvm. The libsvm implementation has the advantage of providing high flexibility to easily change kernel functions and plug in their specific parameters.

1) Heart Disease Dataset: Figure 23 shows the learning curve of SVM using the heart disease dataset.

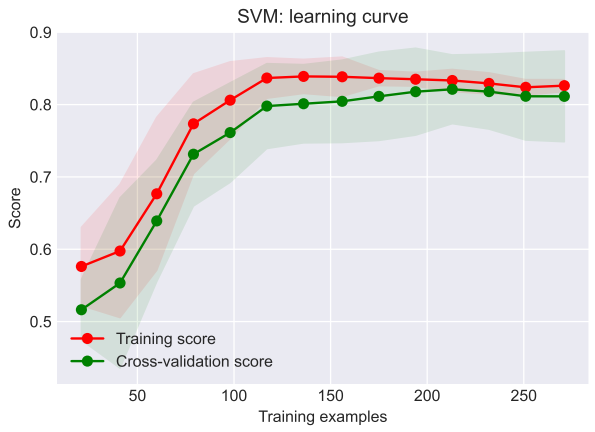

Fig. 23. SVM learning curve [heart disease]

It is clear from that figure that at the start the SVM benefited from having more samples. However, its average cross-validation accuracy tended to remain almost the same after 150 samples. Thus, it is not clear that more samples are needed to improve the model's performance.

In addition, figure 24 shows the effect of regularization parameter C on the SVM model's performance using different types of kernels. The parameter C controls the trade-off between a smooth hyperplane and a hyperplane that aims to classify all points correctly. The small value of C produces a model that is lower bias and higher variance which is more sensitive to the training samples. On the other hand, a high value of C produces a model that is a higher bias but lower variance. It is clear from figure 24 that a very low value of C results in low performance which may suggest that there are many outliers or noise in the dataset. Similarly, the large value of C reduces the performance of the models for most kernel functions which is expected as a high value of C risks making the model underfit. It is also interesting to see that the linear kernel seems to be the best performing kernel in this dataset which suggest that the dataset is mostly linearly separable excluding some outliers.

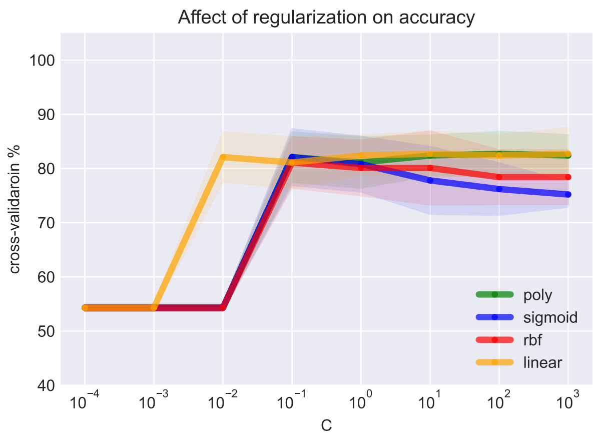

Fig. 24. Effect of regularization [heart disease]

Furthermore, figure 25 shows the effect of kernel coefficient gamma on the SVM model's performance using a different types of kernels. The gamma parameter is used to decide the weight of each point in deciding the hyperplane margin. Increasing the value of gamma, increase the influence of points that are far away from the hyperplane. It can be seen from figure 25 the RBF and sigmoid kernels are greatly affected by gamma, while the effect on the polynomial kernels was minimized since the degree of the polynomial that was used is 1 which is effectively linear. In addition, the gamma parameters only affect RBF, sigmoid, and polynomial kernels. Thus, the linear kernel is gamma invariant. Furthermore, it can be seen that the low value of gamma for RBF and sigmoid kernels produced a model that is likely to overfit the points near the boundaries of the hyperplane. These points could be outliers or noise, and therefore, giving them high weight could negatively affect the performance of the model on new data.

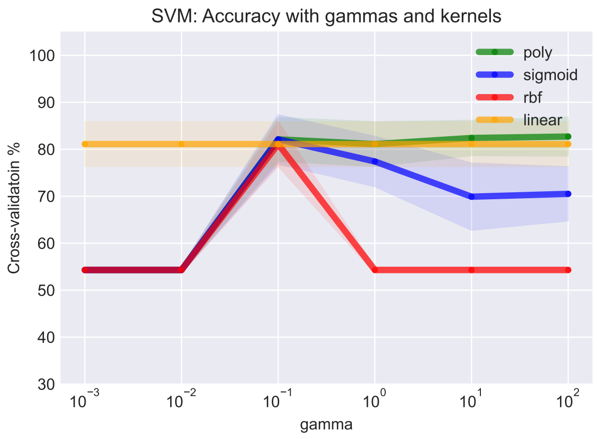

Fig. 25. Effect of Gamma [heart disease]

2) Letter Recognition Dataset: The learning curve of SVM on Letter Recognition dataset is shown in figure 26. The learning curve shows that the SVM benefited from adding more samples, but the performance tends to saturate at a higher number of samples. Thus, adding more samples doesn't seem that it will significantly enhance the performance.

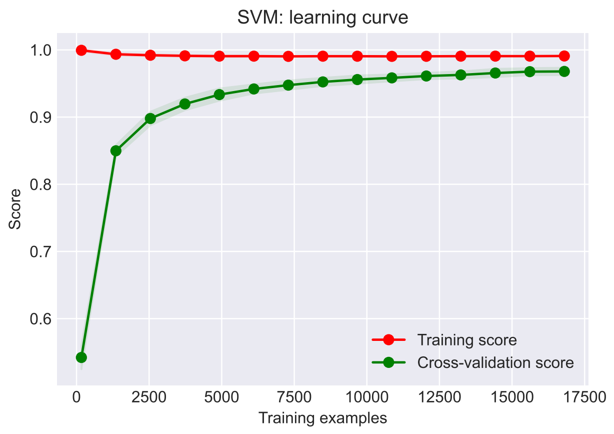

Fig. 26. SVM learning curve [Letter Recognition]

In addition, the effect of regularization parameter C on the SVM model's performance using a different type of kernels is shown in 27. As stated earlier, the parameter C controls the sensitivity of the model to the training data. It can be noticed that the effect of regularization parameter C is much greater in the Letter Recognition dataset than it is in Heart Disease. This is because Letter Recognition is more complex and has more samples. Thus, showing a reasonable value of C is very important to avoid overfitting the training data. Unlike in the Heart Disease dataset, the best kernel in the Letter Recognition dataset was RBF which transforms the data samples into infinite dimensions. Thus, the Letter Recognition data is far from being linearly separable.

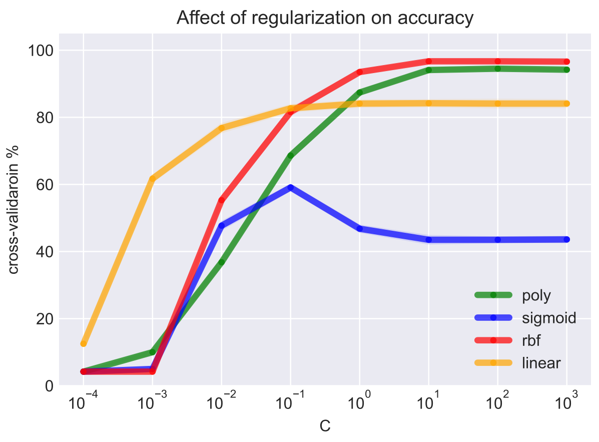

Fig. 27. Effect of regularization [Letter Recognition]

Finally, the effect of kernel coefficient gamma on the SVM model's performance using a different types of kernels is shown in figure 28. As stated earlier, the gamma parameter is used to decide the weight of each point in deciding the hyperplane margin of the SVM model. It can also be seen that the effect of gamma parameters becomes more obvious in the Letter Recognition than in the Heart Disease dataset. The RBF and Sigmoid kernels seem to be the most sensitive to gamma changes. On the contrary, the linear kernel is unaffected by the value of gamma, as it is not used in the linear kernel. At higher gamma values, the kernels give more weights for far points which may not always produce the best result as can be seen in figure 28. Therefore, choosing an appropriate value of gamma for RBF, polynomial, and sigmoid kernels is very crucial for obtaining a good-performing model.

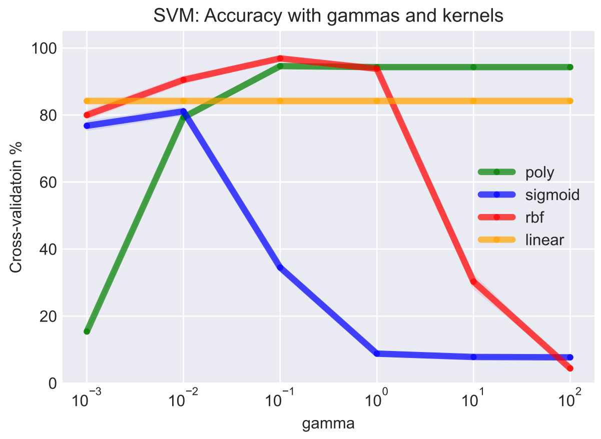

Fig. 28. Effect of Gamma [Letter Recognition]

E. Artificial Neural Network

Artificial Neural Networks (ANN) is a machine learning algorithm that was originally inspired to mimic biological neural networks. In this paper, a fully connected multi-layer perceptrons (MLP) is used from Sklearn. MLP classifier is a special form of ANN which minimize the prediction error by utilizing gradient descent and backpropagation learning algorithms. MLP consists of an input layer, hidden layers, and an output layer. There are several crucial hyper-parameters to be tuned in MLP such as number of layers, number of neurons per layer, learning rate, optimizer method, etc.

1) Heart Disease Dataset: Figure 29 shows the learning curve for ANN. One hidden layers of 10 neurons, and 3000 maximum iterations were used in the training process. It can be seen from the figure that the training accuracy started around 100%. This indicates that the model overfitted the training data with at a low number of samples. However, with more samples, the model was able to generalize better. Inferring from the upward average cross-validation score, it is likely that increasing the number of samples will result in better performance.

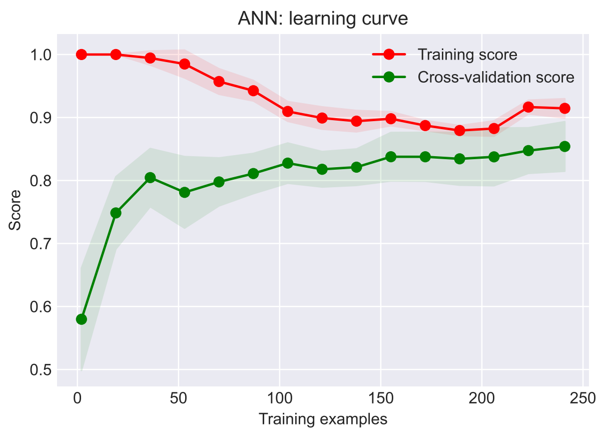

Fig. 29. ANN learning curve [heart disease]

The loss curve of ANN during training is shown in figure 30. The loss curve is useful to track the progress of the ANN during training. The loss curve could be used to identity any instability or oscillations during training. It is clear from the figure that ANN is learning rapidly at the start for both optimizer (Adam, and SGD). As expected, It can observed from the figure that Adam optimizer seems to learn and converge much faster than SGD. In addition, there is no clear instability in the training but the loss then converges at 0.2 for SGD and 0.1 for Adam.

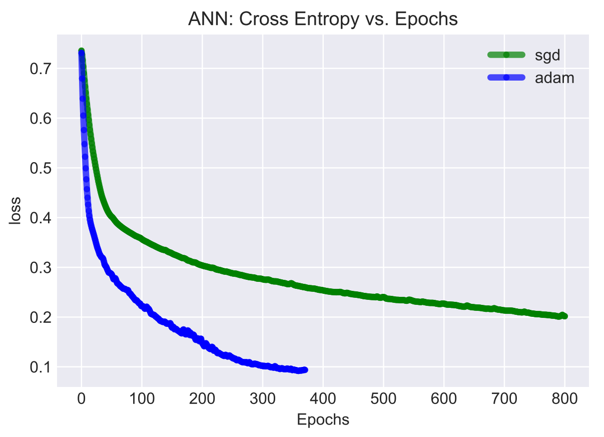

Fig. 30. Loss curve [heart disease]

Table II shows the results of varying the number of layers and neurons in ANN. The training and cross-validation accuracies, and training and testing(single split) time in milliseconds are recorded and shown in table II. The table is sorted in ascending order of the number of parameters in the model (weights and biases) which indicates the complexity of the model. Interestingly, the table shows that the least complicated model with only one hidden layer and 10 neurons was able to achieve the highest cross-validation accuracy. This is likely because the model was not too complex to overfit the training data, and not too simple that it was able to generalize to new data. It can be also concluded from table II that more layers and neurons are computationally expensive and do not always provide a better generalization to new data. In addition, it was interesting to observe that the model with similar number of parameters but more layers take less time in both training and inference. This can be noticed by comparing (500,) with (100,100).

TABLE II
HEART DISEASE DATASET:LAYER AND NEURONS

Number of layers	**n_params**	**Accuracy %**		**Time [ms]**	
		Training	***Cross-valid***	***Train***	**Test**
(10,)	241	95.9	83.8	1007	1.5
(10,10)	351	98.8	81.8	971	1.4
(100,)	2401	98.8	82.1	2607	1.6
(500,)	12001	99.2	82.8	6618	1.9
(100,100)	12501	100.	82.1	2370	1.6
(100,100,100)	22601	100.	81.4	1440	2.0
(500,500)	262501	100.	81.8	19857	10.9

Figure 31 shows the effect of learning rate on the cross-validation accuracy and different optimization solvers. It is clear from figure 31 that choosing an appropriate learning rate is very crucial for ’sgd’ and ’adam’ solvers, while LBFGS solver is not affected. A too low value of learning rate for ’adam’ and ’sgd’ will result in very slow learning. On the contrary, a too high value of learning rate tends to cause high oscillating in changing the weights which usually results in instability in the training, and hence lower performance. Although ’adam’ is a very common default choice for many machine learning practitioners, it is very important to try different solvers to obtain the best performing model.

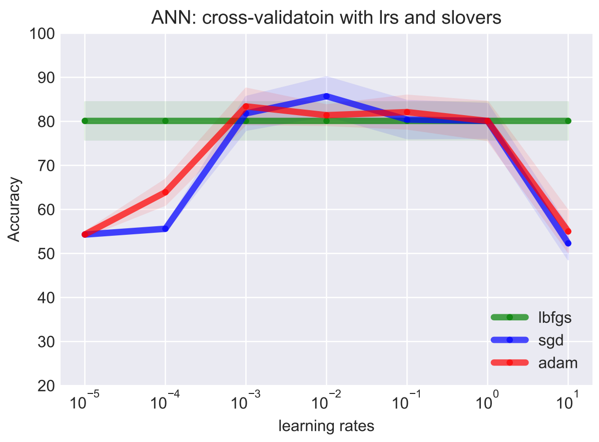

Fig. 31. Effect of learning rate [heart disease]

2) *Letter Recognition Dataset:* The learning curve of the ANN model when using the Letter Recognition dataset is shown in figure 32. The model is configured with two hidden layers with 500 neurons each. In general, the performance of the model increased with adding more samples. However, it is interesting to see that in the end, more samples have caused

the model performance to decrease. This could suggest that a more complicated model is needed if the number of samples continued to increase.

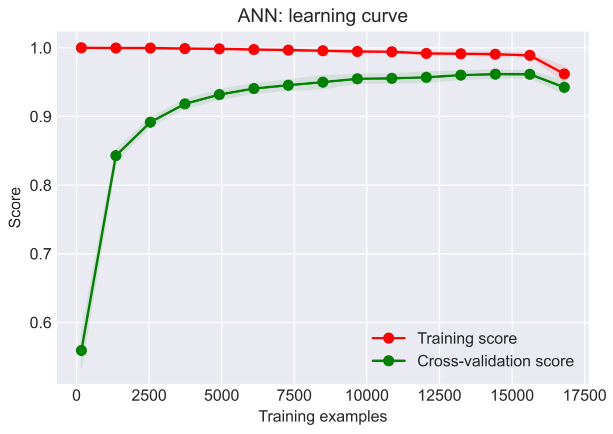

Fig. 32. ANN learning curve [Letter Recognition]

The loss curve of ANN during training is shown in figure 33. The loss curve in both dataset have the same overall trend. It could be seen from figure 33 that Adam optimizer shows some instability during training. However, it is interesting to observe that Adam optimizer still converges with lower loss than SGD.

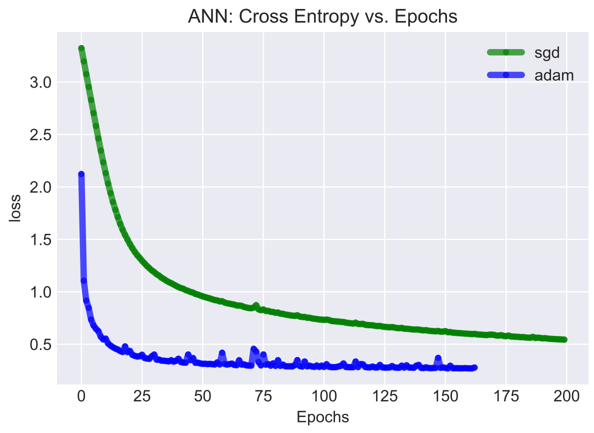

Fig. 33. Loss curve [Letter Recognition]

The effect of the learning rate on the cross-validation accuracy and different optimization solvers is shown in figure 34. It can be observed that LBFGS optimizer is independent of the learning rate, and usually gives decent results. On the contrary, Adam and Stochastic gradient descent is very sensitive to the learning rate. A too low learning rate can cause very slow learning and might need many epochs to get decent performance. On the other hand, too high a learning rate can lead to instability in the training, and hence low performance.

Another important parameter for the ANN is the number of neurons and layers. It can be seen from table III that less

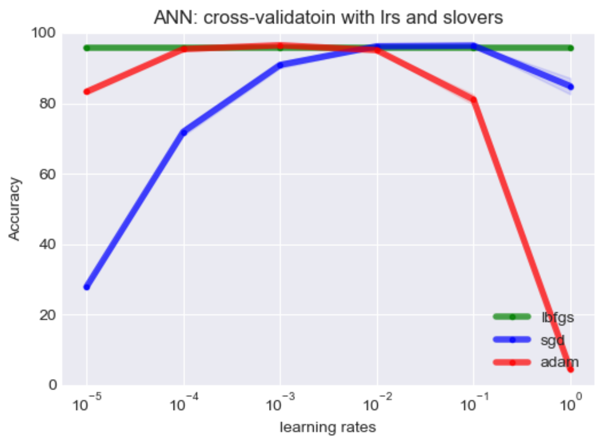

Fig. 34. Effect of learning rate [Letter Recognition]

complex ANN models such as (10,) and (10,10) produced good but lower performance than the more complex models. This indicates that the Letter Recognition data is more complex than the Heart Disease dataset, and therefore requires a more complex model. In addition, by comparing (100,100) and (500) we could conclude that adding more layers may produce better accuracy than adding more neurons. However, it is certain that the performance will be saturated after adding a certain number of layers. In addition, the best performing model was the one with 2 layers and 500 neurons each (500,500). However, it has a higher average inference time than the other models, so it might be not the best option for real-time application, although the inference time is still very small per sample.

TABLE III
LETTER RECOGNITION DATASET:LAYER AND NEURONS

Number of layers	**n_params**	**Accuracy %**		**Time [ms]**	
		Training	***Cross-valid***	***Train***	**Test**
(10,)	456	78.9	78.1	25791	3.14
(10,10)	566	80.4	80.7	43771	2.2
(100,)	4326	94.8	94.1	28342	4.3
(100,100)	14426	96.1	96.2	44656	9.4
(500,)	21526	95.5	95.4	36059	21.3
(100,100,100)	24526	96.3	96.1	44656	12.2
(500,500)	272026	96.6	96.4	27497	84.64

IV. CONCLUSION

In this section, above-mentioned algorithms will be evaluated and compared to each other using both accuracy and time complexity. The best model was selected using a grid search approach. The average cross-validation using five-fold was used as the primary indicator for accuracy. It must be mentioned that the average cross-validation accuracy is obtained by taking the mean of the accuracy of five different models. Each model is trained on a unique combination of four out of the five folds, and it was tested on the remaining untouched fold. The average performance of these five models was recorded. The reason why the average cross-validation was used instead of a single test split is that cross-validation usually provides a better measure (less bias) than a single split.

For completeness, each algorithm was also trained again on 75% of the datasets, and was tested on the remaining 25% (never used for training). This single split accuracy is also recorded.

Furthermore, another important criterion that will be evaluated is the time complexity (time spent in training and inference).

A. Heart Disease Dataset

TABLE IV
SUMMARY TABLE FOR HEART DISEASE DATASET

Algorithm Name	**Accuracy %**		**Time [ms]**	
	Avg Cross-valid	***single test split***	***Train***	**Test**
DT	71 ± 5	75.0	2	1
Ada-Boosting	81.1 ± 7.4	84.2	65	8
KNN	79.8 ± 4.7	82.8	1.6	3
SVM	82.1 ± 4.7	81.6	4	2
ANN	83.8 ± 3.3	82.9	877	2

Table IV shows the accuracy and time complexity of the best model of Decision Tree, Ada-Booting, KNN, SVM, and ANN. Selecting the best performing model is application dependent. For example, if accuracy is the most important criterion, and training time is not important, then the ANN will be chosen since it has the highest average cross-validation and also the lowest standard deviation. However, if the training time is also an important factor, then SVM would be a better choice since it has a very similar performance to ANN but a much lower training time. In addition, the inference time (testing) is also important in a real-time application, so in this case, the ANN might be slightly preferred.

B. Letter Recognition Dataset

TABLE V
SUMMARY TABLE FOR LETTER RECOGNITION DATASET

Algorithm Name	**Accuracy %**		**Time [s]**	
	Avg Cross-valid	***single test split***	***Train***	**Test**
DT	83.8 ± 0.6	83.2	0.042	0.002
Ada-Boosting	86.6 ± 0.6	85.9	0.18	0.004
KNN	93.55 ± 0.3	93.0	0.013	0.2
SVM	96.7 ± 0.3	96.4	2.4	4.8
ANN	96.4 ± 0.3	94.0	98	0.073

A similar analysis could be made for the Letter Recognition dataset. One note that should be mentioned is that time complexity is measured here in seconds as opposed to milliseconds. This is because this dataset is much larger than the Heart Disease dataset so it takes a much longer time to train and test. The SVM algorithm seems to be performing best in this dataset but only a bit higher than the ANN. The SVM would be preferred if the training and accuracy are the primary factors, whereas the ANN would be chosen if the model is optimized for inference speed.

REFERENCES

[1] Andras Janosi et al. *Heart Disease Data Set*. 1988. URL: https://archive.ics.uci.edu/ml/datasets/Heart+Disease.

[2] David J. Slate. *Letter Recognition Data Set*. 1991. URL: https://archive.ics.uci.edu/ml/datasets/letter+recognition.