Classification problems chosen

All experiments have been performed on the following datasets provided by UCI Machine Learning:

1. Breast Cancer Wisconsin (Diagnostic) Data Set [1]: Given features computed from a digitized image of a fine needle aspirate (FNA) of a breast mass, the task is to predict the diagnosis of the tissue (benign or malignant). The class distribution is as follows: 62.74% benign (negative), 37.26% malignant (positive). Thus, the dataset is balanced. This dataset is referred to henceforth as dataset 1.
2. Wine Quality Data Set [2]: Given physicochemical properties and sensory data of the Portuguese "Vinho Verde" wine, the task is to predict wine quality. The dataset related to the white variant of the wine was chosen for experiments since it contains a larger amount of data. The labels in the original dataset are the wine quality measured on a scale from 0 to 10. To facilitate analysis, a cut-off of 6 was chosen. Values above it were relabeled as '1' (good) and those below it as '0' (bad). The dataset contains 66.52% positive examples and 33.48% negative examples. Thus, this dataset is also balanced. This dataset is referred to henceforth as dataset 2.

Both datasets are non-trivial as they involve hundreds or thousands of samples and the label is determined by a complicated interplay between the factors.

Analysis

5 classifiers, viz. decision tree, neural network, AdaBoost, SVM and kNN were analyzed. The scikit-learn Python library [3] was used to source the implementations of the classifiers. Hyperparameter tuning was carried out using grid search and 5-fold cross-validation.

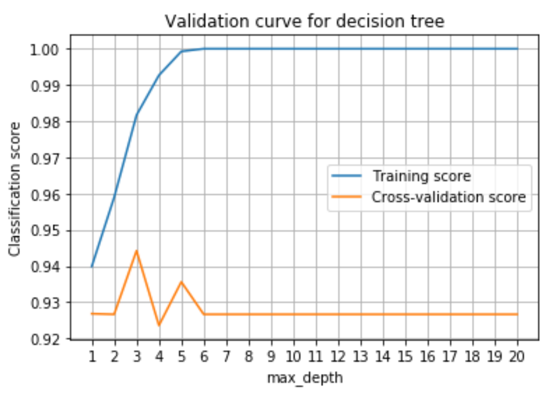
The report is organized as follows: the classifiers are first analyzed individually and their performance is compared in the end.

Decision tree

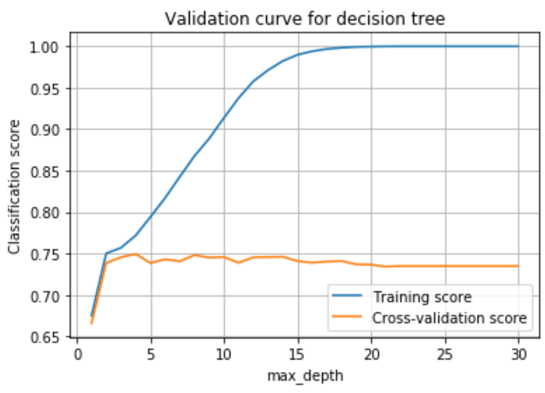
To combat overfitting, the decision tree was pre-pruned by limiting its maximum depth. A tree with a large depth will try to fit the training data exactly and is likely to result in overfitting. The criterion used to split a node is Gini index, a measure of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset.

Validation curves

For low values of max_depth, the tree is excessively pruned and thus suffers from underfitting as both the training and cross-validation scores are low. The training score keeps increasing as max_depth increases. However, the cross-validation score first increases but starts decreasing at a certain point and then plateaus. Thus, at higher values of max_depth, the tree starts to overfit.



Dataset 1

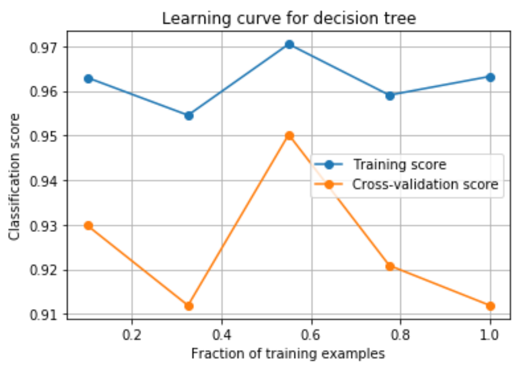


Dataset 2

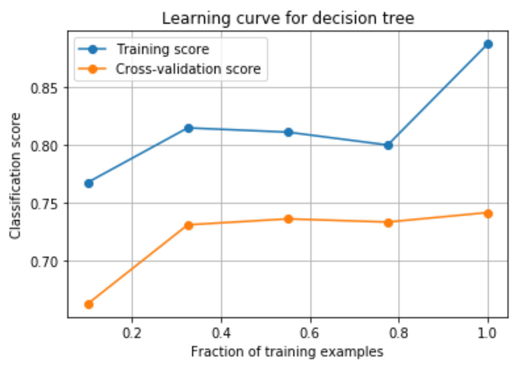
Hyperparameter tuning

After performing grid search, the optimal value of max_depth was found to be 3 and 4 for datasets 1 and 2 respectively, yielding best accuracies of 92.98% and 75.51%.

Learning curves



Dataset 1



Dataset 2

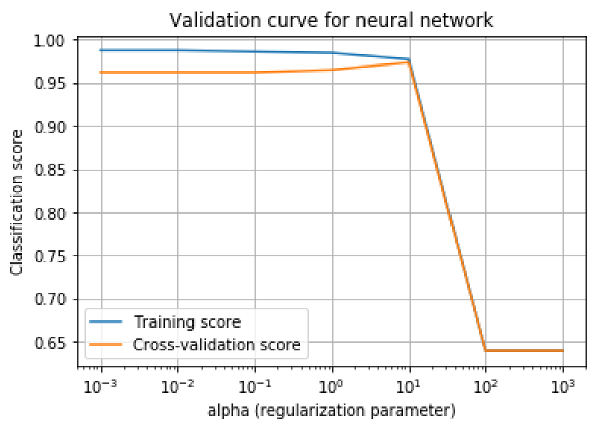
For both the datasets, there is a big difference between the training and cross-validation scores at the maximum data size. This suggests that the classifier is suffering from high variance and will likely benefit from more training data. For dataset 2, both the scores are low indicating that it is also suffering from high bias.

Neural network

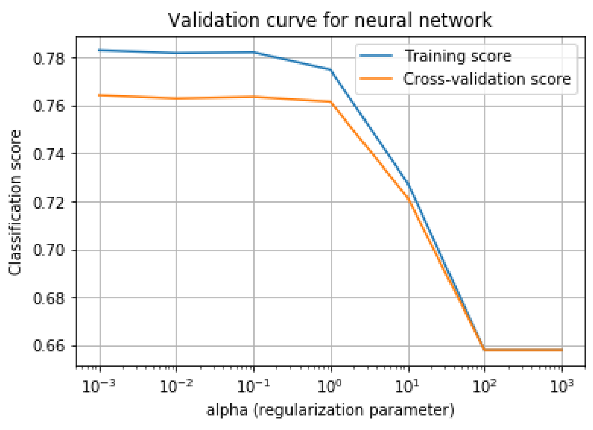
For ease of analysis, a small neural network with 2 hidden layers containing 5 and 2 nodes respectively with ReLU activations was used.

Validation curves

Performance of a neural network is very sensitive to its learning rate and regularization parameter. Validation curves for both the parameters are plotted below.

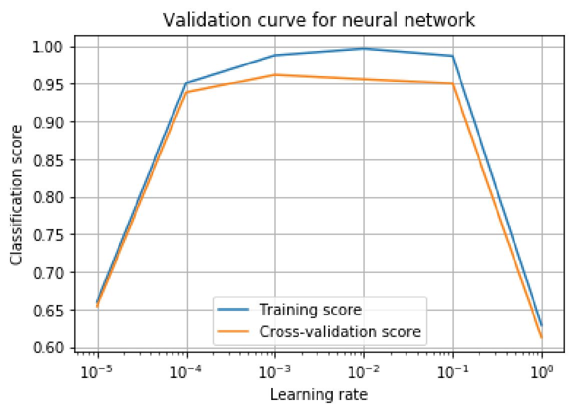


Dataset 1

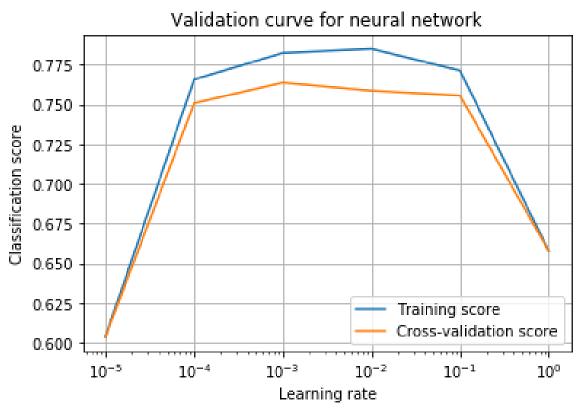


Dataset 2

We can observe when the regularization parameter is too high, the performance is bad because the network suffers from high bias. When the learning rate is too high or too low, the performance goes down because the training cannot converge.



Dataset 1



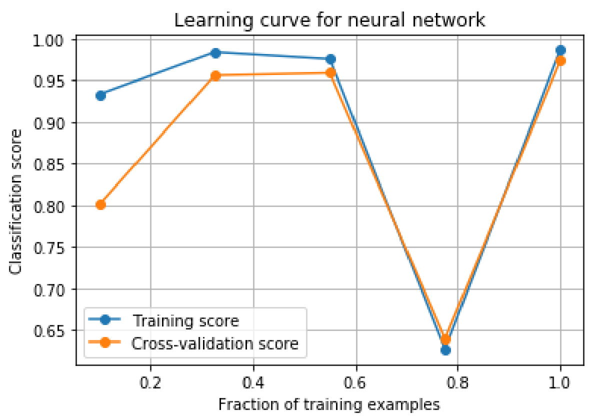
Dataset 2

Hyperparameter tuning

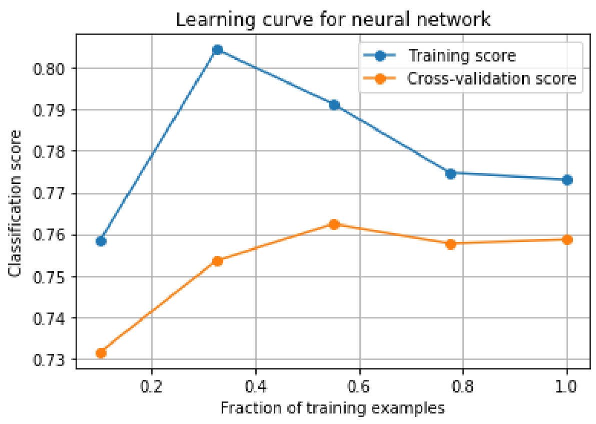
After hyperparameter tuning, the optimal values of the hyperparameters were found to be as follows:

Dataset	Alpha	Learning rate
1	3.16	0.1
2	0.1	0.01

Learning curves



Dataset 1

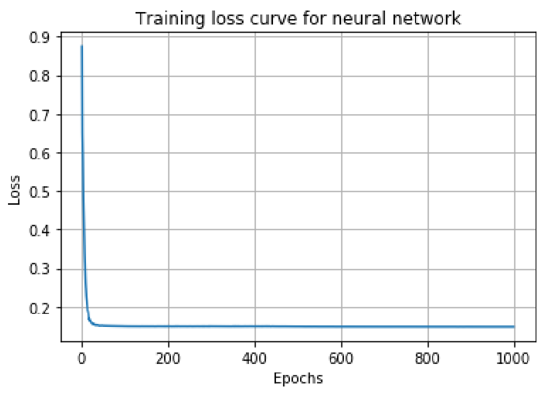


Dataset 2

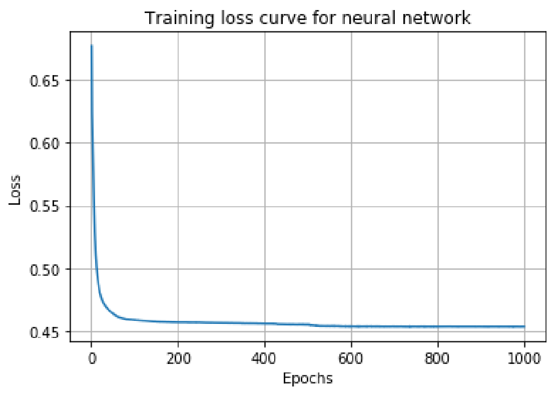
For dataset 1, both training and cross-validation scores are high at the maximum training data size. Hence, there is neither high bias nor high variance and the classifier is performing well.

For dataset 2, the training cross-validation scores converge to a low value, indicating that the network is underfitting. Adding more layers and/or increasing the size of the layers is likely to make the network learn better.

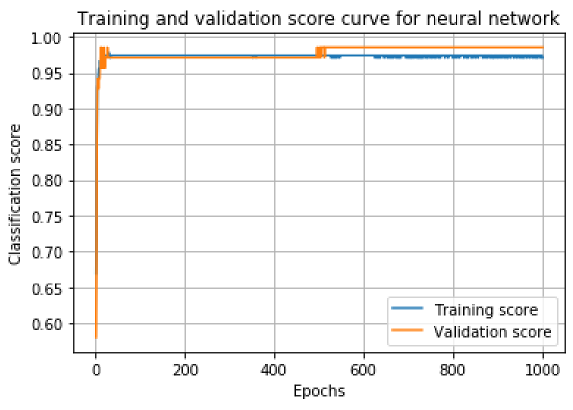
Loss curves



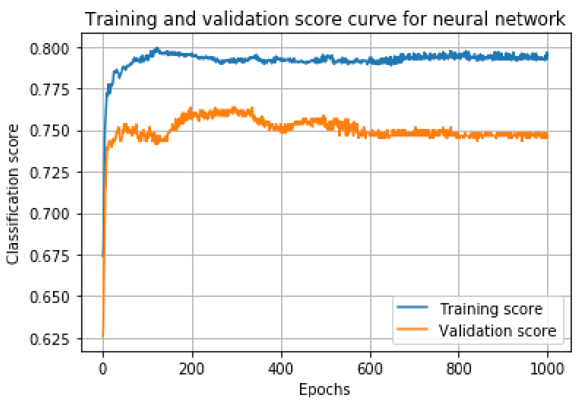
Dataset 1



Dataset 2



Dataset 1



Dataset 2

For dataset 1, training loss decreases with epochs and then converges to a low value. The validation score is high when training has converged, indicating that the network is performing well.

For dataset 2, the training and cross-validation scores increase with more epochs but both of them converge to a low value, suggesting that the network is suffering from high bias (underfitting)

Best accuracies

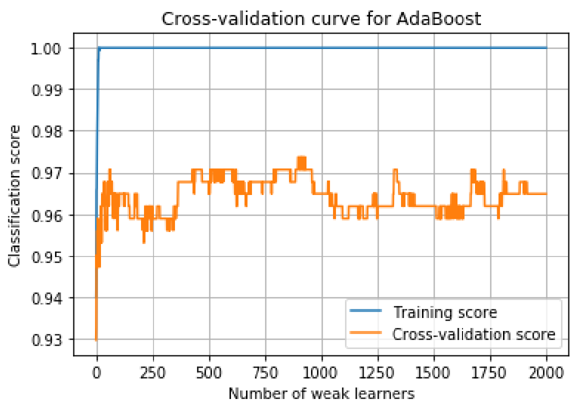
The best accuracy of neural network with optimal hyperparameter values is 98.25% and 76.63% for datasets 1 and 2 respectively.

Boosting (AdaBoost)

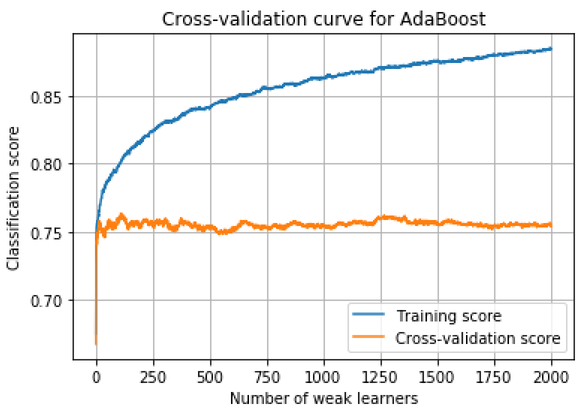
Decision stumps were used as weak learners.

Validation curves

Number of weak learners is a hyperparameter which needs to be tuned. To understand how it affects the performance of the classifier, training and cross-validation scores were plotted wrt the number of learners.



Dataset 1



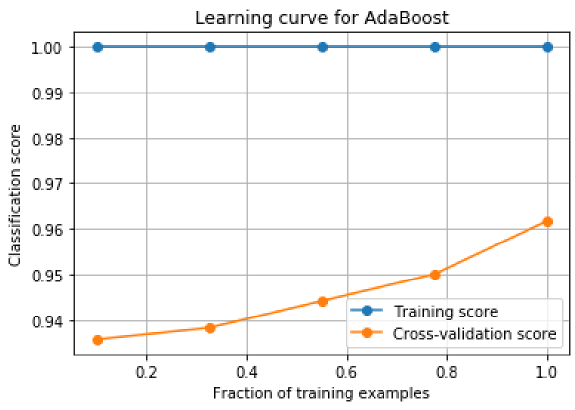
Dataset 2

For both the datasets, the training accuracy first increases with an increase in the number of weak learners and then converges to 1. For dataset 1, the cross-validation score also increases and attains its maximum somewhere between 750 and 1000. For dataset 2, the cross-validation score first increases then plateaus. This might be because of the presence of noise/outliers in dataset 2.

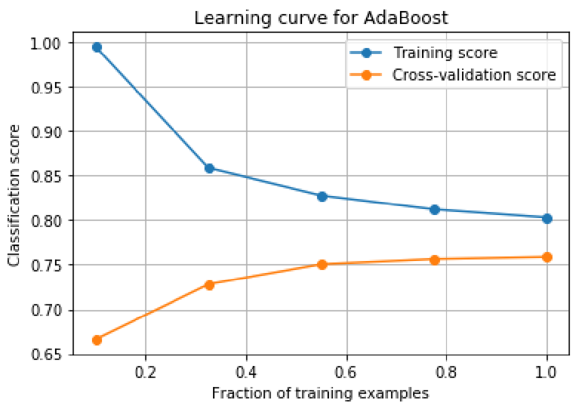
Best accuracy

The number of learners for which cross-validation score was the best was chosen as the optimal value. Using that, the best accuracies were 97.37% and 77.04% respectively.

Learning curves



Dataset 1



Dataset 2

As we can observe in the learning curves plotted above, cross-validation score increases with an increase in the training data size. This can be explained by a property of boosting according to which it can fail to perform well if the given data is insufficient.

For dataset 1, the training score is 1 because boosting learnt the training data exactly after a sufficient number of rounds. Since there is a gap between the training and cross-validation scores at the maximum training data size, more training data is likely to improve cross-validation score further.

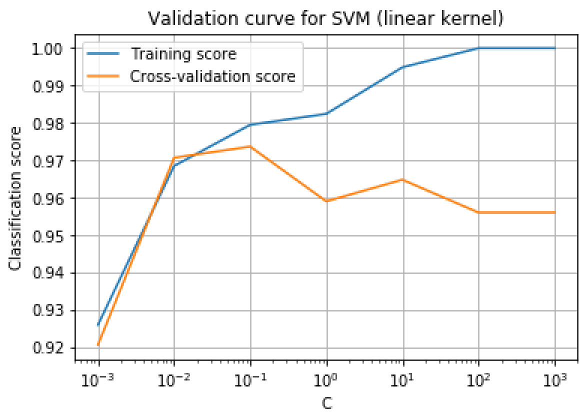
For dataset 2, the training and cross-validation scores converged to a low value. As mentioned earlier, this might be because of the presence of noise/outliers in the data.

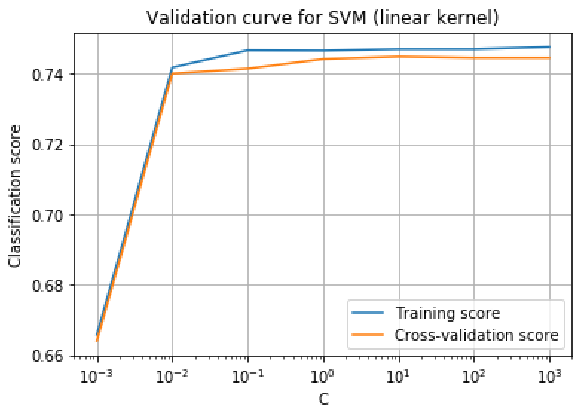
SVM

Experiments were conducted using SVMs with a linear kernel.

Validation curve

The penalty parameter C controls regularization. It was varied to check how it affects the performance.





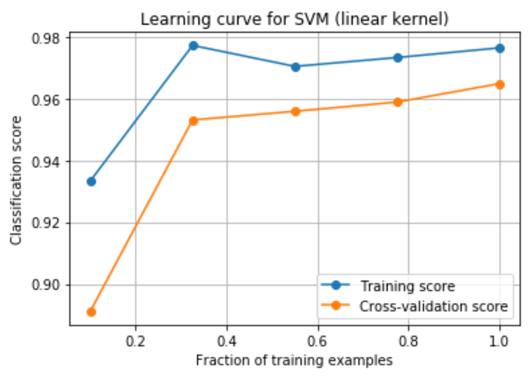
For dataset 1, we can observe that both training and cross-validation scores first increase as C increases. However, after C = 0.1, training score continues to increase while cross-validation score decreases, indicating overfitting.

For dataset 2, both training and cross-validation scores increase as C increases and converge to a low value. Thus, even with weak regularization, the SVM isn't able to learn well which indicates underfitting.

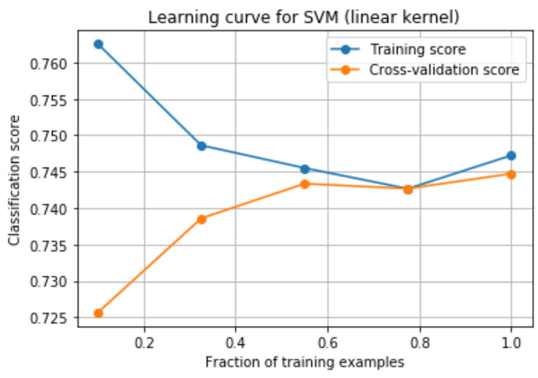
Hyperparameter tuning

After grid search, the optimal values of C were found to be 0.1 and 0.046 with best accuracies 97.37% and 75.71% for datasets 1 and 2 respectively.

Learning curve



Dataset 1



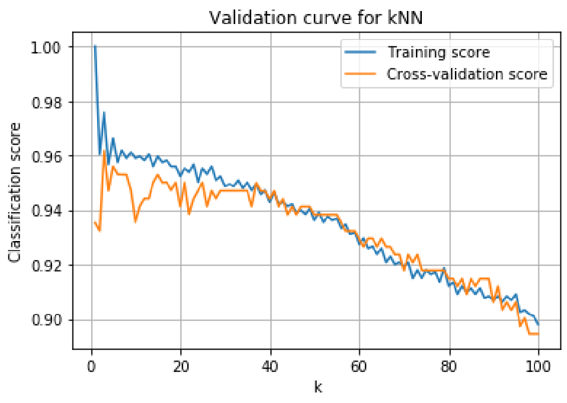
Dataset 2

For dataset 1, we can observe that SVM works well even with a small amount of data. The accuracy increases with the number of training examples and is still increasing when the training data size is at its maximum. Cross-validation accuracy hasn't converged to the training accuracy, indicating high variance. This suggests that test accuracy may improve further if more training data is added.

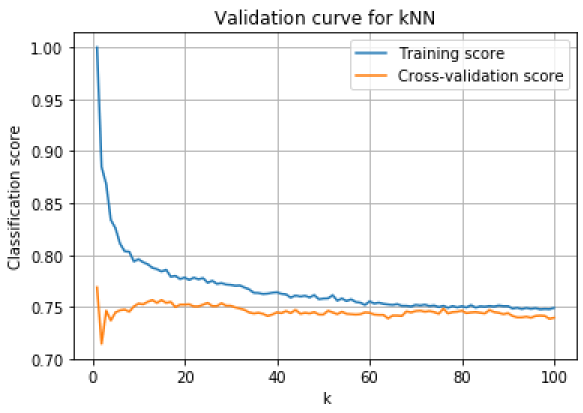
For dataset 2, the training and cross-validation error converge to a low value indicating underfitting. This might be because the data is not linearly separable and a non-linear kernel might improve performance.

kNN

Validation curve



Dataset 1



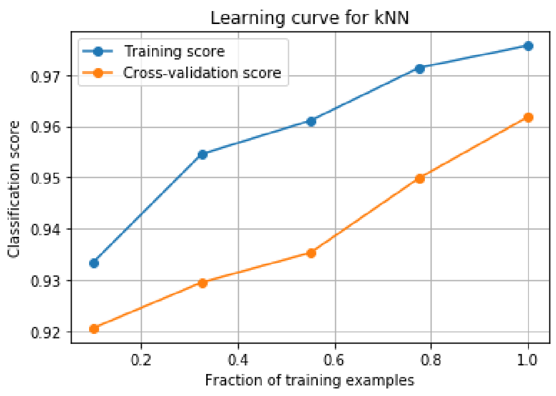
Dataset 2

For low values of k , training score is high whereas cross-validation score is low, indicating that the model is suffering from high variance (overfitting). As k increases, training score drops and cross-validation score increases. After a certain point, both the scores start dropping because of high bias (underfitting).

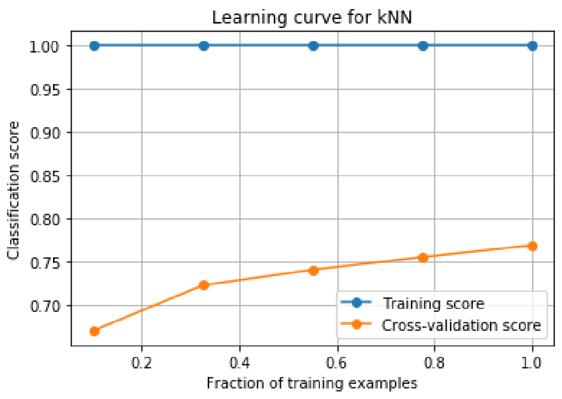
Optimal model

The optimal value of k is 3 and 1 with best accuracies 97.81% and 76.99% for the two datasets respectively.

Learning curve



Dataset 1



Dataset 2

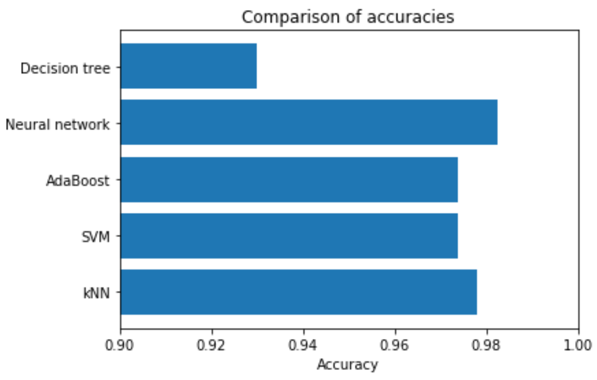
For dataset 1, both training and cross-validation score increase with an increase in the number of training examples. As the training score is much greater than the validation score for the maximum training data size, adding more training data will most likely increase generalization as the classifier is suffering from high variance.

For dataset 2, the training score is always 1 since k = 1. The cross-validation score increases with an increase in the number of training examples but is still much lower than the training score at the maximum data size. This indicates that the classifier is overfitting and more training data will help it generalize.

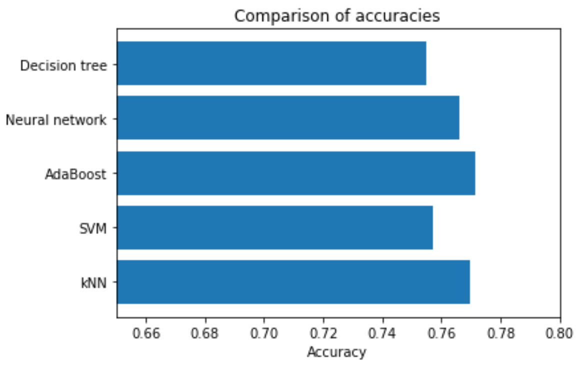
Comparing the classifiers

The 5 classifiers were compared wrt accuracy, training time and inference time.

Accuracy



Dataset 1

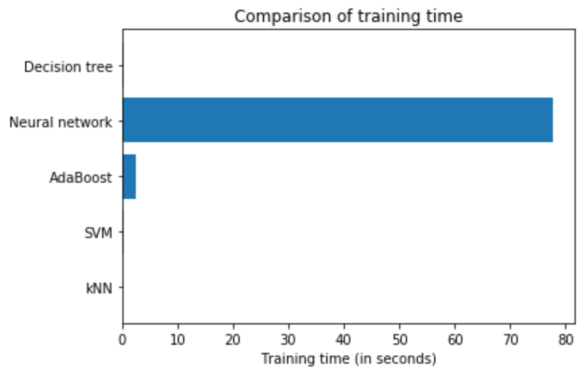


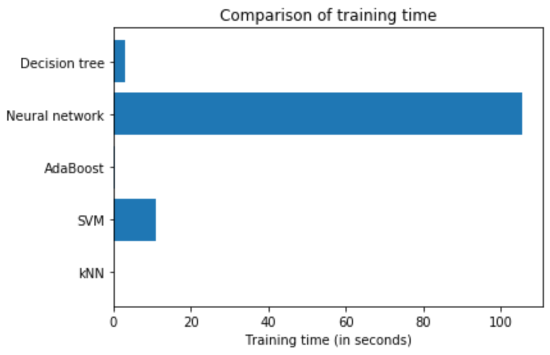
Dataset 2

Dataset 1: Decision tree has the lowest test accuracy because it is the least expressive among the 5 models. Neural network is a powerful learning model and has the best accuracy. It is surprisingly followed by kNN. Adaboost and SVM have the least accuracy among the models which is counter-intuitive. One possible explanation is the amount of data --- the learning curves show that they can potentially improve if more training data is available.

Dataset 2: AdaBoost gives the best accuracy followed by kNN and neural network. Decision tree and SVM give similar accuracies and they are the lowest among the 5. However, all the accuracies are quite low. One possible reason is that the current models aren't complicated enough and underfit with the exception of kNN. Other possible reasons include the presence of noise/outliers.

Training time



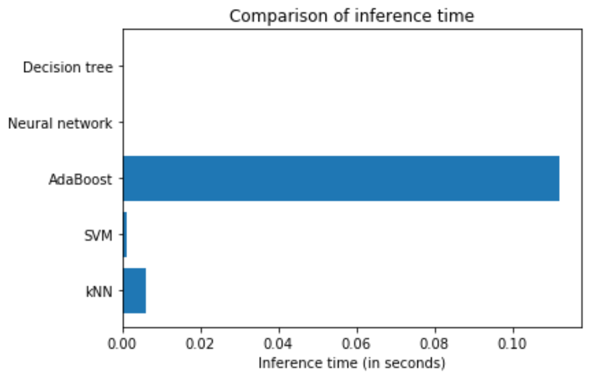


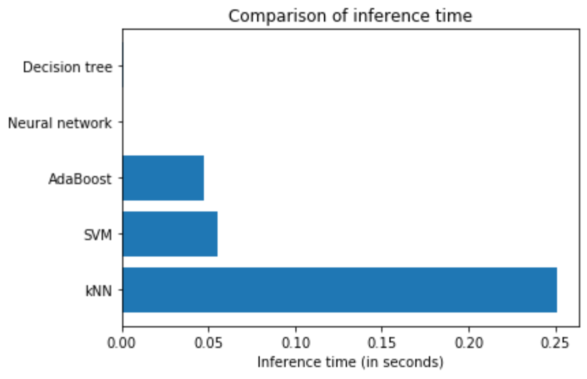
Dataset 1

Dataset 2

Decision tree and kNN take a trivial amount of time to train. For the given dataset, the depth of the decision tree is low due to pre-pruning which is why training time is low. Training kNN is trivial since it just involves storing the training data. Training SVMs involves quadratic optimization which is faster than iterative algorithms such as neural network backpropagation or boosting. Neural network is the slowest as gradient computations in backprop are computationally expensive. The amount of time needed to train AdaBoost depends on the number of weak learners (or in other words, the number of boosting rounds). For dataset 1, the number of boosting rounds is high which is why training takes slightly longer.

Query time





Dataset 1

Dataset 2

Querying a decision tree involves tree traversal. For both the datasets, tree depths are low and hence, querying is fast. Querying a neural network is also fast as once the weights are computed, the forward pass is not computationally expensive. AdaBoost is the slowest for dataset 1 because of the high number of learners. Linear SVMs are quite fast during inference as they just have to compute the value of a linear function. kNN inference is the slowest as it involves distance computation wrt each point in the training data.

References

[1] Street, W. N., Wolberg, W. H., & Mangasarian, O. L. (1993, July). Nuclear feature extraction for breast tumor diagnosis. In Biomedical Image Processing and Biomedical Visualization (Vol. 1905, pp. 861-871). International Society for Optics and Photonics.

[2] Cortez, P., Cerdeira, A., Almeida, F., Matos, T., & Reis, J. (2009). Modeling wine preferences by data mining from physicochemical properties. Decision Support Systems, 47(4), 547-553.

[3] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. Journal of machine learning research, 12(Oct), 2825-2830.