Yidan Long
V00898887
SENG 474 Assignment 1 Report

Part 1 Second Classification Problem Description

The second classification problem that I chose is about diabetes. This dataset includes the following eight features:

-Pregnancies - past pregnancy history (measured in the number of pregnancies)
-Glucose - glucose level after the meal (measured in mg/dl)
-Blood Pressure - diastolic blood pressure (measured in millimeters of mercury)
-Skin Thickness - subcutaneous tissue thickness (measured in mm)
-Insulin - Insulin level (measured in mlU/L)
-BMI - body mass index
-Diabetes Pedigree Function - occurrence and appearance or phenotypes of diabetes from one generation to the next
-Age - age of sample patients
-Outcome - 1 indicates diabetes, 0 indicates no diabetes

I find this topic especially interesting among the others because two of my family members had suffered from diabetes. I have paid close attention to this topic for a long time. After doing lots of research on this common disease among elders and buying them several diabetes-related health care products overseas, diabetes has become one of my concerns for recent years. Besides the basic knowledge I have gained from the research, I am worried about the main causes of diabetes, and what early symptoms we should pay attention to. Also, since diabetes topic is very likely as the provided heart disease dataset, I may create good comparisons of these two problems.

The reason I think this dataset is non-trivial is this dataset has well-chosen, a proper number of features and all the data are real numbered which saved lots of adjustment work. After a deep-down look at this dataset, I found this dataset is very well organized and reasonable. Also, it has large enough data which is very helpful for me to perform the decision tree, random forest, and neural networks methods.

The reason that this dataset makes a good comparison of three methods is because this dataset has:

- Reasonable data: this dataset was taken from the hospital Frankfurt, Germany, the reliability is guaranteed, so the training error and test error are practical and authentic. Therefore, this diabetes dataset with high facticity makes the results clear and reasonable with smaller test error instead of zero test error.

- Binary Classification: the outcome of this dataset is like the given heart disease, which is already included in features in the dataset, with 0 and 1 for either no diabetes or

has diabetes. It is easy for my classification task. After my analysis, this is indeed dataset has ideally test error well below 0.5.

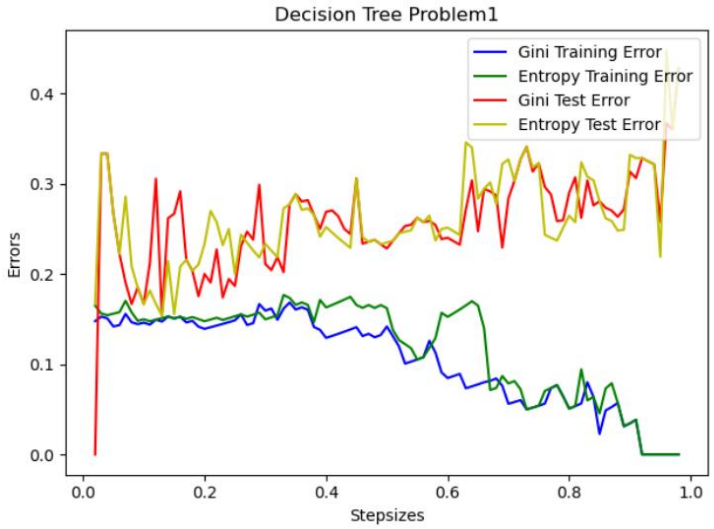
- Large enough data: since I need to split dataset for the training set and test set, to get a fair outcome, it is significant to make sure I learn from a large enough learning set which would reduce test errors.

Part 2 Performance of methods on two problems

1. Problem 1 Heart disease dataset

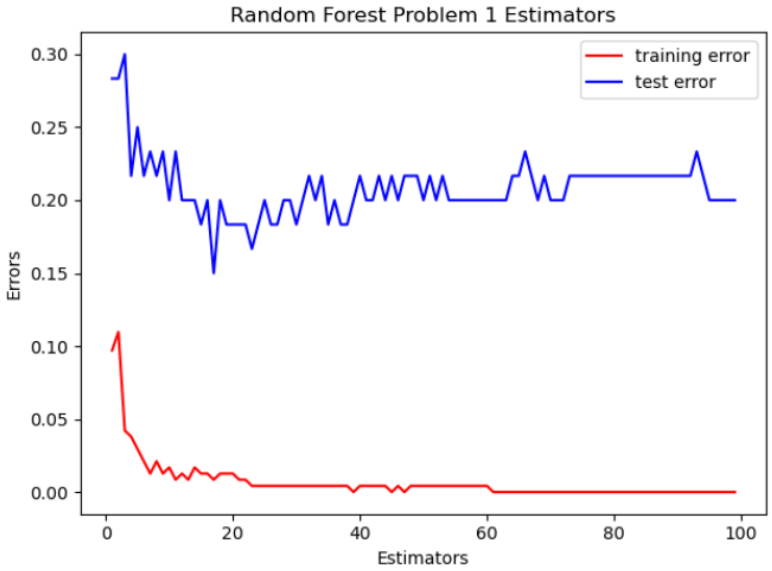
Decision Tree method:

I used the Gini index and Entropy for my decision tree split criterion. The following graph illustrates that Entropy is more concise and better compared to the Gini index since it came with relatively smaller test error with smooth fluctuation in the graph.
The line graph below shows the comparison of training error and test error calculated by the Gini Index and Entropy in the heart disease dataset. The step size I chose is from 0.02 to 0.99 with increment of 0.01. As the increment of step sizes, the training error either using the Gini index and entropy is decreasing. However, the test error is increasing in undulating growth. This graph is generated by DTp1.py file



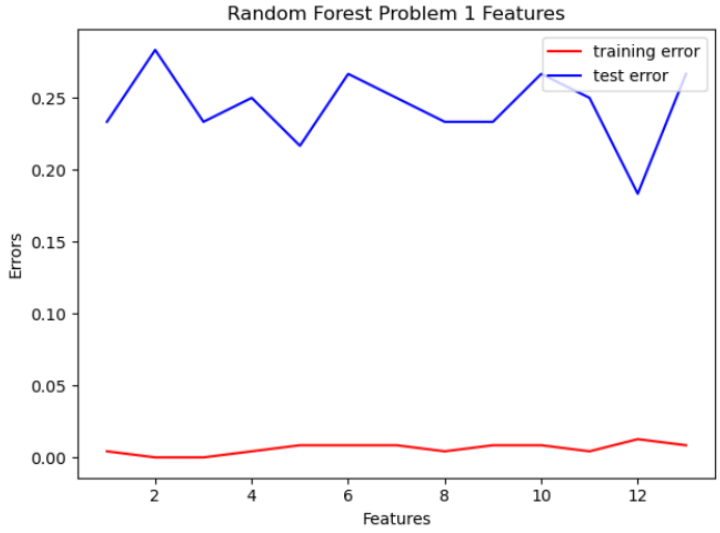
Random Forest method:

I used a different number of estimators (square root of d features), ranged from 1 to 200 with an increment of 1. The line graph below shows that the training and test error changes using a different number of estimators. It is clear to see that when the estimator is 20, the test error is lowest. Hence, I chose 20 estimators when finding the best number of features.
This graph is generated by RFp1_estimators.py



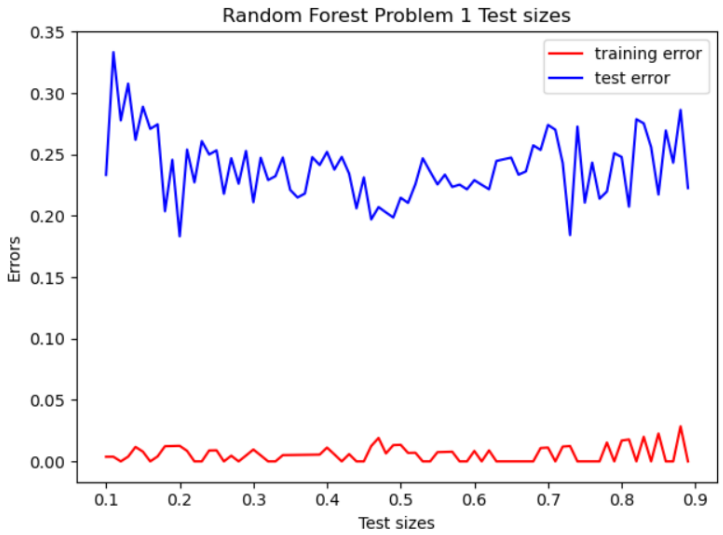
Then I try a different number of features to get the best choice. I set num_features ranged from 1 to max number of features of the data set. The test error is the lowest when number of features is 12.

This graph is generated by RFp1_features.py file



At last, I tried a different number of test sizes instead of 0.2 as recommended. I set test_sizes ranged from 0.1 to 0.9 with an increment of 0.01. The following chart shows that when the test size is about either 0.2 or 0.75, the test errors are the lowest.

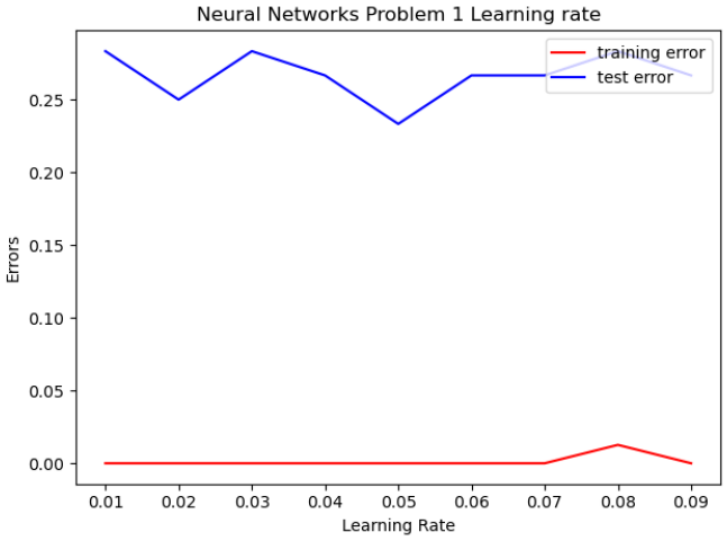
This graph is generated by RFp1_test_sizes.py file



Neural Networks method:

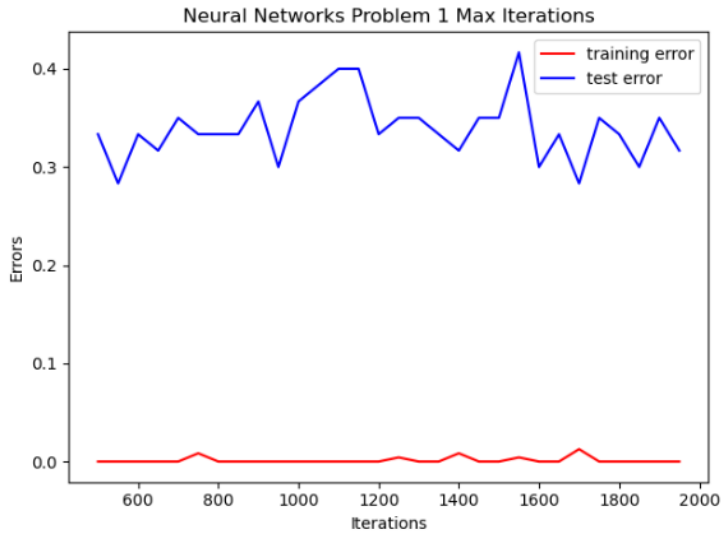
First, I try a different number of learning rates to find the best one with lowest test error, I set learning_rate ranged from 0.01 to 0.1, the increment of 0.01. When the learning rate is 0.05, the test error reached the lowest point, so I chose 0.05 as my learning rate for further operations.

This graph is generated by NNp1_learning_rate.py file



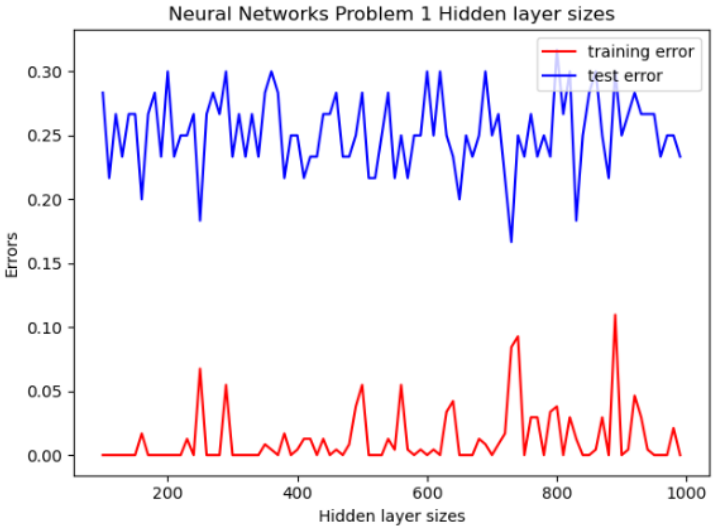
Secondly, I change a different number of iterations of training, I set max_iterations ranged from 500 to 2000 with 50 increments. The following graph indicates test error is lowest at about 1700 iterations, so I chose 1700 iterations of training as my max_iterations when trying a different number of nodes in the hidden layer.

This graph is generated by NNp1_max_iterations.py file



At last, I try a different number of nodes in the hidden layer, I set the hiddenlayer_sizes ranged from 100 to 1000 with an increment of 10. The following graph shows that when there are about 750 nodes in the hidden layer, the test error would be the lowest.

This graph is generated by NNp1_hiddenlayer_size.py file



2. Problem 2 Diabetes dataset

Decision Tree method:

I used the Gini index and Entropy for my decision tree split criterion. The following graph illustrates that Entropy is still more concise and accurate compared to the Gini index since it came with a relatively lower test error.

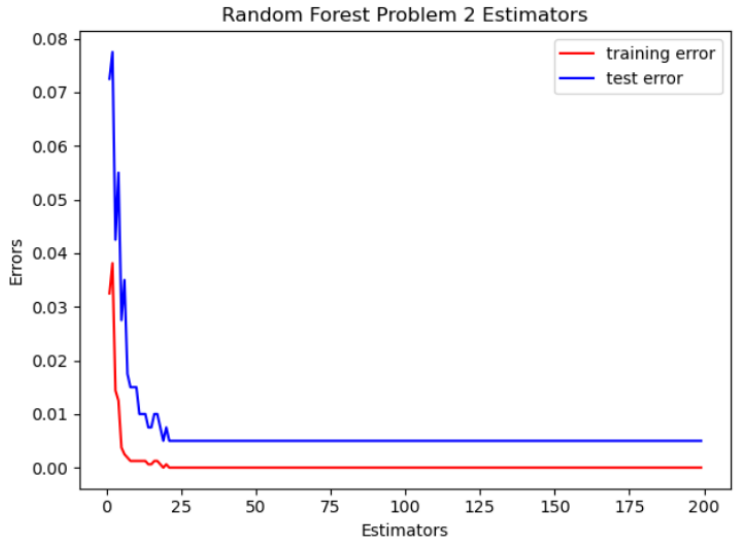
The line graph below shows the comparison of training error and test error calculated by the Gini Index and Entropy in the diabetes dataset. The step size I chose is from 0.02 to 0.99 with a 0.01 increment.

As the increment of step sizes, the training error either using the Gini index and entropy is relatively smooth in 0.02 to 0.6 step size range. However, the Gini and Entropy test errors increasing as the step sizes increase from 0,6. The lowest test error using the Gini criterion is about using a step size of 0.95, and the lowest test error using Entropy is about step size of 0.75.

This graph is generated by DTp2.py file

Random Forest method:

I used a different number of estimators (square root of d features), ranged from 1 to 200 with an increment of 1, same as the heart disease dataset. The line graph below shows that the training and test error changes using a different number of estimators. It is clear to see that when the estimator is bout 25, the test error is lowest. Hence, I chose 25 estimators when finding the best number of features.
This graph is generated by RFp2_estimators.pyh

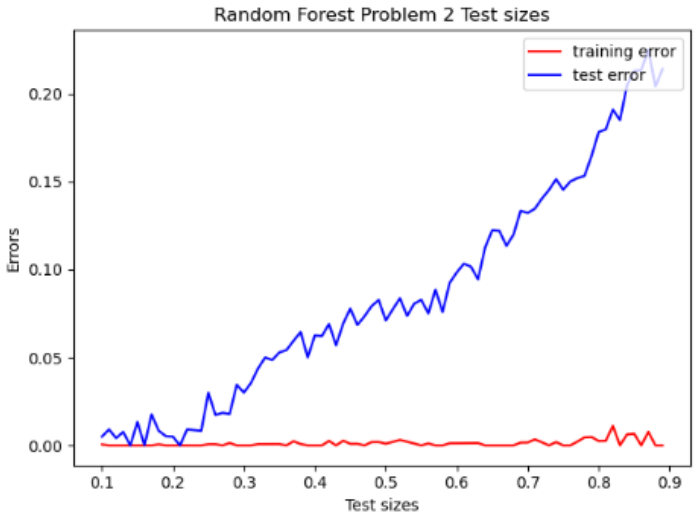


Then I try a different number of features to get the best one, num_features ranged from 1 to the max number of features of diabetes dataset 8 plus one. The test error is the lowest when num_features is either 4 or 6.

This graph is generated by RFp2_features.py

At last, I tried a different number of test sizes instead of 0.2 as recommended. I set test_sizes ranged from 0.1 to 0.9 with an increment of 0.01. The following chart shows that when the test size is about either 0.2 or 0.75, the test errors are the lowest.

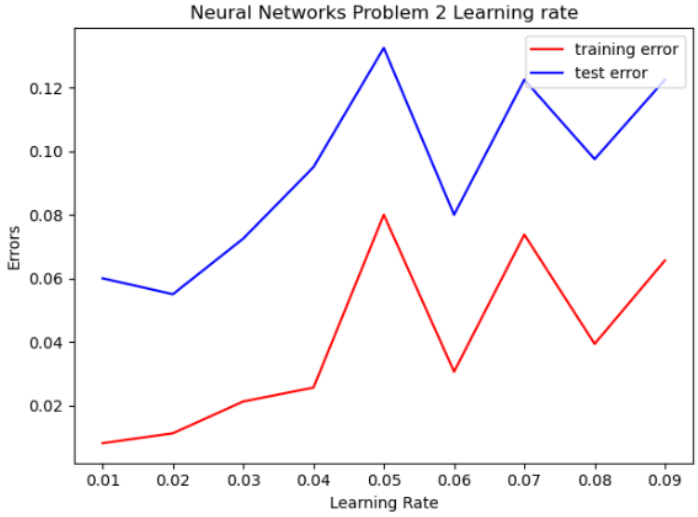
This graph is generated by RFp2_test_sizes.py file



Neural Networks method:

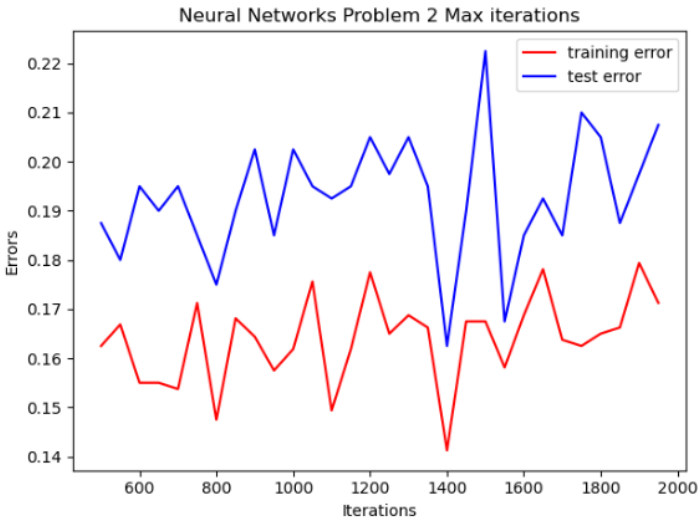
First, I try a different number of learning rates to find the best one with lowest test error, I set learning_rate ranged from 0.01 to 0.1, the increment of 0.01. When the learning rate is about 0.06, the test error reached the lowest point, so I chose 0.06 as my learning rate for further operations.

This graph is generated by NNp2_learning_rate.py file

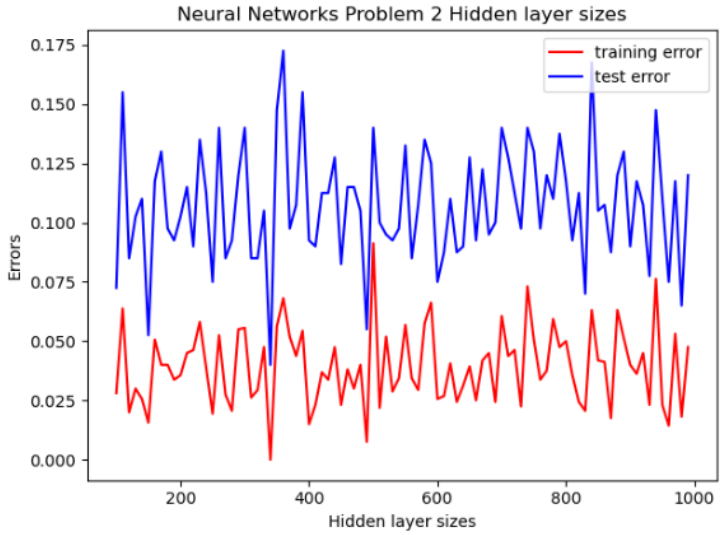


Secondly, I change a different number of iterations of training, I set max_iterations ranged from 500 to 2000 with 50 increments. The following graph indicates test error is lowest at about 1700 iterations, so I chose 1400 iterations of training as my max_iterations when trying a different number of nodes in the hidden layer.

This graph is generated by NNp2_max_iterations.py file



At last, I try a different number of nodes in the hidden layer, I set the hiddenlayer_sizes ranged from 100 to 1000 with an increment of 10. The following graph shows that when there are about 750 nodes in the hidden layer, the test error would be the lowest.
This graph is generated by NNp2_hiddenlayer_size.py file



Part 3 Detailed Analysis of Results

Design choices:

- The split criterion for decision tree: I used the Gini Index and Entropy as my criterion and changed use different step sizes to get test error set. Also, I got errors from the equation: error = 1-accuracy. After analyzing all the graphs using two separate datasets, I concludes that entropy is a more accurate criterion since it has an overall lower test error.

- Random forest: I tried a different number of estimators to find the number with the lowest test error with the same number of features in the heart disease dataset, then I tried a different number of features, using the best number of estimators found before to get test errors.

- Neural network: First I tried different learning rates on both datasets, unsurprisingly the best learning rates are 0.05 and 0.06, so I used 0.05 learning rates for my further operation on both datasets. Secondly, I tried different iterations in the range from 500 to 2000, then I found when using 1400 iterations for heart disease and 1700 iterations for diabetes lead to the lowest test error. Finally, I tried 100 to 1000 nodes in hidden layers for two datasets with their own best choice of learning rate and maximum iterations.

The result from the experiment and Compare method:

- Decision tree adjustment:

I adjusted the heart disease dataset's step size range from 0.02 to 0.99 with an increment of 0.01 to 0.1 to 0.9 with an increment of 0.01 (heart disease dataset), the adjustment illustrates both the Gini test error and Entropy test error are relatively higher as a whole. Therefore, the larger the range of step size, the lower the test error.

- Random forest adjustment:

I adjusted the heart disease dataset's test size range from 0.1 to 0.9 with an increment of 0.01 to 0.1 to 0.99 with an increment of 0.01. However, the test error does not fluctuate a lot, still in the interval of 0.16 to 0.17, so the change of test size will not lead to a large difference when using random forest method.

- Neural network adjustment:

I adjusted the learning rate higher and lower several times, and I conclude that a higher learning rate leads to an unstable growth of iteration graph, the similar lowest points are in the large interval of iterations,(1000 and 1600 iterations with a learning rate of 0.09 in heart disease dataset). Also, a lower learning rate leads to a large interval on iterations (1000 and 1200 iterations with a learning rate of 0.02 in the diabetes dataset).

I also adjusted the number of nodes in hidden layers, when finding the best learning rate and max iterations, I chose to use 10 nodes. As I set the number of nodes to be 100, the max iteration graph has three lowest points (700, 1400, 1600 iterations are having similar test errors but none of them are as low as the one I picked from my learning_rate method). Then I

set the number of nodes to be 1000, it's easy to find the lowest number of iterations and compare with using 10 nodes, the test errors are lower as a whole.

I compared all the test errors generated by these three methods on two datasets, the neural network method is the most accurate in generating test errors compared, second best is random forest method, then the decision tree.

The following chart is the test error obtained from three methods of two datasets.

TESTERROR	Decision Tree	Random Forest	Neural Network
Heart disease dataset	Gini: 0.25~0.28 Entropy: 0.2~0.23	0.178~0.180	0.16~0.17
Diabetes dataset	Gini: 0.28~0.31 Entropy: 0.25~0.28	0.15~0.20	0.03~0.05

How to improve

Ideally, I would try more criteria for decision trees, for example using information gain. A strong relationship between input features will lead the decision tree method to be efficient. For random forest, I would try to use high-quality features in my dataset, or training and evaluating significant features. For neural networks, I would try more algorithms to find the best performing method and check train and test accuracy to avoid overfitting.