Assignment 1 Report

An exploration of various machine learning methods.

Student:

Amaan Makhani

[redacted]

Teacher:

Nishant Mehta

Course:

Seng 474 - Data Mining

1. Second Classification Problem

Description

The second data set I selected was the Breast Cancer Wisconsin (Diagnostic) Data Set [1]. This data set classifies if a tumor is benign or malignant. The data set consists of 357 benign tumor samples and 212 malignant tumor samples. The features contained in the data set are measurements of the cell nuclei present in the image scans. The measured features of the cell include its radius, texture perimeter, area, smoothness (variation in radius lengths), compactness, concavity, concave points, symmetry ratio, and fractal dimension. For these ten measurements their mean, standard error and largest values are computed. This results in the creation of the 30 features. The diagnosis is the target and can take on the classifier of benign or malignant. In order to maintain consistent data types for use in my python code the diagnosis value was changed to have zero represent a malignant cell and one to represent a benign cell.

Importance

Breast cancer diagnosis is of huge importance as it is the best way to combat breast cancer and increase survival rates [2]. The American Cancer Society research shows the decline in survival rates as the severity of stage of the tumor increases [2]:

- Stage 0 – 93% 5-year survival rate
- Stage I — 88% 5-year survival rate
- Stage IIA — 81% 5-year survival rate
- Stage IIB — 74% 5-year survival rate
- Stage IIIA — 67% 5-year survival rate
- Stage IIIB — 41% 5-year survival rate
- Stage IIIC — 49% 5-year survival rate
- Stage IV — 15% 5-year survival rate

This reinforces the need and the difference early diagnosis has on survival rates. This early detection will not only allow us to decrease the fatality rate of breast cancer but assist those women around us who suffer from this cancer. We will likely know many women affected by this because 1 in 8 women are shown to suffer from breast cancer [2]. The reason tumor detection is so difficult is because not all lumps are tumors [2]. It is also rather difficult determine the cells classification by sizes as cell abnormalities can take on different sizes in the human body, resulting in this classification being prone to outliers. Currently, tumors are biopsied in order to be classified. This process can be improved to allow testing of more women and quicker diagnosis. This research is vital, continually ongoing, and dear to my heart as we are all affected by breast cancer.

This data set is a terrific set to be used in model comparison. This is because this data set involves a large number of features which will allow in a sizable number of possibilities for each model. The diversity of the features and its quantities will test the model's ability to split data, classify continuous numerical values, combat overfitting, and as mentioned above the ability to handle outlier data points. Since research is still conducted on this classification there is a wide variety of exploration that can be done in the selection if these models. Due to all these factors this dataset will be an effective comparer of machine learning methods.

References

1. "Breast Cancer Wisconsin (Diagnostic) Data Set", Kaggle.com, 2020. [Online]. Available: https://www.kaggle.com/uciml/breast-cancer-wisconsin-data. [Accessed: 30- Jun- 2020].
2. "The Stages of Breast Cancer & the Importance of Early Detection", Maurer Foundation, 2020. [Online]. Available: https://www.maurerfoundation.org/the-stages-of-breast-cancer-the-importance-of-early-detection/. [Accessed: 30- Jun- 2020].

2. Performance and Tests

Decision Trees

1) This experiment uses the default parameters below:
   - Test sizes between 0.1 and 0.9.
   - No randomness of the estimator.
   - The Gini index (impurity) is used to measure the quality of a split.
   - No max depth of the tree.
   - Two samples required to split an internal node.
   - No maximum features.
   - If the split induces a decrease of the impurity greater than or equal to zero a node will be split.

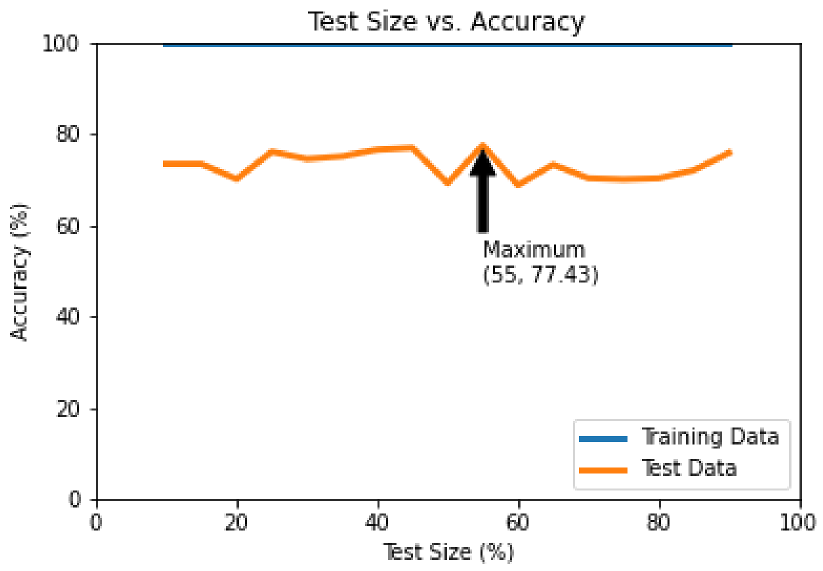

Figure 1: Data set 1 and experiment 1

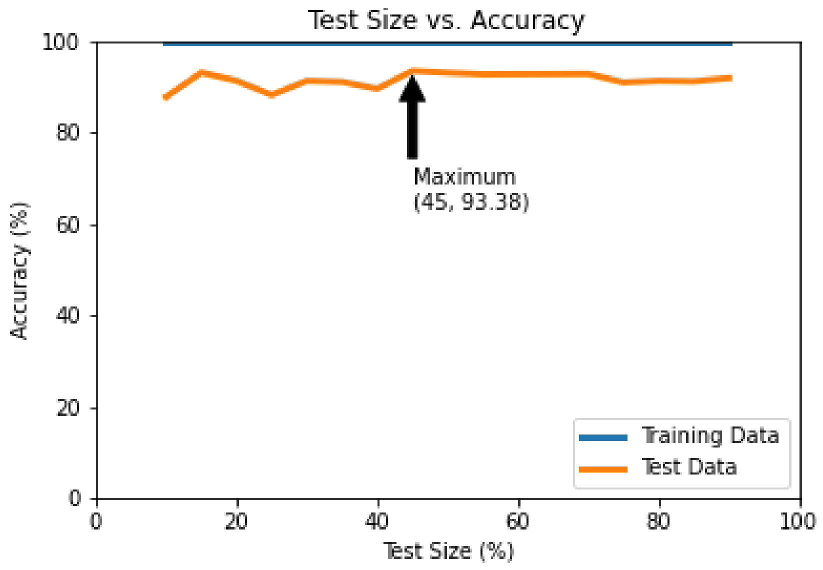

Figure 2: Data set 2 and experiment 1

2) This experiment uses the parameters below:
- Test sizes between 0.1 and 0.9.
- No randomness of the estimator.
- The entropy (information gain) is used to measure the quality of a split.
- No max depth of the tree.
- Two samples required to split an internal node.
- No maximum features.
- If the split induces a decrease of the impurity greater than or equal to zero a node will be split.

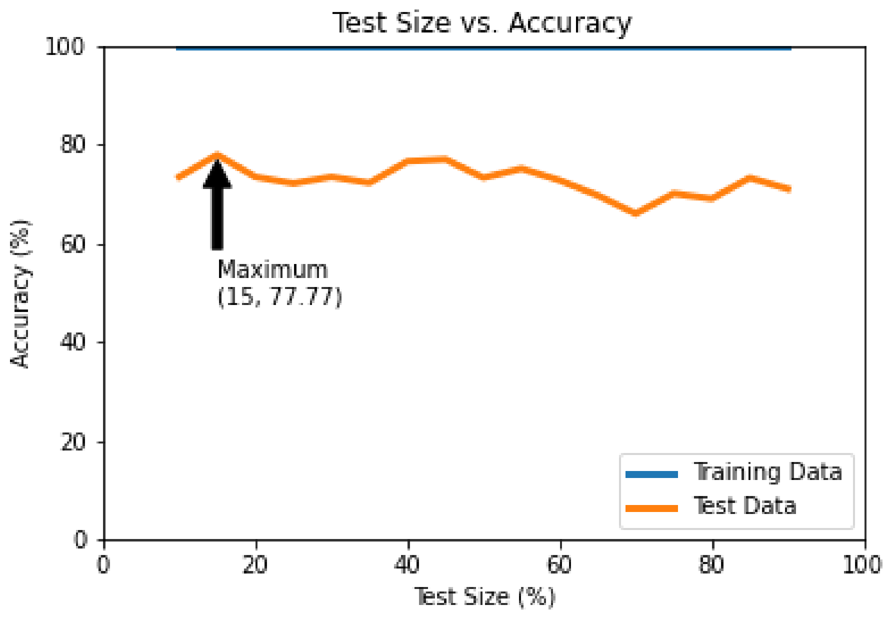

Figure 3: Data set 1 and experiment 2

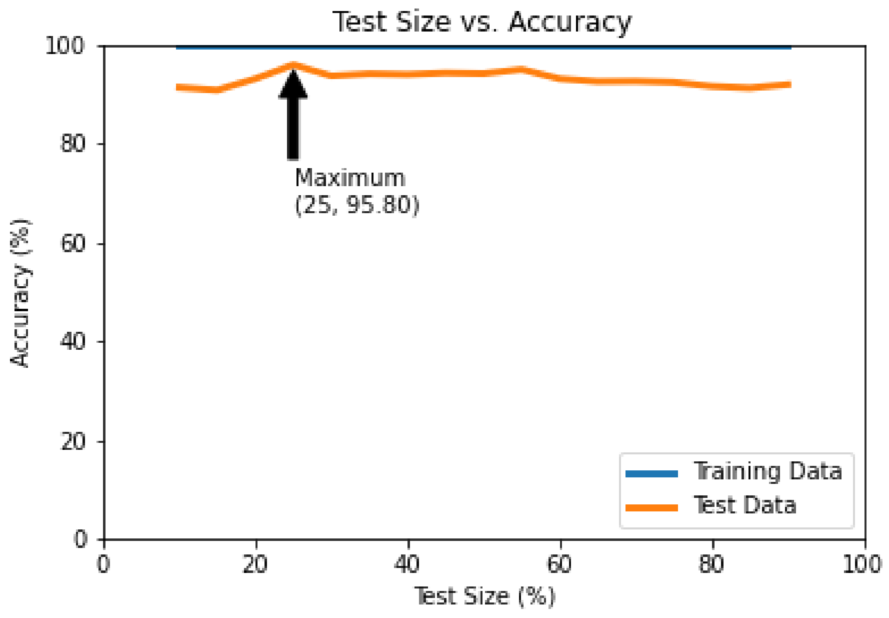

Figure 4: Data set 2 and experiment 2

3) This experiment uses the parameters below:
- Test size of 0.2.
- The Gini index (impurity) is used to measure the quality of a split.
- Varying max depth of the tree from 1-8.
- Two samples required to split an internal node.
- No maximum features.
- If the split induces a decrease of the impurity greater than or equal to zero a node will be split.
- No randomness of the estimator.

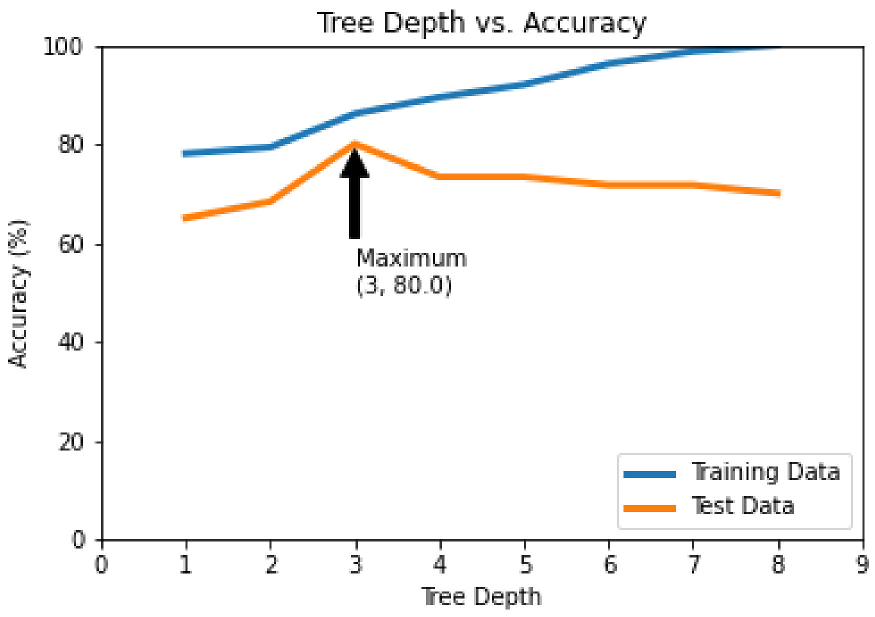

Figure 5: Data set 1 and experiment 3

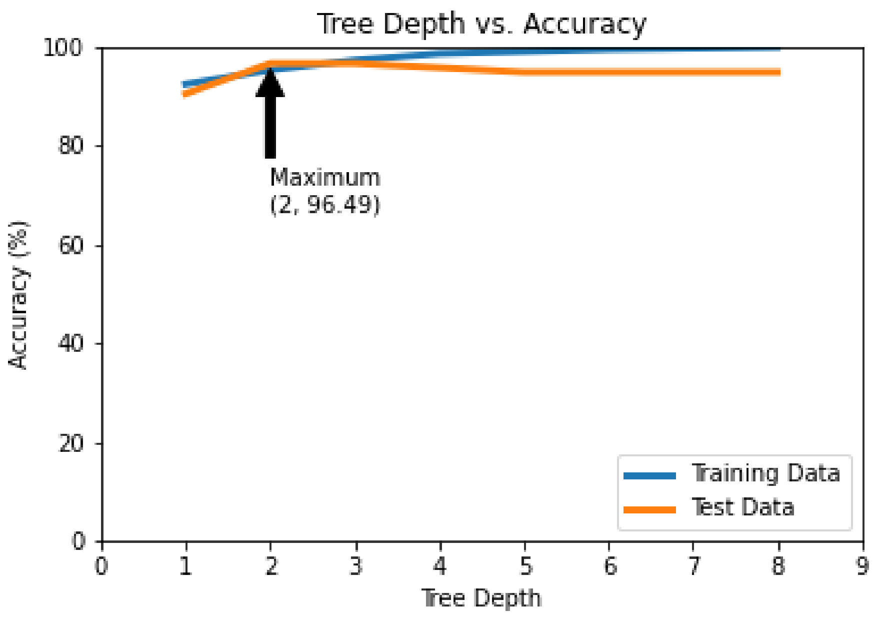

Figure 6: Data set 2 and experiment 3

4) This experiment uses the parameters below:
   - Test size of 0.2.
   - The entropy (information gain) is used to measure the quality of a split.
   - Varying max depth of the tree from 1-8.
   - Two samples required to split an internal node.
   - No maximum features.
   - If the split induces a decrease of the impurity greater than or equal to zero a node will be split.
   - No randomness of the estimator.

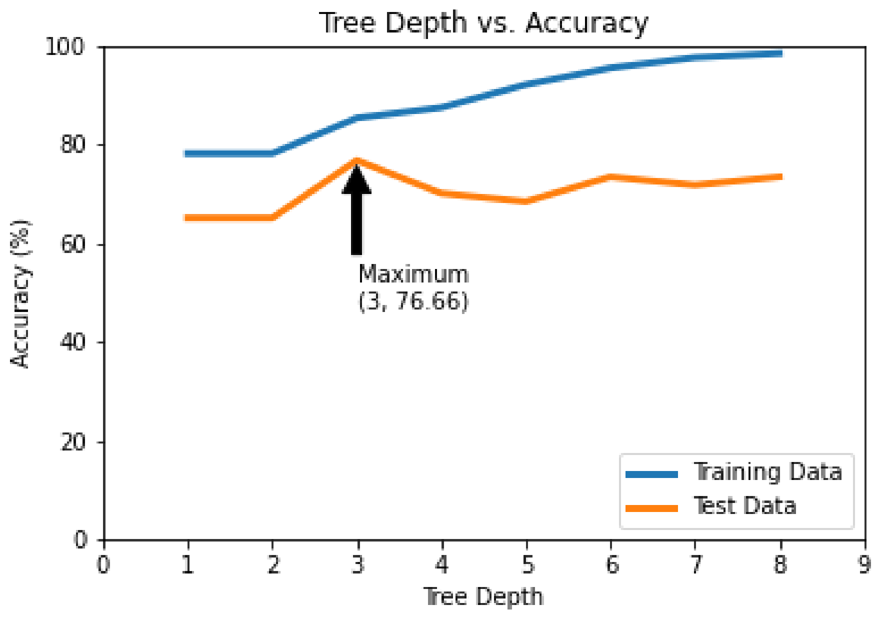

Figure 7: Data set 1 and experiment 4

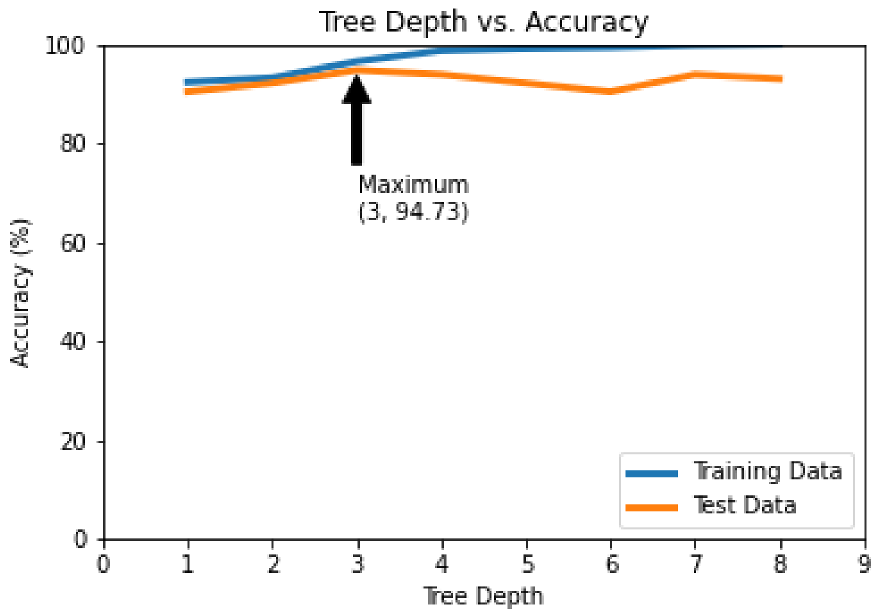

Figure 8: Data set 2 and experiment 4

5) This experiment uses the parameters below:
- Test size of 0.2.
- No randomness of the estimator.
- The Gini index (impurity) is used to measure the quality of a split.
- No max depth of the tree.
- Vary samples required to split an internal.
- No maximum features.
- If the split induces a decrease of the impurity greater than or equal to zero a node will be split.

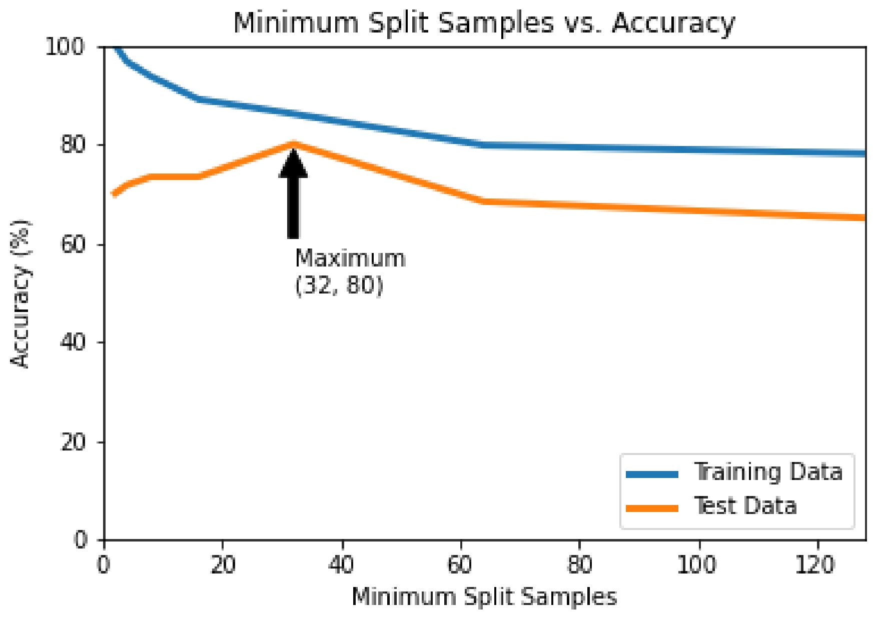

Figure 9: Data set 1 and experiment 5

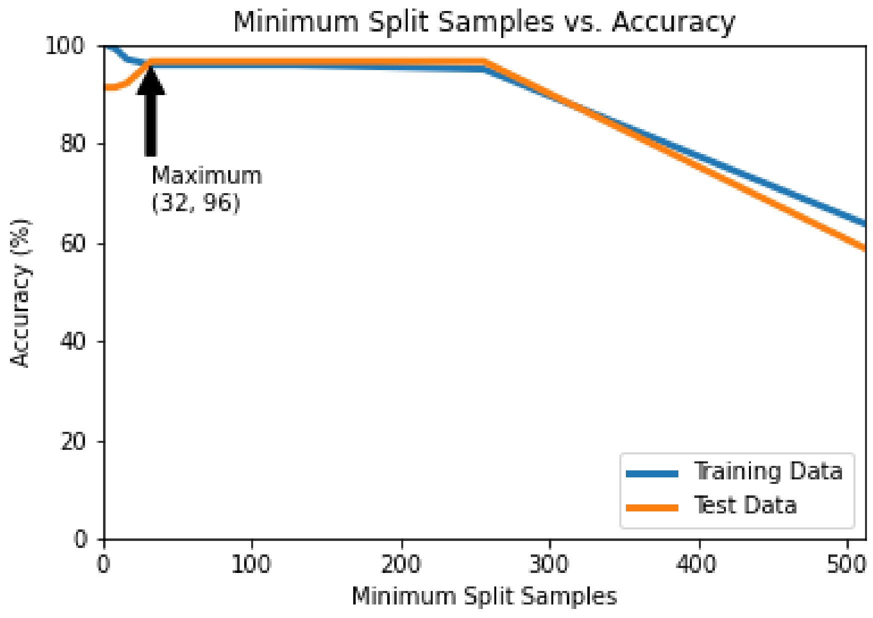

Figure 10: Data set 2 and experiment 5

6) This experiment uses the parameters below:
- Test size of 0.2.
- No randomness of the estimator.
- The entropy (information gain) is used to measure the quality of a split.
- No max depth of the tree.
- Vary samples required to split an internal.
- No maximum features.
- If the split induces a decrease of the impurity greater than or equal to zero a node will be split.

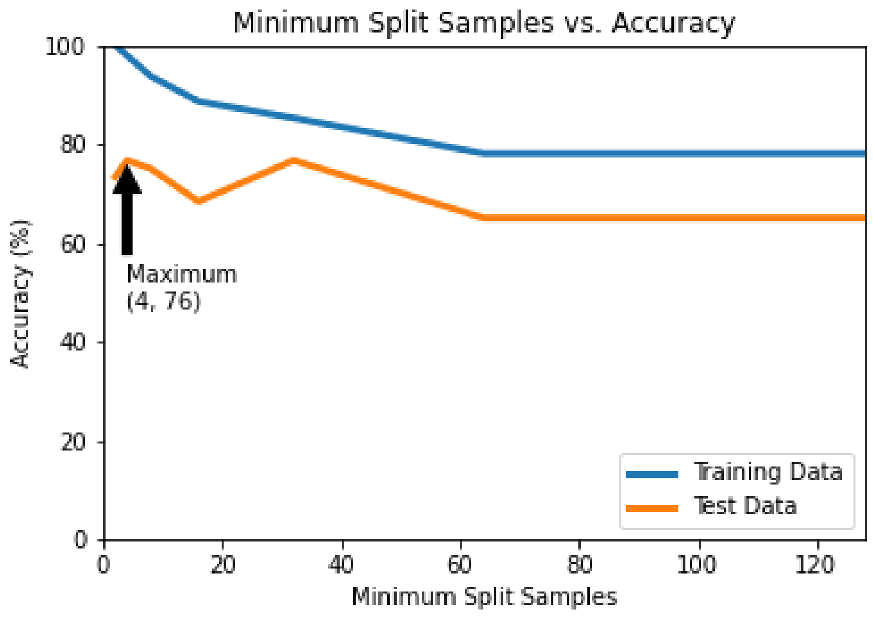

Figure 11: Data set 1 and experiment 6

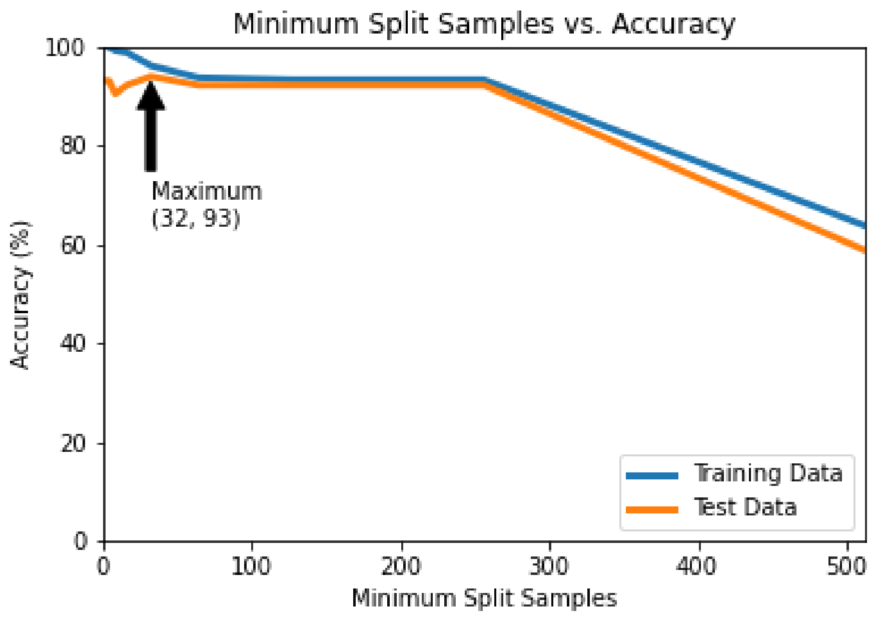

Figure 12: Data set 2 and experiment 6

7) This experiment uses the parameters below:
- Test size of 0.2.
- Varied randomness of the estimator.
- The Gini index (impurity) is used to measure the quality of a split.
- No max depth of the tree.
- Two samples required to split an internal node.
- No maximum features.
- If the split induces a decrease of the impurity greater than or equal to zero a node will be split.

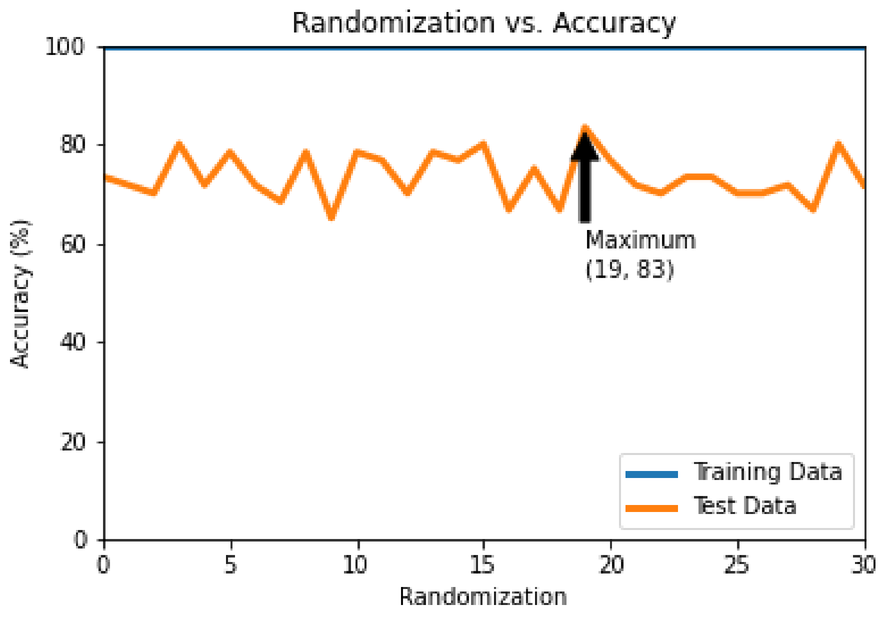

Figure 13: Data set 1 and experiment 7

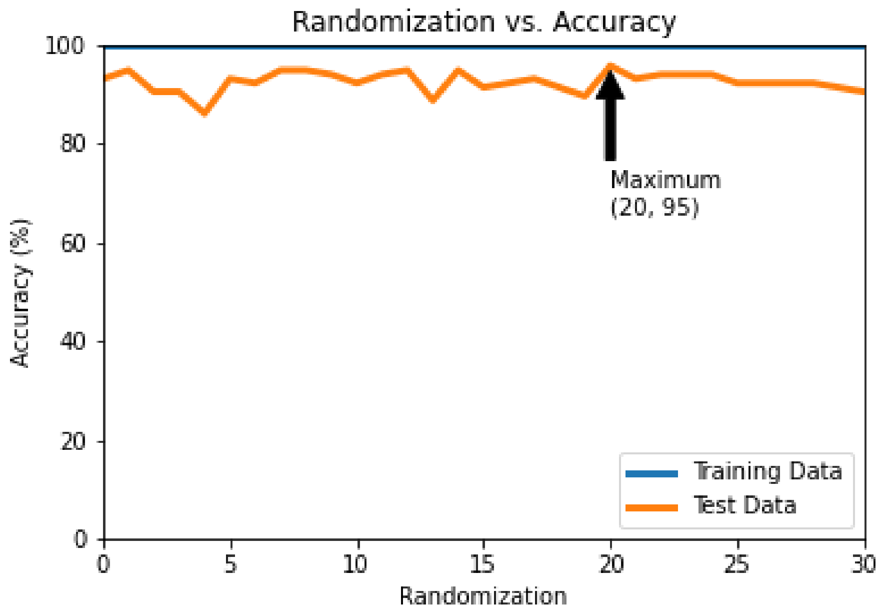

Figure 14: Data set 2 and experiment 7

8) This experiment uses the parameters below:
- Test size of 0.2.
- Varied randomness of the estimator.
- The entropy (information gain) is used to measure the quality of a split.
- No max depth of the tree.
- Two samples required to split an internal node.
- No maximum features.
- If the split induces a decrease of the impurity greater than or equal to zero a node will be split.

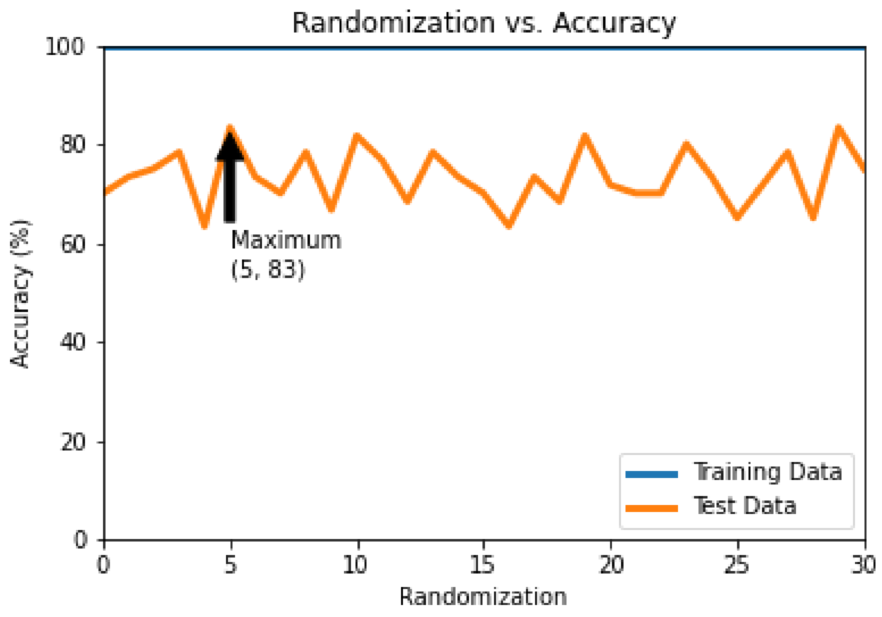

Figure 15: Data set 1 and experiment 8

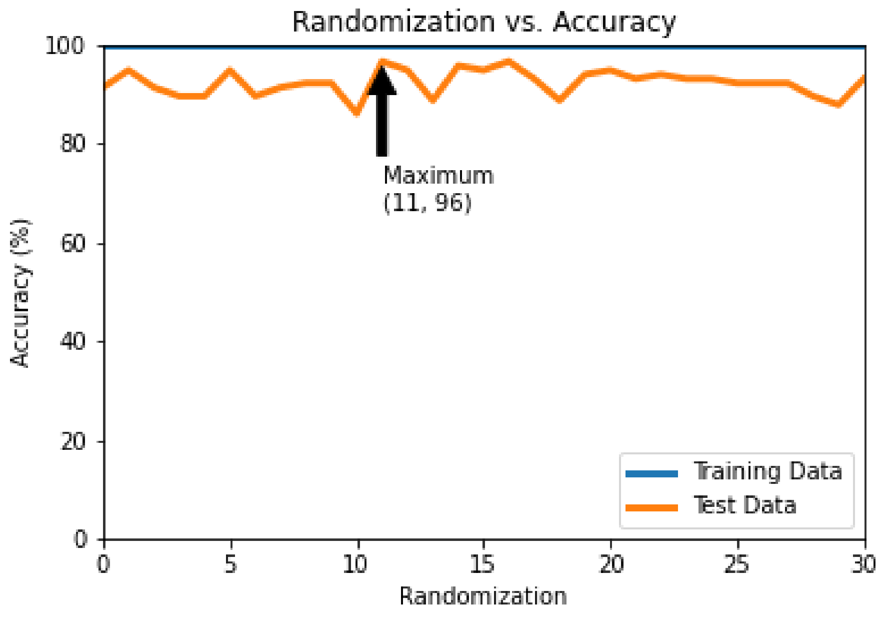

Figure 16: Data set 2 and experiment 8

9) This experiment uses the parameters below:
   - Test size of 0.2.
   - No randomness of the estimator
   - The Gini index (impurity) is used to measure the quality of a split.
   - No max depth of the tree.
   - Two samples required to split an internal node.
   - Varying maximum features.
   - If the split induces a decrease of the impurity greater than or equal to zero a node will be split.

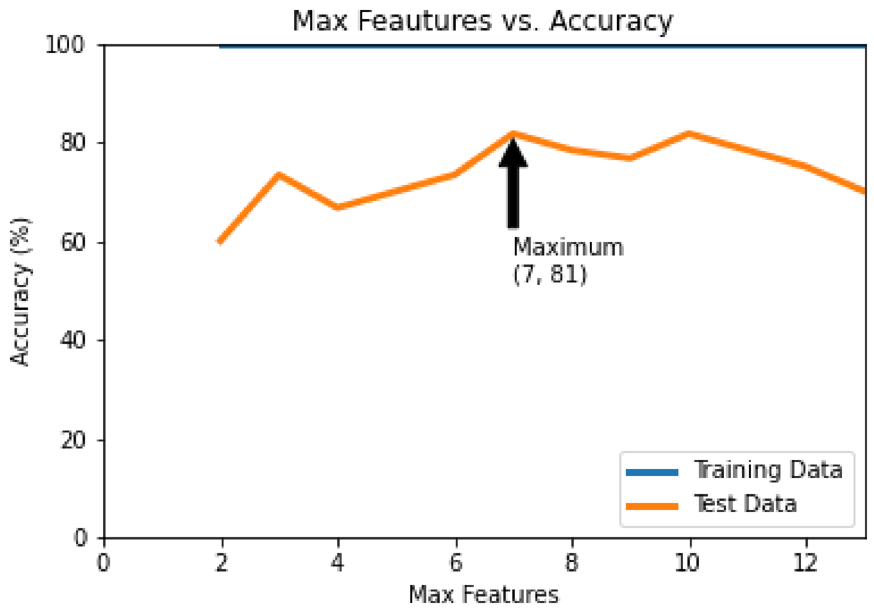

Figure 17: Data set 1 and experiment 9

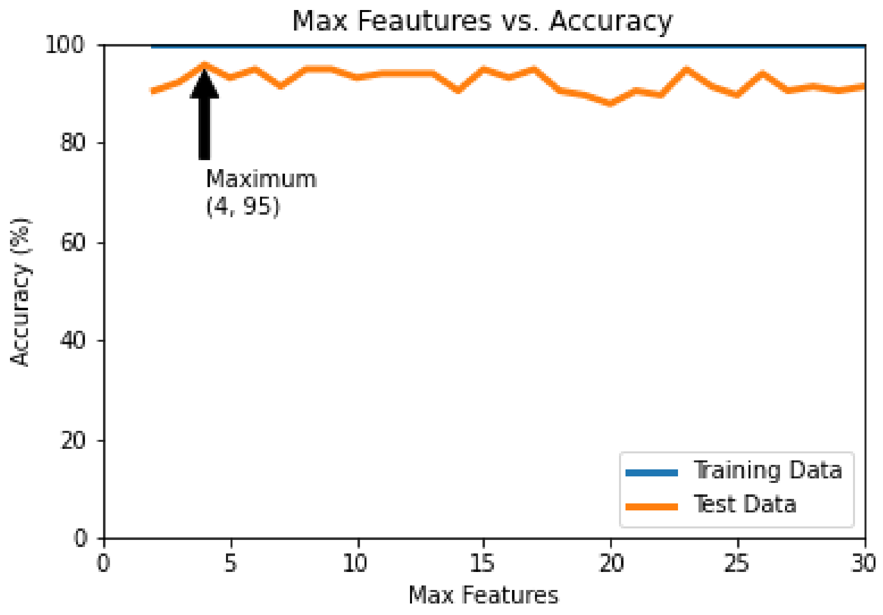

Figure 18: Data set 2 and experiment 9

10) This experiment uses the parameters below:

- Test size of 0.2.
- No randomness of the estimator
- The entropy (information gain) is used to measure the quality of a split.
- No max depth of the tree.
- Two samples required to split an internal node.
- Varying maximum features.
- If the split induces a decrease of the impurity greater than or equal to zero a node will be split.

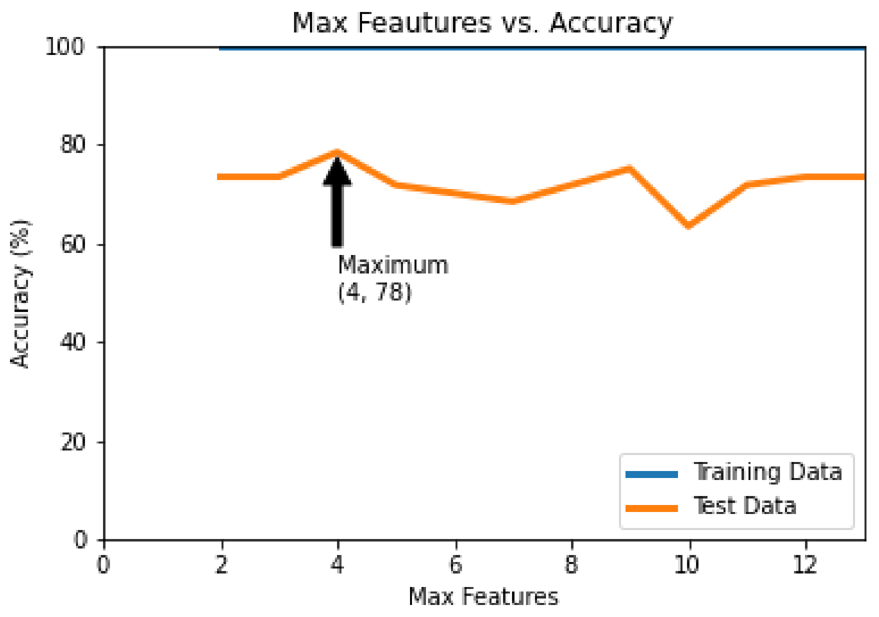

Figure 19: Data set 1 and experiment 10

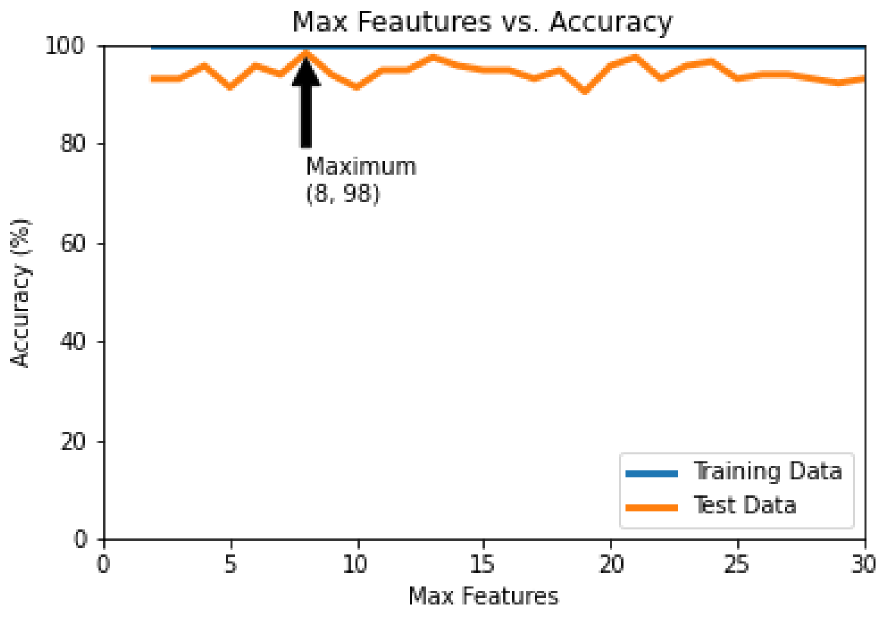

Figure 20: Data set 2 and experiment 10

11) This experiment uses the parameters below:

- Test size of 0.2.
- No randomness of the estimator
- The Gini index(impurity) is used to measure the quality of a split.
- No max depth of the tree.
- Two samples required to split an internal node.
- No maximum features.
- Varying minimum impurity split decrease.

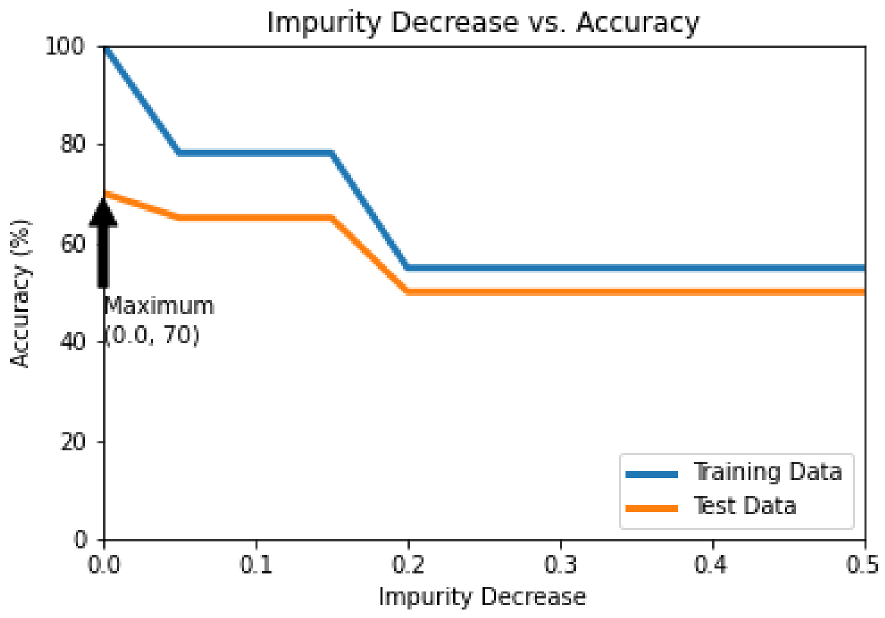

Figure 21: Data set 1 and experiment 11

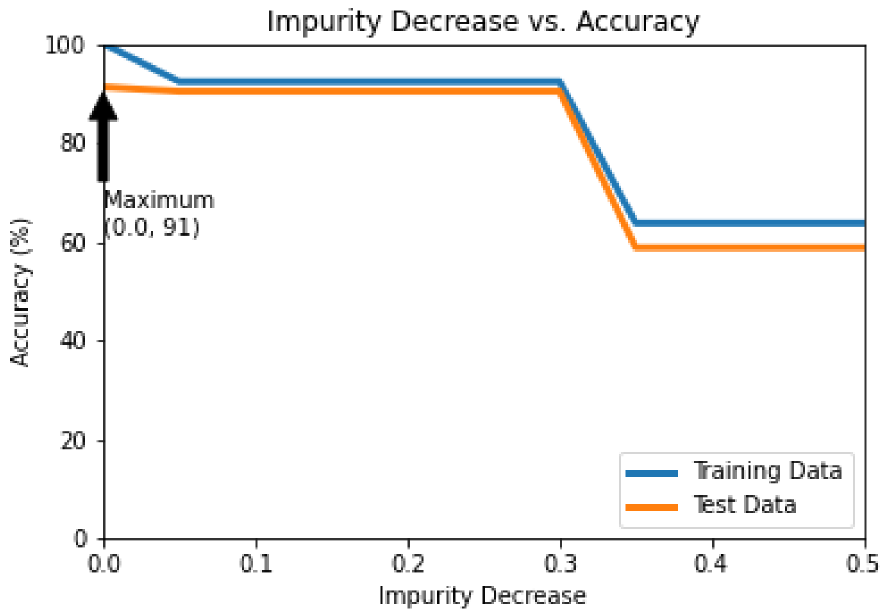

Figure 22: Data set 2 and experiment 11

12) This experiment uses the parameters below:

- Test size of 0.2.
- No randomness of the estimator
- The entropy (information gain) is used to measure the quality of a split.
- No max depth of the tree.
- Two samples required to split an internal node.
- No maximum features.
- Varying minimum impurity split decrease.

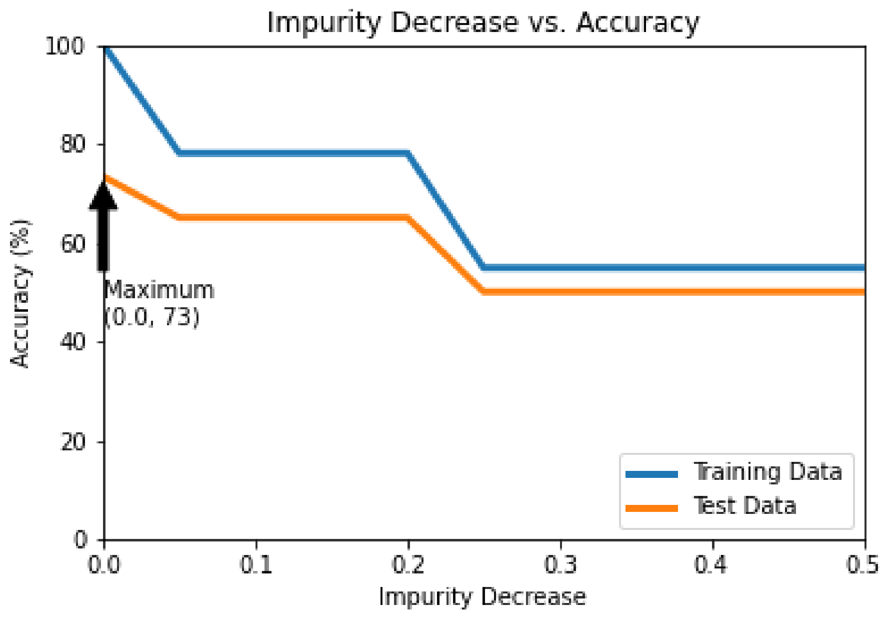

Figure 23: Data set 1 and experiment 12

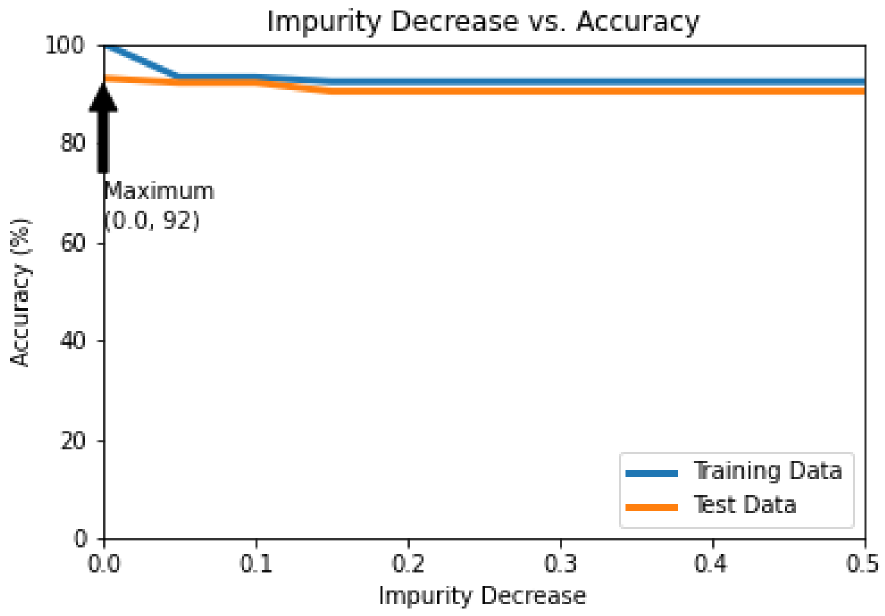

Figure 24: Data set 2 and experiment 12

Random Forests

13) This experiment uses the parameters below:

- Varying test size.
- 100 estimators.
- No randomness of the estimator
- The Gini index(impurity) is used to measure the quality of a split.
- No max depth of the tree.
- Two samples required to split an internal node.
- Maximum features are the square root of the features.
- If the split induces a decrease of the impurity greater than or equal to zero a node will be split.

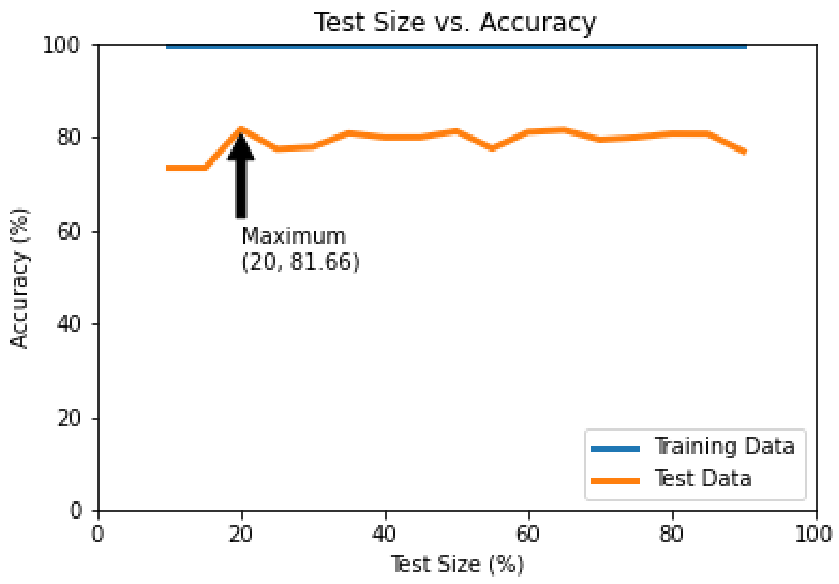

Figure 25: Data set 1 and experiment 13

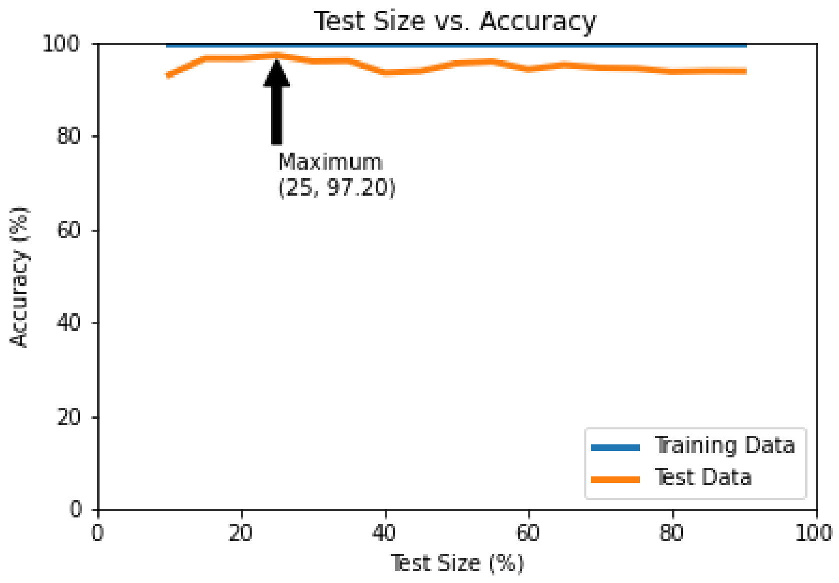

Figure 26: Data set 2 and experiment 13

14) This experiment uses the parameters below:

- Varying test size.
- 100 estimators.
- No randomness of the estimator
- The entropy (information gain) is used to measure the quality of a split.
- No max depth of the tree.
- Two samples required to split an internal node.
- Maximum features are the square root of the features.
- If the split induces a decrease of the impurity greater than or equal to zero a node will be split.

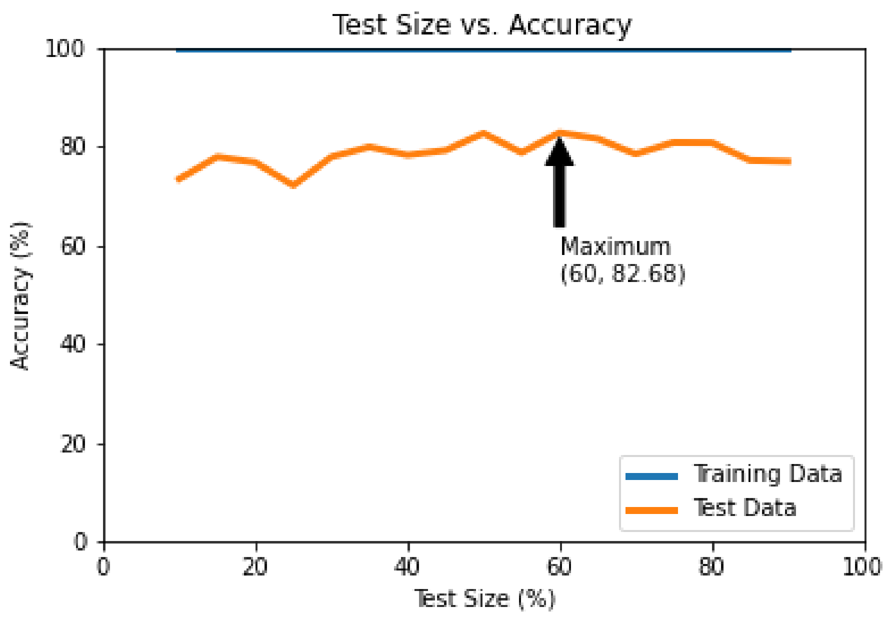

Figure 27: Data set 1 and experiment 14

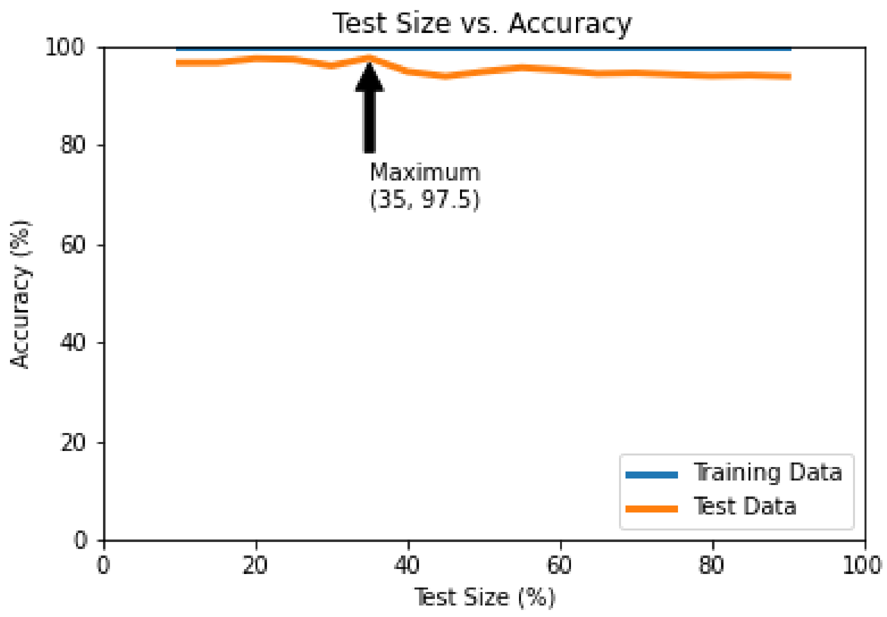

Figure 28: Data set 2 and experiment 14

15) This experiment uses the parameters below:

- Test size of 0.2.
- Varying number of estimators.
- No randomness of the estimator
- The Gini index(impurity) is used to measure the quality of a split.
- No max depth of the tree.
- Two samples required to split an internal node.
- Maximum features are the square root of the features.
- If the split induces a decrease of the impurity greater than or equal to zero a node will be split.

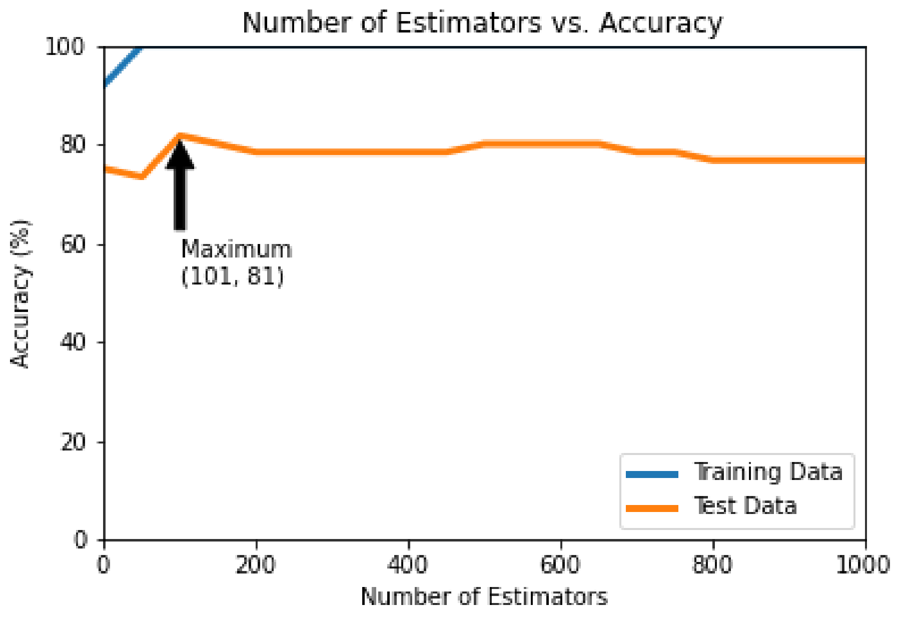

Figure 29: Data set 1 and experiment 15

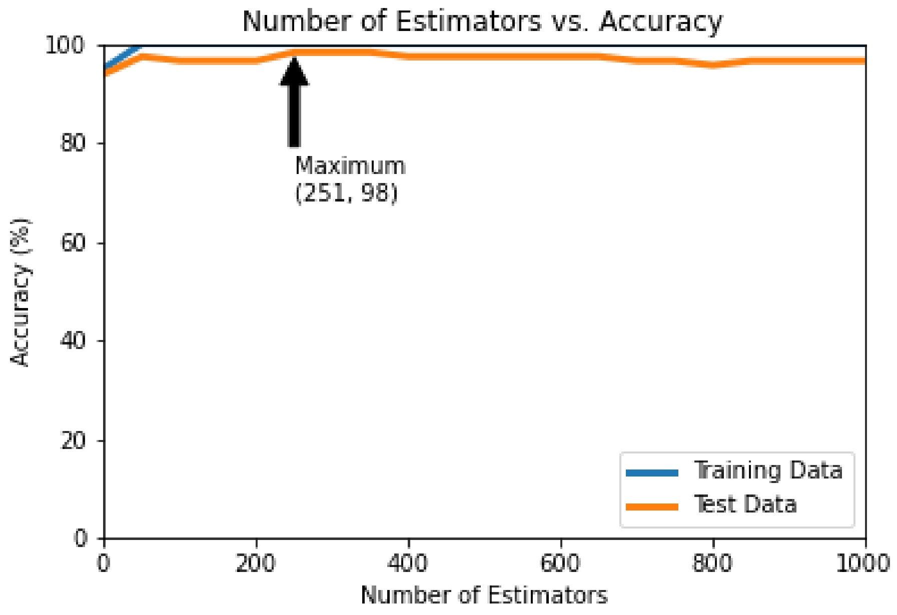

Figure 30: Data set 2 and experiment 15

16) This experiment uses the parameters below:

- Test size of 0.2.
- Varying number of estimators.
- No randomness of the estimator
- The entropy (information gain) is used to measure the quality of a split.
- No max depth of the tree.
- Two samples required to split an internal node.
- Maximum features are the square root of the features.
- If the split induces a decrease of the impurity greater than or equal to zero a node will be split.

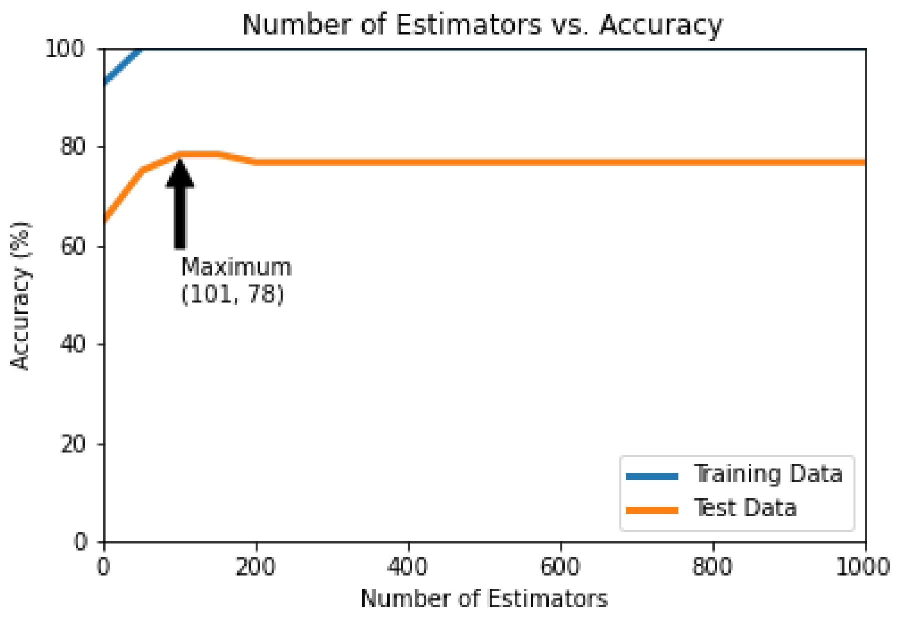

Figure 31: Data set 1 and experiment 16

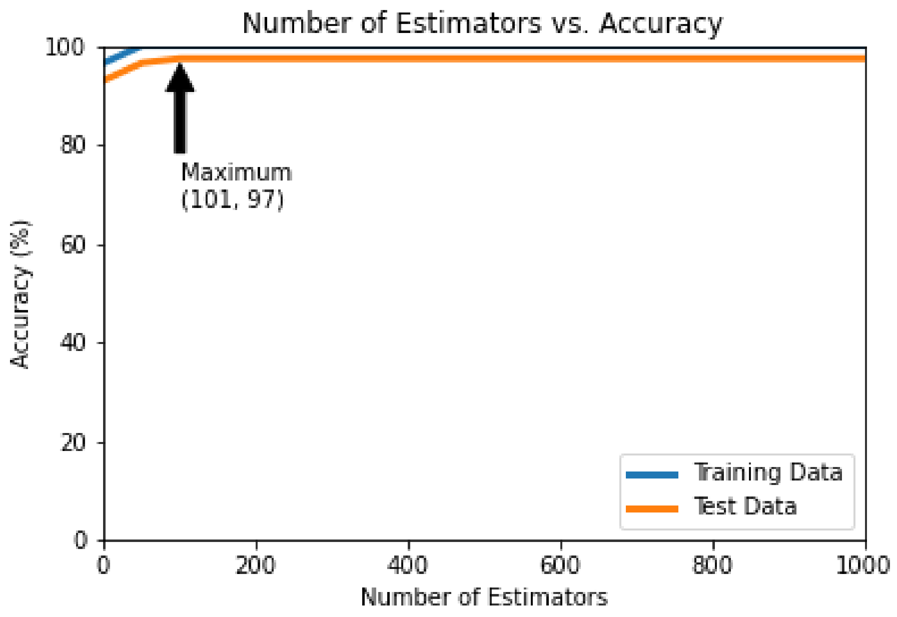

Figure 32: Data set 2 and experiment 16

17) This experiment uses the parameters below:

- Test size of 0.2.
- 100 estimators.
- No randomness of the estimator
- The Gini index(impurity) is used to measure the quality of a split.
- Varying max depth of the tree.
- Two samples required to split an internal node.
- Maximum features are the square root of the features.
- If the split induces a decrease of the impurity greater than or equal to zero a node will be split.

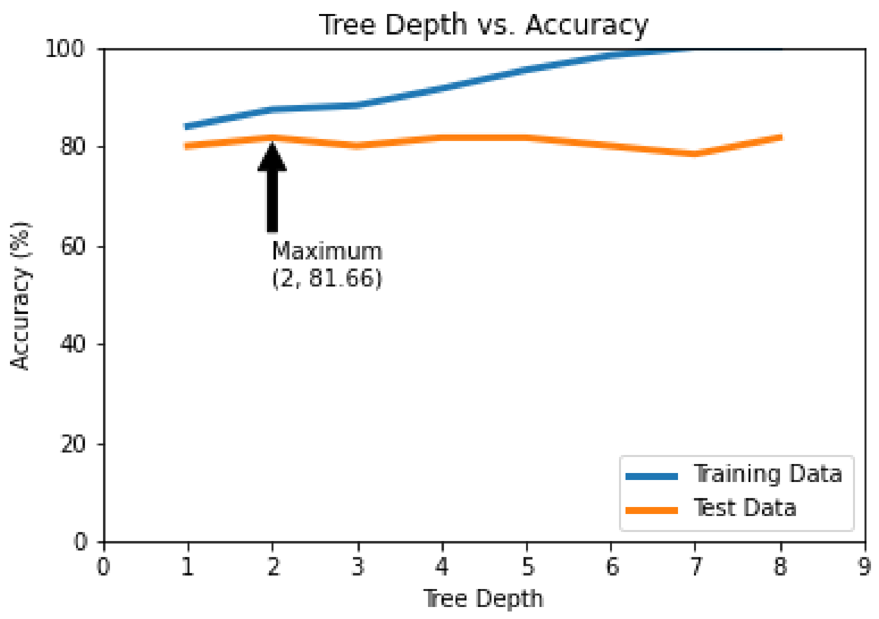

Figure 33: Data set 1 and experiment 17

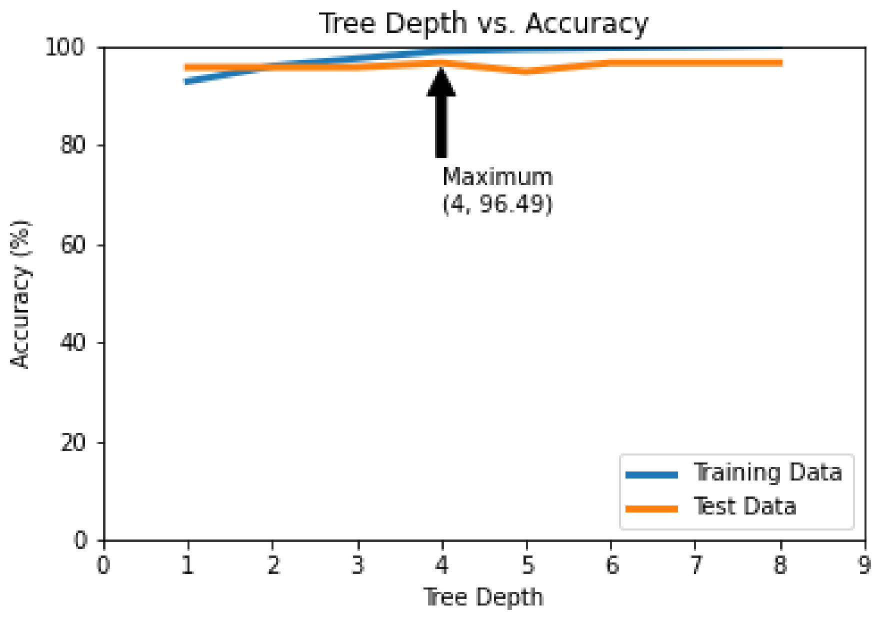

Figure 34: Data set 2 and experiment 17

18) This experiment uses the parameters below:

- Test size of 0.2.
- 100 estimators.
- No randomness of the estimator
- The entropy (information gain) is used to measure the quality of a split.
- Varying max depth of the tree.
- Two samples required to split an internal node.
- Maximum features are the square root of the features.
- If the split induces a decrease of the impurity greater than or equal to zero a node will be split.

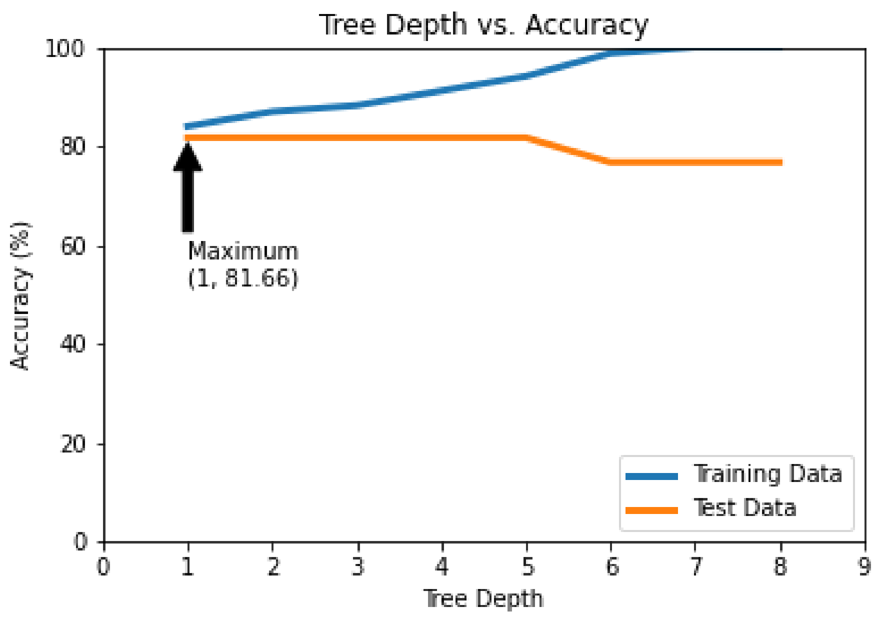

Figure 35: Data set 1 and experiment 18

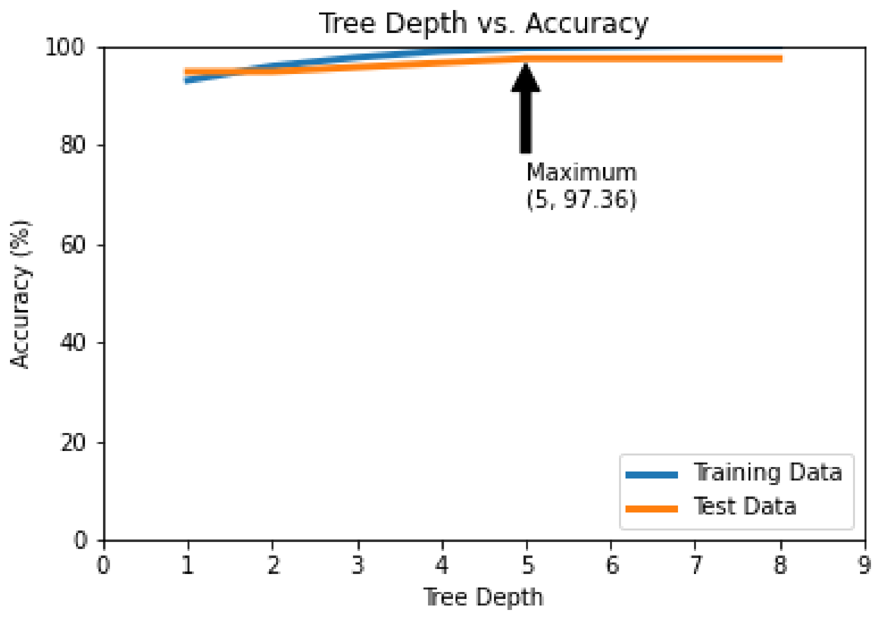

Figure 36: Data set 2 and experiment 18

Neural Networks

19) This experiment uses the parameters below:

- Varying test size.
- 1 hidden layer.
- No randomness of the estimator
- Adam solver type.
- Constant learning rate of 0.001.
- 200 iterations.
- No early stopping.
- Tolerance of 0.0001.

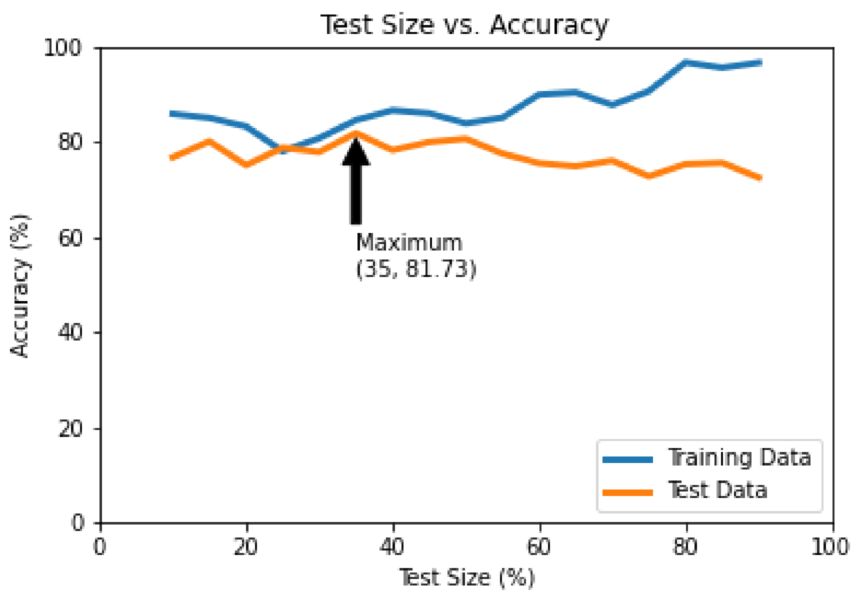

Figure 37: Data set 1 and experiment 19

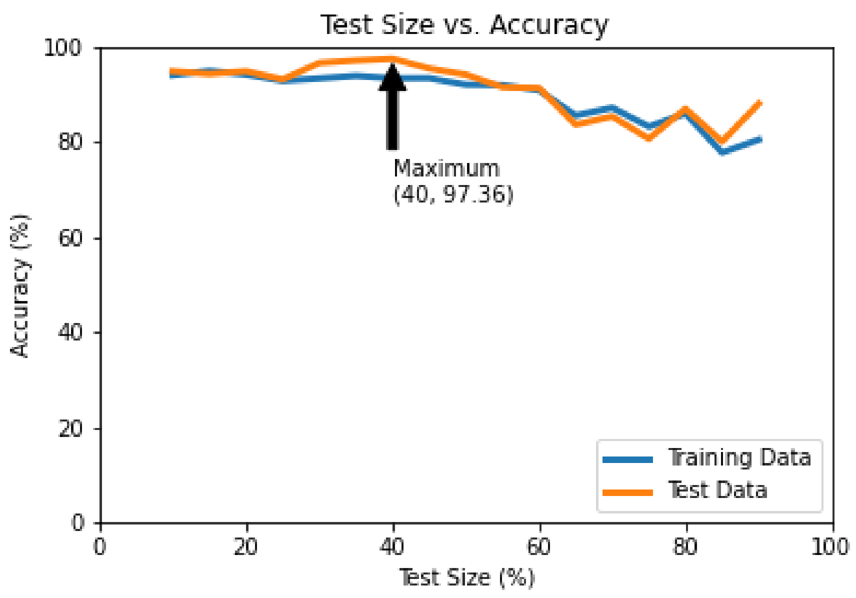

Figure 38: Data set 2 and experiment 19

20) This experiment uses the parameters below:

- Test size of 0.2.
- Varying number of layers of 100 neurons.
- No randomness of the estimator
- Adam solver type.
- Constant learning rate of 0.001.
- 200 iterations.
- No early stopping.
- Tolerance of 0.0001.

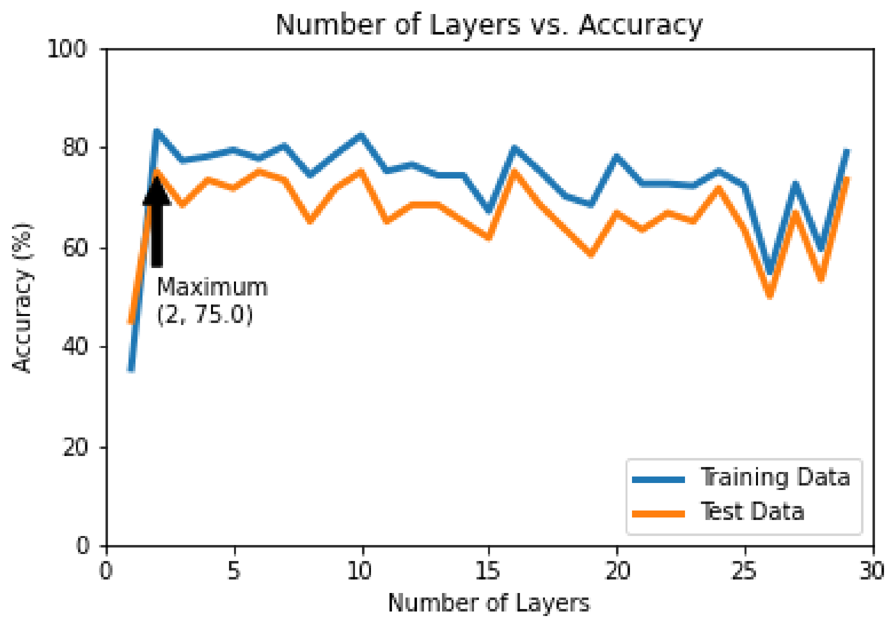

Figure 39: Data set 1 and experiment 20

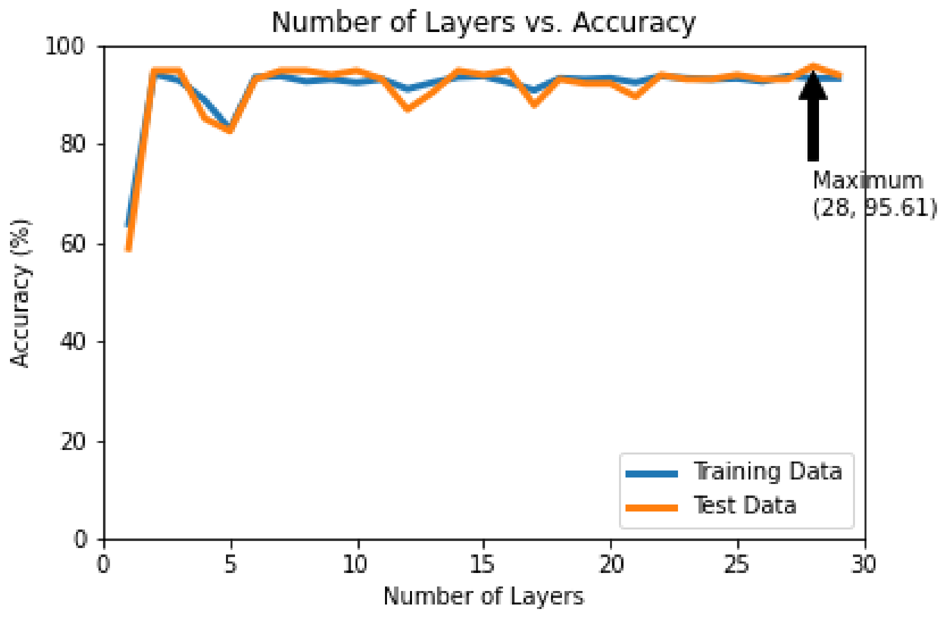

Figure 40: Data set 2 and experiment 20

21) This experiment uses the parameters below:

- Test size of 0.2.
- 1 hidden layer.
- No randomness of the estimator
- Socratic Gradient Decent solver type.
- Constant learning rate of 0.001.
- Varying number of iterations.
- No early stopping.
- Tolerance of 0.0001.

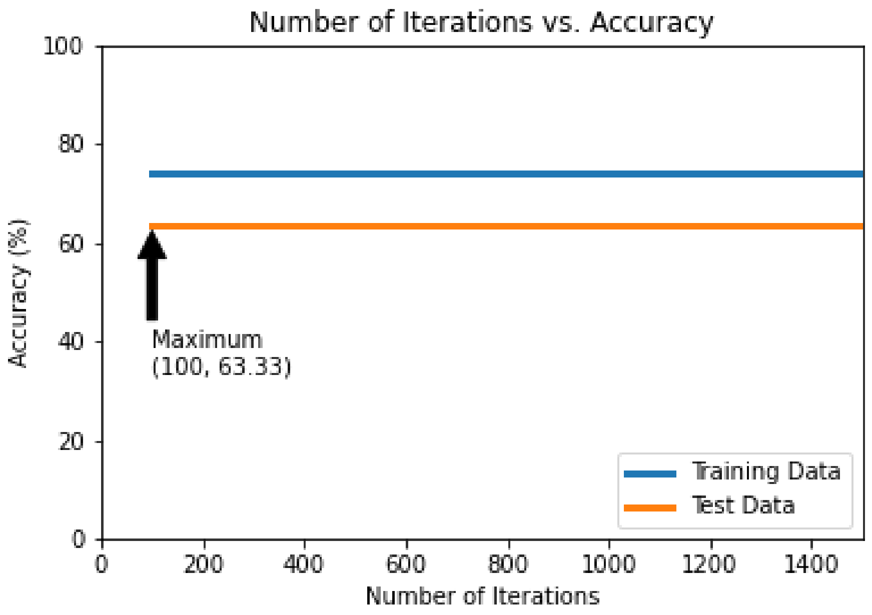

Figure 41: Data set 1 and experiment 21

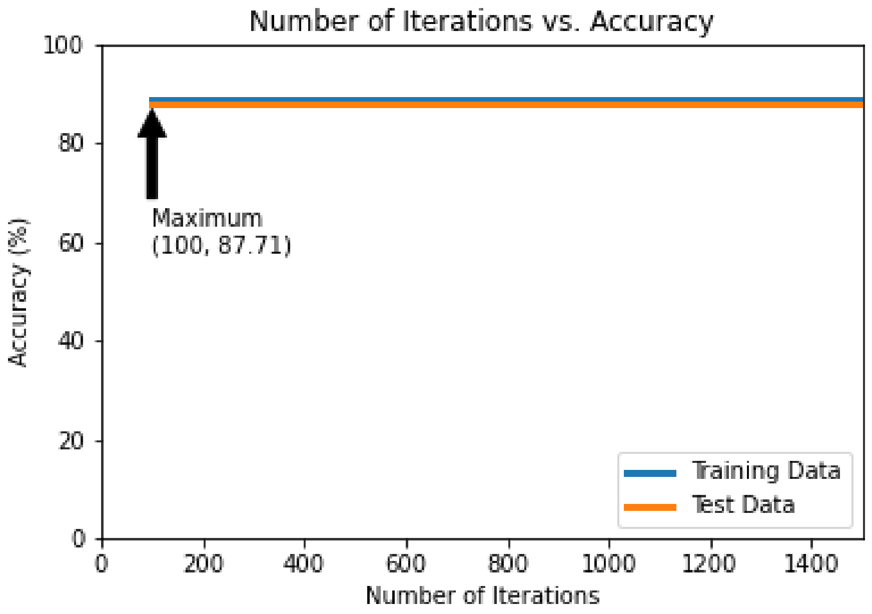

Figure 42: Data set 2 and experiment 21

22) This experiment uses the parameters below:

- Test size of 0.2.
- 1 hidden layer.
- No randomness of the estimator
- Adam solver type.
- Constant learning rate of 0.001.
- Varying number of iterations.
- No early stopping.
- Tolerance of 0.0001.

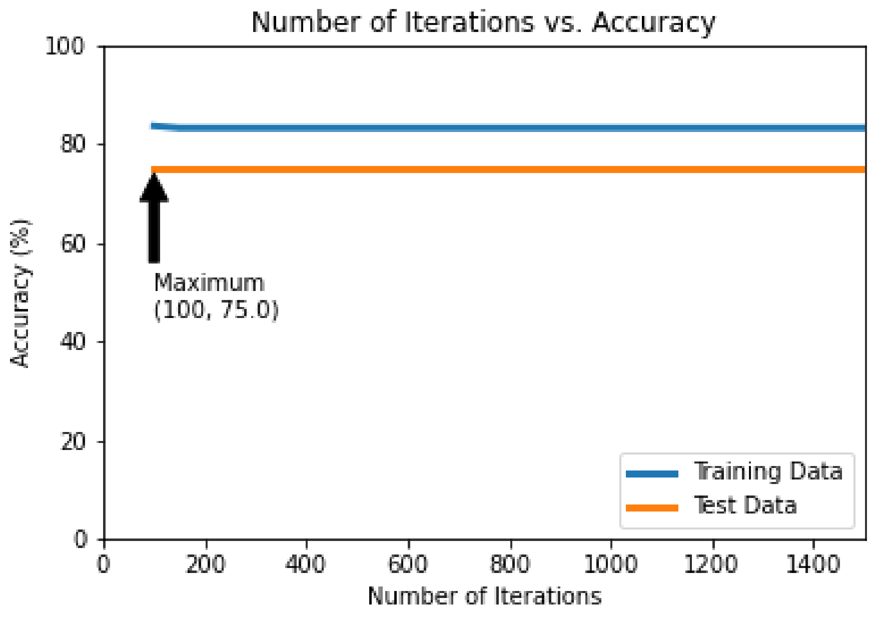

Figure 43: Data set 1 and experiment 22

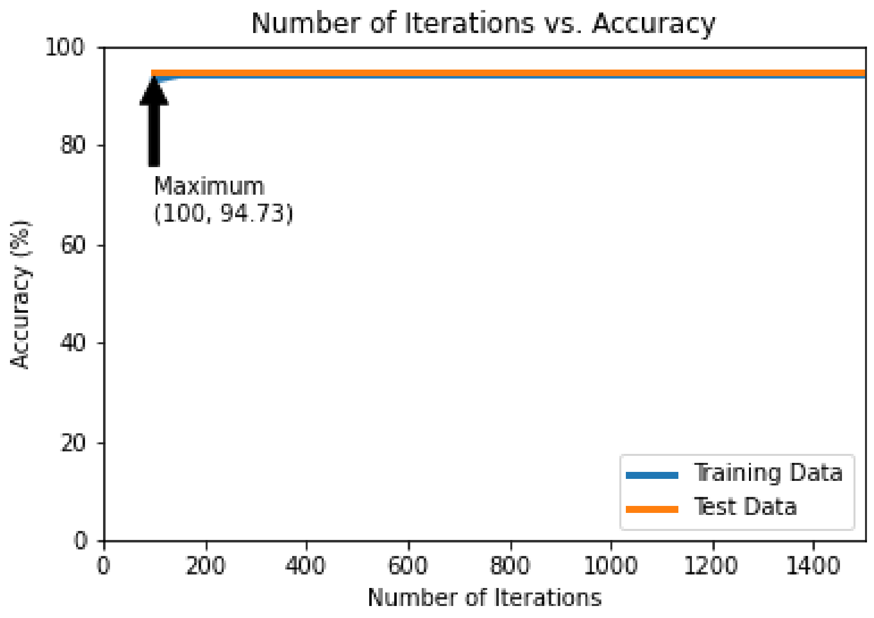

Figure 44: Data set 2 and experiment 22

23) This experiment uses the parameters below:

- Test size of 0.2.
- 1 hidden layer.
- No randomness of the estimator
- Adam solver type.
- Constant learning rate of 0.001.
- 200 iterations.
- No early stopping.
- Varying alpha value.
- Tolerance of 0.0001.

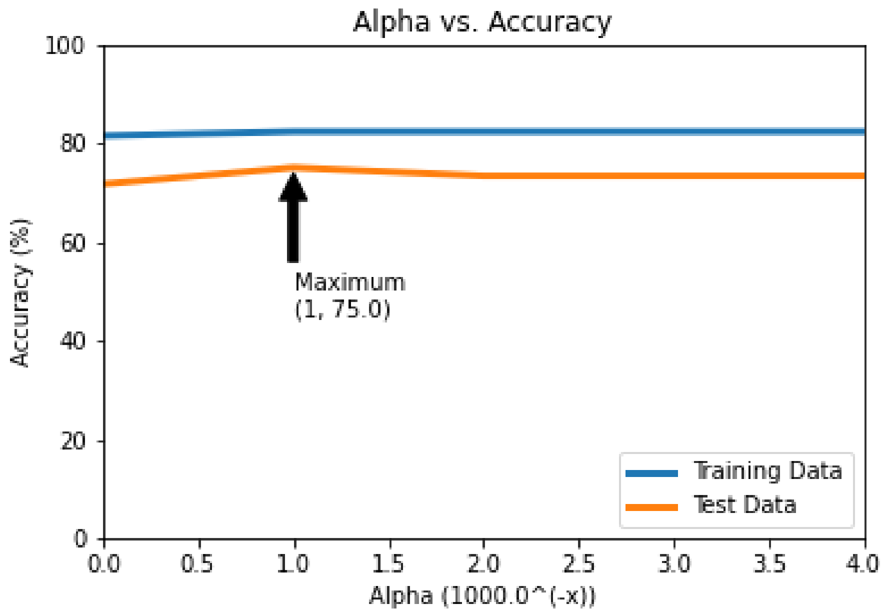

Figure 45: Data set 1 and experiment 23

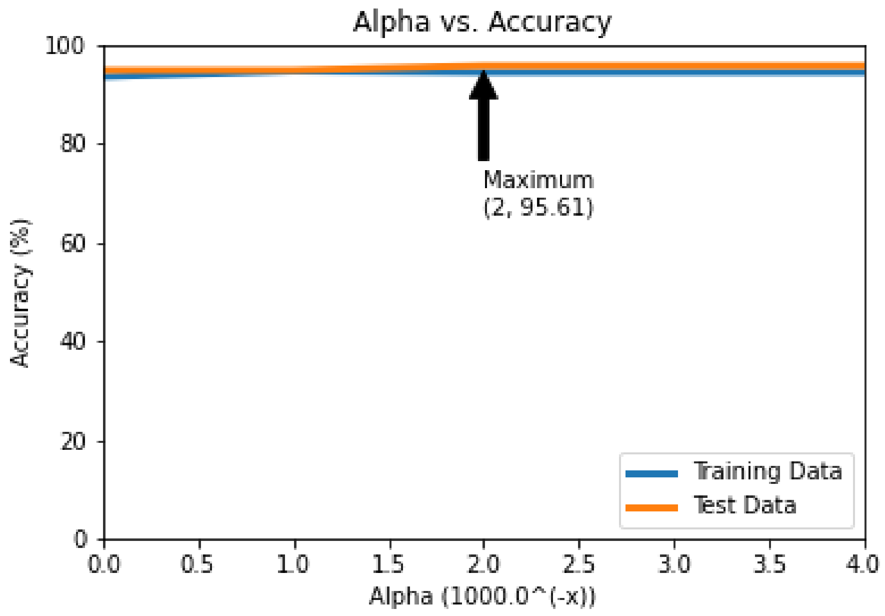

Figure 46: Data set 2 and experiment 23

24) This experiment uses the parameters below:

- Test size of 0.2.
- 2 hidden layers of varying size.
- Logistic activation function
- No randomness of the estimator
- Adam solver type.
- Adaptive learning rate of 0.001.
- 200 iterations.
- No early stopping.
- Tolerance of 0.0001.

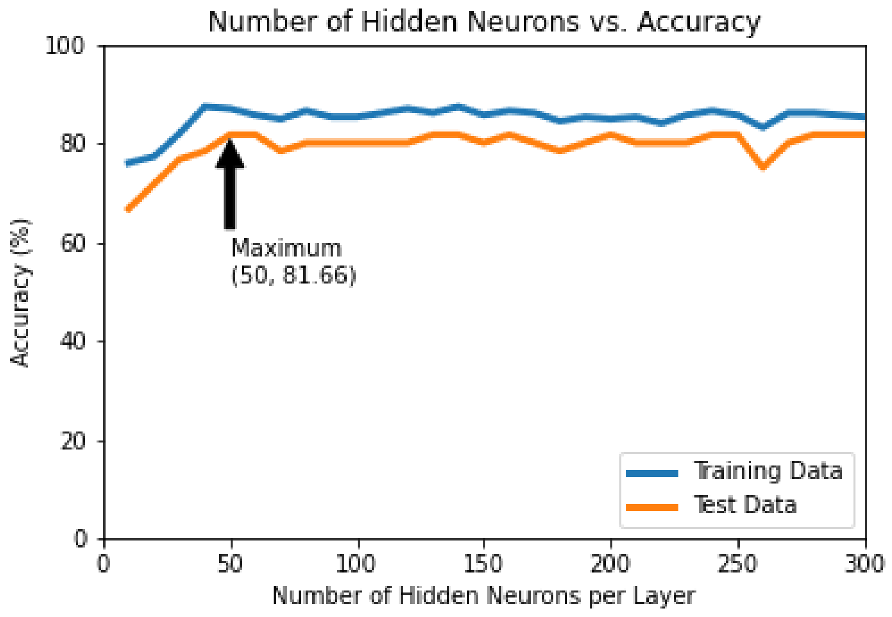

Figure 47: Data set 1 and experiment 24

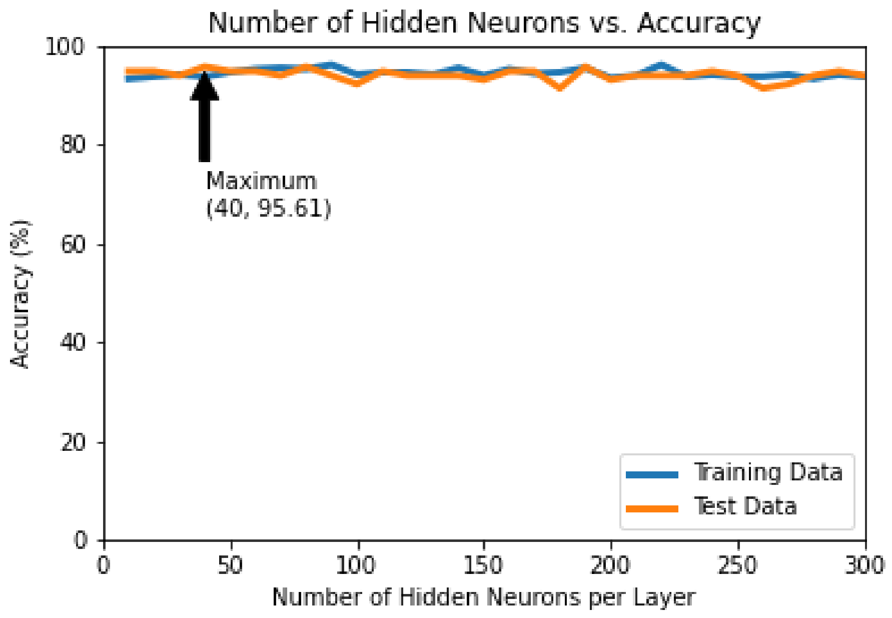

Figure 48: Data set 2 and experiment 24

25) This experiment uses the parameters below:

- Test size of 0.2.
- 1 hidden layer.
- No randomness of the estimator
- Adam solver type.
- Varying value of the constant learning rate.
- 200 iterations.
- No early stopping.
- Tolerance of 0.0001.

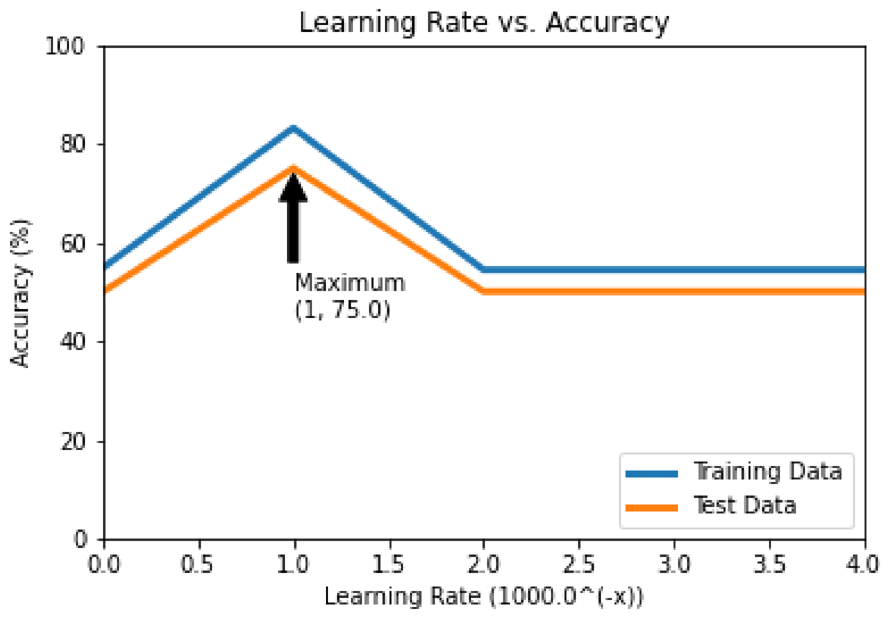

Figure 49: Data set 1 and experiment 25

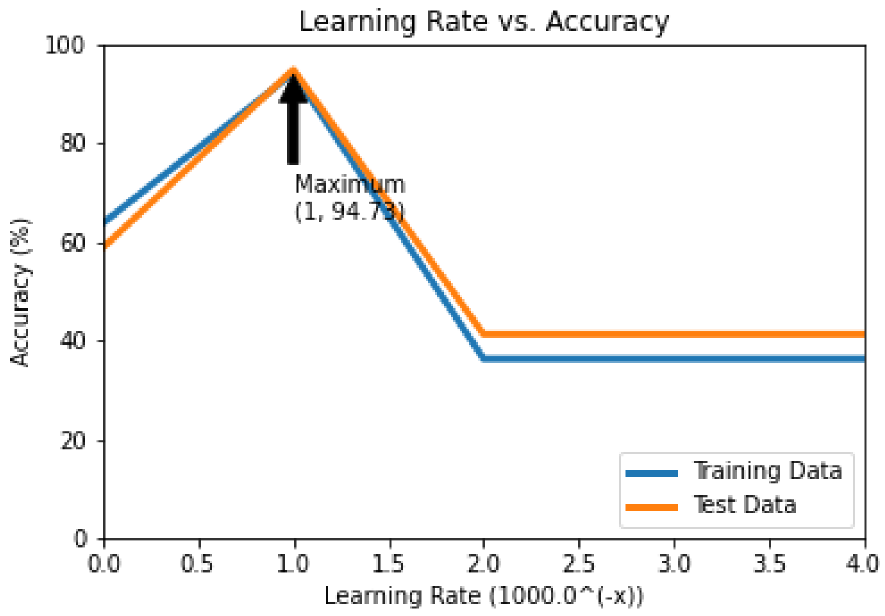

Figure 50: Data set 2 and experiment 25

26) This experiment uses the parameters below:

- Test size of 0.2.
- 2 hidden layers of varying size.
- No randomness of the estimator
- SGD solver type.
- Adaptive learning rate.
- 200 iterations.
- No early stopping.
- Tolerance of 0.0001.

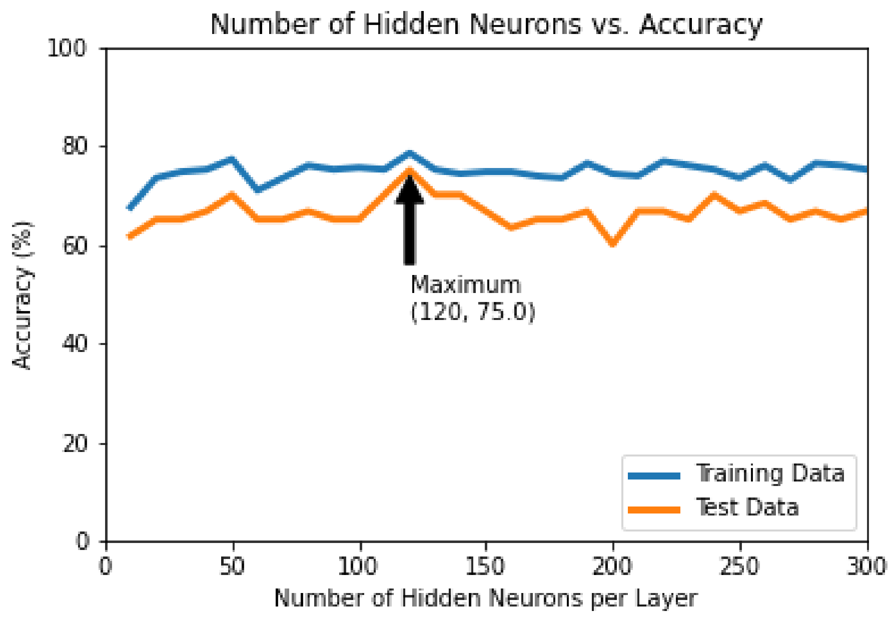

Figure 51: Data set 1 and experiment 26

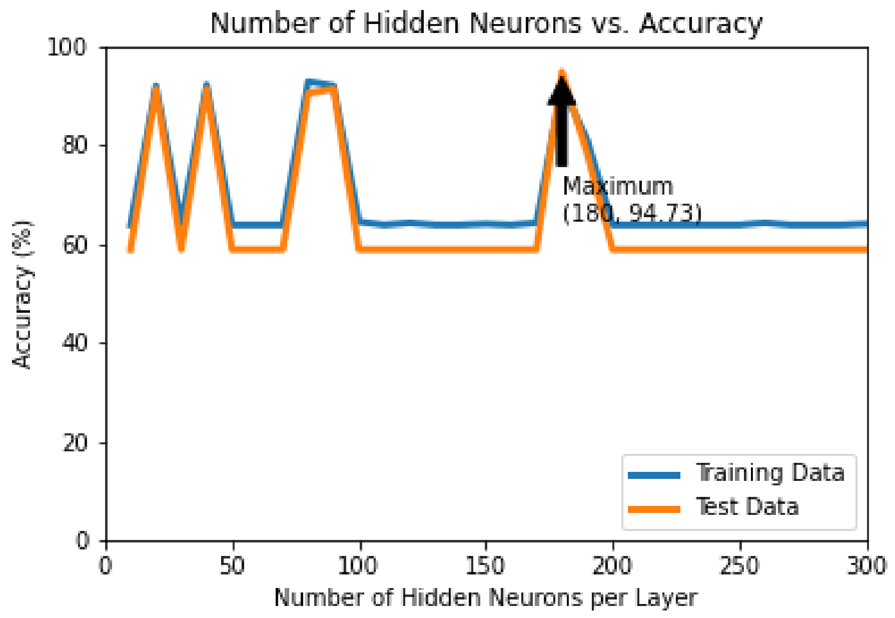

Figure 52: Data set 2 and experiment 26

27) This experiment uses the parameters below:

- Test size of 0.2.
- 2 hidden layers of varying size.
- No randomness of the estimator
- SGD solver type.
- Invscaling learning rate.
- 200 iterations.
- No early stopping.
- Tolerance of 0.0001.

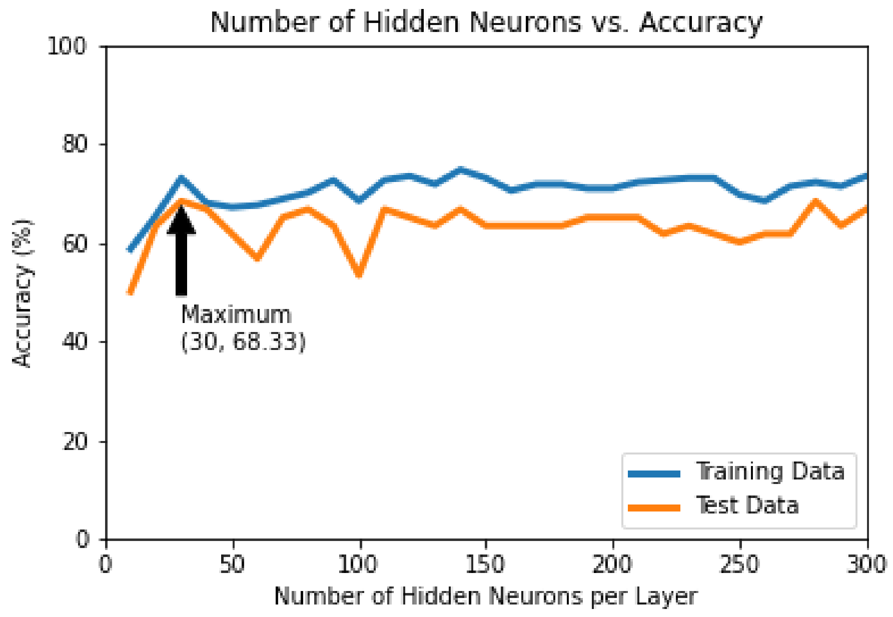

Figure 51: Data set 1 and experiment 27

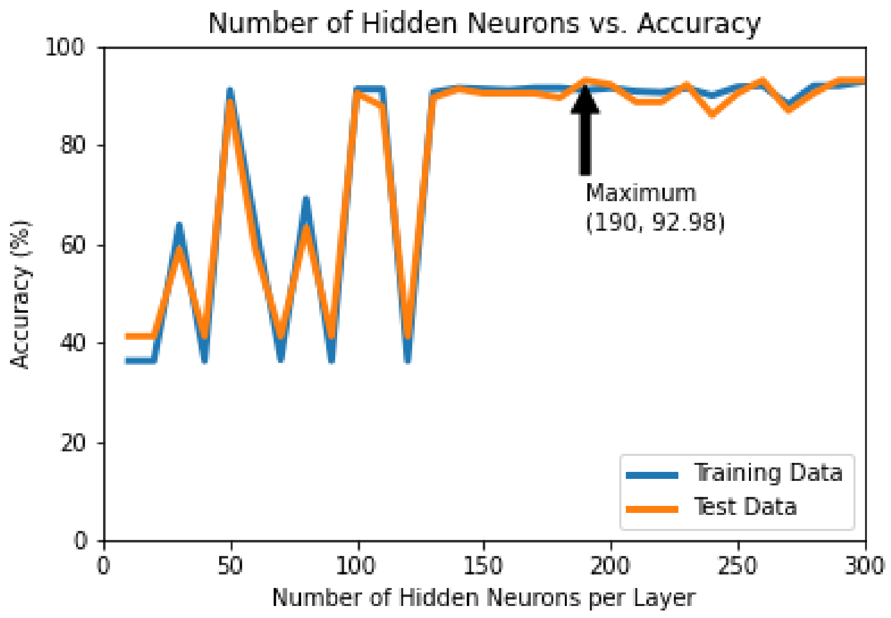

Figure 52: Data set 2 and experiment 27

3. Analysis

This report contains various experiments run with decision trees, random forests, and neural networks. This section will analyze those results, explain the design choices of these tests, examine best model parameters, and make observations on possible model improvements that can be made.

Decision Trees

The parameters I varied when creating decision tree experiments were test size, split criterion, max depth, max feature, randomization, and impurity decrease.

When varying test size, it could be seen that the Gini index preformed best on an equal split, whereas entropy performed best around the rule of thumb eighty to twenty split. I chose to run these varying size tests to view possible overfitting as the ratio of data changes. This experiment allowed me to confirm the most accurate and efficient test size is eighty to twenty, and I carried these results through my other experiments with decision trees.

I then varied the depth of the trees, as shown in **Figure 5 -8**. Varying the depth produced a maximum accuracy at a tree depth of 3 in 3 of 4 tests. The maximum accuracy for data set one was 80% using the Gini index as the split criterion and a tree depth of 3. The maximum accuracy for data set two was 96% using the Gini index as the split criterion and a tree depth of 2. This test was run to see how different split criterions dealt with increasing tree depth; however, they both produced similar results.

I then varied the number of minimum samples to induce a split. The minimum number of samples needed to split optimally for dataset one was found to be 40 samples in which Gini had an 80% accuracy, and entropy produced a 76% accuracy. Also, after the increase of accuracy at 40 samples, the datasets had a sharp decrease than a continuous line as the varying the value become ineffective. The optimal minimum number of samples was 32 samples in which Gini had a 96% accuracy, and entropy produced a 93% accuracy.
In addition to 32 producing the max accuracy, there was no sharp decline in accuracy until 300 samples where reached, which provides a broad range in which to select a value. The data set trends seen after the optimized value is unique between datasets due to their different distribution and size. Varying the number of samples in order to split was intended to examine the differences between these values in different datasets, which was observed in this experiment.

The next parameter that was varied in **Figures 13-16** was the randomization value. The tests run with a varying random value produced an erratic graph. Much like a heartbeat rhythm, these graphs increased then decreased. When examining the optimized randomization, we can see that the test run using Gini has a similar peak around a randomized value of 1, whereas, when using entropy, the optimized randomization values had no similarities between the datasets. I ran this randomization test to investigate if there was ever an optimized range or a range where the randomized value does not change the accuracy.

I then tested the variation of the maximum number of features. I found that the optimized values of features were either ¼ or ¾ of the number of features. Dataset one was the most accurate at 81% when using the Gini index as the split criterion and seven features for the maximum number of features. I found that the dataset's entropy was the most accurate at 98% when using entropy as the split criterion and eight as the max features. I wanted to examine the relationship between number features considered and accuracy to prove a random subset of the features can improve the maximum accuracy.

The impurity decrease produced a maximum at 0 and declined after that. The impurity decreased sloped downward until it flattens out until it hits a constant rate, as shown in **Figures 11 and 12**. Therefore an impurity of 0 is most efficient.

Random Forests

Since random forests are constructed of decision trees, I chose to skip some tests as they were inefficient when analyzing decision trees and continued to remain in random forests. In random forests, I varied the test size, number of estimators, and the tree depth.

When varying the test size for both datasets, I found that while using the Gini index for the split criterion, the test size of that follows the 80-20 rule of thumb was the best size. When using entropy, the optimized test size was around 40. It seems random forests using entropy require a larger test size. This is the opposite finding than what I found for decision trees.

When varying the number of estimators, I found that 3 of the tests had a max accuracy when using a hundred estimators, and after that point, the maximum the line converged to a constant as it did not affect the number of estimators grew. The number of trees is the emphasis of a random forest; this was a crucial experiment for me to investigate.

Due to a large number of trees in a random forest, I thought it would be beneficial to see if the max depth had an increased impact on a multi-tree dependant model. The results of the optimal depth were between 1—5. Dataset one had an optimal value at a lower depth, whereas dataset 2 had an optimal value at a depth value of 4 or 5. This result was different from decision trees as there is no average tree depth that the datasets share an optimal value at.

Neural Networks

When implementing a multi-layer neural network, I varied the learning rate, the alpha value, number of layers, neurons in a layer, test size, solver type, and the maximum number of iterations.

When varying the test size, both datasets performed better at a training size of around 35-40. This implies that neural networks might need more samples base on it is extensive training methods. This test is essential and intended to explore the optimal split of neural network needs vs. other models.

Using the default value of 100 neurons per layer, I varied the number of layers from 1 to 30. This caused multiple spikes in the first data set, as seen in Figure 39 and 40. These spikes had a maximum accuracy at around two layers. It is preferred to have a lower number of hidden layers due to the time it takes to train a neural network. This test was intended to explore how the number of neurons would impact the output layer.

When varying the number of iterations, I went from 100 to 1500 iterations; however, for both datasets, this did not affect. This leads me to believe that the number of iterations must be much higher as the current number of iterations are not sufficient for convergence of the data. This ineffectiveness of the varying number of iterations presented itself when using both sgd and Adam as the solvers. This test was run to view the test time of a neural network vs. other models.

When changing the alpha values, as seen in Figure 45-46, a very slight increase in accuracy occurred between 0.001 and 0.001, which is near it's the default value. This was run to view the degree of impact that alpha value had and see if it was worth overriding the default value. This proved ineffective.

When varying the number of neurons in a two hidden layer network, the graph is seen to have a small amplitude shift between different values. Dataset two's line is relatively smooth and bounded, whereas dataset one is bounded regions but multiple more drastic variations. This test was intended to explore the relationship between the number of neurons and the datasets. This proved that the size of the layers must be dependant on the dataset size.

Changing the learning rate was very similar to the alpha value, where it was seen that the default value was the optimal value; however, there is a significant spike inaccuracy when using this value compared to the small effect of varying alpha. When varying the learning rate after maximum accuracy, the graph is seen to have a sharp drop to form a constant value. This showed the ineffectiveness of varying the learning rate initial value. This test was run to see if it the benefit of changing the default learning rate value.

When sgd is used instead of Adam as the solver and an adaptive learning rate over a constant one, there are more optimal neuron values produced. This is seen in Figure 52, as there are four near-optimal neuron values. This test was intended to explore another type of learning rate other than a constant one. The adaptive learning rate performed less efficiently than a constant one.

Optimal Parameters

I ran an optimizer on the decision tree for dataset one with various possible parameters, and then I received the following.

Best decison tree parameters and score, Dataset 1
{'criterion': 'gini', 'max_depth': 3, 'max_features': 9, 'min_impurity_decrease': 0.0, 'min_samples_split': 31}, 0.8346327683615818

This shows that the best criterion is the Gini index, with a max depth of 3, 9 features, impurity decrease of 0, and 31 samples needed to split. These findings line up with mine gathered above. Also, randomization was left out as I found that it only helps assist in the prevention of data clustering and does not have similar effects on other datasets. These optimal parameters may change based on the distribution of the data set. When the test size changes, the max features and minimum samples needed to be split should be modified. When training the tree to a more extensive test data, the tree must be allowed to shift or increase their parameter based on the test size change. To effectively choose the best leaf a more significant number of features should be looked at to see a trend in the occurrence of a feature. If a feature is not viewed, the training function might miss a meaningful correlation.

When optimizing random forests, the optimal features were as below.

Best random forest paramaters and score, Dataset 1
{'criterion': 'entropy', 'max_depth': 1, 'max_features': 'log2', 'min_impurity_decrease': 0.1, 'min_samples_split': 3, 'n_estimators': 100}, 0.8584180790960453

Similar to decision trees, these parameters match my observations except when the tree depth. If you use a low depth tree, this means there are a few features that best predicts the classifier. This was not consistent with my experiments above. Another interesting parameter seen as optimal was considering the square root of the max features rather than all the features. I would have expected more features to be optimal in a lower depth tree to avoid the limiting of your scope. If the test size changes, the number of estimators should increase to allow for more processing and account for the higher distribution of samples in each tree. The other key takeaway is the nearly 2 percent increase in accuracy even though the trees in the random forest only have an optimal depth of one. This speaks to the wisdom of the crowd phenomenon.

I was unable to run the optimizer on my neural network model due to a large number of possibilities. However, through tests, I found that two hidden layers in most efficient and in each hidden layer, the number of hidden neurons should be 2/3 the size of the input layer, plus the size of the output layer. In terms of the solver, my experiments found Adam to be more effective than SGD. Also, a constant learning rate was more efficient in the experiments run above. If the data set size changes, it would be beneficial to use early stopping parameters and increase the validation size accordingly. The max features and neurons should not change as the outer layers aren't varied.

Model Comparison and Improvement

When comparing the models, we can see that the models performed roughly at the same accuracy on dataset two. When experimenting using dataset two on a decision tree, random forest, and neural network result in a 98%, 97.5%, and a 97% percent accuracy, respectively. When experimenting using dataset one on a decision tree, random forest, and neural network result in 83%, 83%, and 81% percent accuracy, respectively. The maximum accuracy of dataset one can be improved, as shown by the optimizer. It is likely based on an accuracy score; a neural would be the best predictor given no resources or time constraints.

I believe to improve the neural network's method; I could have run many more iterations as I was unaware the data would not converge. This would significantly improve the neural network's maximum accuracy value for both datasets. I wish that I could further limit parameters that I had the optimizer function run for neural networks as I was unable to get any results to a vast number of possible parameters. Furthermore, I would like to visualize the effect of reducing the hidden layers from many to few.

These models were compared through many experiments, and the analysis of these should help improve future models of similar nature.