**Assignment Three**

I chose to complete programming questions. I will provide the code, and images of my results to demonstrate this in the document here.

DISCLAIMER: The data is generated in a specific way, however if we alter the train data ratio to test data the accuracy of the classifiers will change drastically. As instructed, I try to keep a large portion of data for validation and testing.

**Programming with Supervised Learning**

I generated data such as in the images 1, 5 and 6 in the assignment questions.

Text

Description automatically generated with medium confidence

Text

Description automatically generated

Graphical user interface

Description automatically generated with medium confidence

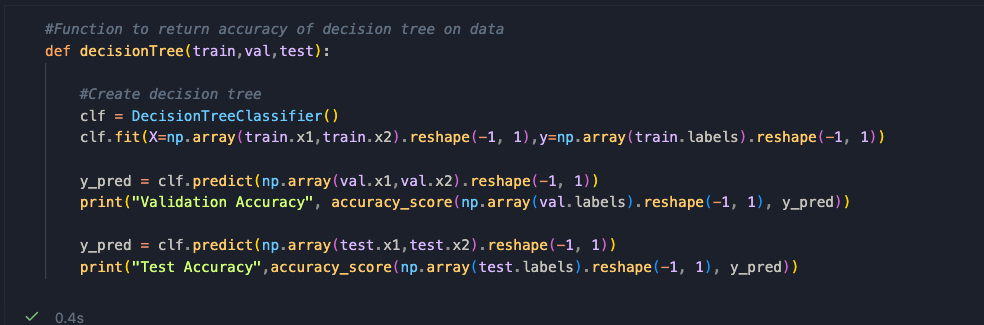
I programmed the K-NN algorithm, and then performed cross-validation on for the three pieces of data. I demonstrate the results below:

Text

Description automatically generated

The accuracy on the data is very high, however the data was generated in a manner where a high accuracy such as this is likely to happen. If we run K-NN with less optimal K values, we achieve less favourable results. My code has the specific implementation in it alongside comments to see how I approached the problem.

**Comparison of K-NN and Decision Tree**

I used sklearn as a framework to create a decision tree model I could easily run on each dataset. I create the model quite easily with this chosen framework.****

I made a function to make it very simple to run an entire training/cross-validation quickly on a dataset. I ran the function on each dataset and got the following results:

**Text

Description automatically generated**

Generally, the KNN algorithm has great classification, whereas the decision tree struggles. The accuracy of the tree on test data never is higher than 63% on any dataset. Changing decision tree parameters and options, plus making a deeper tree may help to increase accuracy here. The specific implementation can be seen in the ipynb file.

**Deep Neural Networks**

I tested multiple types of networks with different structures in this question. Here we can see the structure of the two networks.

|  |  |
| --- | --- |
| Model One | Diagram  Description automatically generated  Model Two |

Model two is a deeper model than model one. I constructed more hidden layers for model two, to see how this might affect accuracy. I tested each model on each dataset.

Text

Description automatically generated

Text

Description automatically generated

Model two has better accuracy generally for test and validation data. Dataset two isn’t well suited to the neural network constructed, and we can see even lower accuracy with the deeper model here. Specific implementation is available in the ipynb file. Based on the testing results, the model to use is data dependant. Model two should be used on datasets one, and three, whereas model one should be used for dataset two. This is based on a visual inspection of the test accuracy.

**Noisy Data**

I added a variety of noise to the different data. I chose datasets one and two to generate noise with. Below I show some noise added to each.

|  |  |
| --- | --- |
| Chart, scatter chart  Description automatically generated | Chart, scatter chart  Description automatically generated |

Here above I added 10% noise to the data on the right. The data on the left is without noise to demonstrate the effects. Additionally, dataset two was also modified.

|  |  |
| --- | --- |
| Chart, scatter chart  Description automatically generated | Chart, scatter chart  Description automatically generated |

Similarly, the left dataset demonstrates the data with no noise, where the right dataset involves noise. I made a list of noise values to test, and then iterated through each dataset applying noise. In each iteration I ran the K-NN and Decision tree functions I defined earlier which gave me validation and test accuracy. Below we can see the output of this.

|  |  |
| --- | --- |
| Text  Description automatically generated | Text  Description automatically generated |

The addition of noise to each of the datasets make classification more difficult. The KNN algorithm seems to find classification much more difficult as we increase noise in each case, but the decision tree accuracy doesn’t decrease too much. The KNN algorithm additionally keeps changing to a different optimal K value. The noise can help to determine good hyperparameters for the KNN algorthim. For dataset one, 9 is likely the best candidate as even with 25% noise, an accuracy of 67% is achieved. For dataset two, 3 is certainly a good candidate as even with 25%(247.5 is the scaled value with the data) noise, we achieve an accuracy of 86% on unseen data despite noisiness in training data.