ARTIFICIAL INTELLIGENCE ASSIGNMENT 2

Law Ho Fong 10107321

Kiki Mutiara 10031014

Hyperparameters Search Methodology

Nearest Neighbours Classifier

For Nearest Neighbours Classifier, we need to tune the *number of neighbours* parameter. We can achieve this by first declaring the options of parameter that we want to test, in this case it's between 1 and 30, and then by using *GridSearchCV* function from *sklearn*, we can find the best *number of neighbours*. We also cross validate the data by separating it into 10 subsets. Later we fit the training data to the tuned model, and we can get a higher accuracy. The result of the tuning can be different on each run, so we print the result to the *stdout* so the user can see which *number of neighbours* is being used in that specific run.

Decision Trees Classifier

In this classifier, we tune the *minimum sample leaf* parameter. The method that we use to find the hyperparameter is, declare the number of leaves that we are going to check, which are 1, 2, and 4, then use *GridSearchCV* to find the most optimum *minimum sample leaf* and do cross validation for 10 subsets. After that, we can fit the training data to the model and see the result. We also print the best sample leaf to *stdout,* since the result can be different each time we run the code. So, by printing it out, it can help the user to see what *sample leaf* is being used.

Support Vector Machine Classifier

For SVM Classifier, we tune the *C* parameter of the model. We want to check between 0.1, 1, 10, and 100, and pick which *C* parameter is the best. In this case, we use *GridSearchCV* function from sklearn and cross validate the data into 10 subsets. After we get the best model, we can fit the training data into that model and check the accuracy. We print the best *C* parameter so the user can know which *C* parameter is the best for that run.

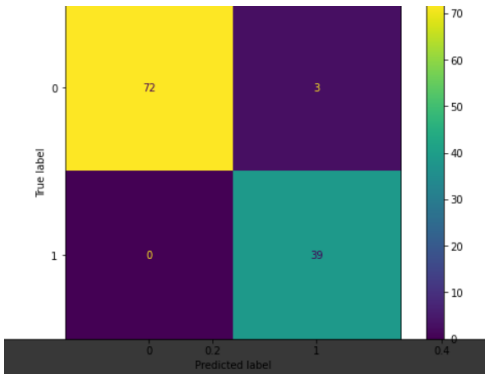
Neural Network Classifier

Last, in Neural Network Classifier, we tune the *hidden layer sizes* parameter. We give 2 options for the layer sizes, (20 and 10), and (30 and 20). This means there are 2 dense layer in the model, the first number represents the neuron size for the first layer, the second number represents the neuron size of the second layer. *GridSearchCV* function is being used again in this hyperparameter search, also with 10 subsets of cross validation. The best layer sizes is being printed out to the *stdout*, which helps the user to see the best parameter used in that run.

Classifier Performance Comparison

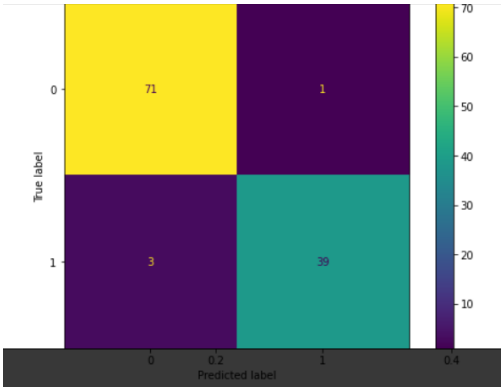
The best accuracy goes to the Neural Network Classifier with 97.37% of accuracy and layer size (20,10). The result and confusion matrix are being shown below:

```
Best hidden layer size: (20, 10)
The accuracy of Neural Network:0.9736842105263158
```



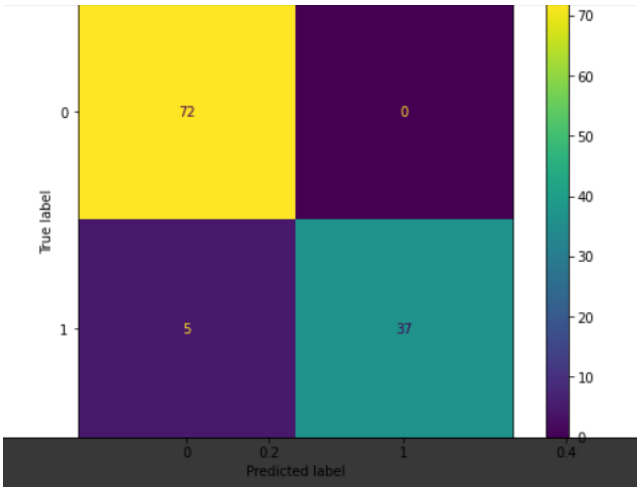
The second highest performance classifier is SVM with 96.49% of accuracy and C parameter of 10. The result confusion matrix for this classifier are being shown below:

```
Best C: 10
The accuracy of SVM:0.9649122807017544
```



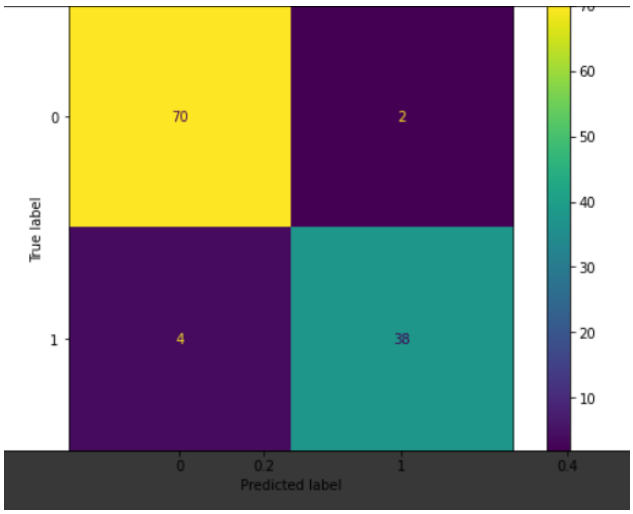
Later, we have the accuracy of Nearest Neighbours Classifier at 95.61% with n_neighbors of 11. The result and confusion matrix are as shown below:

```
Best n_neighbors: 11
The accuracy of KNN:0.956140350877193
```



Last, with the lowest accuracy of four classifiers is the Decision Tree Classifier with the accuracy of 94.74% and minimum sample leaf of 4. We can see the result and confusion matrix below:

```
Best sample leaf: 4
The accuracy of decision tree classifier:0.9473684210526315
```



All four classifiers have a very high accuracy and only about 1% different each. This means that the data can be predicted accurately on the test data and the models that are being used fits the data very well.