1 Data

1.1 Describing the data

The MNIST dataset for handwritten numbers, is made up of 10 classes, with a total of 1797 total samples. It is made up of an eight by eight image of a number. All input attributes are integers within the range of zero to sixteen, with the last attribute being zero to nine [1]. An example of the number zero from the MNIST database is shown in figure 1. Figure 2 shows array value when the database is loaded.

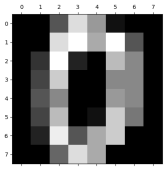
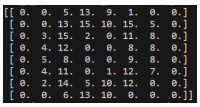


Figure 1: Image for zero



```
[[ 0.  0.  5. 13.  9.  1.  0.  0.]
 [ 0.  0. 13. 15. 10. 15.  5.  0.]
 [ 0.  3. 15.  2.  0. 11.  8.  0.]
 [ 0.  4. 12.  0.  0.  8.  8.  0.]
 [ 0.  5.  8.  0.  0.  9.  8.  0.]
 [ 0.  4. 11.  0.  1. 12.  7.  0.]
 [ 0.  2. 14.  5. 10. 12.  0.  0.]
 [ 0.  0.  6. 13. 10.  0.  0.  0.]]
```

Figure 2: Array value for zero

1.2 What is done with the data ?

I saved the dataset into two arrays, y, which holds the labels for each of the numbers, for example, the image for the number one, has the label 1. The variable X, holds all the data, which is an 8 by 8 array, containing the layout of the number. Using train_test_split the data was split into testing and training data.For this we use a training size of eighty five percent as well as a test size of 15 percent. With this we can then use these to train out classifiers using the sklearn method fit in order to build a classifier. This is done for the decision tree, random forest and nearest neighbour classifiers. For the classifiers as well as training and testing of the data, I used the seed 97, to ensure the repeatability of results.

2 Classifiers

The classifiers that were chosen to be tested were decision trees, random forest and nearest neighbour. Each classifier takes the training data, looks through its features, and looks for patterns. Each classifier method has its own rules to find accurate predictions.After that the code uses the test set in order to evaluate the performance for the trained classifiers, and when the classifier is applied to the testing set, with its predictions then being compared to that of the real values, from the dataset.[2]

Once this is done it can then be used to predict the labels of future values. We used cross validation [4] score, which is a Kfold validation [5] method to estimate the skill of the dataset on new data [3].

2.1 Outputs

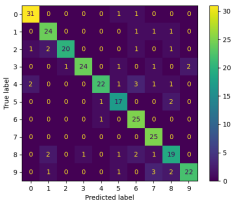


Figure 3: Confusion matrix for decision trees

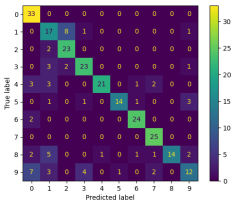


Figure 4: Confusion matrix for random forest

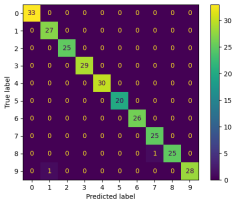


Figure 5: Confusion matrix for nearest neighbour

2.1.1 Decision tree

max depth	6	7	8	9	10
Mean Accuracy	0.66	0.73	0.76	0.76	0.78
Standard Deviation	0.12	0.12	0.11	0.10	0.07

For the decision tree classifier, the mean accuracy turned out to be seventy eight percent, which is the lowest out of every classifier, with the confusion matrix figure 3 showing that there is a higher proportion of predictions that were incorrect, in comparison to the other classifiers. The table shows that when increasing the max depth from 6 to 10, the mean accuracy would increase and the standard deviation would decrease as shown in figure 6, showing that to get the most precise prediction, however increasing the max depth would increase over fitting, increasing the amount of testing errors as it will overlook patterns. Decision trees are used in random forest, and more efficient as it does not need the same amount of computational power as the other 2 classifiers.

2.1.2 Random forest

max samples	6	7	8	9	10
Mean Accuracy	0.66	0.71	0.74	0.74	0.78
Standard Deviation	0.13	0.16	0.11	0.11	0.10

This classifier fits a number of decision tree classifiers on a number of sub data sets, using the average to improve the prediction and to control over-fitting, with sample size controlled with max_samples. As we see in the confusion matrix in figure 4, for 10 samples, it shows that there is a seventy eight percent, however, increasing the number of samples, increases the mean accuracy as well as decreases the standard deviation, while not having the same problem as Decision tree classifiers, but needs substantially more computational power, as it is making multiple decision trees, as well as using their average to increase accuracy. This is shown visually in figure 7.

2.1.3 Nearest neighbour

No. Neighbours	6	7	8	9	10
Mean Accuracy	0.97	0.97	0.97	0.96	0.96
Standard Deviation	0.03	0.03	0.04	0.04	0.04

KNN is a lazy learner, needing next to no training time, so instead relying on memorising, does not train the parameters. It is the most complex classifier needing the most computational power, resulting in a high complexity, however, this is also the most accurate out of the three classifiers I have chosen, showing the highest mean accuracy, and boasting the lowest standard deviation. As for the effects on the the accuracy, according the to the table as well as the graph in figure 8, the lower the number of neighbours the higher the accuracy. This is as the closer samples have similar classes.

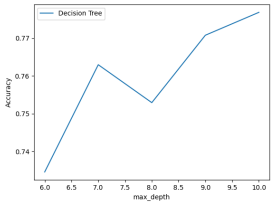


Figure 6: Max depth v accuracy graph

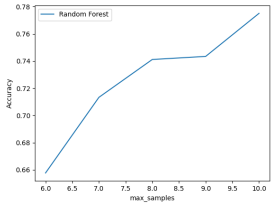


Figure 7: max samples v accuracy graph

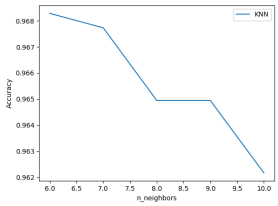


Figure 8: number of neighbours v accuracy graph

3 Evaluation

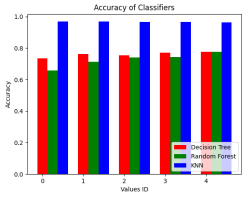


Figure 9: accuracy of each classifier

Looking at Figure 9 in terms of overall accuracy, It would be best to use KNN classifier, to get the most accurate results. for decision trees, there is a risk of over fitting, if the depth is too high, and with random forest, although the classifier can be extremely accurate, it requires a lot of samples in order to be able to reach a high precision, KNN is relatively high even when the variables are changed. Therefore, with decision tree having the lowest accuracy overall, and random forest, needing a large number of samples in order to reach a comparable accuracy and standard deviation to that of KNN, even though KNN needs the highest computational power it is the most accurate and precise.

References

[1] UC Irvine. *Optical Recognition of Handwritten Digits*. https://archive.ics.uci.edu/dataset/80/optical+recognition+of+handwritten+digits. 1998.

[2] Kashish. *KNN vs Decision Tree vs Random Forest for handwritten digit recognition*. https://medium.com/analytics-vidhya/knn-vs-decision-tree-vs-random-forest-for-handwritten-digit-recognition-470e864c75bc.

[3] Shanthababu Pandian. *K-Fold Cross Validation Technique and its Essentials*. K-Fold Cross Validation Technique and its Essentials.

[4] Sklearn. https://scikit-learn.org/stable/modules/generated/sklearn.model$_s election.cross_v al_s core.html$.

[5] sklearn. https://scikit-learn.org/stable/modules/generated/sklearn.model$_s election.KFold.html$.