----------------------------------------------------------------------------------------------------------------

Usage:

comparison.py contains the answer from decision tree to multilayer perception

$ python comparison.py

comparison\_split.py contains the answer for data split

$ python comparison\_split.py

In both programs, from line 86 to 95, different classifier can be chosen by commenting and uncommenting the line. It arranges in the same order of the assignment

-----------------------------------------------------------------------------------------------------------------

2.2

Decision Tree

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Classifier | precision | accuracy | f1 score | recall | training time(s) |
| Gini  (depth = 5) | 0.68 | 0.697 | 0.67 | 0.70 | 1.875 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Classifier | precision | accuracy | f1 score | recall | training time(s) |
| Entropy  (depth = 10) | 0.80 | 0.793 | 0.79 | 0.79 | 7.223 |

When comparing Gini versus entropy, entropy perform better with a higher precision, accuracy, f1 score and recall. The reason is that entropy allows the maximum tree depth to be 10 while Gini only 5. A limited tree depth may cause underfitting when there are a lot of attributes (in this case, a 28 x 28 matrix), which decreases the accuracy and other index.

In terms of running time, entropy is expected to have longer running time due to logarithm computation, while Gini is simpler and faster without such computation.

KNN, SVM, Random Forest

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Classifier | precision | accuracy | f1 score | recall | training time(s) |
| KNN  (neighbors=5) | 0.84 | 0.836 | 0.84 | 0.84 | 17.06 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Classifier | precision | accuracy | f1 score | recall | training time(s) |
| SVM  (gamma=0.001,  kernel = "poly") | 0.84 | 0.837 | 0.84 | 0.84 | 20.10 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Classifier | precision | accuracy | f1 score | recall | training time(s) |
| Random Forest  (depth = 10) | 0.81 | 0.815 | 0.81 | 0.81 | 0.815 |

After changing the kernel from default “linear” to “poly”, these three classifiers have similar high precision (>80%). There is only 0.1 precision if using “linear” kernel.

As SVM needs to tackle polynomial equations with n unknowns, where n is the number of attributes, SVM is generally slower than KNN and Random Forest. As mention in sklearn document, the SVM fit time complexity is more than quadratic with the number of samples which makes it hard to scale to dataset with more than a couple of 10000 samples. For Random Forest, since it uses Gini index to split, it is generally faster without expensive computation like cracking equation or logarithm in SVM and entropy respectively. For KNN, as Euclidean distances are computed to selected 5 nearest neighbors for classification which including square root computation. Therefore, the running time will be longer than Random Forest but still faster than SVM.

Multilayer perception

1 hidden layer, 50 neurons

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Classifier | precision | Accuracy | f1 score | recall | training time(s) |
| MLP | 0.77 | 0.762 | 0.76 | 0.76 | 15.11 |

1 hidden layer, 100 neurons

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Classifier | precision | Accuracy | f1 score | recall | training time(s) |
| MLP | 0.82 | 0.813 | 0.81 | 0.81 | 24.76 |

2 hidden layers, [100, 10] neurons

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Classifier | precision | accuracy | f1 score | recall | training time(s) |
| MLP | 0.06 | 0.135 | 0.06 | 0.14 | 10.77 |

2 hidden layers, [50, 20] neurons

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Classifier | precision | accuracy | f1 score | recall | training time(s) |
| MLP | 0.76 | 0.763 | 0.74 | 0.74 | 15.34 |

MLP Classifier gives an interesting and unstable result. When comparing 1 hidden layer with 2, 1 hidden layer generally has a much higher and more stable precision. Although 2 hidden layers with 50 and 20 neurons seems good, its precision various greatly range from about 0.4 to 0.7. The reason causes high error in 2 hidden layers is due to overfitting and high variance in over-complex model. 1 hidden layer gives more general result.

Data Split

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Classifier | precision | accuracy | f1 score | recall | training time(s) |
| KNN  (neighbors=5) | 0.82 | 0.814 | 0.81 | 0.81 | 16.09 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Classifier | precision | accuracy | f1 score | recall | training time(s) |
| SVM  (gamma=0.001,  kernel = "poly") | 0.83 | 0.829 | 0.83 | 0.83 | 19.82 |

When comparing to non-split data, their precision, accuracy, f1 score and recall are similar. The reason is due to large training set of input. There are 10000 training images and 1000 test images for learning 10 digits. Merge and split them may not have significant effect on training which turn out have similar result.