Machine Learning for Computer Vision Coursework 3

Hao Ding (CID: 00734091), Tanat Sanpaveeravong (CID: 00733970),
Department of Electrical and Electronic Engineering, Imperial College London, SW7 2AZ
Email: hd1812@ic.ac.uk, ts3912@ic.ac.uk

PART 1: TRAINING DECISION FOREST

Bagging (Boostrap Aggregating):

Given a training data set, multiple data subsets can be generated using the Bagging method (Bootstrap Aggregating algorithm). Bagging is a special case of model averaging, which trains multiple models with bootstrapped dataset. This method is commonly used to obtain a better prediction accuracy as it reduces the variance through averaging the results from obtained from each model. Bootstrap refers to the technique used to construct additional data sets. This is particularly useful when training data is not sufficiently large. By sampling uniformly with replacement from the original training set, an additional data set is created as an estimator of the original data set. Some samples in the each data subset may be repeated. If the size of a data subset is equal to the size of the original training set, then 63.2% $(1 - \frac{1}{e})$ of the observations in that data subset is expected to be unique while the rest are duplicates.

Figure 1 shows four of the data subsets obtained from Bagging. The algorithm is configured in such a way that the size of the each data subset is the same as that of the training data set. It can be observed that within all four data sets, there are a few duplicate samples as expected.

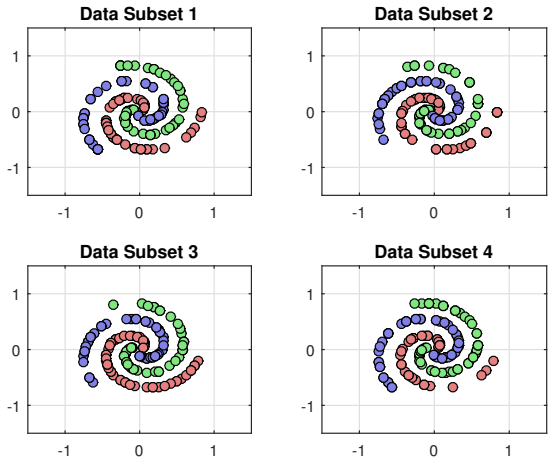

Fig. 1. Four data subsets obtained from bootstrapping

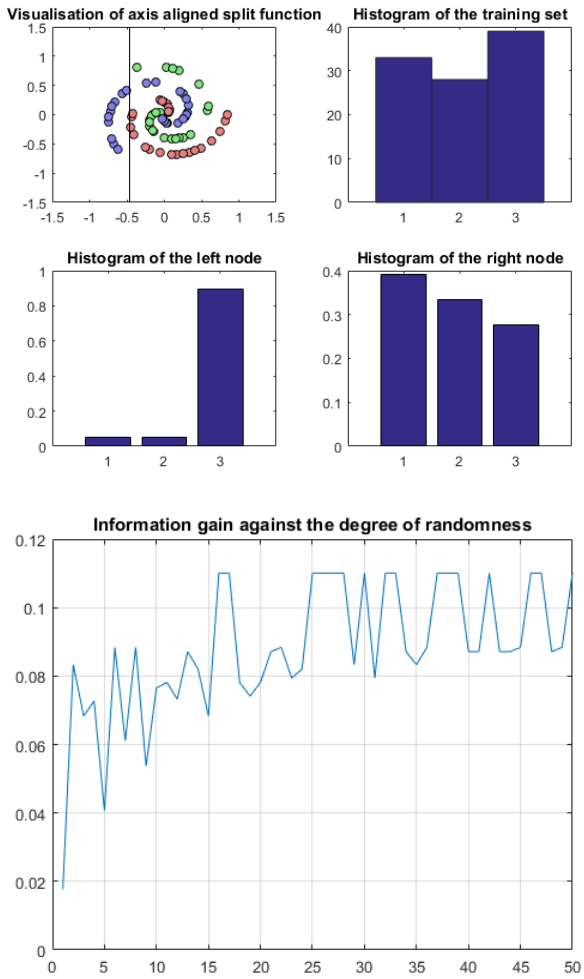

Fig. 2. Axis-aligned weak learner and histograms of the child nodes (top). The Information gain against the degree of randomness(bottom).

Splitting Functions for Decision Tree:

Using one of the data subset generated, a decision tree can be grown by splitting this data subset, which is represented as a root node. Several feature splitting functions (weak leaner) and their corresponding threshold values will be investigated. The results obtained from different split functions and threshold values will be compared in term of information gain.

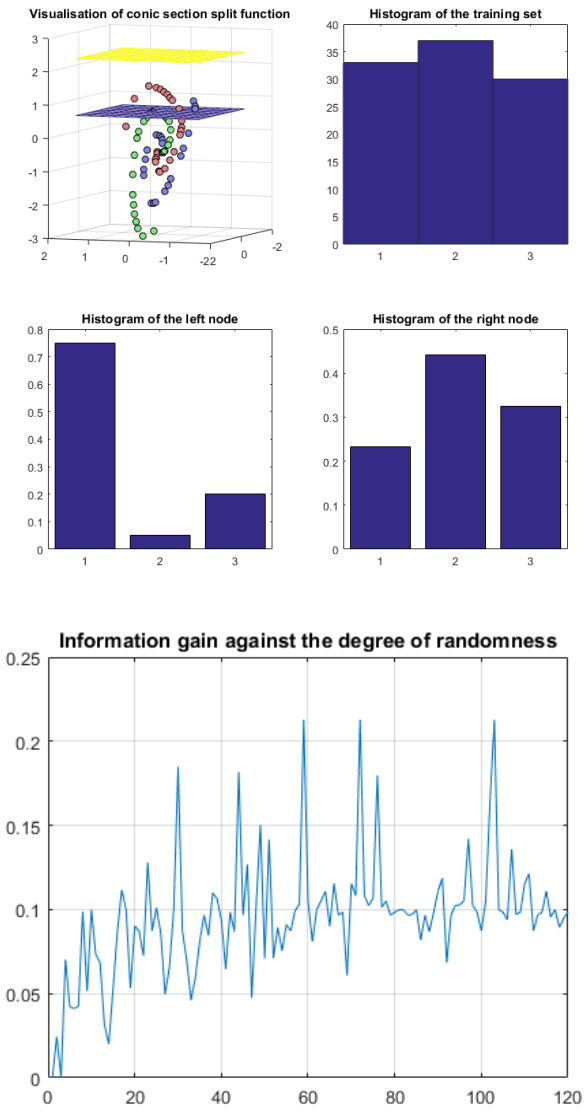

Fig. 3. Conic section weak learner and histograms of the child nodes (top). The Information gain against the degree of randomness(bottom).

To start with, let us consider the axis-aligned weak learner. This split function uses a one dimensional plane to split data in the root node of the tree. The first four plots of Figure 2 shows the respective class histograms of the root node and its two child nodes. The splitting threshold (degree of randomness ρ) is determined by randomly select N values and select the one that yields the highest information gain. In this experiment, the degree of randomness is set to 5, i.e, try 5 thresholds before determining the best value. The relationship between the information gain and the number of attempts is also in bottom plot of Figure 2. It can be observed that the information gain increase as the degree of randomness increases. However, it converges after a certain value of ρ is reached. It is at the convergence where the information gain is maximised. Thus, there is no need to increase ρ beyond the value that allows the information gain to converge since higher degree of randomness implies high computational complexity. From the information gain plot in Figure 2, setting the randomness to 10 seems to give a good trade-off between complexity and information gain.

The next split function that will be discussed is the conic section learner, which is a non-linear hyperplane. It gives a relatively good performance while incurring low computational complexity. The first plot of Figure 3 illustrate this split function while the next three plots shows the respective class histograms of the root node and the two child nodes. The bottom plot shows the information gain curve obtained using the conic section learner. It can be observed that the conic section learner has a larger possibility to obtain higher information gain than the axis-aligned split function.

Apart from the two split functions that have already been discussed, there are also other split functions that were implemented and tested in this experiment. These split functions include: 2D linear decision line, distance learner and two-pixel test. The 2D linear decision line is similar to the axis-aligned weak learner but it uses the 2D surface to split data. The distance learner uses the distance between test points and the base point to categorise data. The two-pixel test randomly selects two input features and uses the difference to split data. Their performance will be discussed at the end of part 2.

Tree Growth & Stopping Criteria:

By recursively splitting the nodes, a complete decision tree can be constructed. The Axis-aligned split function is selected for this task and the degree of randomness is set to 10. Figure 4 shows class distribution of some leaf nodes.

In order to determine a stopping criteria, which terminates the tree growth, several conditions can be considered Theoretically, the tree building process will be terminated when a split is pure, meaning that all members in the left or right child node are the same output value. However, since a large number of splits is usually required to achieve this, the process can become computationally expensive and potentially lead to overfitting. Alternatively, the stopping criterion can be determined using the information gain. If the information gain obtained from a splitting process is negative, the tree building should be stopped since the splitting will no longer provide information. Again, this process can be computationally expensive. Therefore, instead of stopping when the information gain is negative, the building process is terminated when the depth of the tree reaches an arbitrary threshold, which is set to 5. The reason why 5 is chosen is that this dept already gives a reasonable trade-off between the complexity and information gain obtained. Additionally, it is very unlikely that this dept will cause overfitting.

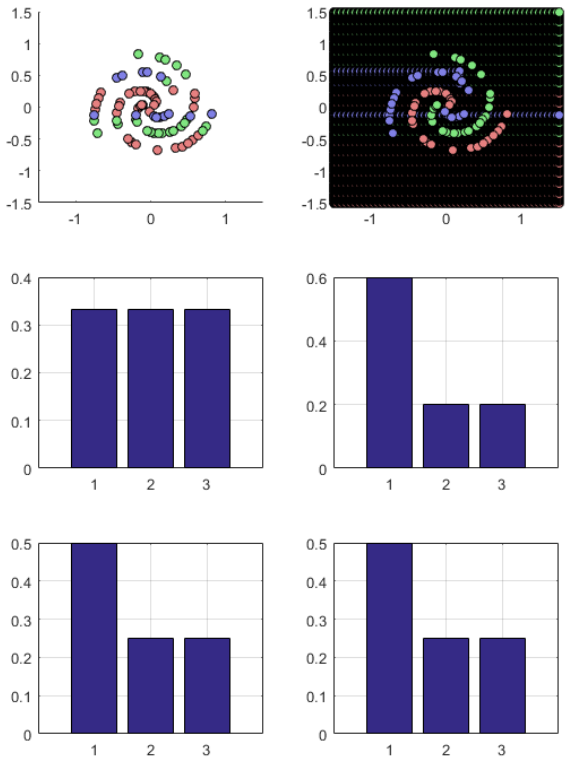

Fig. 4. Tree grown by Axis-aligned weak learners. Classification results (Top-left), Testing result (Top-right), samples of child node histograms.

PART 2: EVALUATING DECISION FOREST ON THE TEST DATA

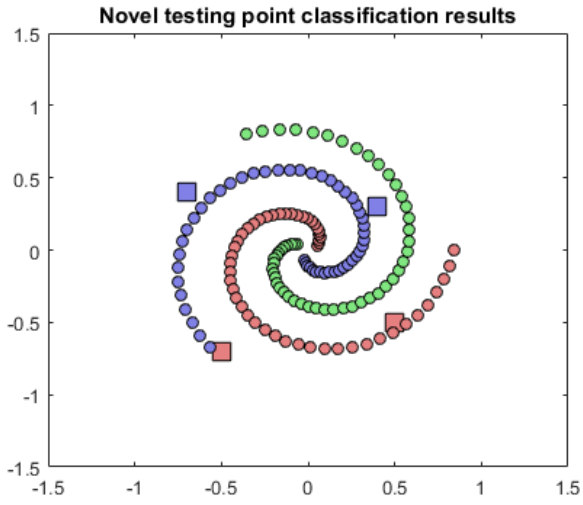

Fig. 5. Classification results for novel test points.

The axis-aligned weak learner is selected to grow the decision tree due to its good performance and low computational complexity. The degree of randomness is set to 10 in order to obtain a reasonable information gain for each split. Figure 5 shows the classification result for some novel test points selected near the training dataset. It can be observed that the random forest has shown reasonable performance on the close testing points. The test point on the bottom left is an example of misclassification.

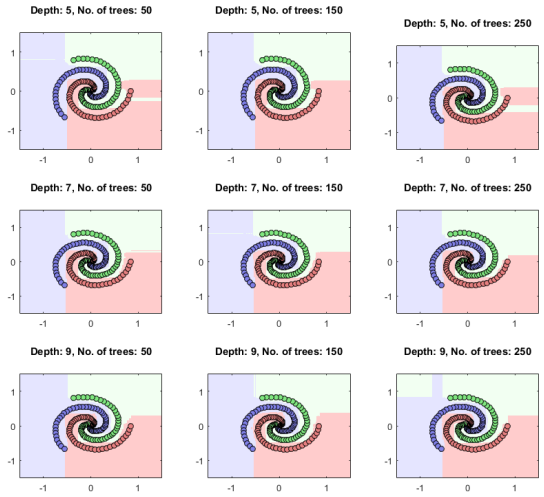

Fig. 6. Visualisation of classification results using axis-aligned weak learner with different parameter settings

Figure 6 shows the effect of varying tree dept and the number of trees used. It can be observed that larger tree depth value increases the training accuracy. This is due to the fact that the classification results from various decision trees are averaged. At the same time, increasing the dept also increases the computational complexity significantly. It should be noted that for a tree depth of d, the number of node is $2^{d-1} - 1$.

The training accuracy for these nine tests are close to 1. It can be observed that the random forest generally perform well on the training data, giving very high prediction accuracy. However, for testing data, especially those further awary from the spiral training data, the axis-aligned split function cannot perform classification well. This is due to the property of the spiral data that it is not linearly separable using the axis-aligned split function.

For the purpose of comparison, non-linear weak learners are also used with the the spiral data. These learners include the conic section learner and an distance learner. As mentioned earlier, the distance learner uses the difference between test points and a base point to categorise data. Thus, the decision boundary will be a circle. it should be noted that both of these algorithms incur high computational complexity.

From Figure 7 and Figure 8, it can be observed that when the non-linear weak learners are used, the classification becomes more accurate. This is expected as the data is also non-linear in nature. The distance learner is shown to

perform particularly well on this specific task. Looking at the right plot of Figure 9, the 2D dcision leaner also gives good classification result while the two-pixel test shows poor performance as can be seen in the left plot.

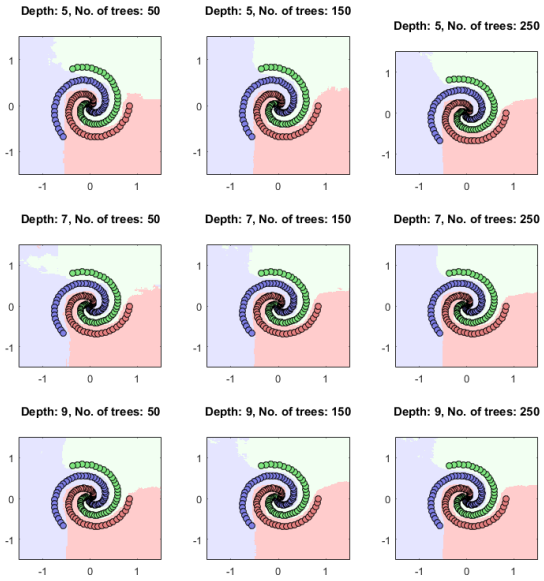

Fig. 7. Visualisation of classification results using conic section weak learner with different parameter settings

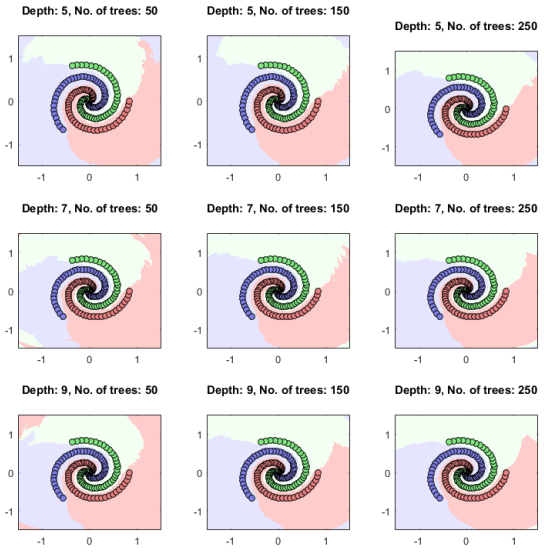

Fig. 8. Visualisation of classification results using distant weak learner with different parameter settings

Now, let use compare the performance of the SVM and random forest. For ths spiral data classification task, the SVM shows and the random forest shows similar performance. The spiral data is closely clustered to each other and is not linearly separable. The non-linear functions, in particular

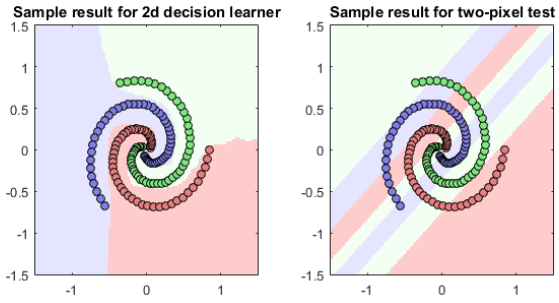

Fig. 9. Sample result of 2D decision learner (Left) and two-pixel test (Right)

the RBF kernel in SVM and the distance learner in random forest, show outstanding performance compared to other functions. As mentioned earlier, this is due to the non-linear nature of the spiral training data. In terms of computational complexity, random forest is generally faster than the SVM.

PART 3: EXPERIMENT WITH CALTECH 101 DATASET FOR IMAGE CATEGORISATION

The random forest algoirthm is applied to the subset of Caltech101 dataset for image categorisation. Its performance will be compared to that of the SVM. The images are converted to high dimensional vectors using the bag-of-words technique. The random forest algoirthm is then applied in the same way as in the previous case (toy spiral data experiment). It should be noted that the dataset and bag-of-word representations used are the same as in the previous coursework (coursework 2 on BoW and SVM). In this part, the axis-aligned weak learner and the two-pixel weak learner will be investigated.

Figure 10 shows a 3D plot of the prediction accuracy against various parameters. Each two-pixel test weak learner selects two dimensions and compute the difference between the two. It can be seen that the accuracy obtained from the random forest algorithm (75.3%) is slightly higher the accuracy obtained from the SVM that uses RBF kernel and one-against-all algorithm (74.7%). The confusion matrix in 10 also shows satisfying result. It should be noted that unlike the non-linear kernels, it is not very likely that random forest is less likely to cause overfitting. Since the Caltech 101 data set contains only 15 training images for each class and each training data has 75 dimensions, the number of training image is considered to be small. This means that the data is sparse and so overfitting is not very desirable. Therefore, random forest gives a more accurate result. An example of success and failure can bee seen in Figure 11.

The performance of the axis-aligned weak learner has been shown in Figure 12. The prediction accuracy of the axis-aligned learner (74.67%) is shown to be very close to that of the SVM that uses RBF kernel and one-against-all algorithm. The associated confusion matrix and the example of success and failure can bee seen in Figure 12 and 13 respectively.

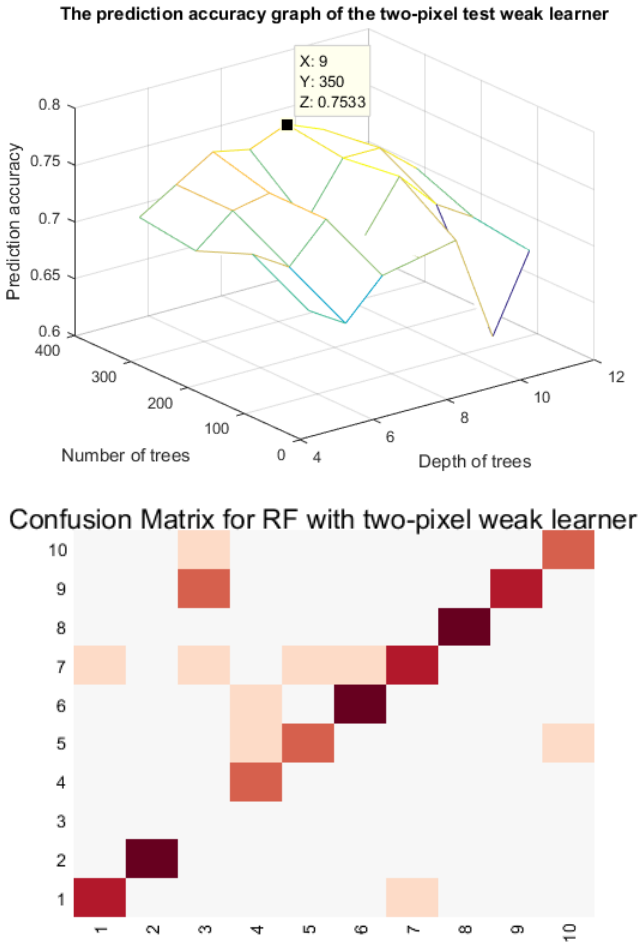

Fig. 10. Accuracy plot (Top) and confusion matrix (Bottom) for two-pixel weak learner.

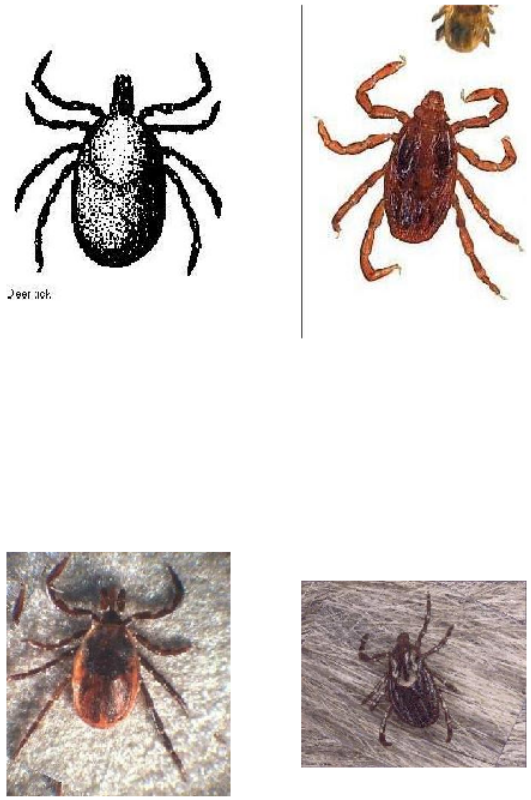

Fig. 11. The example of successful (Top) and failed (Bottom) classification

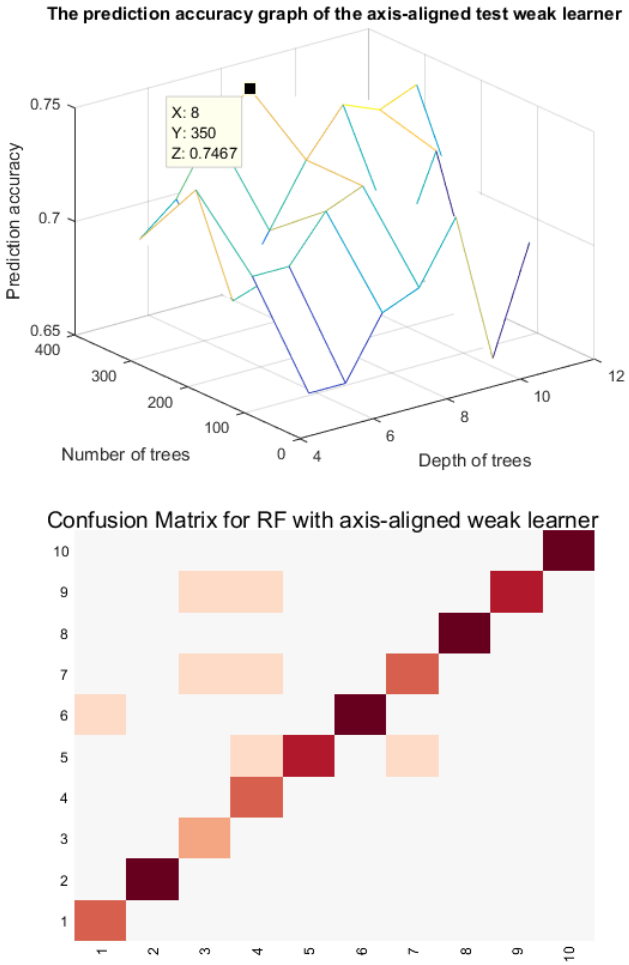

Fig. 12. Accuracy plot (Top) and confusion matrix (Bottom) for axis-aligned weak learner.

Fig. 13. The example of successful (Top) and failed (Bottom) classification