**Section 1.2**

1. Confusion matrix:   
   [[ 59 4]  
    [ 2 106]]   
   Accuracy: 0.9649122807017544   
   Precision: 0.9636363636363636   
   Recall: 0.9814814814814815   
   Average 10-fold cross validation accuracy: 0.9067355889724309
2. The accuracy is really just a measurement of all the correct classifications. The precision is calculated with: (true positive / (true positive + false negatives)), giving us a better insight into how we’re predicting. The recall is calculated with: (true positive / (true positive + false negatives)), giving us the ratio of true positives / relevant elements. For the class distribution makes the precision and recall important when false positives / negatives are crucial. Therefore, the precision and recall give a better measure of the incorrectly classified cases.
3. The 10-fold cross validation generates 10 different models and returns 10 accuracy scores. The accuracy, on the other hand, just gives us an accuracy score for this particular test data. The CV accuracy gives us an insight into how accurate we might predict other new data. Accuracy returns the accuracy for this particular run, while the CV accuracy gives us accuracy for upcoming runs. Therefore, the CV method generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split. But the reason for the accuracy actually being higher than cross\_val is the train\_test\_split(), where we suffer from selection bias using the function.
4. Confusion matrix: Sum up all the matrices into one. At each fold, a confusion matrix from the test set can be generated and all such confusion matrices can be added to give the overall confusion matrix.   
   Precision and recall: For the CV method, each fold is a model with its own precisions and recall, so taking the average of them to get a mean performance metric over all the folds can be done. But one thing to note, since recall is the proportion of true positives out of all positives, each fold should be weighted by the number of positives.

**Section 2.1**

1. Confusion matrix:   
   [[ 60 3]  
    [ 0 108]]   
   Accuracy: 0.9824561403508771   
   Precision: 0.972972972972973   
   Recall: 1.0   
   Average 10-fold cross validation accuracy: 0.9068609022556391
2. Max\_features = 'sqrt' n\_estimators >= 10. Adding a lot of trees can slow down the training process considerably.   
   Metric results:   
   Confusion matrix:   
   [[ 61 2]  
    [ 1 107]]   
   Accuracy: 0.9824561403508771   
   Precision: 0.981651376146789   
   Recall: 0.9907407407407407

Chart, histogram

Description automatically generated**Section 2.2**

**Section 2.3**

1. Feature importance is calculated as the decrease in node impurity weighted by the probability of reaching that node. The node probability can be calculated by the number of samples that reach the node, divided by the total number of samples. Higher value means that the feature is more important.
2. ['mean radius' 'mean texture' 'mean perimeter' 'mean area'  
    'mean smoothness' 'mean compactness' 'mean concavity'  
    'mean concave points' 'mean symmetry' 'mean fractal dimension'  
    'radius error' 'texture error' 'perimeter error' 'area error'  
    'smoothness error' 'compactness error' 'concavity error'  
    'concave points error' 'symmetry error' 'fractal dimension error'  
    'worst radius' 'worst texture' 'worst perimeter' 'worst area'  
    'worst smoothness' 'worst compactness' 'worst concavity'  
    'worst concave points' 'worst symmetry' 'worst fractal dimension']   
     
   [7, 27, 20, 22, 23, 6, 26, 13, 2, 3, 0, 21, 12, 1, 24, 10, 25, 29, 16, 19, 4, 28, 11, 5, 15, 18, 8, 17, 14, 9]   
     
   Most important; feature 7; mean concave points   
   Least important: feature 9; mean fractal dimension

Chart

Description automatically generated

**Section 2.4**

**Section 2.5**

1. As the number of estimators increase the out of bag error decreases and some people use the out of bag error to decide how many estimators they want for their random forest but after a certain point it fluctuates again so there seems to be some interval where it's the most stable. Growing a larger forest will improve predictive accuracy, although there are usually diminishing returns once you get up to several hundreds of trees.
2. Yes, there is some n\_estimators interval, for all of the ensembles, where the OOB error stabilises. It makes sense in a way, when you're encountering new data, the amount of n\_estimators you use isn't the same amount you used for some other dataset. You'll have to find that new interval where you get the most stabilised answers.

Chart, bar chart, histogram

Description automatically generated**Section 3.1**

1. Confusion matrix:   
   [[ 60 3]  
    [ 0 108]]   
   Accuracy: 0.9824561403508771   
   Precision: 0.972972972972973   
   Recall: 1.0   
   Cross validation accuracy: 0.9122180451127818
2. ['mean radius' 'mean texture' 'mean perimeter' 'mean area'  
    'mean smoothness' 'mean compactness' 'mean concavity'  
    'mean concave points' 'mean symmetry' 'mean fractal dimension'  
    'radius error' 'texture error' 'perimeter error' 'area error'  
    'smoothness error' 'compactness error' 'concavity error'  
    'concave points error' 'symmetry error' 'fractal dimension error'  
    'worst radius' 'worst texture' 'worst perimeter' 'worst area'  
    'worst smoothness' 'worst compactness' 'worst concavity'  
    'worst concave points' 'worst symmetry' 'worst fractal dimension']   
     
   [27, 20, 7, 6, 26, 23, 2, 3, 0, 22, 13, 21, 5, 25, 10, 12, 1, 24, 28, 16, 8, 4, 17, 29, 14, 15, 11, 18, 9, 19]   
     
   Most important: Feature 27; worst concave points   
   Least important: Feature 19; fractal dimension error

**Section 3.2**

Chart

Description automatically generated