CITS5508 Machine Learning - Assignment 2

Franco Meng 23370209

Outline:

This report is aimed at building a binary classification algorithm by using decision tree and random forrest. The dataset was from on the Breast Cancer Wisconsin (diagnostic) available on Scikit-Learn, where containing:

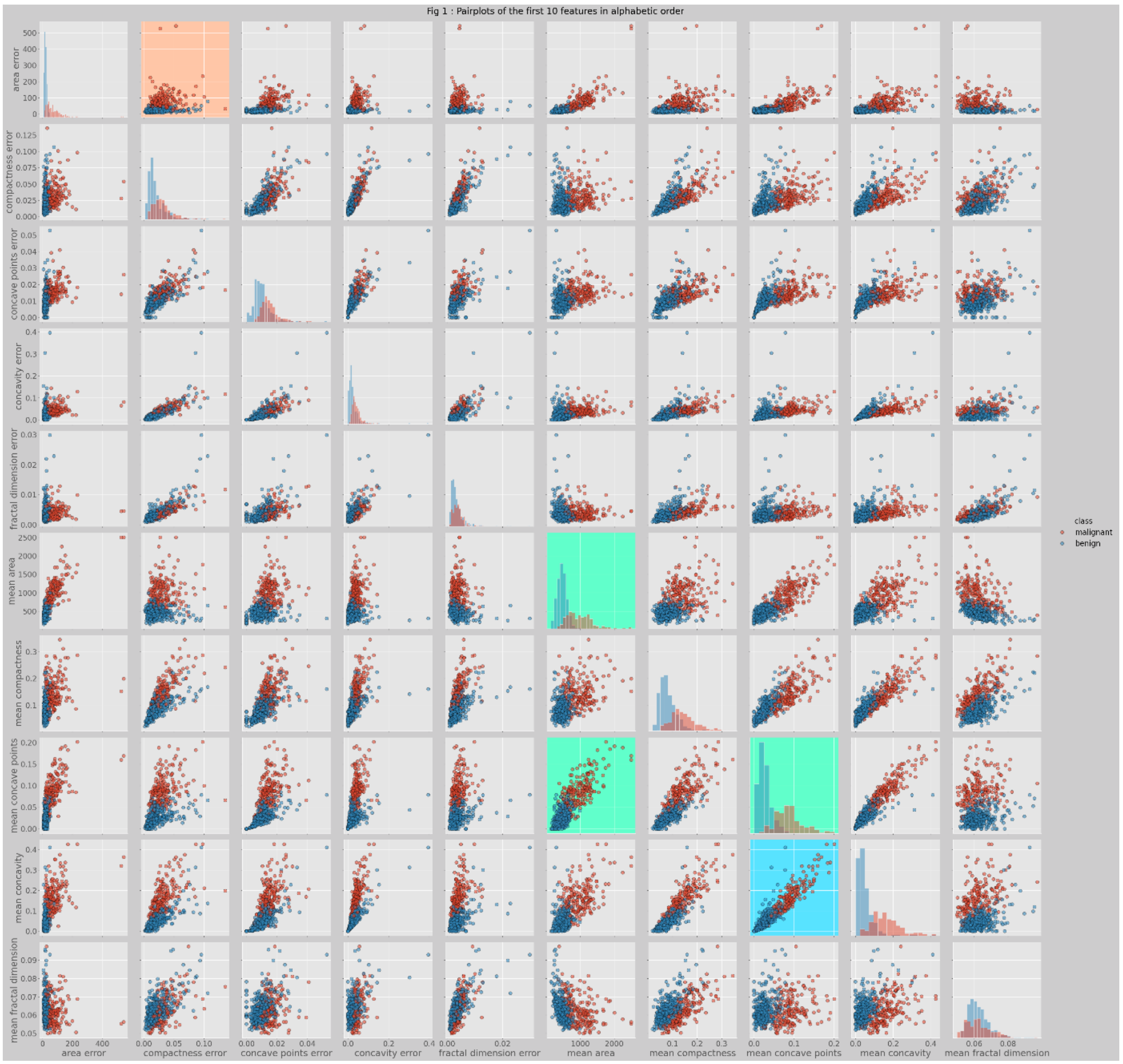
- malignant (212 instances, class value 0)
- benign (357 instances, class value 1).

569 instances in total, with 30 features for each instances.

Deliverables:

D1: Re-order the columns in feature matrix. Provide a scatter plot inspect first 10 features of the data. Use different colour for each classes.

- Answer: figure 1.



D2: Provide comment for above.

1. There are some highly correlated features, when the certain pair plots look like a narrow diagonal line, for example, the blue cell of the plots showing a strong correlation between mean concave point with mean concavity.

2. The presence of clustered groups is evident, the green cell shows malignant class (red scatters) appears having greater measurements in both 'mean area' and 'mean concave points'. It can also be seen from

the histogram of these corresponding features, where malignant class (the red bars) are mostly on the right-hand side of benign class (the blue bars)

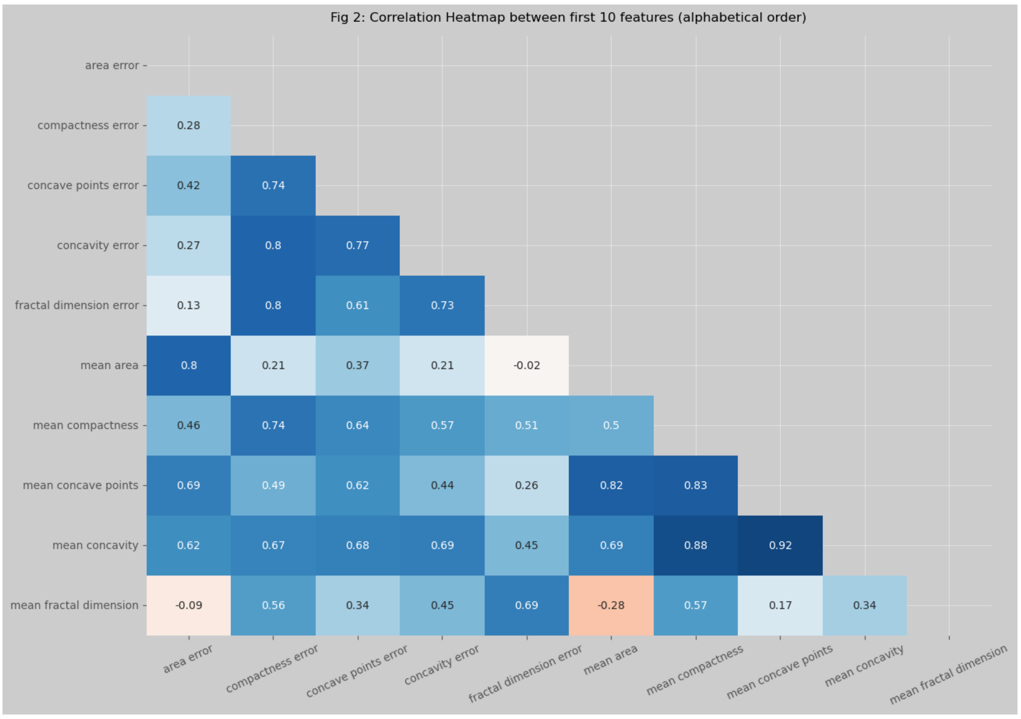
3. Yes, for example, the orange cell showing a couple of possible outliers in feature area error may existed in malignant class instances, where majority of area error is under 200, but these possible outlier are over 500.

4. Maybe highly correlated features could be removed, e.g. the blue cell, we may keep one of the features (concave point or mean concavity), as these features may serve very similar functionality in the decision tree slitting.

   However, due to the nature of the decision tree algorithm, it will perform feature selection during the training process, as calculating the metrics like Gini impurity or information gain by considering each feature individually, if two feature (highly correlated) providing same effectiveness of loss reduction, the algorithm would use either feature (depends on random state seed). Removing feature may still speed up the computational time though, but overall, it is not very crucial that we have to deal with multicollinearity issue in data preparation stage, unlike some linear regression models.

D3: Correlation matrix between the corresponding pair of features.

- Answer : Figure 2.



D4: Correlation map vs Observation in D2.

- Answer: As shown in Figure 2, the correlation coefficients support my previous observations, where the feature mean concave points and mean concavity has the highest correlation of 0.92. And many highly correlated features where the correlation coefficients are over 0.7 or 0.8.

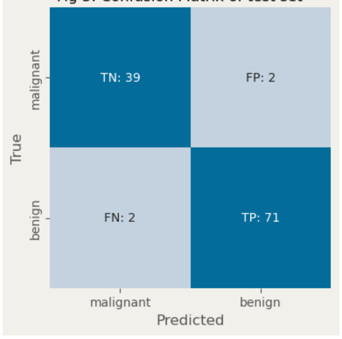
D5. Drop the features: mean perimeter, mean radius, worst radius, worst perimeter, and radius error.

D6. Fit a decision tree classifier using default hyperparameters using 80% of the data, random state = 5508. Provide accuracy, precision and recall scores for training and testing, along with testing set confusion matrix.

- Answer: Figure 3

	Training Set:	Test Set:
Accuracy:	1.00	0.96
Precision:	1.00	0.97
Recall:	1.00	0.97

Fig 3: Confusion Matrix of test set



D13. Comment

- Answer: it is interesting to see the FP and FN both have increased comparing with D6, without fine tuning hyper parameters. However as Fine-tuning process using all the combination of the hyperparameters candidates, along with the cross validation to result in an overall better model. Simply by comparing the result, there are only 1 or 2 increase in the misclassification, however this doesn't mean the model has not been improved. This new model with optimal hyperparameter most likely will generalise better with other new and bigger testing size.
- Moreover, With 3 being the max_depth, it is definitely more preferred comparing the depth of 8 in D6 overfitted model ..

D8. Visualising the Decision tree model

- Answer: Figure 4 as below.

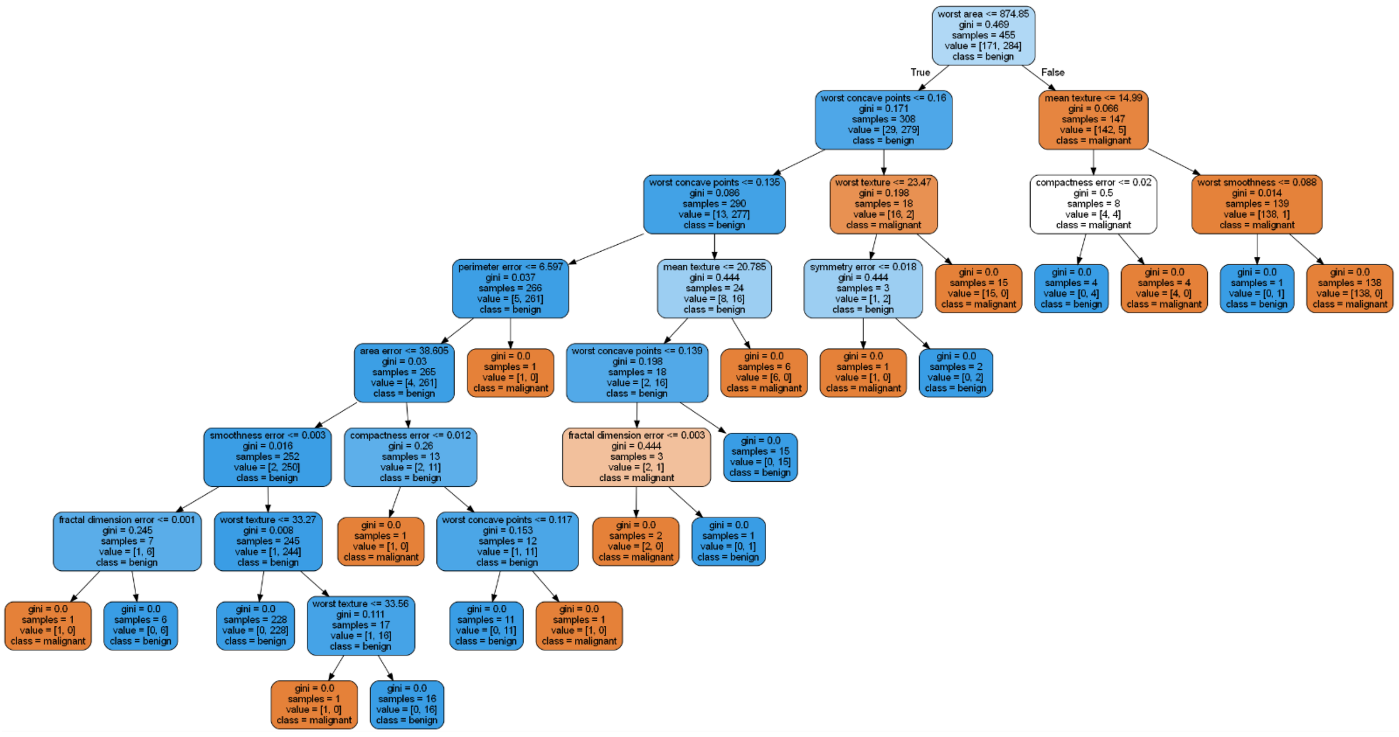

worst area <= 874.85 gini = 0.469 samples = 455 value = [171, 284] class = benign
True
False
worst concave points <= 0.16 gini = 0.171 samples = 308 value = [29, 279] class = benign
mean texture <= 14.99 gini = 0.066 samples = 147 value = [142, 5] class = malignant
worst concave points <= 0.135 gini = 0.086 samples = 290 value = [13, 277] class = benign
worst texture <= 23.47 gini = 0.198 samples = 18 value = [16, 2] class = malignant
compactness error <= 0.02 gini = 0.5 samples = 8 value = [4, 4] class = malignant
worst smoothness <= 0.088 gini = 0.014 samples = 139 value = [138, 1] class = malignant
perimeter error <= 6.597 gini = 0.037 samples = 266 value = [5, 261] class = benign
mean texture <= 20.785 gini = 0.444 samples = 24 value = [8, 16] class = benign
symmetry error <= 0.018 gini = 0.444 samples = 3 value = [1, 2] class = benign
gini = 0.0 samples = 15 value = [15, 0] class = malignant
gini = 0.0 samples = 4 value = [0, 4] class = benign
gini = 0.0 samples = 4 value = [4, 0] class = malignant
gini = 0.0 samples = 1 value = [0, 1] class = benign
gini = 0.0 samples = 138 value = [138, 0] class = malignant
area error <= 38.605 gini = 0.03 samples = 265 value = [4, 261] class = benign
gini = 0.0 samples = 1 value = [1, 0] class = malignant
worst concave points <= 0.139 gini = 0.198 samples = 18 value = [2, 16] class = benign
gini = 0.0 samples = 6 value = [6, 0] class = malignant
gini = 0.0 samples = 1 value = [1, 0] class = malignant
gini = 0.0 samples = 2 value = [0, 2] class = benign
smoothness error <= 0.003 gini = 0.016 samples = 252 value = [2, 250] class = benign
compactness error <= 0.012 gini = 0.26 samples = 13 value = [2, 11] class = benign
fractal dimension error <= 0.003 gini = 0.444 samples = 3 value = [2, 1] class = malignant
gini = 0.0 samples = 15 value = [0, 15] class = benign
fractal dimension error <= 0.001 gini = 0.245 samples = 7 value = [1, 6] class = benign
worst texture <= 33.27 gini = 0.008 samples = 245 value = [1, 244] class = benign
gini = 0.0 samples = 1 value = [1, 0] class = malignant
worst concave points <= 0.117 gini = 0.153 samples = 12 value = [1, 11] class = benign
gini = 0.0 samples = 2 value = [2, 0] class = malignant
gini = 0.0 samples = 1 value = [0, 1] class = benign
gini = 0.0 samples = 1 value = [1, 0] class = malignant
gini = 0.0 samples = 6 value = [0, 6] class = benign
gini = 0.0 samples = 228 value = [0, 228] class = benign
worst texture <= 33.56 gini = 0.111 samples = 17 value = [1, 16] class = benign
gini = 0.0 samples = 11 value = [0, 11] class = benign
gini = 0.0 samples = 1 value = [1, 0] class = malignant
gini = 0.0 samples = 1 value = [1, 0] class = malignant
gini = 0.0 samples = 16 value = [0, 16] class = benign

Fig.4

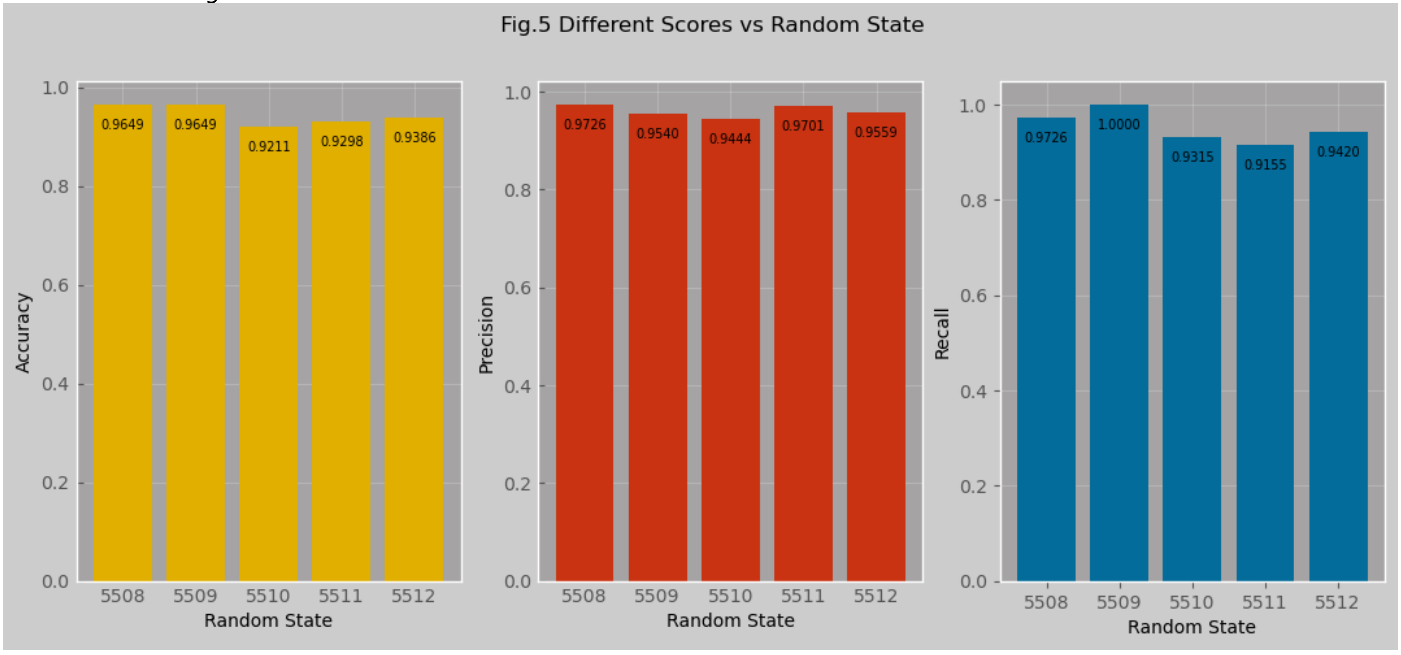
D9: Comment on the Fig 4:

1. From the decision tree visualisation, we can see there are 8 levels/depth in the tree.
2. The diagram can confirm the overfitting issue, where all the terminal nodes are pure, also there are instances where after the final split, the left and right terminal nodes combined sample size is only 3. These are obvious indicators that the model is overfitted.

3. Again, we can see all the leaves are pure, some leave has 288 samples with same class maybe quite a good split, but some leaves having very low sample size within, indicated the overfit issues.
4. As decision tree is a white box algorithm, therefore it can be interpretated clearly. We can use the splitting rules to explain what feature was used for each split, as well as the threshold. If there is a new instance, we could easily use the visualisation to work out which class it predicted to be.

D10: Repeat the data split another four times, each using 80% of the data to train the model and the remaining 20% for testing. For these splits, set the seed of the random state to the values "5509", "5510", "5511" and "5512". The random state of the model can be kept at "5508". For each of these four splits, fit the decision tree classifier and use the trained classifier to perform predictions on the test set. Provide three plots to show the accuracy, precision and recall scores for the test set for each split and comment on the consistency of the results in the five splits (including the original split with random state "5508").

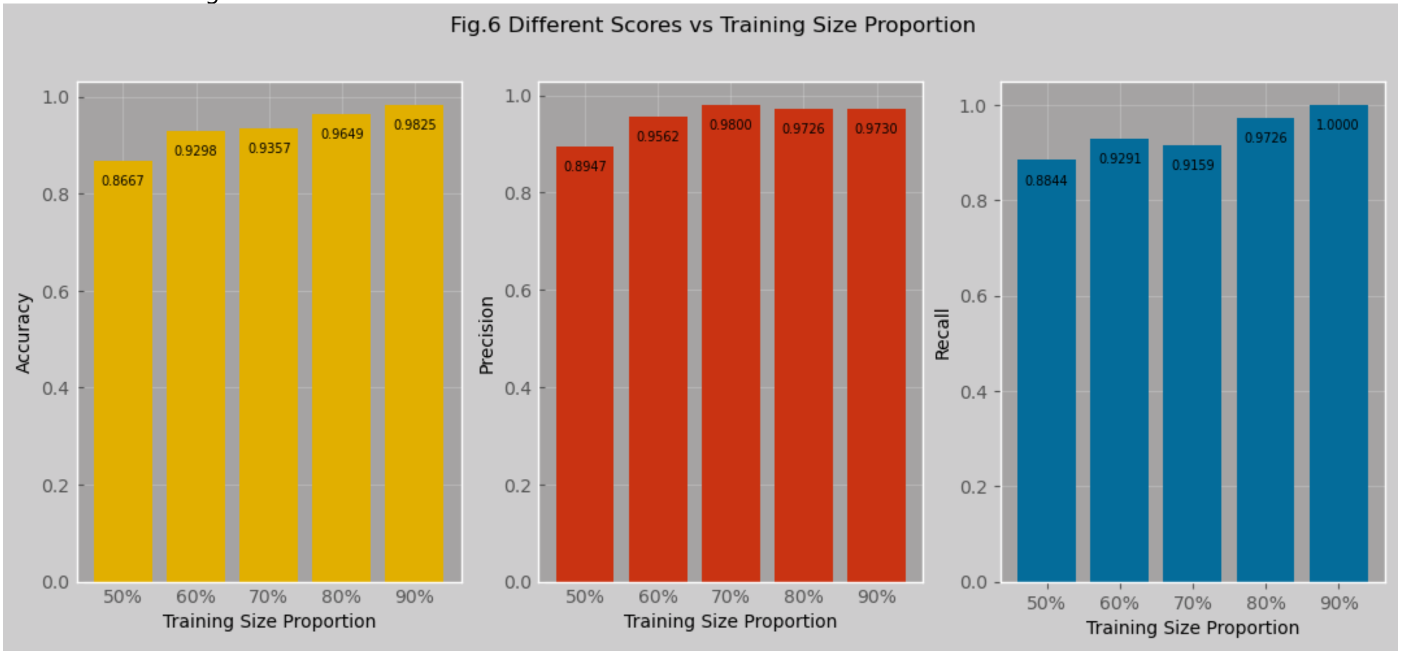
- Answer: Figure 5



Comment : We can see from above, different random states would affect the model results differently, mainly due differences in training set, also the outliner may cause further differences depending on whether training or testing set got split into.

D11: Investigate the impact of the training size on the performance of the model. On 5 different splits: 50%-50% (training the model on 50% of the data and testing on the remaining 50%), 60%-40%, 70%-30%, 80%-20% and 90%-10%. For each of these data splits, set back the seed of the random state to the value "5508".

- Answer: Figure 6



Comment: As expected, overall, more training set would lead to better result in testing set across three different scores. However need to be careful still maintain a reasonable number of instances in testing set.

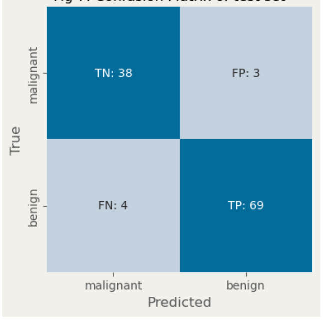
D12: Create a training set using 80% of the data and a test set with the remaining 20%. Use a 10-fold cross-validation and grid-search to find the optimal combination of hyperparameters max depth — using values [2, 3, 4, 5], min samples split — using values [2, 4, 5,10], and min samples leaf — using values [2, 5] of a decision tree model. Remember to set the seed of the random state of the data split function and model class to the value "5508". For the cross-validation, set the value of the random state to "42". Use accuracy for the scoring argument of the grid-search function.

- Answer. Figure 7

  Optimal Hyperparameters : 'max_depth': 3, 'min_samples_leaf': 2, 'min_samples_split': 2

	Training Set:	Test Set:
Accuracy:	0.96	0.94
Precision:	0.95	0.96
Recall:	0.99	0.95

Fig 7: Confusion Matrix of test set



D13: Comment

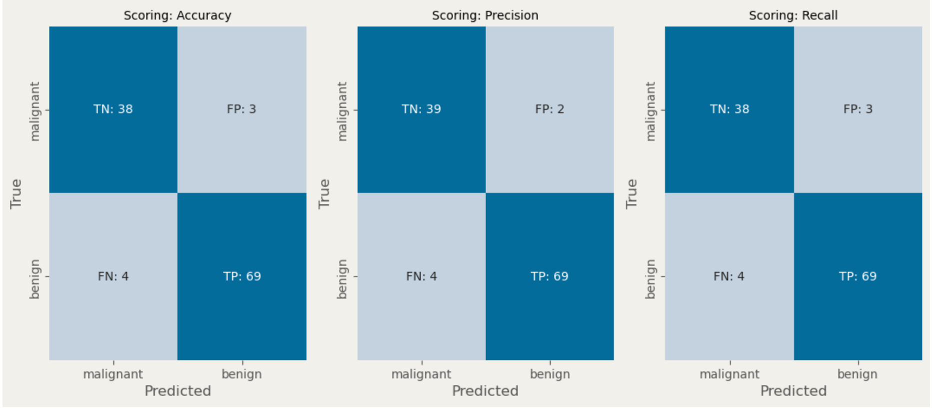
- It is interesting to see the FP and FN both have increased comparing with D6, without fine tuning hyper parameters. However as Fine tuning process using all the combination of the hyperparameters candidates, along with the cross validation to result in an overall better model. simply by comparing the result, there are only 1 or 2 increase in the misclassification, however this doesn't mean the model has not been improved. This new model with optimal hyperparameter most likely will generalise better with other new and bigger testing size.
- With 3 being the max_depth, it is more preferred comparing the depth of 8 in D6 overfitted model.

D14: Repeat the training of task D12 twice: one considering the scoring argument of the grid-search function as precision and the other recall.

- Answer: Figure 8.

	Max_depth	Min_samples_leaf	Min_samples_split
Accuracy	3	2	2
Precision	4	2	2
Recall	3	5	2

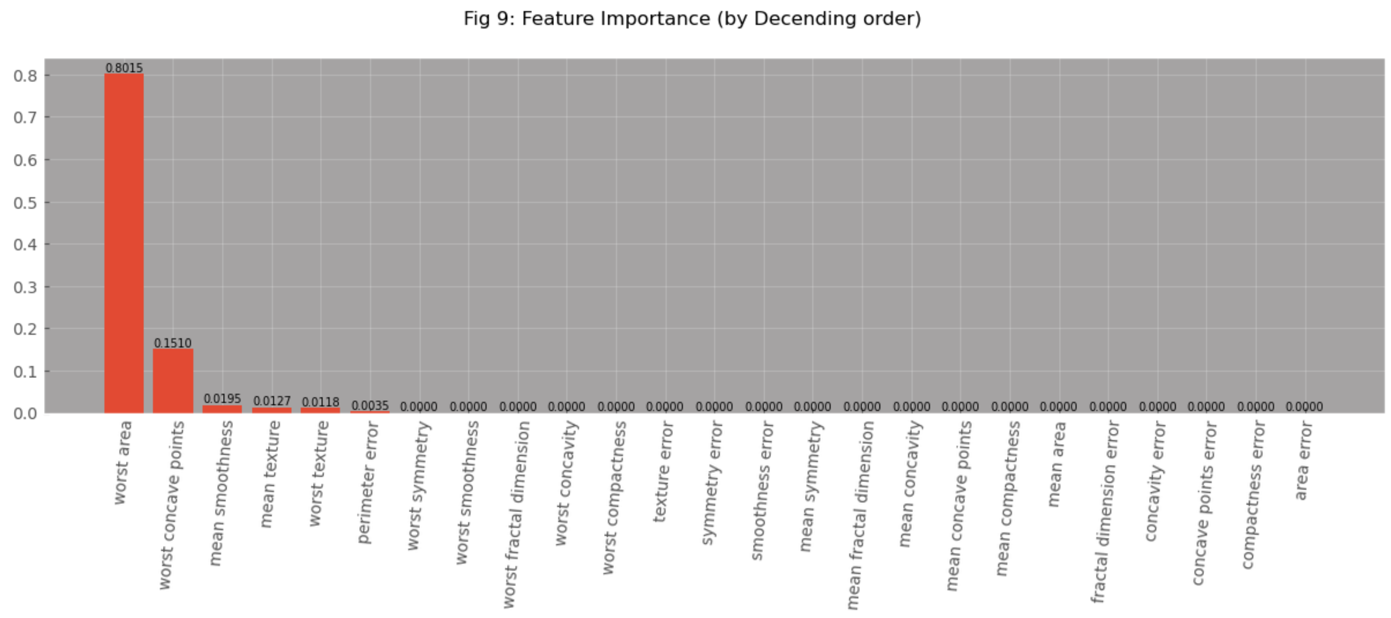
Fig 8: Confusion Matrice



- Comment: Overall, with different scoring options, we can see the regardless which scoring option we use, the model struggle to lower the false negative. However the using Precision as scoring method, the false positive rate has been lowered as expected.
  There maybe some outlier in the test sample feature space in malignant class, where the model constantly incorrectly predicted into Benign class.
  It may be worth to use export_graphviz function again, to visualise the new model with optimal hyperparameters. To see which instances in the testing data constantly being mistakenly classified as benign where in fact they were malignant class. These 4 misclassification instances may not be the same for all three model though.

D15. Using the model with fine-tuned hyperparameters based on accuracy (in D12), display the feature importance for each feature obtained from the training process.

- Answer: Figure 9



D16. Using the feature importance you calculated in the previous task, trim the feature dimension of the data. That is, you should retain only those features whose importance values are above 1% (i.e., 0.01). Report what features were retained and removed in the above process. Also report the total feature importance value that is retained after your dimension reduction step.

- Answer.

  Features remained:
  'mean smoothness', 'mean texture', 'worst area', 'worst concave points', 'worst texture'
  Features removed:
  'area error', 'compactness error', 'concave points error', 'concavity error', 'fractal dimension error', 'mean area','mean compactness', 'mean concave points', 'mean concavity','mean fractal dimension', 'mean symmetry', 'perimeter error', 'smoothness error', 'symmetry error', 'texture error', 'worst compactness', 'worst concavity', 'worst fractal dimension','worst smoothness', 'worst symmetry'

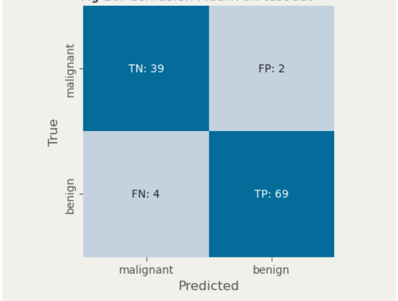
  Total feature importance score after feature reduction and model retrained :
  1

D17. Compare the model's performance (accuracy, precision, recall) on training and test sets when using the reduced set of features and the model trained on the complete set of features. Also, report the corresponding confusion matrices on the test sets.

- Answer: Figure 10.
  Yes, we would need to repeat the cross-validation process to find the optimal hyperparameters, after feature reduction.

	Training Set:	Test Set:
Accuracy:	0.96	0.96
Precision:	0.95	0.95
Recall:	0.95	0.99

Fig 10: Confusion Matrix on test set*



* after feature reduction, with new GridSearch for optimal hyperparameters

D18. Comment:

- Answer: The optimal hyperparameter were changed after the feature reduction, especially the increase on the min_samples_split.

	Max_depth	Min_samples_leaf	Min_samples_split
After Feature reduction	4	2	5

  The model performance on the testing set were really similar with the model without feature reduction, again as mentioned above, decision tree itself can perform the feature reduction during the training process.
  The training process would be a lot more faster after feature reduction, as we are feed in all the important features for the Algorithm to perform the split. The increase on Min_Sample_Split also shows the benefit of the feature reduction.

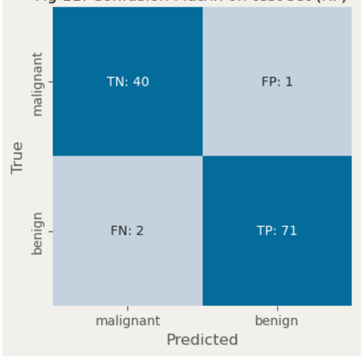
Random Forrest:

D19. Comment:

- Answer:
  Optimal Hyperparameters : max_depth: 5, min_samples_leaf: 2, n_estimators: 1000

	Training Set:	Test Set:
Accuracy:	0.99	0.97
Precision:	0.99	0.99
Recall:	1	0.97

Fig 11: Confusion Matrix on test set (RF)



D20. Comment

- Answer: Comparing with D12, the model performance was improved, both FN and FP were decreased, meaning the model perform better on the testing set with lower variance. This is the expected result by using random forest over decision trees, which reduce the over fitting issue to perform better on the new data. The random forest tree is more robust and generalized.

D21. Comment:

- Do you think these models are good enough and can be trusted to be used for real?

  Even the model performs well on the various scoring and testing set, I don’t believe this is good enough for real world deployments, due to the low dataset.

The total of 569 instance is a very low number for machine learning algorithms. The model may discovered / tested the intrinsic characteristics of these 569 instances, however whether these 569 instances can represent the population left unverified.

- Do you think a more complex model is necessary?

  For our training data and testing data, it is not necessary to have more complex model, as the random forest is robust to handle the overfitting issue.

- Do you think using a machine learning algorithm for this task is a good idea? That is, should this decision process be automated?

  I don’t think so, as it is not recommended to reply on one single model to conclude the final decision, multidisciplinary analysis are crucial to understand any problems.
  The random forest is very powerful tool, based on the data we have. However the knowledge, or the reason why these features were collected is unknown. We may some really important aspects in order to classifying cancer.
  It is important to understand more domain knowledge, rather than build a model with data available to us.
  There are always some patterns in any data, but whether this pattern is meaningful or significant, further study would be required.

- Are there considerations with the used dataset?

  We may able to further inspect the data with the outliner issue, however the decision tree / random forest are not sensitive to normalisation and scaling , then here use the original data in its original scale, which leads to better interpretation of the model.