**CS3481 Fundamentals of Data Science**

**Assignment 2**

# **Random Forest and Naive Bayes Classifier Report**

**Student Name: LUO Peiyuan**

**SID: 56642728**

Table of Contents

[Random Forest and Naive Bayes Classifier Report 1](#_Toc131371286)

[Question A 3](#_Toc131371287)

[**Question A Data Generalization Parameter Setting** 3](#_Toc131371288)

[**Question A Data Generalization** 3](#_Toc131371289)

[**Question A Analysis** 6](#_Toc131371290)

[**Analysis 1: Random Forest model with The Best Classification Performance** 6](#_Toc131371291)

[**Analysis 2: The Performance of Models with Different n\_estimators** 6](#_Toc131371292)

[Question B 7](#_Toc131371293)

[**Question B Component Trees Accuracy Scores Data** 7](#_Toc131371294)

[**Question B Component Trees Accuracy Scores Analysis** 9](#_Toc131371295)

[**Annotation of the Figures** 9](#_Toc131371296)

[**Observations** 10](#_Toc131371297)

[**Findings** 10](#_Toc131371298)

[**Question B Component Trees Details Structure** 10](#_Toc131371299)

[**Component Trees** 10](#_Toc131371300)

[**Findings** 11](#_Toc131371301)

[Question C 12](#_Toc131371302)

[**Question C Feature Importance For six Features for each tree** 12](#_Toc131371303)

[**Question C Analysis** 14](#_Toc131371304)

[**Annotation** 14](#_Toc131371305)

[**Observation From Figure 1 (above one)** 15](#_Toc131371306)

[**Observation from Figure 2 (above one)** 15](#_Toc131371307)

[**Mean Feature Importance Data** 16](#_Toc131371308)

[**Observation from Figure 3 and Mean Feature Importance Data (above one)** 16](#_Toc131371309)

[**Findings** 16](#_Toc131371310)

[Question D 17](#_Toc131371311)

[**Question D Analysis 1 with The Selected Model in Question B** 17](#_Toc131371312)

[**Observations** 17](#_Toc131371313)

[**Findings** 17](#_Toc131371314)

[**Potential Reasons (Referred from internet)** 17](#_Toc131371315)

[**Question D Analysis 2 with Test Size ranging from 0.1 to 0.9** 17](#_Toc131371316)

[**Methodology** 17](#_Toc131371317)

[**Observation** 18](#_Toc131371318)

[Potential Reasons (Referred from internet) 18](#_Toc131371319)

[Appendix 18](#_Toc131371320)

[**My Source GitHub** 18](#_Toc131371321)

# **Question A**

## **Question A Data Generalization Parameter Setting**

I generate the results with the following parameters:

* Test Size: 0.1
* Train Split Random State: 27
* Model Random State: 0
* Criterion: gini
* Max Depth: None
* Mini Samples Split: 2
* Max Features: “sqrt”

## **Question A Data Generalization**

The following is the data I generated for Question 1, which covers the n\_estimator from 1 to 200 with step of 1 and 200 to 1000 with step of 10. Their accuracy scores is also displayed as follows.

n\_estimator: 1 Accuracy Scores: 0.7419354838709677

n\_estimator: 2 Accuracy Scores: 0.8064516129032258

n\_estimator: 3 Accuracy Scores: 0.7741935483870968

n\_estimator: 4 Accuracy Scores: 0.8387096774193549

n\_estimator: 5 Accuracy Scores: 0.7741935483870968

n\_estimator: 6 Accuracy Scores: 0.8064516129032258

n\_estimator: 7 Accuracy Scores: 0.7741935483870968

n\_estimator: 8 Accuracy Scores: 0.8064516129032258

n\_estimator: 9 Accuracy Scores: 0.7741935483870968

n\_estimator: 10 Accuracy Scores: 0.8387096774193549

n\_estimator: 11 Accuracy Scores: 0.8387096774193549

n\_estimator: 12 Accuracy Scores: 0.8709677419354839

n\_estimator: 13 Accuracy Scores: 0.8387096774193549

n\_estimator: 14 Accuracy Scores: 0.8387096774193549

n\_estimator: 15 Accuracy Scores: 0.8387096774193549

n\_estimator: 16 Accuracy Scores: 0.8709677419354839

n\_estimator: 17 Accuracy Scores: 0.8387096774193549

n\_estimator: 18 Accuracy Scores: 0.8387096774193549

n\_estimator: 19 Accuracy Scores: 0.8387096774193549

n\_estimator: 20 Accuracy Scores: 0.8387096774193549

n\_estimator: 21 Accuracy Scores: 0.8387096774193549

n\_estimator: 22 Accuracy Scores: 0.8709677419354839

n\_estimator: 23 Accuracy Scores: 0.8064516129032258

n\_estimator: 24 Accuracy Scores: 0.8387096774193549

n\_estimator: 25 Accuracy Scores: 0.8387096774193549

n\_estimator: 26 Accuracy Scores: 0.8387096774193549

n\_estimator: 27 Accuracy Scores: 0.8064516129032258

n\_estimator: 28 Accuracy Scores: 0.8709677419354839

n\_estimator: 29 Accuracy Scores: 0.8064516129032258

n\_estimator: 30 Accuracy Scores: 0.8709677419354839

n\_estimator: 31 Accuracy Scores: 0.8387096774193549

n\_estimator: 32 Accuracy Scores: 0.8709677419354839

n\_estimator: 33 Accuracy Scores: 0.8709677419354839

n\_estimator: 34 Accuracy Scores: 0.8709677419354839

n\_estimator: 35 Accuracy Scores: 0.8709677419354839

n\_estimator: 36 Accuracy Scores: 0.8709677419354839

n\_estimator: 37 Accuracy Scores: 0.8709677419354839

n\_estimator: 38 Accuracy Scores: 0.8709677419354839

n\_estimator: 39 Accuracy Scores: 0.8709677419354839

n\_estimator: 40 Accuracy Scores: 0.8709677419354839

n\_estimator: 41 Accuracy Scores: 0.8709677419354839

n\_estimator: 42 Accuracy Scores: 0.8709677419354839

n\_estimator: 43 Accuracy Scores: 0.8387096774193549

n\_estimator: 44 Accuracy Scores: 0.8709677419354839

n\_estimator: 45 Accuracy Scores: 0.8387096774193549

n\_estimator: 46 Accuracy Scores: 0.8387096774193549

n\_estimator: 47 Accuracy Scores: 0.8387096774193549

n\_estimator: 48 Accuracy Scores: 0.8387096774193549

n\_estimator: 49 Accuracy Scores: 0.8387096774193549

n\_estimator: 50 Accuracy Scores: 0.8387096774193549

n\_estimator: 51 Accuracy Scores: 0.8387096774193549

n\_estimator: 52 Accuracy Scores: 0.8709677419354839

n\_estimator: 53 Accuracy Scores: 0.8387096774193549

n\_estimator: 54 Accuracy Scores: 0.8387096774193549

n\_estimator: 55 Accuracy Scores: 0.8387096774193549

n\_estimator: 56 Accuracy Scores: 0.8709677419354839

n\_estimator: 57 Accuracy Scores: 0.8709677419354839

n\_estimator: 58 Accuracy Scores: 0.8709677419354839

n\_estimator: 59 Accuracy Scores: 0.8709677419354839

n\_estimator: 60 Accuracy Scores: 0.8709677419354839

n\_estimator: 61 Accuracy Scores: 0.8709677419354839

n\_estimator: 62 Accuracy Scores: 0.8709677419354839

n\_estimator: 63 Accuracy Scores: 0.8387096774193549

n\_estimator: 64 Accuracy Scores: 0.8709677419354839

n\_estimator: 65 Accuracy Scores: 0.8387096774193549

n\_estimator: 66 Accuracy Scores: 0.8387096774193549

n\_estimator: 67 Accuracy Scores: 0.8387096774193549

n\_estimator: 68 Accuracy Scores: 0.8709677419354839

n\_estimator: 69 Accuracy Scores: 0.8709677419354839

n\_estimator: 70 Accuracy Scores: 0.8709677419354839

n\_estimator: 71 Accuracy Scores: 0.8387096774193549

n\_estimator: 72 Accuracy Scores: 0.8387096774193549

n\_estimator: 73 Accuracy Scores: 0.8387096774193549

n\_estimator: 74 Accuracy Scores: 0.8387096774193549

n\_estimator: 75 Accuracy Scores: 0.8387096774193549

n\_estimator: 76 Accuracy Scores: 0.8387096774193549

n\_estimator: 77 Accuracy Scores: 0.8387096774193549

n\_estimator: 78 Accuracy Scores: 0.8387096774193549

n\_estimator: 79 Accuracy Scores: 0.8387096774193549

n\_estimator: 80 Accuracy Scores: 0.8709677419354839

n\_estimator: 81 Accuracy Scores: 0.8709677419354839

n\_estimator: 82 Accuracy Scores: 0.8709677419354839

n\_estimator: 83 Accuracy Scores: 0.8709677419354839

n\_estimator: 84 Accuracy Scores: 0.8709677419354839

n\_estimator: 85 Accuracy Scores: 0.8709677419354839

n\_estimator: 86 Accuracy Scores: 0.8709677419354839

n\_estimator: 87 Accuracy Scores: 0.8709677419354839

n\_estimator: 88 Accuracy Scores: 0.8709677419354839

n\_estimator: 89 Accuracy Scores: 0.8709677419354839

n\_estimator: 90 Accuracy Scores: 0.8709677419354839

n\_estimator: 91 Accuracy Scores: 0.8709677419354839

n\_estimator: 92 Accuracy Scores: 0.8709677419354839

n\_estimator: 93 Accuracy Scores: 0.8709677419354839

n\_estimator: 94 Accuracy Scores: 0.8709677419354839

n\_estimator: 95 Accuracy Scores: 0.8709677419354839

n\_estimator: 96 Accuracy Scores: 0.8709677419354839

n\_estimator: 97 Accuracy Scores: 0.8709677419354839

n\_estimator: 98 Accuracy Scores: 0.8709677419354839

n\_estimator: 99 Accuracy Scores: 0.8709677419354839

n\_estimator: 100 Accuracy Scores: 0.8709677419354839

n\_estimator: 101 Accuracy Scores: 0.8709677419354839

n\_estimator: 102 Accuracy Scores: 0.8709677419354839

n\_estimator: 103 Accuracy Scores: 0.8709677419354839

n\_estimator: 104 Accuracy Scores: 0.8709677419354839

n\_estimator: 105 Accuracy Scores: 0.8709677419354839

n\_estimator: 106 Accuracy Scores: 0.8709677419354839

n\_estimator: 107 Accuracy Scores: 0.8709677419354839

n\_estimator: 108 Accuracy Scores: 0.8709677419354839

n\_estimator: 109 Accuracy Scores: 0.8709677419354839

n\_estimator: 110 Accuracy Scores: 0.8709677419354839

n\_estimator: 111 Accuracy Scores: 0.8709677419354839

n\_estimator: 112 Accuracy Scores: 0.8709677419354839

n\_estimator: 113 Accuracy Scores: 0.8709677419354839

n\_estimator: 114 Accuracy Scores: 0.8709677419354839

n\_estimator: 115 Accuracy Scores: 0.9032258064516129

n\_estimator: 116 Accuracy Scores: 0.8709677419354839

n\_estimator: 117 Accuracy Scores: 0.8709677419354839

n\_estimator: 118 Accuracy Scores: 0.8709677419354839

n\_estimator: 119 Accuracy Scores: 0.9032258064516129

n\_estimator: 120 Accuracy Scores: 0.9032258064516129

n\_estimator: 121 Accuracy Scores: 0.9032258064516129

n\_estimator: 122 Accuracy Scores: 0.9032258064516129

n\_estimator: 123 Accuracy Scores: 0.9032258064516129

n\_estimator: 124 Accuracy Scores: 0.9032258064516129

n\_estimator: 125 Accuracy Scores: 0.9032258064516129

n\_estimator: 126 Accuracy Scores: 0.9032258064516129

n\_estimator: 127 Accuracy Scores: 0.9032258064516129

n\_estimator: 128 Accuracy Scores: 0.9032258064516129

n\_estimator: 129 Accuracy Scores: 0.9032258064516129

n\_estimator: 130 Accuracy Scores: 0.9032258064516129

n\_estimator: 131 Accuracy Scores: 0.9032258064516129

n\_estimator: 132 Accuracy Scores: 0.9032258064516129

n\_estimator: 133 Accuracy Scores: 0.9032258064516129

n\_estimator: 134 Accuracy Scores: 0.9032258064516129

n\_estimator: 135 Accuracy Scores: 0.9032258064516129

n\_estimator: 136 Accuracy Scores: 0.9032258064516129

n\_estimator: 137 Accuracy Scores: 0.9032258064516129

n\_estimator: 138 Accuracy Scores: 0.9032258064516129

n\_estimator: 139 Accuracy Scores: 0.9032258064516129

n\_estimator: 140 Accuracy Scores: 0.9032258064516129

n\_estimator: 141 Accuracy Scores: 0.9032258064516129

n\_estimator: 142 Accuracy Scores: 0.9032258064516129

n\_estimator: 143 Accuracy Scores: 0.9032258064516129

n\_estimator: 144 Accuracy Scores: 0.9032258064516129

n\_estimator: 145 Accuracy Scores: 0.9032258064516129

n\_estimator: 146 Accuracy Scores: 0.9032258064516129

n\_estimator: 147 Accuracy Scores: 0.9032258064516129

n\_estimator: 148 Accuracy Scores: 0.9032258064516129

n\_estimator: 149 Accuracy Scores: 0.9032258064516129

n\_estimator: 150 Accuracy Scores: 0.9032258064516129

n\_estimator: 151 Accuracy Scores: 0.9032258064516129

n\_estimator: 152 Accuracy Scores: 0.9032258064516129

n\_estimator: 153 Accuracy Scores: 0.9032258064516129

n\_estimator: 154 Accuracy Scores: 0.9032258064516129

n\_estimator: 155 Accuracy Scores: 0.9032258064516129

n\_estimator: 156 Accuracy Scores: 0.8709677419354839

n\_estimator: 157 Accuracy Scores: 0.8709677419354839

n\_estimator: 158 Accuracy Scores: 0.8709677419354839

n\_estimator: 159 Accuracy Scores: 0.8709677419354839

n\_estimator: 160 Accuracy Scores: 0.8709677419354839

n\_estimator: 161 Accuracy Scores: 0.8709677419354839

n\_estimator: 162 Accuracy Scores: 0.8709677419354839

n\_estimator: 163 Accuracy Scores: 0.8709677419354839

n\_estimator: 164 Accuracy Scores: 0.8709677419354839

n\_estimator: 165 Accuracy Scores: 0.8709677419354839

n\_estimator: 166 Accuracy Scores: 0.8709677419354839

n\_estimator: 167 Accuracy Scores: 0.8709677419354839

n\_estimator: 168 Accuracy Scores: 0.8709677419354839

n\_estimator: 169 Accuracy Scores: 0.8709677419354839

n\_estimator: 170 Accuracy Scores: 0.8709677419354839

n\_estimator: 171 Accuracy Scores: 0.8709677419354839

n\_estimator: 172 Accuracy Scores: 0.8709677419354839

n\_estimator: 173 Accuracy Scores: 0.8709677419354839

n\_estimator: 174 Accuracy Scores: 0.8709677419354839

n\_estimator: 175 Accuracy Scores: 0.8709677419354839

n\_estimator: 176 Accuracy Scores: 0.8709677419354839

n\_estimator: 177 Accuracy Scores: 0.8709677419354839

n\_estimator: 178 Accuracy Scores: 0.8709677419354839

n\_estimator: 179 Accuracy Scores: 0.8709677419354839

n\_estimator: 180 Accuracy Scores: 0.8709677419354839

n\_estimator: 181 Accuracy Scores: 0.8709677419354839

n\_estimator: 182 Accuracy Scores: 0.8709677419354839

n\_estimator: 183 Accuracy Scores: 0.8709677419354839

n\_estimator: 184 Accuracy Scores: 0.8709677419354839

n\_estimator: 185 Accuracy Scores: 0.8709677419354839

n\_estimator: 186 Accuracy Scores: 0.8709677419354839

n\_estimator: 187 Accuracy Scores: 0.8709677419354839

n\_estimator: 188 Accuracy Scores: 0.8709677419354839

n\_estimator: 189 Accuracy Scores: 0.8709677419354839

n\_estimator: 190 Accuracy Scores: 0.8709677419354839

n\_estimator: 191 Accuracy Scores: 0.8709677419354839

n\_estimator: 192 Accuracy Scores: 0.8709677419354839

n\_estimator: 193 Accuracy Scores: 0.8709677419354839

n\_estimator: 194 Accuracy Scores: 0.8709677419354839

n\_estimator: 195 Accuracy Scores: 0.8709677419354839

n\_estimator: 196 Accuracy Scores: 0.8709677419354839

n\_estimator: 197 Accuracy Scores: 0.8709677419354839

n\_estimator: 198 Accuracy Scores: 0.8709677419354839

n\_estimator: 199 Accuracy Scores: 0.8709677419354839

n\_estimator: 200 Accuracy Scores: 0.8709677419354839

n\_estimator: 210 Accuracy Scores: 0.9032258064516129

n\_estimator: 220 Accuracy Scores: 0.9032258064516129

n\_estimator: 230 Accuracy Scores: 0.9032258064516129

n\_estimator: 240 Accuracy Scores: 0.9032258064516129

n\_estimator: 250 Accuracy Scores: 0.9032258064516129

n\_estimator: 260 Accuracy Scores: 0.9032258064516129

n\_estimator: 270 Accuracy Scores: 0.9032258064516129

n\_estimator: 280 Accuracy Scores: 0.9032258064516129

n\_estimator: 290 Accuracy Scores: 0.9032258064516129

n\_estimator: 300 Accuracy Scores: 0.9032258064516129

n\_estimator: 310 Accuracy Scores: 0.9032258064516129

n\_estimator: 320 Accuracy Scores: 0.9032258064516129

n\_estimator: 330 Accuracy Scores: 0.9032258064516129

n\_estimator: 340 Accuracy Scores: 0.9032258064516129

n\_estimator: 350 Accuracy Scores: 0.9032258064516129

n\_estimator: 360 Accuracy Scores: 0.9032258064516129

n\_estimator: 370 Accuracy Scores: 0.9032258064516129

n\_estimator: 380 Accuracy Scores: 0.9032258064516129

n\_estimator: 390 Accuracy Scores: 0.9032258064516129

n\_estimator: 400 Accuracy Scores: 0.9032258064516129

n\_estimator: 410 Accuracy Scores: 0.9032258064516129

n\_estimator: 420 Accuracy Scores: 0.9032258064516129

n\_estimator: 430 Accuracy Scores: 0.9032258064516129

n\_estimator: 440 Accuracy Scores: 0.9032258064516129

n\_estimator: 450 Accuracy Scores: 0.9032258064516129

n\_estimator: 460 Accuracy Scores: 0.9032258064516129

n\_estimator: 470 Accuracy Scores: 0.9032258064516129

n\_estimator: 480 Accuracy Scores: 0.9032258064516129

n\_estimator: 490 Accuracy Scores: 0.9032258064516129

n\_estimator: 500 Accuracy Scores: 0.9032258064516129

n\_estimator: 510 Accuracy Scores: 0.9032258064516129

n\_estimator: 520 Accuracy Scores: 0.9032258064516129

n\_estimator: 530 Accuracy Scores: 0.9032258064516129

n\_estimator: 540 Accuracy Scores: 0.9032258064516129

n\_estimator: 550 Accuracy Scores: 0.9032258064516129

n\_estimator: 560 Accuracy Scores: 0.9032258064516129

n\_estimator: 570 Accuracy Scores: 0.9032258064516129

n\_estimator: 580 Accuracy Scores: 0.9032258064516129

n\_estimator: 590 Accuracy Scores: 0.9032258064516129

n\_estimator: 600 Accuracy Scores: 0.9032258064516129

n\_estimator: 610 Accuracy Scores: 0.9032258064516129

n\_estimator: 620 Accuracy Scores: 0.9032258064516129

n\_estimator: 630 Accuracy Scores: 0.9032258064516129

n\_estimator: 640 Accuracy Scores: 0.9032258064516129

n\_estimator: 650 Accuracy Scores: 0.9032258064516129

n\_estimator: 660 Accuracy Scores: 0.9032258064516129

n\_estimator: 670 Accuracy Scores: 0.9032258064516129

n\_estimator: 680 Accuracy Scores: 0.9032258064516129

n\_estimator: 690 Accuracy Scores: 0.9032258064516129

n\_estimator: 700 Accuracy Scores: 0.9032258064516129

n\_estimator: 710 Accuracy Scores: 0.9032258064516129

n\_estimator: 720 Accuracy Scores: 0.9032258064516129

n\_estimator: 730 Accuracy Scores: 0.9032258064516129

n\_estimator: 740 Accuracy Scores: 0.9032258064516129

n\_estimator: 750 Accuracy Scores: 0.9032258064516129

n\_estimator: 760 Accuracy Scores: 0.9032258064516129

n\_estimator: 770 Accuracy Scores: 0.9032258064516129

n\_estimator: 780 Accuracy Scores: 0.9032258064516129

n\_estimator: 790 Accuracy Scores: 0.9032258064516129

n\_estimator: 800 Accuracy Scores: 0.9032258064516129

n\_estimator: 810 Accuracy Scores: 0.9032258064516129

n\_estimator: 820 Accuracy Scores: 0.9032258064516129

n\_estimator: 830 Accuracy Scores: 0.9032258064516129

n\_estimator: 840 Accuracy Scores: 0.9032258064516129

n\_estimator: 850 Accuracy Scores: 0.9032258064516129

n\_estimator: 860 Accuracy Scores: 0.9032258064516129

n\_estimator: 870 Accuracy Scores: 0.9032258064516129

n\_estimator: 880 Accuracy Scores: 0.9032258064516129

n\_estimator: 890 Accuracy Scores: 0.9032258064516129

n\_estimator: 900 Accuracy Scores: 0.9032258064516129

n\_estimator: 910 Accuracy Scores: 0.9032258064516129

n\_estimator: 920 Accuracy Scores: 0.9032258064516129

n\_estimator: 930 Accuracy Scores: 0.9032258064516129

n\_estimator: 940 Accuracy Scores: 0.9032258064516129

n\_estimator: 950 Accuracy Scores: 0.9032258064516129

n\_estimator: 960 Accuracy Scores: 0.9032258064516129

n\_estimator: 970 Accuracy Scores: 0.9032258064516129

n\_estimator: 980 Accuracy Scores: 0.9032258064516129

n\_estimator: 990 Accuracy Scores: 0.9032258064516129

## **Question A Analysis**

### **Analysis 1: Random Forest model with The Best Classification Performance**

After generating the data above, I sorted the data record by accuracy scores, and get the random forest model corresponding to the best classification performance is when n\_estimators of 115.

Max Scores: 0.9032258064516129 With n\_estimators: 115

### **Analysis 2: The Performance of Models with Different n\_estimators**

#### **Observations**

**The Variations of the Model Performance:** From the below graph, the conclusion is drawn that the performance is changing a lot when the n\_estimators is ranging between 1 to 200, and the model performance is steady when the n\_estimators is reaching above 200.

**Distribution of model accuracy：** From the below graph, it’s easy to find the accuracy scores of the model performance is ranging from 0.74 to. 0.904. And it finally reaches a step scores of 0.904.

Chart

Description automatically generated

#### **Findings**

* From the above figures, it’s easy to find that the n\_estimator parameters plays a very important role in influencing the model performance.
* Increasing the number of trees in a random forest model can improve its performance by reducing the model's variance and increasing its stability. However, increasing the number of trees beyond a certain point may not necessarily improve the model's performance and can even lead to overfitting.
* From the above observations, in this case, it’s easy to reach a conclusion that in order to get the models with a good performance and not time-consuming, we can train the models with n\_estimators ranging from 1 to 200, and choose the relatively best one. This would reduce some unnecessary computation time.

# **Question B**

## **Question B Component Trees Accuracy Scores Data**

From the Question A, the best classification model is found with the n\_estimators of 115, which contains 115 component trees, I have put the generated accuracy scores for different component trees as below.

DecisionTreeClassifier(max\_features='sqrt', random\_state=209652396)

0.7419354838709677

DecisionTreeClassifier(max\_features='sqrt', random\_state=398764591)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=924231285)

0.6774193548387096

DecisionTreeClassifier(max\_features='sqrt', random\_state=1478610112)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=441365315)

0.7096774193548387

DecisionTreeClassifier(max\_features='sqrt', random\_state=1537364731)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=192771779)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=1491434855)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=1819583497)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=530702035)

0.7419354838709677

DecisionTreeClassifier(max\_features='sqrt', random\_state=626610453)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=1650906866)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=1879422756)

0.7419354838709677

DecisionTreeClassifier(max\_features='sqrt', random\_state=1277901399)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=1682652230)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=243580376)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=1991416408)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=1171049868)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=1646868794)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=2051556033)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=1252949478)

0.9032258064516129

DecisionTreeClassifier(max\_features='sqrt', random\_state=1340754471)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=124102743)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=2061486254)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=292249176)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=1686997841)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=1827923621)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=1443447321)

0.9032258064516129

DecisionTreeClassifier(max\_features='sqrt', random\_state=305097549)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=1449105480)

0.6451612903225806

DecisionTreeClassifier(max\_features='sqrt', random\_state=374217481)

0.6451612903225806

DecisionTreeClassifier(max\_features='sqrt', random\_state=636393364)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=86837363)

0.9032258064516129

DecisionTreeClassifier(max\_features='sqrt', random\_state=1581585360)

0.7419354838709677

DecisionTreeClassifier(max\_features='sqrt', random\_state=1428591347)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=1963466437)

0.967741935483871

DecisionTreeClassifier(max\_features='sqrt', random\_state=1194674174)

0.9032258064516129

DecisionTreeClassifier(max\_features='sqrt', random\_state=602801999)

0.7419354838709677

DecisionTreeClassifier(max\_features='sqrt', random\_state=1589190063)

0.7096774193548387

DecisionTreeClassifier(max\_features='sqrt', random\_state=1589512640)

0.7419354838709677

DecisionTreeClassifier(max\_features='sqrt', random\_state=2055650130)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=2034131043)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=1284876248)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=1292401841)

0.7096774193548387

DecisionTreeClassifier(max\_features='sqrt', random\_state=1982038771)

0.7096774193548387

DecisionTreeClassifier(max\_features='sqrt', random\_state=87950109)

0.6129032258064516

DecisionTreeClassifier(max\_features='sqrt', random\_state=1204863635)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=768281747)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=507984782)

0.7096774193548387

DecisionTreeClassifier(max\_features='sqrt', random\_state=947610023)

0.8709677419354839

DecisionTreeClassifier(max\_features='sqrt', random\_state=600956192)

0.7096774193548387

DecisionTreeClassifier(max\_features='sqrt', random\_state=352272321)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=615697673)

0.6774193548387096

DecisionTreeClassifier(max\_features='sqrt', random\_state=160516793)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=1909838463)

0.7096774193548387

DecisionTreeClassifier(max\_features='sqrt', random\_state=1110745632)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=93837855)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=454869706)

0.7419354838709677

DecisionTreeClassifier(max\_features='sqrt', random\_state=1780959476)

0.7096774193548387

DecisionTreeClassifier(max\_features='sqrt', random\_state=2034098327)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=1136257699)

0.6129032258064516

DecisionTreeClassifier(max\_features='sqrt', random\_state=800291326)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=1177824715)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=1017555826)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=1959150775)

0.7096774193548387

DecisionTreeClassifier(max\_features='sqrt', random\_state=930076700)

0.8709677419354839

DecisionTreeClassifier(max\_features='sqrt', random\_state=293921570)

0.8709677419354839

DecisionTreeClassifier(max\_features='sqrt', random\_state=580757632)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=80701568)

0.8709677419354839

DecisionTreeClassifier(max\_features='sqrt', random\_state=1392175012)

0.5161290322580645

DecisionTreeClassifier(max\_features='sqrt', random\_state=505240629)

0.7096774193548387

DecisionTreeClassifier(max\_features='sqrt', random\_state=642848645)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=481447462)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=954863080)

0.6129032258064516

DecisionTreeClassifier(max\_features='sqrt', random\_state=502227700)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=1659957521)

0.9354838709677419

DecisionTreeClassifier(max\_features='sqrt', random\_state=1905883471)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=1729147268)

0.6451612903225806

DecisionTreeClassifier(max\_features='sqrt', random\_state=780912233)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=1932520490)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=1544074682)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=485603871)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=1877037944)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=1728073985)

0.967741935483871

DecisionTreeClassifier(max\_features='sqrt', random\_state=848819521)

0.8709677419354839

DecisionTreeClassifier(max\_features='sqrt', random\_state=426405863)

0.6774193548387096

DecisionTreeClassifier(max\_features='sqrt', random\_state=258666409)

0.8709677419354839

DecisionTreeClassifier(max\_features='sqrt', random\_state=2017814585)

0.7419354838709677

DecisionTreeClassifier(max\_features='sqrt', random\_state=716257571)

0.7419354838709677

DecisionTreeClassifier(max\_features='sqrt', random\_state=657731430)

0.9032258064516129

DecisionTreeClassifier(max\_features='sqrt', random\_state=732884087)

0.8709677419354839

DecisionTreeClassifier(max\_features='sqrt', random\_state=734051083)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=903586222)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=1538251858)

0.9354838709677419

DecisionTreeClassifier(max\_features='sqrt', random\_state=553734235)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=1076688768)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=1354754446)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=463129187)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=1562125877)

0.8709677419354839

DecisionTreeClassifier(max\_features='sqrt', random\_state=1396067212)

0.6451612903225806

DecisionTreeClassifier(max\_features='sqrt', random\_state=301492857)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=165035946)

0.967741935483871

DecisionTreeClassifier(max\_features='sqrt', random\_state=1883779156)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=576702667)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=2097549636)

0.8709677419354839

DecisionTreeClassifier(max\_features='sqrt', random\_state=1971172102)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=438279108)

0.7096774193548387

DecisionTreeClassifier(max\_features='sqrt', random\_state=656229423)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=897118847)

0.8387096774193549

DecisionTreeClassifier(max\_features='sqrt', random\_state=580073460)

0.7096774193548387

DecisionTreeClassifier(max\_features='sqrt', random\_state=692819075)

0.7741935483870968

DecisionTreeClassifier(max\_features='sqrt', random\_state=2127295436)

0.7419354838709677

DecisionTreeClassifier(max\_features='sqrt', random\_state=657595236)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=351544500)

0.8064516129032258

DecisionTreeClassifier(max\_features='sqrt', random\_state=1087879144)

0.6774193548387096

## **Question B Component Trees Accuracy Scores Analysis**

**Annotation of the Figures** As the below figures show, the red one represents the accuracy scores of the random forest model, and the yellow one represents the accuracy scores of the component trees.

Chart, line chart

Description automatically generated

### **Observations**

* The accuracy scores of the individual component trees varies a lot between each other. And the individual scores ranges from 0.5+ to 0.9+, and the mean score calculated by averaging all the scores of each individuals is 0.7865357643758767.
* Compared to the mean scores, the actual model score is much higher, and is 0.9032258064516129.

### **Findings**

* Concluded from the above observations, the random forest model scores are much higher than the mean of the component trees and the majority of the individual component trees. Combined with my understanding of the random forest classifier model, it makes final predictions by aggregating the predictions of multiple trees and taking a majority vote, which helps to reduce the impact of any individual tree's errors and improve the overall accuracy of the model.
* Concern: if the trees in the random forest are quite similar, then even though it would choose the majority choice, but under this situation, the random forest classifier may lead to a bad performance. Hence, we should make the trees in the random forest differs to achieve a higher performance.

## **Question B Component Trees Details Structure**

### **Component Trees**

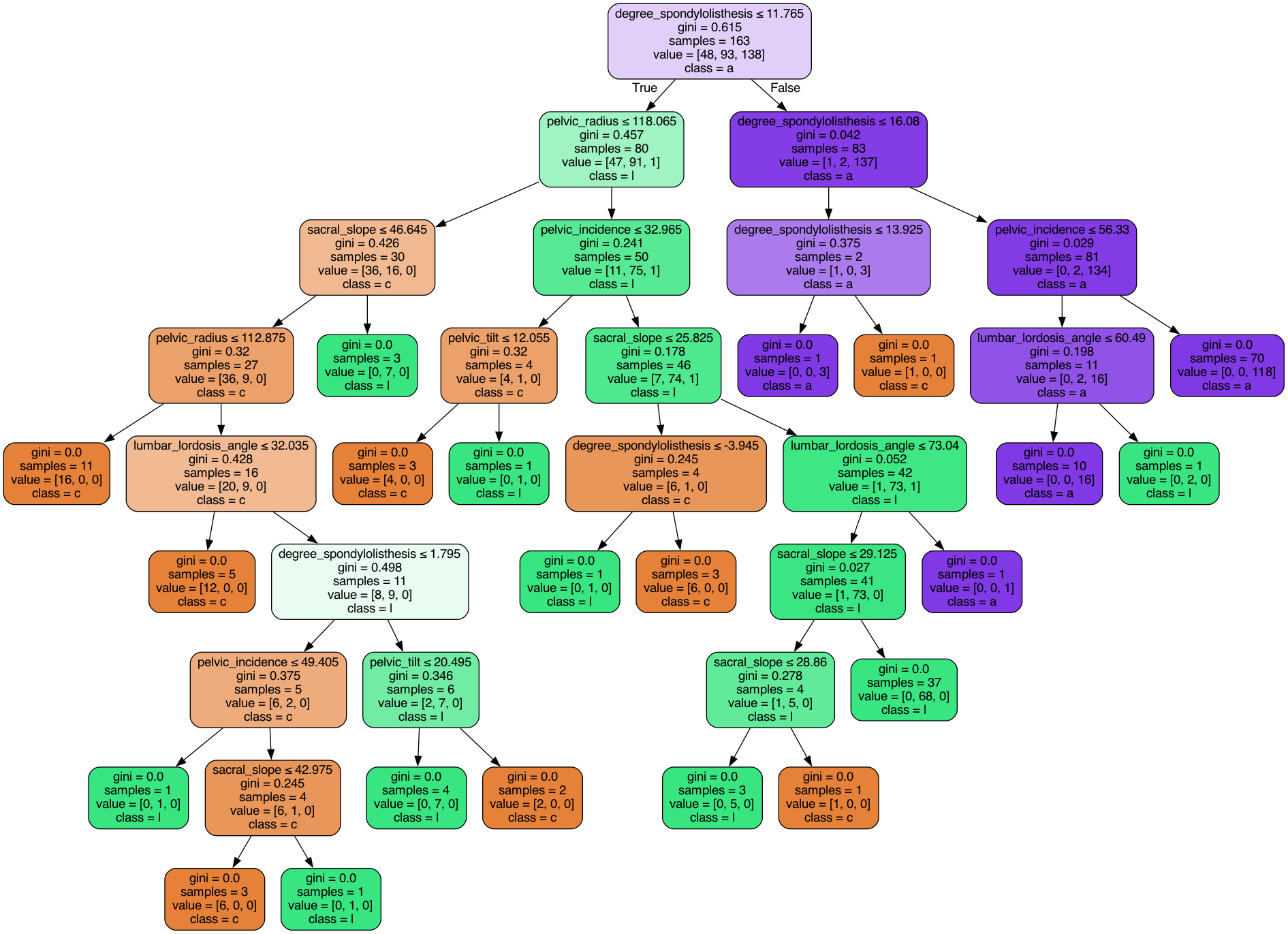
I select several component tree graphs with relatively high performance of the visualized decision trees here. It has a very good performance.

**Component Tree with accuracy score of 0.967741935483871**

Chart

Description automatically generated

**Component Tree with accuracy score of 0.9354838709677419**



**Component Tree with accuracy score of 0.967741935483871**

**Timeline

Description automatically generated**

### **Findings**

**Actually, I found there are several identical trees in the forest.**

# **Question C**

## **Question C Feature Importance For six Features for each tree**

Using the selected component trees in Question B, I generated their associated lists of relative attribute importance values as below.

DecisionTreeClassifier(max\_features='sqrt', random\_state=209652396)

[0.34743972 0.06688596 0.09613247 0.11083523 0.14073868 0.23796794]

DecisionTreeClassifier(max\_features='sqrt', random\_state=398764591)

[0.0704545 0.12277568 0.11591574 0.34788493 0.08497838 0.25799077]

DecisionTreeClassifier(max\_features='sqrt', random\_state=924231285)

[0.07819602 0.07496975 0.07340742 0.36696011 0.16622329 0.24024341]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1478610112)

[0.10069467 0.098169 0.03968471 0.36453001 0.07268564 0.32423597]

DecisionTreeClassifier(max\_features='sqrt', random\_state=441365315)

[0.36794413 0.10335586 0.07686015 0.09926287 0.05461244 0.29796456]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1537364731)

[0.22079137 0.07188466 0.05164534 0.07793813 0.17189905 0.40584146]

DecisionTreeClassifier(max\_features='sqrt', random\_state=192771779)

[0.0549225 0.11671801 0.03003166 0.05624246 0.13228544 0.60979992]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1491434855)

[0.04202879 0.13918054 0.10559717 0.31951049 0.12454384 0.26913917]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1819583497)

[0.14524999 0.11394059 0.31645584 0.03366849 0.11418708 0.27649801]

DecisionTreeClassifier(max\_features='sqrt', random\_state=530702035)

[0.02510466 0.06998405 0.0432837 0.12051246 0.14227029 0.59884482]

DecisionTreeClassifier(max\_features='sqrt', random\_state=626610453)

[0.06570792 0.19321436 0.11215693 0.08513924 0.10106282 0.44271873]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1650906866)

[0.32145655 0.13794529 0.04047477 0.05339755 0.12036099 0.32636485]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1879422756)

[0.0632979 0.06045249 0.0508958 0.07462037 0.05938524 0.6913482 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1277901399)

[0.05267909 0.08956992 0.19038568 0.03426295 0.05718535 0.57591701]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1682652230)

[0.41919304 0.14903491 0. 0.11603395 0.06621477 0.24952334]

DecisionTreeClassifier(max\_features='sqrt', random\_state=243580376)

[0.05765295 0.0322834 0.05611742 0.15057501 0.11509883 0.5882724 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1991416408)

[0.05704782 0.11788846 0.29789974 0.08013284 0.14927086 0.29776027]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1171049868)

[0.05834843 0.08615375 0.07441213 0.3310427 0.16314021 0.28690279]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1646868794)

[0.02729737 0.09027287 0.13939624 0.06807388 0.05738474 0.6175749 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=2051556033)

[0.05778124 0.05567375 0.06761095 0.07610747 0.07638609 0.6664405 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1252949478)

[0.11131064 0.06321312 0.07040658 0.30010276 0.05978129 0.39518562]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1340754471)

[0.38016551 0.08868927 0. 0.08639426 0.14093518 0.30381578]

DecisionTreeClassifier(max\_features='sqrt', random\_state=124102743)

[0.31983352 0.16391856 0.06020317 0.14154513 0.10926201 0.20523761]

DecisionTreeClassifier(max\_features='sqrt', random\_state=2061486254)

[0.36005177 0.08621668 0.0701134 0.05126966 0.09935207 0.33299642]

DecisionTreeClassifier(max\_features='sqrt', random\_state=292249176)

[0.01068674 0.07563093 0.35135043 0.14908363 0.0959677 0.31728057]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1686997841)

[0.01980846 0.06818435 0.03972797 0.14065926 0.09618113 0.63543882]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1827923621)

[0.12746359 0.03479536 0.0034941 0.284094 0.09373704 0.45641591]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1443447321)

[0.0263564 0.05321395 0.07230605 0.05800153 0.10634329 0.68377877]

DecisionTreeClassifier(max\_features='sqrt', random\_state=305097549)

[0.08074532 0.14812023 0.0488653 0.05712348 0.10170317 0.56344249]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1449105480)

[0.286469 0.13844493 0.12545862 0.10501089 0.18304802 0.16156854]

DecisionTreeClassifier(max\_features='sqrt', random\_state=374217481)

[0.05511151 0.26149464 0.12011367 0.21592076 0.04978345 0.29757598]

DecisionTreeClassifier(max\_features='sqrt', random\_state=636393364)

[0.0533855 0.11336019 0.30462837 0.09351601 0.12623913 0.30887081]

DecisionTreeClassifier(max\_features='sqrt', random\_state=86837363)

[0.07865872 0.08573112 0.11232419 0.13983481 0.16212177 0.42132939]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1581585360)

[0.05920415 0.08787007 0.41387913 0.13019566 0.08740737 0.22144362]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1428591347)

[0.10658044 0.24124526 0.12467387 0.13263493 0.16887361 0.22599189]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1963466437)

[0.04085715 0.06479862 0.36548217 0.08185986 0.175853 0.2711492 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1194674174)

[0.36976261 0.10800231 0.05704066 0.15412902 0.192055 0.11901041]

DecisionTreeClassifier(max\_features='sqrt', random\_state=602801999)

[0.0484159 0.0487353 0.10991839 0.46939448 0.12409158 0.19944434]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1589190063)

[0.0179187 0.06573194 0.07225228 0.12618913 0.05863493 0.65927302]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1589512640)

[0.125002 0.0465147 0.06522041 0.26043697 0.19923626 0.30358965]

DecisionTreeClassifier(max\_features='sqrt', random\_state=2055650130)

[0.03465266 0.08834858 0.06736404 0.06986443 0.08967668 0.65009362]

DecisionTreeClassifier(max\_features='sqrt', random\_state=2034131043)

[0.0452859 0.04120036 0.0679091 0.10695237 0.12131399 0.61733829]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1284876248)

[0.10108784 0.08004884 0.29631522 0.08167436 0.16283487 0.27803886]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1292401841)

[0.17026779 0.07282673 0.04343398 0.3088798 0.08531244 0.31927926]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1982038771)

[0.12068012 0.04879925 0.42469528 0.06722838 0.07227117 0.2663258 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=87950109)

[0.13362681 0.11715664 0.34591486 0.12893029 0.05559194 0.21877945]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1204863635)

[0.04753238 0.09140618 0.03929189 0.12603119 0.07366376 0.6220746 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=768281747)

[0.15550292 0.03828647 0.10568638 0.38101589 0.04769744 0.27181089]

DecisionTreeClassifier(max\_features='sqrt', random\_state=507984782)

[0.03930105 0.12590563 0.28008018 0.20503982 0.1398194 0.20985391]

DecisionTreeClassifier(max\_features='sqrt', random\_state=947610023)

[0.01665355 0.02193916 0.0331283 0.16554322 0.18662717 0.57610859]

DecisionTreeClassifier(max\_features='sqrt', random\_state=600956192)

[0.04003656 0.08779017 0.06334442 0.11139794 0.07487894 0.62255196]

DecisionTreeClassifier(max\_features='sqrt', random\_state=352272321)

[0.10669355 0.11942664 0.03081629 0.28553516 0.13628694 0.32124142]

DecisionTreeClassifier(max\_features='sqrt', random\_state=615697673)

[0.3386003 0.08711497 0.13081975 0.15886308 0.06668229 0.21791961]

DecisionTreeClassifier(max\_features='sqrt', random\_state=160516793)

[0.12629416 0.04419408 0.02822362 0.33818226 0.21520308 0.2479028 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1909838463)

[0.13783948 0.05936108 0.19789719 0.18580645 0.09283705 0.32625874]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1110745632)

[0.11675543 0.08953569 0.10927283 0.26145753 0.10008016 0.32289837]

DecisionTreeClassifier(max\_features='sqrt', random\_state=93837855)

[0.14294937 0.07843927 0.08695443 0.31343329 0.1332696 0.24495405]

DecisionTreeClassifier(max\_features='sqrt', random\_state=454869706)

[0.08159374 0.03382821 0.35622231 0.08878521 0.17675858 0.26281194]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1780959476)

[0.07885916 0.10443928 0.11089514 0.05015521 0.05453662 0.60111459]

DecisionTreeClassifier(max\_features='sqrt', random\_state=2034098327)

[0.04393128 0.10780342 0.09557158 0.03702596 0.11812886 0.5975389 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1136257699)

[0.21559404 0.02093023 0.05852361 0.29777726 0.21294479 0.19423007]

DecisionTreeClassifier(max\_features='sqrt', random\_state=800291326)

[0.27573203 0.15269326 0.02272751 0.10843996 0.11826573 0.32214152]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1177824715)

[0.41057976 0.11311207 0.04149588 0.0433736 0.09971018 0.29172851]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1017555826)

[0.0509112 0.06489457 0.06563315 0.03745332 0.17069819 0.61040958]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1959150775)

[0.34404316 0.06772821 0.10321543 0.16716273 0.08667082 0.23117965]

DecisionTreeClassifier(max\_features='sqrt', random\_state=930076700)

[0.05276 0.06897285 0.07584642 0.12826921 0.11462991 0.5595216 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=293921570)

[0.03796185 0.13105751 0.06597808 0.0778511 0.05790695 0.62924451]

DecisionTreeClassifier(max\_features='sqrt', random\_state=580757632)

[0.0775731 0.08978542 0.03256879 0.16963629 0.07126006 0.55917635]

DecisionTreeClassifier(max\_features='sqrt', random\_state=80701568)

[0.07526654 0.11555722 0.36041927 0.04917359 0.08858618 0.31099721]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1392175012)

[0.14733084 0.08341027 0.20156191 0.36051311 0.11492905 0.09225482]

DecisionTreeClassifier(max\_features='sqrt', random\_state=505240629)

[0.27279999 0.03761398 0.07020699 0.11980252 0.1698699 0.32970662]

DecisionTreeClassifier(max\_features='sqrt', random\_state=642848645)

[0.31058243 0.09132156 0.03934485 0.11477177 0.10699218 0.3369872 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=481447462)

[0.0206101 0.12690045 0.08954202 0.08828914 0.0248403 0.64981799]

DecisionTreeClassifier(max\_features='sqrt', random\_state=954863080)

[0.0765737 0.06444468 0.21498127 0.28725213 0. 0.35674822]

DecisionTreeClassifier(max\_features='sqrt', random\_state=502227700)

[0.13176494 0.06042738 0.19425201 0.08805615 0.16429983 0.36119969]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1659957521)

[0.03734525 0.02747262 0.05508266 0.11905828 0.13071207 0.63032912]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1905883471)

[0.08221859 0.11144661 0.17030589 0.26937061 0.11611361 0.2505447 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1729147268)

[0.06069946 0.16187134 0.04092704 0.09347168 0.15980699 0.48322349]

DecisionTreeClassifier(max\_features='sqrt', random\_state=780912233)

[0.06402494 0.04436765 0.06252052 0.0554821 0.1295037 0.64410109]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1932520490)

[0.02909393 0.15310076 0.20874867 0.10527422 0.09217223 0.41161018]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1544074682)

[0.36051173 0.09992588 0.07360433 0.07993071 0.13840348 0.24762387]

DecisionTreeClassifier(max\_features='sqrt', random\_state=485603871)

[0.0893434 0.06761247 0.01639248 0.12384878 0.10996592 0.59283694]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1877037944)

[0.24701794 0.04702269 0.03416974 0.12037961 0.18573915 0.36567088]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1728073985)

[0.088103 0.09655952 0.04377152 0.07498743 0.11610588 0.58047265]

DecisionTreeClassifier(max\_features='sqrt', random\_state=848819521)

[0.11836996 0.04856513 0.07374171 0.11085206 0.04823387 0.60023727]

DecisionTreeClassifier(max\_features='sqrt', random\_state=426405863)

[0.28897332 0.13447774 0.11486443 0.08353963 0.1558868 0.22225808]

DecisionTreeClassifier(max\_features='sqrt', random\_state=258666409)

[0.08124079 0.03121456 0.09543327 0.06787232 0.08048223 0.64375683]

DecisionTreeClassifier(max\_features='sqrt', random\_state=2017814585)

[0.28833579 0.0293282 0.07492698 0.13570605 0.1857579 0.28594507]

DecisionTreeClassifier(max\_features='sqrt', random\_state=716257571)

[0.05905285 0.1182783 0.31556553 0.12569725 0.04306812 0.33833794]

DecisionTreeClassifier(max\_features='sqrt', random\_state=657731430)

[0.03943664 0.06606328 0.05536218 0.08410091 0.1170502 0.63798679]

DecisionTreeClassifier(max\_features='sqrt', random\_state=732884087)

[0.27763363 0.02201264 0.05194222 0.15828786 0.11552242 0.37460124]

DecisionTreeClassifier(max\_features='sqrt', random\_state=734051083)

[0.12334795 0.06184697 0.08980592 0.21146821 0.09730292 0.41622805]

DecisionTreeClassifier(max\_features='sqrt', random\_state=903586222)

[0.10130044 0.04095862 0.07296618 0.09924924 0.12221017 0.56331534]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1538251858)

[0.06643119 0.02197649 0.02077099 0.10795411 0.13950574 0.64336149]

DecisionTreeClassifier(max\_features='sqrt', random\_state=553734235)

[0.0179042 0.13735373 0.27850272 0.18871573 0.17352385 0.20399978]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1076688768)

[0.03904137 0.0377934 0.1282061 0.06237319 0.06945357 0.66313237]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1354754446)

[0.25415452 0.06811137 0.13621371 0.09847733 0.13553819 0.30750488]

DecisionTreeClassifier(max\_features='sqrt', random\_state=463129187)

[0.32940772 0.06205173 0.07114865 0.12423858 0.16225292 0.25090039]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1562125877)

[0.09162818 0.04730917 0.0090824 0.3349238 0.16398359 0.35307286]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1396067212)

[0.09398372 0.09722893 0.32099875 0.13397398 0.15660989 0.19720474]

DecisionTreeClassifier(max\_features='sqrt', random\_state=301492857)

[0.09028728 0.12434295 0.17985959 0.13360432 0.09487305 0.3770328 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=165035946)

[0.1051823 0.07718866 0.04040011 0.10510364 0.12552292 0.54660237]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1883779156)

[0.03972676 0.03590136 0.03613877 0.35190526 0.04272375 0.49360411]

DecisionTreeClassifier(max\_features='sqrt', random\_state=576702667)

[0.07379895 0.10858406 0.30272978 0.03420779 0.10893643 0.37174299]

DecisionTreeClassifier(max\_features='sqrt', random\_state=2097549636)

[0.04955924 0.02345712 0.39512797 0.0829612 0.11077074 0.33812373]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1971172102)

[0.04778386 0.13399473 0.07455469 0.34455981 0.14238775 0.25671915]

DecisionTreeClassifier(max\_features='sqrt', random\_state=438279108)

[0.11579044 0.06428075 0.3428078 0.15752302 0.1896888 0.1299092 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=656229423)

[0.06587906 0.05607583 0.28908613 0.16855401 0.08452499 0.33587998]

DecisionTreeClassifier(max\_features='sqrt', random\_state=897118847)

[0.09641396 0.06018287 0.10765326 0.37170217 0.13729444 0.2267533 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=580073460)

[0.270719 0.05426182 0.17397563 0.0380249 0.23065356 0.2323651 ]

DecisionTreeClassifier(max\_features='sqrt', random\_state=692819075)

[0.06161059 0.09449211 0.05833471 0.08919481 0.00549685 0.69087093]

DecisionTreeClassifier(max\_features='sqrt', random\_state=2127295436)

[0.04404368 0.11070336 0.26370302 0.16802957 0.07056821 0.34295216]

DecisionTreeClassifier(max\_features='sqrt', random\_state=657595236)

[0.30141542 0.09267196 0.07190997 0.04065186 0.12240575 0.37094504]

DecisionTreeClassifier(max\_features='sqrt', random\_state=351544500)

[0.05452705 0.04608159 0.0579415 0.2747659 0.12308972 0.44359423]

DecisionTreeClassifier(max\_features='sqrt', random\_state=1087879144)

[0.15826342 0.19008542 0.15127259 0.10691329 0.21636501 0.17710026]

Model Scores: 0.9032258064516129 Mean scores: 0.7865357643758767

## 

## **Question C Analysis**

**Annotation** I generated the feature importance of the 6 features (Figure 1) by taking the feature importance result from the 115 component trees. I also merged the feature importance of the 6 features into one graph (Figure 2). I also calculated the mean feature importance for each features by averaging the feature importance from the 115 component trees.

Chart, diagram

Description automatically generated

Figure 1 Features Importance for The 6 features

### **Observation From Figure 1 (above one)**

* The above figure shows the distributions of the feature importance for each feature:
  + Pelvic incidence: 0+ to 0.4+
  + Pelvic tilt: 0+ to 0.2+
  + Lumbar lordosis angle: 0+ to 0.4+
  + Sacral slope: 0+ to 0.4+
  + Pelvic radius: 0+ to 0.2+
  + Degree spondylolisthesis: 0+ to 0.6+

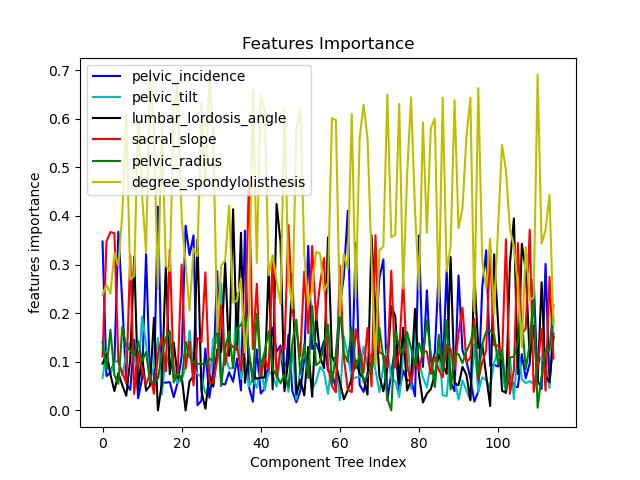


Figure 2:Feature Importance Overview

### **Observation from Figure 2 (above one)**

* The feature importance of degree spondylolisthesis is appreciably higher than that of the other fie features.

Graphical user interface

Description automatically generated

Figure 3: Mean Features Importance for the 6 features

### **Mean Feature Importance Data**

Mean Feature Importance of pelvic\_incidence: 0.12863396105770825

Mean Feature Importance of pelvic\_tilt: 0.08667147442890695

Mean Feature Importance of lumbar\_lordosis\_angle: 0.12414353670988675

Mean Feature Importance of sacral\_slope: 0.15122335909960322

Mean Feature Importance of pelvic\_radius: 0.11613742117405064

Mean Feature Importance of degree\_spondylolisthesis: 0.3931902475298444

### **Observation from Figure 3 and Mean Feature Importance Data (above one)**

* Form the above figure, it’s obvious that the mean feature importance of degree spondylolisthesis is well above that of the other five features
* The mean feature importance of pelvic tilt is below 0.10, which is quite lower than the others
* Except for the above 2 features, the mean feature importance of the rest 4 features are quite close, and is around 0.10-0.15.

### **Findings**

* From the above figures and analysis, it’s easy to conclude that the degree spondylolisthesis plays a relatively high importance among all the six features.
* The feature importance in different trees varies a lot.

# **Question D**

## **Question D Analysis 1 with The Selected Model in Question B**

I generate the results with the following parameters:

* Test Size: 0.1
* Train Split Random State: 27

## **Observations**

* I constructed the Gaussian Naïve Bayes Classifier, and use it to predict the result, and got the accuracy scores of 0.8064516129032258.
* Compared to the random forest classifier model, the Gaussian Naïve Bayes Classifier model has an appreciably lower accuracy scores.
* However, the training and running speed of the Gaussian Naïve Bayes Classifier model is much faster than the random forest model.

## **Findings**

In general, the Naive Bayes model is a simpler and faster model than the random forest model, but it has relatively lower accuracy scores and generalization performance. On the other hand, the random forest model may have higher accuracy and generalization performance, but its computation cost is much higher and harder to interpret.

### **Potential Reasons (Referred from internet)**

In summary, the choice of which classifier to use depends on the specific dataset and problem at hand, as well as the trade-off between performance, complexity, interpretability, and computational resources. While the Gaussian Naive Bayes classifier is a simple and fast algorithm, it may have lower performance than the random forest classifier in some cases, especially if the data has complex relationships or interactions between the features.

## **Question D Analysis 2 with Test Size ranging from 0.1 to 0.9**

### **Methodology**

I generated the best accuracy scores for the random forest classifier with each test size from 0.1 to 0.9, and also generated the Gaussian Naïve Bayes Classifier result with the test size of 0.1-0.9.

### **Observation**

As the graph shown, it’s easy to see that with different test size, the accuracy scores of gaussian naïve bayes classifier is always much poorer than that of random forest classifier.

Chart, line chart

Description automatically generated

### Potential Reasons (Referred from internet)

* Robustness to outliers: The random forest classifier is less sensitive to outliers and noise in the data than the Gaussian Naive Bayes classifier, which can lead to better performance on noisy or outlier-prone datasets.
* Feature selection and importance estimation: The random forest classifier can perform feature selection and estimate feature importance based on the feature's contribution to the model's performance, which can help to identify the most relevant features for the classification task.
* Ensembling and generalization: The random forest classifier combines the predictions of multiple decision trees and can generalize well to new and unseen data, while the Gaussian Naive Bayes classifier may overfit to the training data if the distributional assumptions are violated.

# **Appendix**

## **My Source GitHub**

My code and related figures for this project is store in the GitHub Repository:

<https://github.com/alfreddLUO/Random-Forest-and-Naive-Bayes-Classifier-Assignment>