**Problem 1**

**Compute sensitivity, specificity, precision, accuracy, F-meassure, F2, and MCC measures. You have to show all your calculations.**

|  |  |
| --- | --- |
| TP:254 | FN:36 |
| FP:72 | TN:324 |

**Problem 2**

**Use a significance level of 1%. If there is a significant difference, which one is better?**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Iteration | M1 | M2 | Avg(M1)-Avg(M2) | (M1-M2-(Avg(M1)-Avg(M2)))2 |
| 1 | 0.12 | 0.08 | 0.015 | 0.000625 |
| 2 | 0.12 | 0.1 | 0.015 | 0.000025 |
| 3 | 0.15 | 0.22 | 0.015 | 0.007225 |
| 4 | 0.15 | 0.1 | 0.015 | 0.001225 |
| 5 | 0.03 | 0.07 | 0.015 | 0.003025 |
| 6 | 0.17 | 0.11 | 0.015 | 0.002025 |
| 7 | 0.2 | 0.1 | 0.015 | 0.007225 |
| 8 | 0.14 | 0.11 | 0.015 | 0.000225 |
| 9 | 0.1 | 0.17 | 0.015 | 0.007225 |
| 10 | 0.14 | 0.11 | 0.015 | 0.000225 |
|  | Avg=0.132 | Avg=0.117 |  | Sum=0.02905 |

Because , we can not reject the null hypothesis, and conclude that M1 is more accurate than M2

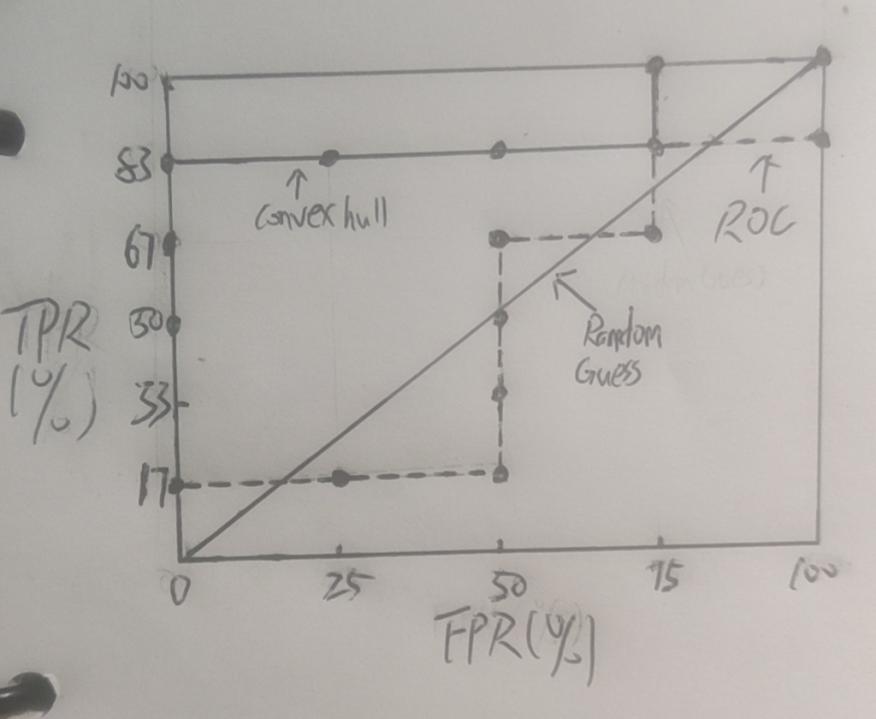
**Problem 3**

|  |  |  |
| --- | --- | --- |
| **Tuple\_id** | **Actual Class** | **Probability** |
| **1** | **P** | **0.92** |
| **2** | **N** | **0.70** |
| **3** | **N** | **0.76** |
| **4** | **P** | **0.92** |
| **5** | **P** | **0.83** |
| **6** | **P** | **0.89** |
| **7** | **N** | **0.79** |
| **8** | **P** | **0.73** |
| **9** | **N** | **0.82** |
| **10** | **P** | **0.96** |

1. **. For each row, compute TP, FP, TN, FN, TPR, and FPR.**

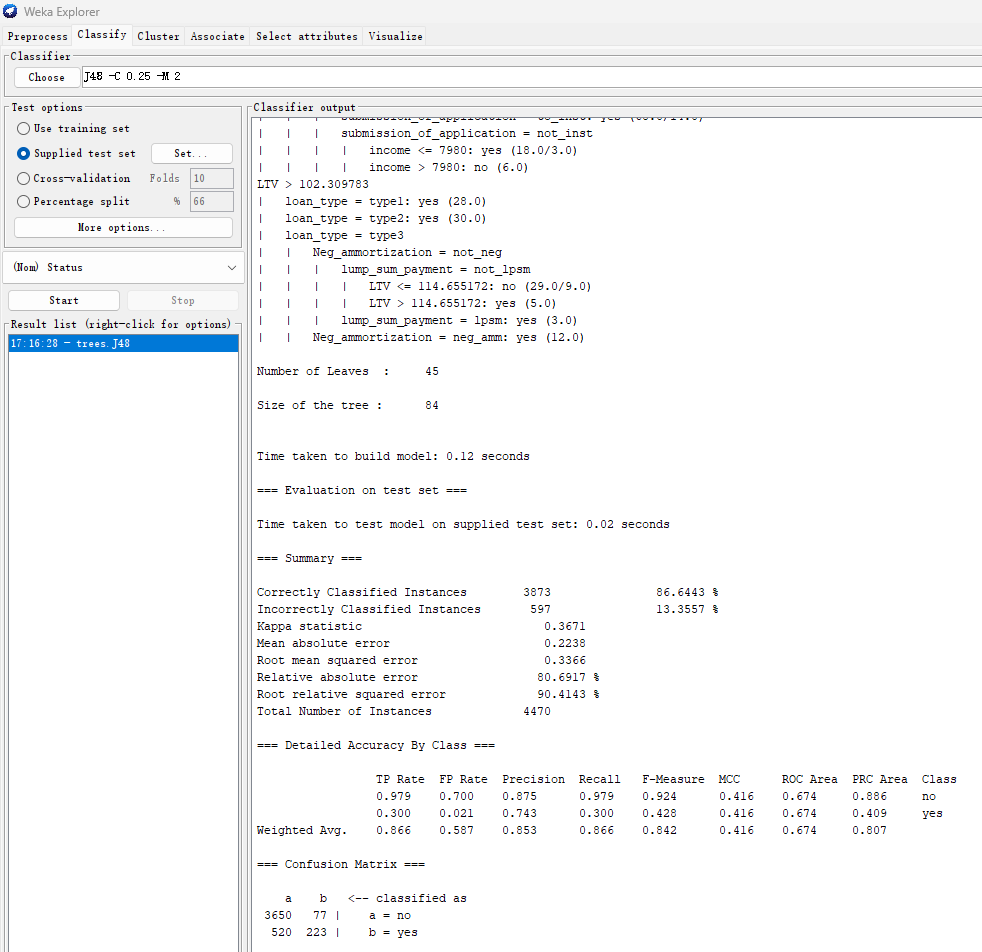
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tuple\_id | Actual Class | Probability | TP | FP | TN | FN | TPR | FPR |
| 1 | P | 0.92 | 3 | 0 | 4 | 3 | 0.5 | 0 |
| 2 | N | 0.70 | 6 | 4 | 0 | 0 | 1 | 1 |
| 3 | N | 0.76 | 5 | 3 | 1 | 1 | 0.83 | 0.75 |
| 4 | P | 0.92 | 3 | 0 | 4 | 3 | 0.5 | 0 |
| 5 | P | 0.83 | 5 | 0 | 4 | 1 | 0.83 | 0 |
| 6 | P | 0.89 | 4 | 0 | 4 | 2 | 0.67 | 0 |
| 7 | N | 0.79 | 5 | 2 | 2 | 1 | 0.83 | 0.5 |
| 8 | P | 0.73 | 6 | 3 | 1 | 0 | 1 | 0.75 |
| 9 | N | 0.82 | 5 | 1 | 3 | 1 | 0.83 | 0.25 |
| 10 | P | 0.96 | 1 | 0 | 4 | 5 | 0.17 | 0 |

1. **. Plot the ROC curve for the dataset. You must draw the curve yourself (i.e., don’t use Weka,R, or other software to generate the curve).**

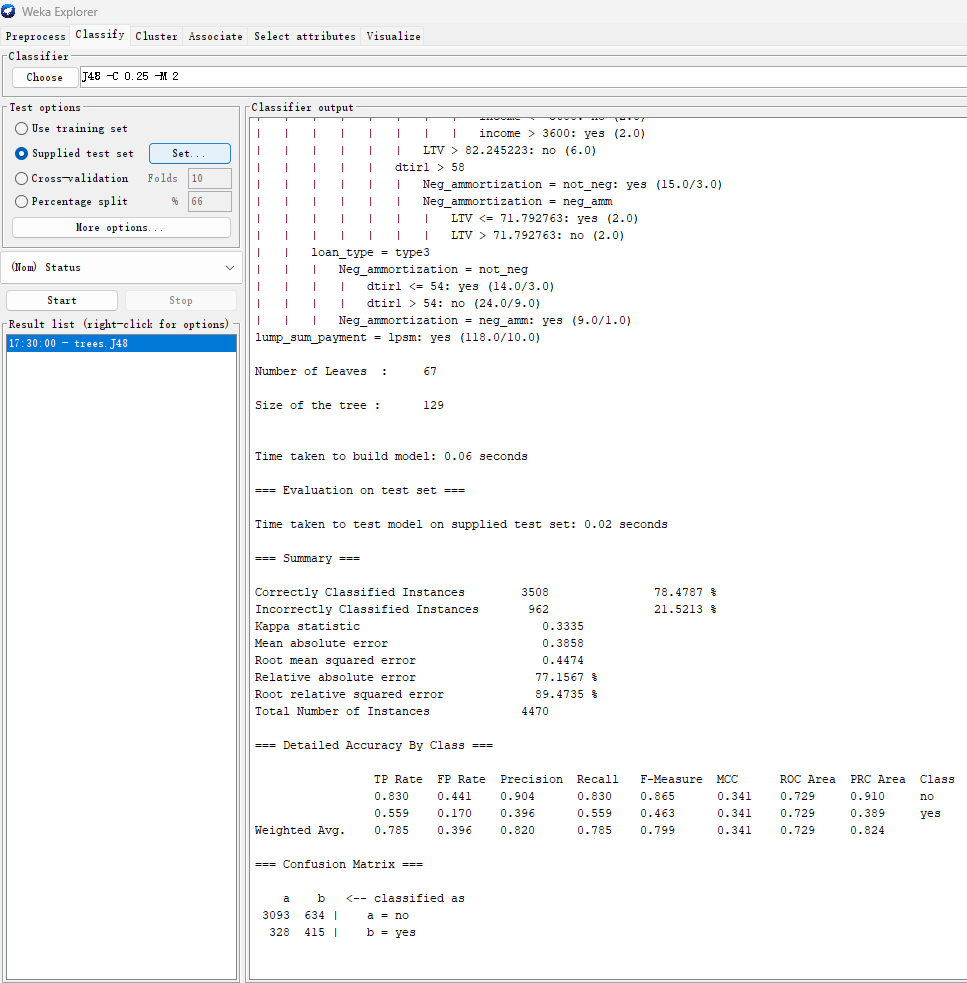
****

**Problem 4**

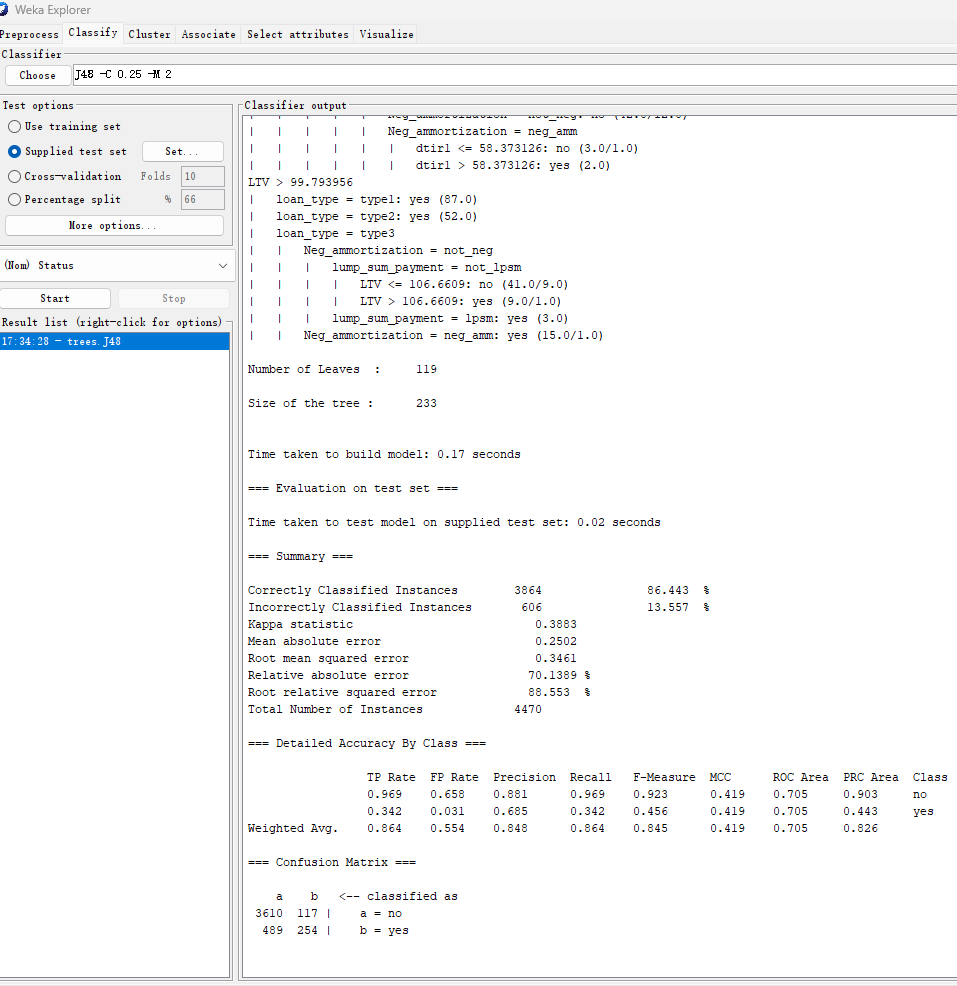
1. **. Build a decision tree model from a4\_p4\_train.arff using J48 and test it on a4\_p4\_test.arff. Include the resulting confusion matrix in your submission.**

****

1. **. Create an undersampled training dataset from a4\_p4\_train.arff and name it a4\_p4\_train\_undersampled.arff. Build a decision tree model from a4\_p4\_train\_undersampled.arff using J48 and test it on a4\_p4\_test.arff. Include the resulting confusion matrix in your submission.**

****

1. **. Create an oversampled training dataset from a4\_p4\_train.arff and name it a4\_p4\_train\_oversampled.arff. Build a decision tree model from a4\_p4\_train\_oversampled.arff using J48 and test it on a4\_p4\_test.arff. Include the resulting confusion matrix in your submission.**

****

1. **. What conclusion can you draw from this experiment?**

First, defined the categories: True Positive (TP): The model correctly predicted the positive class (b = yes).

True Negative (TN): The model correctly predicted the negative class (a = no).

False Positive (FP): The model incorrectly predicted the positive class (b = yes).

False Negative (FN): The model incorrectly predicted the negative class (a = no).

Then, we can calculate the confusion matrices as follows:

Normal training set: TP = 223, TN = 3650, FP = 77, FN = 520

Undersampled training set: TP = 415, TN = 3093, FP = 634, FN = 328

Oversampled training set: TP = 254, TN = 3610, FP = 117, FN = 489

Normal training set

Precision: 223 / (223 + 77) = 0.743

Recall: 223 / (223 + 520) = 0.300

Accuracy: (223 + 3650) / (223 + 3650 + 77 + 520) = 0.860

F1 Score: 2 \* (0.743 \* 0.300) / (0.743 + 0.300) = 0.428

Undersampled training set

Precision: 415 / (415 + 634) = 0.396

Recall: 415 / (415 + 328) = 0.558

Accuracy: (415 + 3093) / (415 + 3093 + 634 + 328) = 0.774

F1 Score: 2 \* (0.396 \* 0.558) / (0.396 + 0.558) = 0.464

Oversampled training set

Precision: 254 / (254 + 117) = 0.685

Recall: 254 / (254 + 489) = 0.342

Accuracy: (254 + 3610) / (254 + 3610 + 117 + 489) = 0.858

F1 Score: 2 \* (0.685 \* 0.342) / (0.685 + 0.342) = 0.456

At last, we can get the conclusions:

The model trained on the normal dataset has the highest accuracy but struggles with identifying positive cases (low recall).

The model trained on the undersampled dataset has the highest recall (good at identifying positive cases) but makes more mistakes overall (lower accuracy) and within predicted positives (lower precision).

The model trained on the oversampled dataset strikes a balance, with accuracy nearly as high as the normal dataset and improved recall.

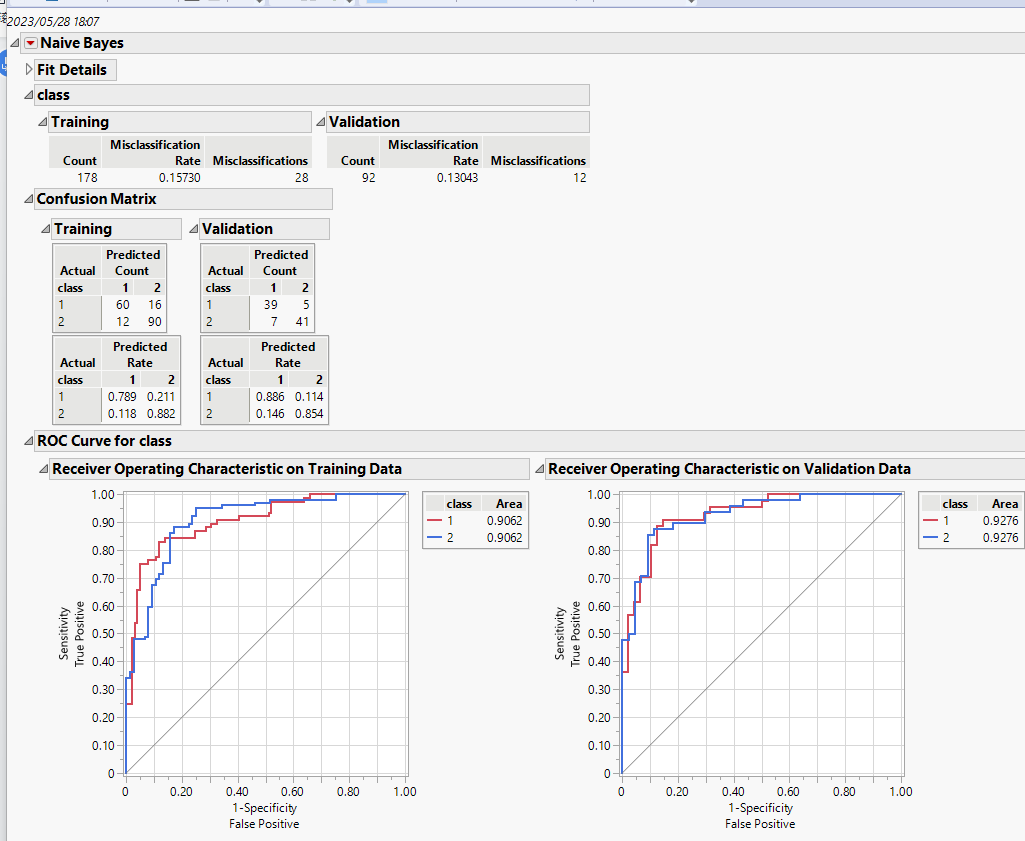
In conclusion, the best model depends on your specific needs and the costs associated with false positives and false negatives. The oversampling approach seems to provide a good balance in this case.

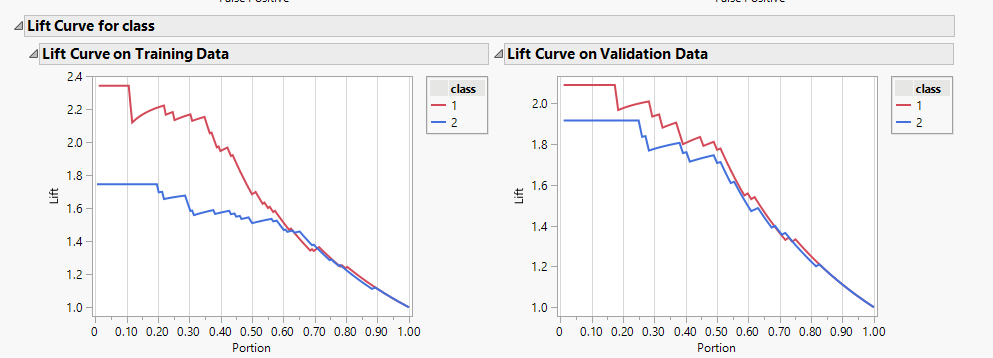
**Problem 5**

|  |  |  |
| --- | --- | --- |
|  | **1** | **2** |
| **1** | **TP** | **FN** |
| **2** | **FP** | **TN** |

1. **Naïve Bayes**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | Class |  |
| 0.89 | 0.15 | 0.85 | 0.89 | 0.87 | 0.74 | 1 | count |
| 0.89 | 0.15 | 0.86 | 0.89 | 0.87 | 0.74 | 1 | rate |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | Class |  |
| 0.85 | 0.11 | 0.89 | 0.85 | 0.87 | 0.74 | 2 | count |
| 0.85 | 0.11 | 0.88 | 0.85 | 0.86 | 0.74 | 2 | rate |

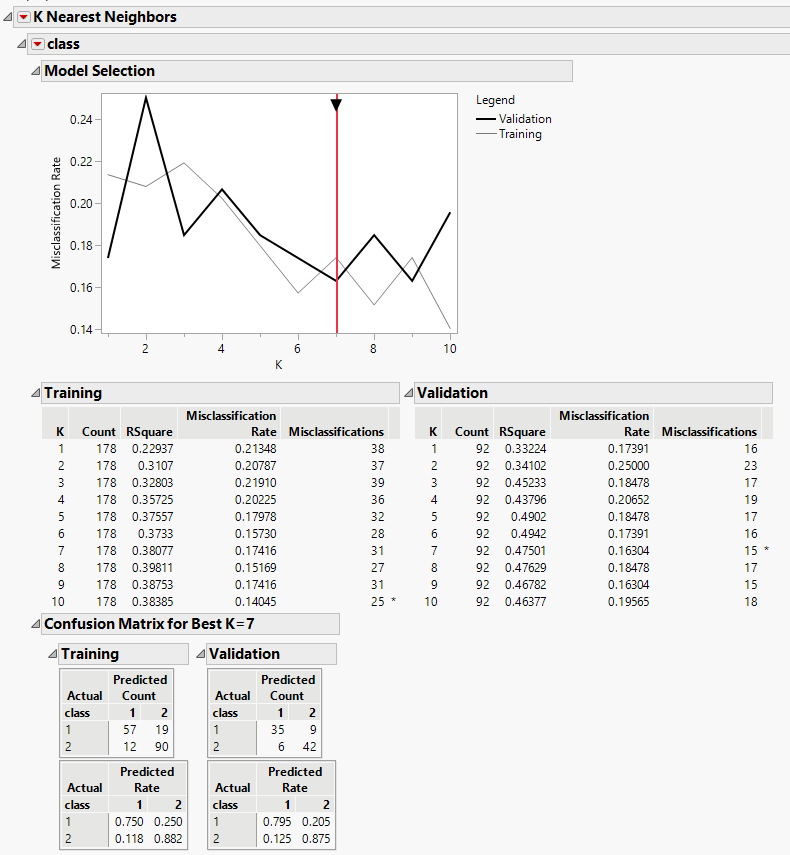
****

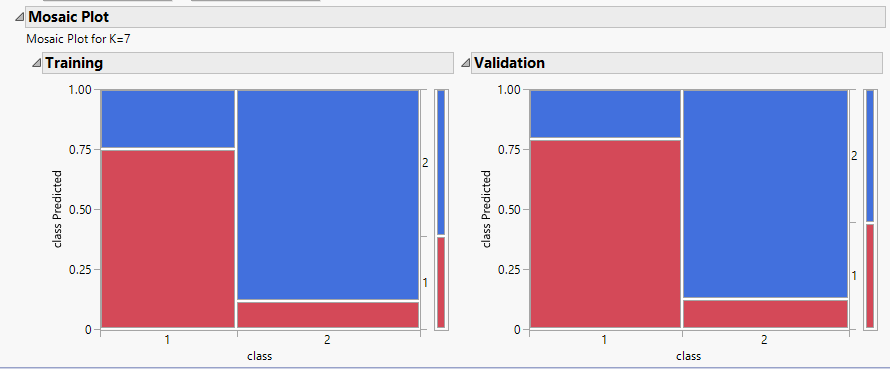
****

1. **KNN**

Best K = 7

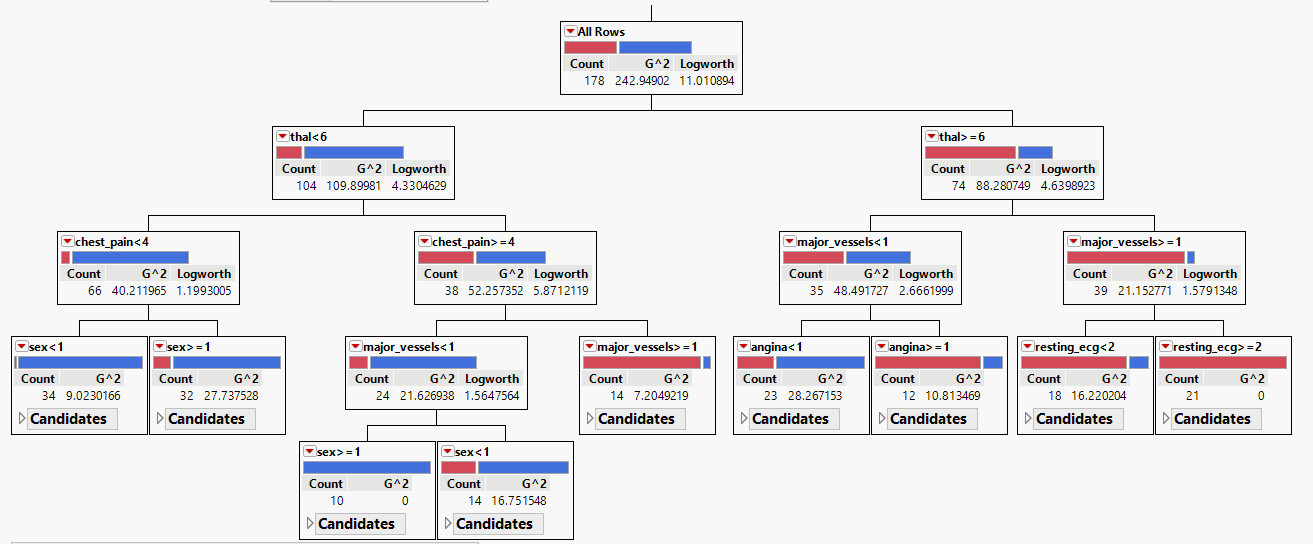
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | Class |  |
| 0.8 | 0.13 | 0.85 | 0.8 | 0.82 | 0.67 | 1 | count |
| 0.8 | 0.13 | 0.86 | 0.8 | 0.83 | 0.67 | 1 | rate |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | Class |  |
| 0.88 | 0.2 | 0.82 | 0.88 | 0.85 | 0.67 | 2 | count |
| 0.88 | 0.21 | 0.81 | 0.88 | 0.84 | 0.67 | 2 | rate |

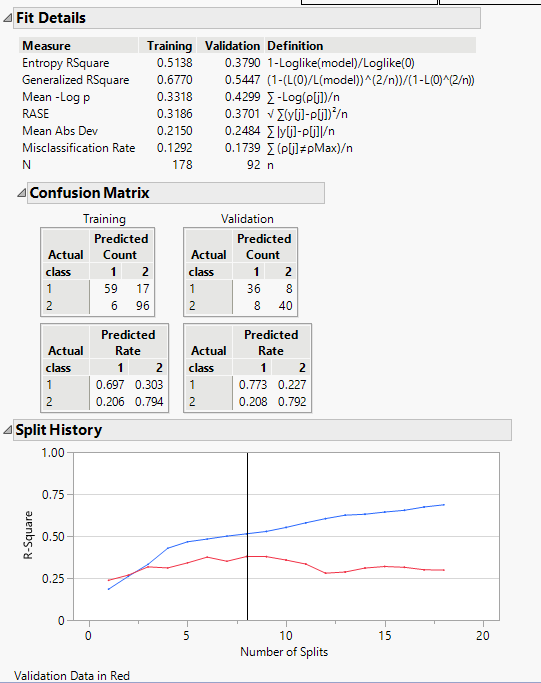
****

****

1. **Partition Model (decision tree)**

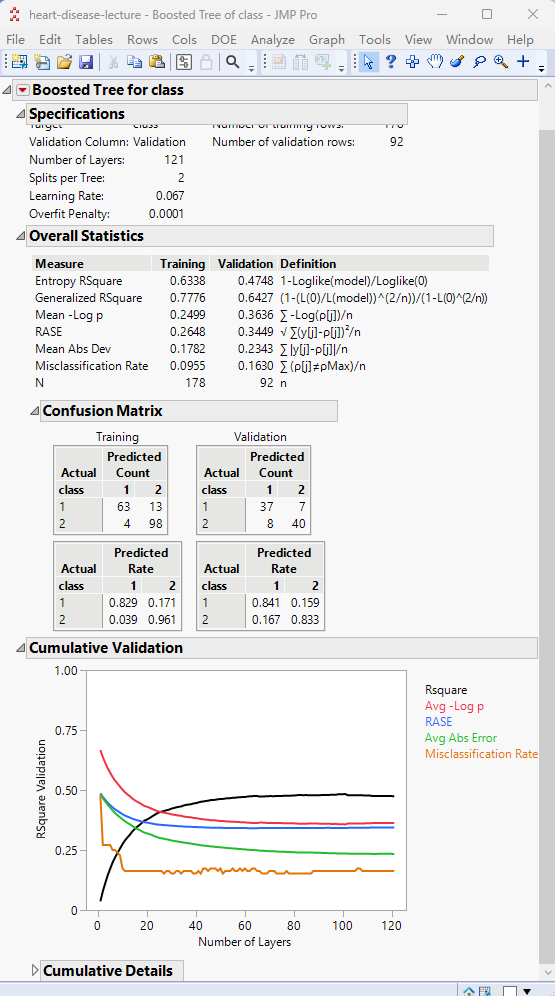
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | Class |  |
| 0.82 | 0.17 | 0.82 | 0.82 | 0.82 | 0.65 | 1 | count |
| 0.77 | 0.21 | 0.79 | 0.77 | 0.78 | 0.57 | 1 | rate |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | Class |  |
| 0.83 | 0.18 | 0.83 | 0.83 | 0.83 | 0.65 | 2 | count |
| 0.79 | 0.23 | 0.78 | 0.79 | 0.78 | 0.57 | 2 | rate |

****

****

1. **Boosted Tree**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | Class |  |
| 0.84 | 0.17 | 0.82 | 0.84 | 0.83 | 0.67 | 1 | count |
| 0.84 | 0.17 | 0.83 | 0.84 | 0.83 | 0.67 | 1 | rate |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | Class |  |
| 0.83 | 0.16 | 0.85 | 0.83 | 0.84 | 0.67 | 2 | count |
| 0.83 | 0.16 | 0.84 | 0.83 | 0.83 | 0.67 | 2 | rate |

****

1. **Neural Network**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | Class |  |
| 0.84 | 0.13 | 0.86 | 0.84 | 0.85 | 0.72 | 1 | count |
| 0.84 | 0.13 | 0.87 | 0.84 | 0.85 | 0.72 | 1 | rate |
| TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | Class |  |
| 0.88 | 0.16 | 0.86 | 0.88 | 0.87 | 0.72 | 2 | count |
| 0.88 | 0.16 | 0.85 | 0.88 | 0.86 | 0.72 | 2 | rate |

Hidden layers=5

