Grant Ikehara, Cameron Healy, Dominic Soares

CPSC 310

HW6

Descriptions of Processes and Results

Step 1:

1. We made a function that takes the tables and determines their length to figure out how many instances will be in the test and remainder set and then partitioned the data based on those size values.
2. We made a bootstrapping function that can take a remainder set of instances and the N variable that specifies the number of trees, which is the number of bootstrap lists. The function returns one list with each sub-list being a bootstrap set of instances, so when we loop to create each tree for the random forest, we also iterate over the bootstrap list to use each sub-list bootstrap sample to create the tree by randomly selecting attributes.
3. We wrote a function to test each tree and find the tree’s accuracy when it is used on the test set. The M variable for the number of trees we want is passed in and we take the M best trees and store them.
4. Using the trees that were stored, we added code that uses them to classify the instance that is being classified by using majority voting.

Step2:

We passed in the titanic and auto data tables with N = 20, M = 7, and F = 2 to the random forest function to find the accuracy of the Random Forest method. The parameters force the program to make 20 trees from random subsets of 2 attributes, then take the 7 best trees to make majority voting predictions when testing on the test set. Each of the predictions and results are put into the confusion matrix. We also made a normal tree that uses the entropy values to order the tree to compare the confusion matrix and accuracy against the random forest’s.

Step 3:

We ran our program with different values of the three parameters N,M,F to determine if those values change the algorithm enough to change the prediction accuracy of the random forest classifier. The most accurate tree used the parameters, N=30, M=15, F=2. Trees with polar opposite N and M values were not accurate, but trees with an M value about half of the N value gave accurate trees.

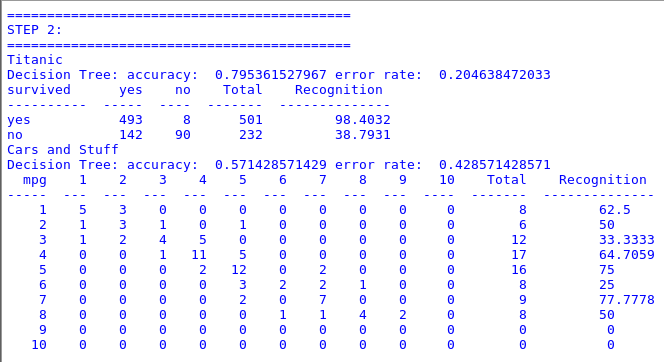
Step 4:

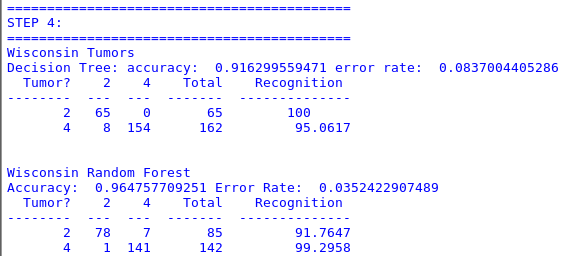
We ensured that our functions for the random forest on the titanic and auto data can also be used on the Wisconsin data, including the read csv function. Using the functions, we ran the Wisconsin text file through the random forests algorithm with multiple parameters to find the most accurate tree and parameters, with a normal decision tree for comparison.

Test Results:

Normal Decision Tree

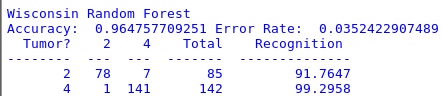
|  |  |  |
| --- | --- | --- |
|  | Accuracy % | Error Rate % |
| Auto Data | 45-60 | 40-55 |
| Titanic | 70-80 | 20-30 |
| Wisconsin | 89-93 | 7-11 |

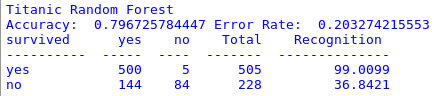


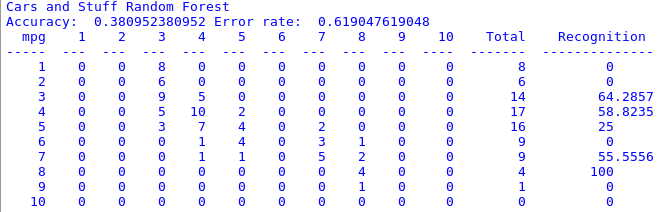


Random Forest - N=20, M=7, F=2

|  |  |  |
| --- | --- | --- |
|  | Accuracy % | Error Rate % |
| Auto Data | 35-50 | 50-65 |
| Titanic | 70-80 | 20-30 |
| Wisconsin | 95-98 | 2-5 |

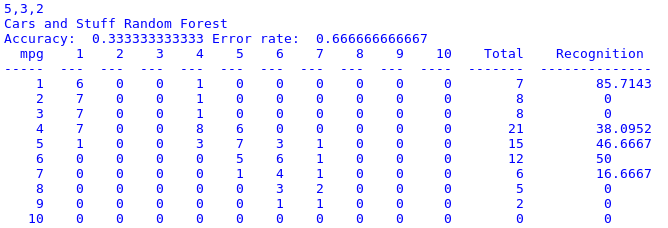


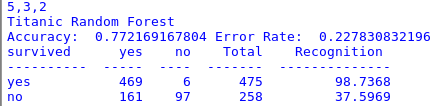


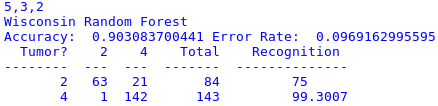


Random Forest - N=5, M=3, F=2

|  |  |  |
| --- | --- | --- |
|  | Accuracy % | Error Rate % |
| Auto Data | 33-41 | 59-67 |
| Titanic | 70-78 | 22-30 |
| Wisconsin | 90-95 | 5-10 |

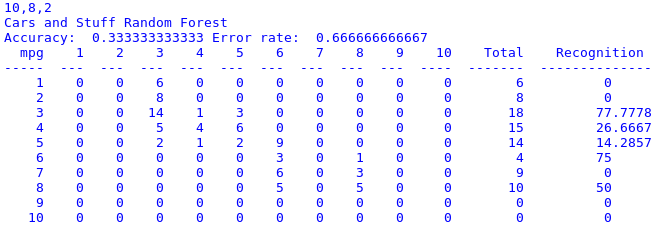


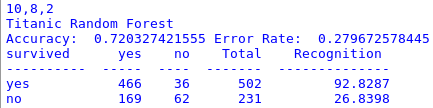


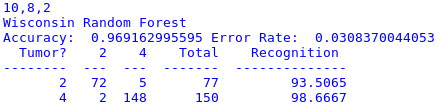


Random Forest - N=10, M=8, F=2

|  |  |  |
| --- | --- | --- |
|  | Accuracy % | Error Rate % |
| Auto Data | 33-42 | 58-67 |
| Titanic | ~72 | ~28 |
| Wisconsin | ~97 | ~3 |

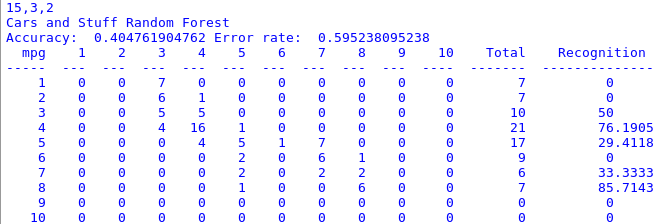


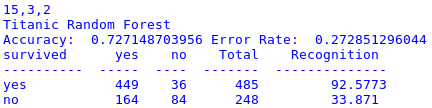


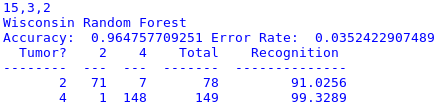


Random Forest - N=15, M=3, F=2

|  |  |  |
| --- | --- | --- |
|  | Accuracy % | Error Rate % |
| Auto Data | 39-44 | 56-61 |
| Titanic | 70-74 | 26-30 |
| Wisconsin | 94-97 | 3-6 |

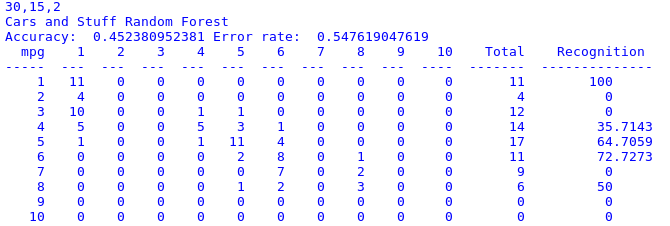


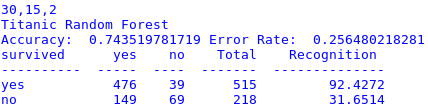


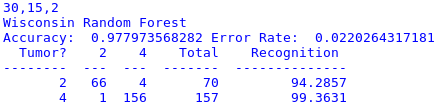


Random Forest - N=30, M=15, F=2

|  |  |  |
| --- | --- | --- |
|  | Accuracy % | Error Rate % |
| Auto Data | 40-45 | 65-60 |
| Titanic | 73-80 | 20-27 |
| Wisconsin | 96-98 | 2-4 |







Random Forest - N=50, M=5, F=2

|  |  |  |
| --- | --- | --- |
|  | Accuracy % | Error Rate % |
| Auto Data | 28-40 | 60-72 |
| Titanic | 71-74 | 26-29 |
| Wisconsin | 96-98 | 2-4 |

