Dorian Benitez
CS 4375.501
February 16, 2020

Homework 4 Report

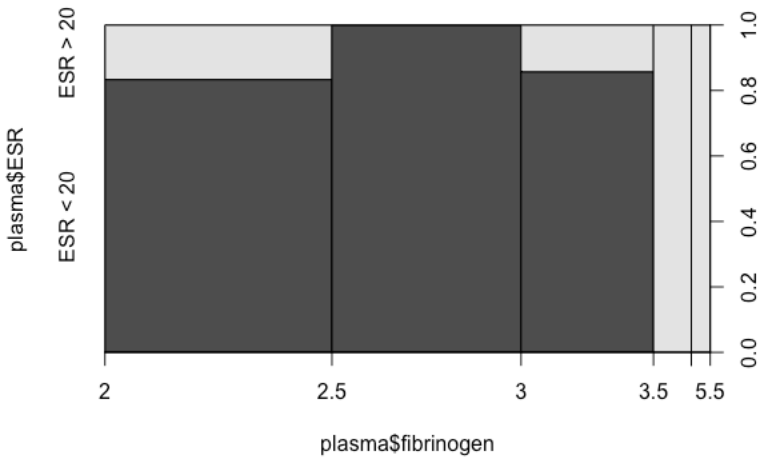
For this assignment, the objective is to implement two classification machine learning algorithms, Logistic Regression and Naïve Bayes, and compare the results and performance to the equivalent functions within C++ and R. A hypothesis of the project outcome is that implementation in R will be much simpler and more accurate when compared to that of C++. Although, I believe a strong benefit of utilizing C++ is that the runtime will be much faster than that of R.

LOGISTIC REGRESSION

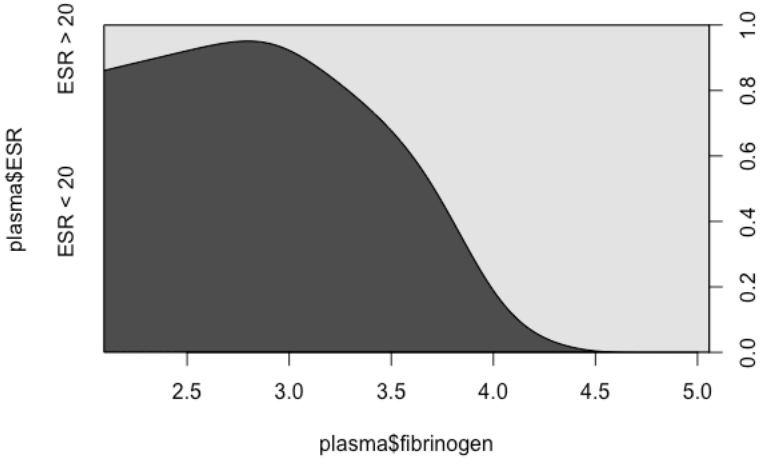
For practicality reasons and to obtain the ground truth values, Logistic Regression was first implemented in R, and then in C++. Although, the results obtained from both executions are fairly similar. The runtimes also went as expected, with a faster execution time found in C++ than in R. Execution time in R was measured by starting time count right before the following for loop began, and time end as soon as the loop was finished executing. The same idea was implemented in C++ to maintain a consistent form of measurement.

```
> for (i in 1:500000){
+   probVec <- sigmoid(data_matrix %*% weights)
+   error <- labels - probVec
+   weights <- weights + learning_rate * t(data_matrix) %*% error
+ }
> endTime <- proc.time()
> weights
          [,1]
[1,] -6.845075
[2,]  1.827081
> print(paste("Total execution time: ", endTime - startTime))
[1] "Total execution time:  4.181"            "Total execution time:  0.327"
[3] "Total execution time:  4.58000000000001" "Total execution time:  0"
[5] "Total execution time:  0"
```

R Code Output (Logistic Regression)



(Plot from R – Logistic Regression)



(CD Plot from R – Logistic Regression)

```
Weights after looping
  -6.8451
   1.8271
Run time of loop: 3.634
```

C++ Code Output (Logistic Regression)

Code used for training data in C++:

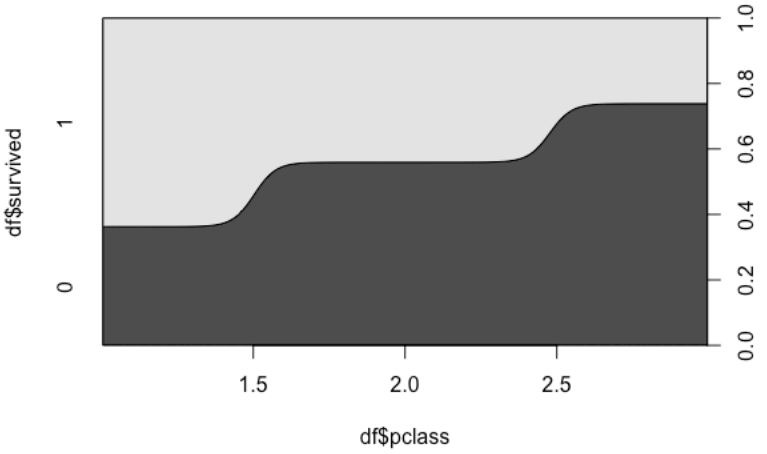
```cpp
float startTime = clock();

// Loop to match that in R
for (int i = 0; i < 500000; i++) {
    mat probVec= DataMatrix * weights;
    for (int j =0; j<32; j++) {
        probVec(j) = sigmoid(probVec(j));
    }
    mat error = labels - probVec;
    weights = weights + learning_rate * DataMatrix.t() * error;
}
float endTime = clock();
```

NAIVE BAYES

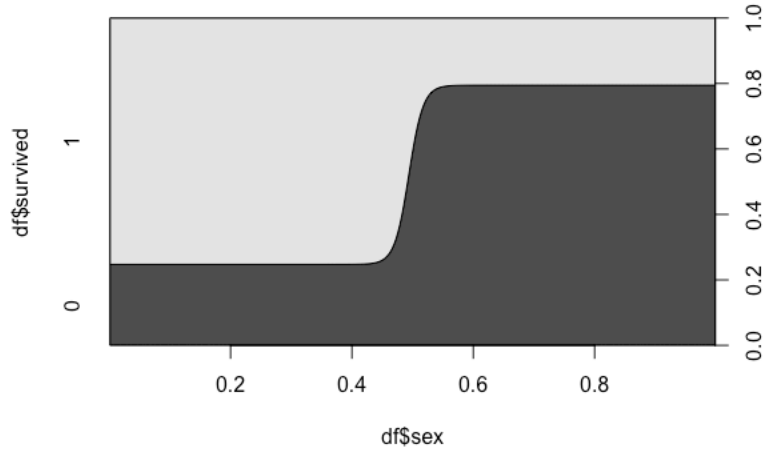
For implementing Naïve Bayes, the same steps were followed as in the logistic regression procedure of the homework. Implementation was first made in R to acquire the ground truth values, followed by implementation in C++. The difference in timing measurement for Naïve Bayes is that it is a measurement of execution of the entire program, rather than just a loop. This was also followed in C++ to obtain equivalent and consistent measurement standards.

```
          Reference
Prediction  0  1
         0 69 25
         1 10 42

               Accuracy : 0.7603
                 95% CI : (0.6827, 0.827)
    No Information Rate : 0.5411
    P-Value [Acc > NIR] : 3.612e-08

                  Kappa : 0.5089

 Mcnemar's Test P-Value : 0.01796

            Sensitivity : 0.6269
            Specificity : 0.8734
         Pos Pred Value : 0.8077
         Neg Pred Value : 0.7340
             Prevalence : 0.4589
         Detection Rate : 0.2877
   Detection Prevalence : 0.3562
      Balanced Accuracy : 0.7501

       'Positive' Class : 1

>
> # Saves and prints the runtime of the script
> endTime <- proc.time()
> totTime <- endTime - startTime
> print(paste("Total execution time: ", totTime))
[1] "Total execution time:  0.181999999999999"
[2] "Total execution time:  0.073"
[3] "Total execution time:  0.29099999999994"
[4] "Total execution time:  0"
[5] "Total execution time:  0"
```

R Code Output (Naïve Bayes)



CD Plot (Survived v. PClass)



CD Plot (Survived v. Sex)

REFERENCES

[1] All PDF and Git examples provided by Dr. Karen Mazidi on Piaza

[2] https://github.com/masumhabib/quest/wiki/How-to-Install-Armadillo

[3] http://arma.sourceforge.net/docs.html

[4] https://www.youtube.com/watch?v=mrP4CyW4tKA

[5] StackOverflow