Question 3)

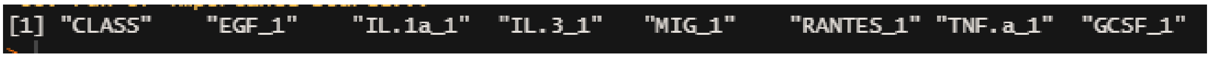
Explanations:

The Boruta package was used to select the features that have the largest impact on the determining the class result. Boruta is a wrapper function that applies the random forest algorithm to the dataset its given, requiring specification of the evalutated attributes and the attribute that is determined by the others, and returns the important attributes in an boruta object. The tentative rough fix function is another method of the Boruta package that tries to identify if features on the edge of important or un important should be kept or not. The getSelectedAttributes function is used to get the names of the features selected in a vector, which is then used to loop through the dataset and remove all other proteins. The CLASS column was looped through replacing any values that were not “AD” with “NON_AD”, the matrix was then written to a file for Weka to use.

For Classfication the Weka software was used to generate a J48 pruned tree using the features selected. The collapseTree setting within the classifier in Weka was set to true as this ensured that unneccesary checks were not being formed. Other values were mainly set to default as they didn’t have a significant impact on the classifiers perforamance.

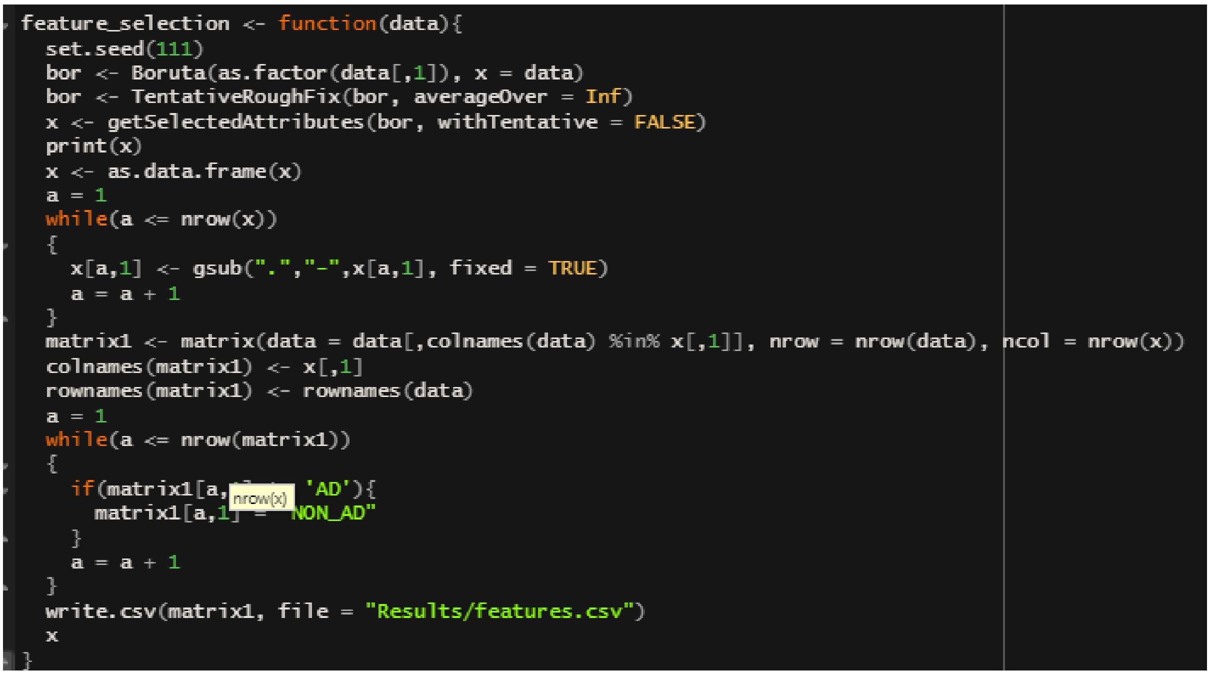
a)

The following proteins were chosen as the most important for determining the outcome of the CLASS value.



```
[1] "CLASS"    "EGF_1"    "IL.1a_1"  "IL.3_1"   "MIG_1"    "RANTES_1" "TNF.a_1"  "GCSF_1"
```

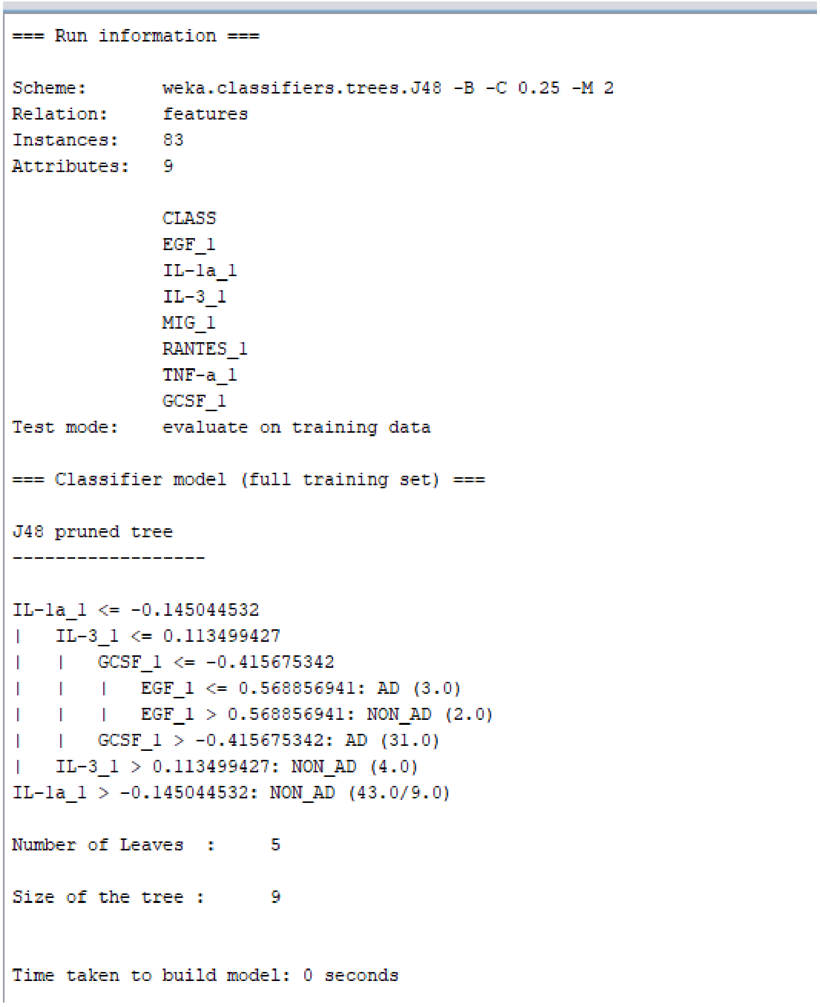
Code**:**



```
feature_selection <- function(data){
  set.seed(111)
  bor <- Boruta(as.factor(data[,1]), x = data)
  bor <- TentativeRoughFix(bor, averageOver = Inf)
  x <- getSelectedAttributes(bor, withTentative = FALSE)
  print(x)
  x <- as.data.frame(x)
  a = 1
  while(a <= nrow(x))
  {
    x[a,1] <- gsub(".","-",x[a,1], fixed = TRUE)
    a = a + 1
  }
  matrix1 <- matrix(data = data[,colnames(data) %in% x[,1]], nrow = nrow(data), ncol = nrow(x))
  colnames(matrix1) <- x[,1]
  rownames(matrix1) <- rownames(data)
  a = 1
  while(a <= nrow(matrix1))
  {
    if(matrix1[a,nrow(x) 'AD'){
      matrix1[a,1] = NON_AD"
    }
    a = a + 1
  }
  write.csv(matrix1, file = "Results/features.csv")
  x
}
```

b)

Weka outputs the classification system below, with the pruned tree being the actual system that will be used to test on the different datasets.

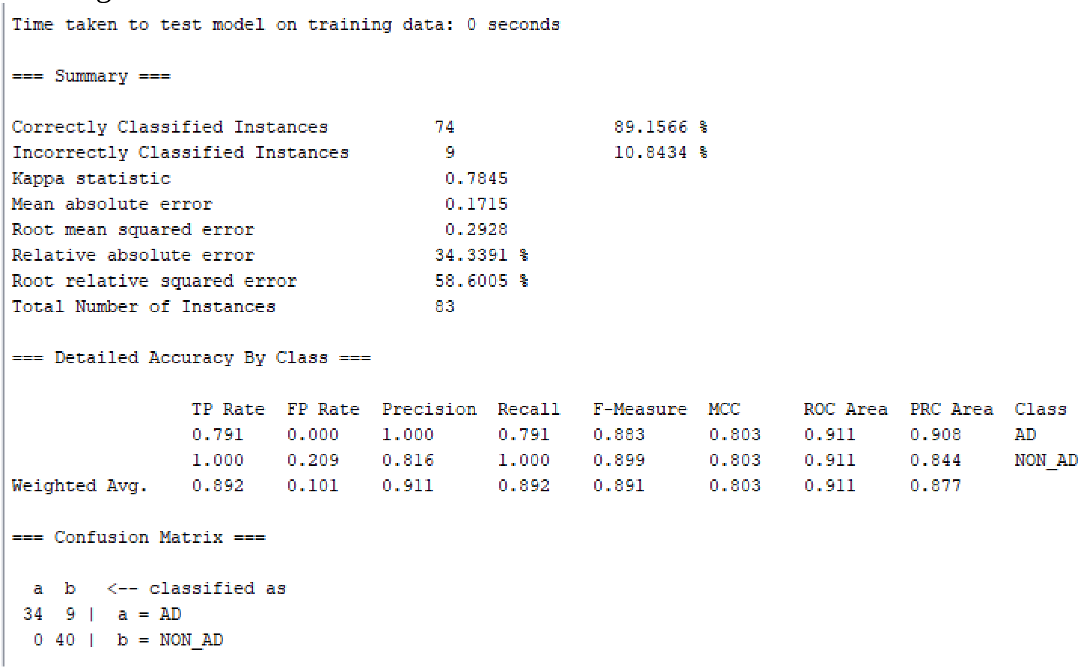


```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -B -C 0.25 -M 2
Relation:     features
Instances:    83
Attributes:   9

              CLASS
              EGF_1
              IL-1a_1
              IL-3_1
              MIG_1
              RANTES_1
              TNF-a_1
              GCSF_1
Test mode:    evaluate on training data

=== Classifier model (full training set) ===

J48 pruned tree
------------------

IL-1a_1 <= -0.145044532
|   IL-3_1 <= 0.113499427
|   |   GCSF_1 <= -0.415675342
|   |   |   EGF_1 <= 0.568856941: AD (3.0)
|   |   |   EGF_1 > 0.568856941: NON_AD (2.0)
|   |   GCSF_1 > -0.415675342: AD (31.0)
|   IL-3_1 > 0.113499427: NON_AD (4.0)
IL-1a_1 > -0.145044532: NON_AD (43.0/9.0)

Number of Leaves  :     5

Size of the tree :      9


Time taken to build model: 0 seconds
```

c)

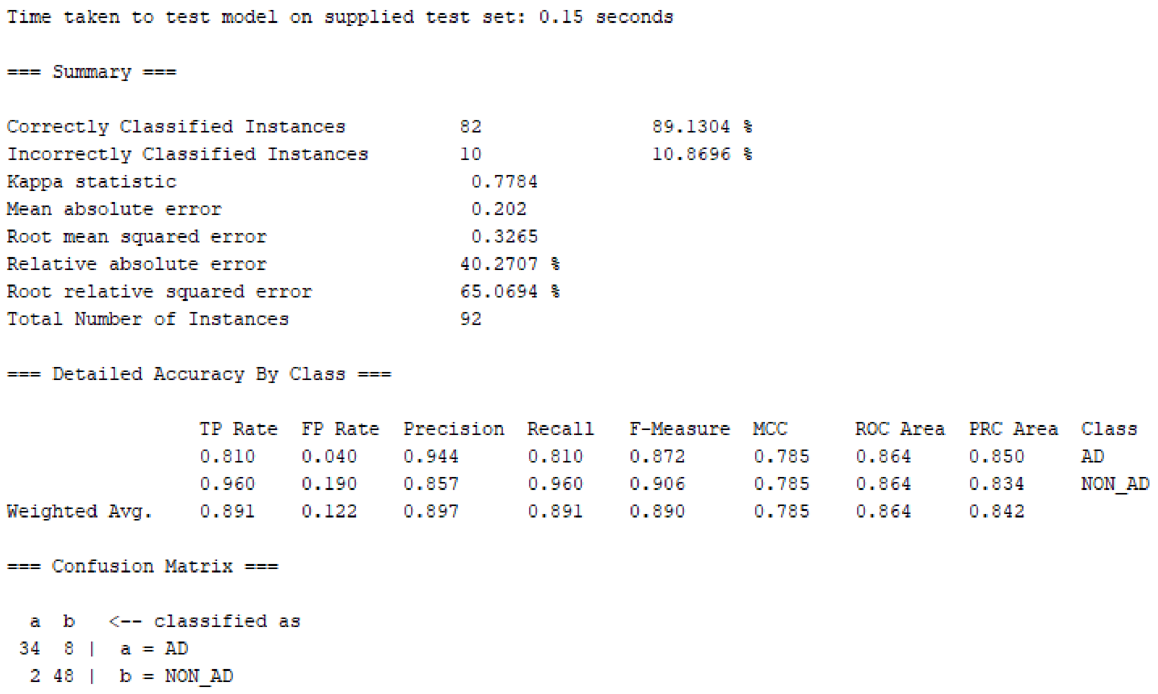
Weka outputs with the classification system.

Training Set Results:



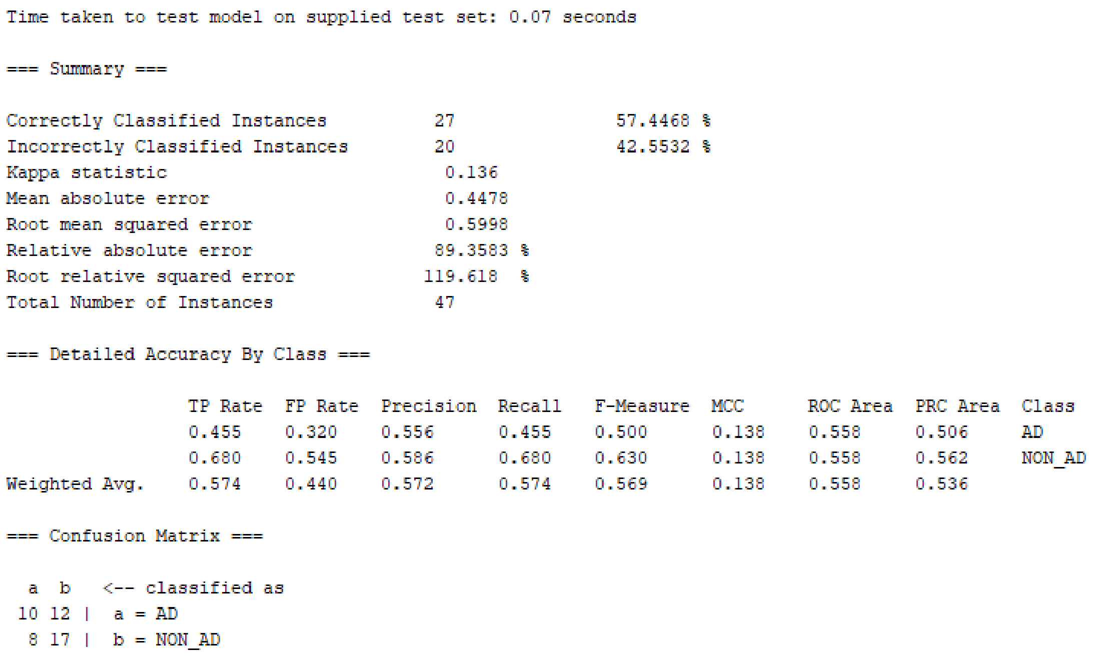
```
Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances          74               89.1566 %
Incorrectly Classified Instances         9               10.8434 %
Kappa statistic                          0.7845
Mean absolute error                      0.1715
Root mean squared error                  0.2928
Relative absolute error                 34.3391 %
Root relative squared error             58.6005 %
Total Number of Instances               83

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.791    0.000    1.000      0.791    0.883      0.803    0.911     0.908     AD
                 1.000    0.209    0.816      1.000    0.899      0.803    0.911     0.844     NON_AD
Weighted Avg.    0.892    0.101    0.911      0.892    0.891      0.803    0.911     0.877

=== Confusion Matrix ===

  a  b   <-- classified as
 34  9 |  a = AD
  0 40 |  b = NON_AD
```

Test Set AD Results:



```
Time taken to test model on supplied test set: 0.15 seconds

=== Summary ===

Correctly Classified Instances          82               89.1304 %
Incorrectly Classified Instances        10               10.8696 %
Kappa statistic                          0.7784
Mean absolute error                      0.202
Root mean squared error                  0.3265
Relative absolute error                 40.2707 %
Root relative squared error             65.0694 %
Total Number of Instances               92

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.810    0.040    0.944      0.810    0.872      0.785    0.864     0.850     AD
                 0.960    0.190    0.857      0.960    0.906      0.785    0.864     0.834     NON_AD
Weighted Avg.    0.891    0.122    0.897      0.891    0.890      0.785    0.864     0.842

=== Confusion Matrix ===

  a  b   <-- classified as
 34  8 |  a = AD
  2 48 |  b = NON_AD
```

Test Set MCI Results:



```
Time taken to test model on supplied test set: 0.07 seconds

=== Summary ===

Correctly Classified Instances          27               57.4468 %
Incorrectly Classified Instances        20               42.5532 %
Kappa statistic                          0.136
Mean absolute error                      0.4478
Root mean squared error                  0.5998
Relative absolute error                 89.3583 %
Root relative squared error            119.618  %
Total Number of Instances               47

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.455    0.320    0.556      0.455    0.500      0.138    0.558     0.506     AD
                 0.680    0.545    0.586      0.680    0.630      0.138    0.558     0.562     NON_AD
Weighted Avg.    0.574    0.440    0.572      0.574    0.569      0.138    0.558     0.536

=== Confusion Matrix ===

  a  b   <-- classified as
 10 12 |  a = AD
  8 17 |  b = NON_AD
```

	Training	Test AD	Test MCI
Sensitivity	0.791	0.81	0.455
Specificity	1.0	0.96	0.68
Accuracy	89.16%	89.13%	57.45%
F1-Score	0.891	0.89	0.569
MCC	0.803	0.785	0.138
Youden’s J	0.791	0.77	0.135

The classifier algorithm showed promise based on its performance with the first Test set with strong scores all around, specifically in its specificity, however the second test set performed very poorly using the same classifier system, with its accuracy only being slightly better then a coin flip. These results could indicate a flaw in the classifier system however the MCI test set also has the lowest sample size and could have been a statistical outlier.