# Sonar Rock or Mine Prediction

Sadia Boksh

30/01/2021

## Introduction

In this project we will use Sonar data from mlbench package. This is the data set used by Gorman and Sejnowski in their study of the classification of sonar signals using a neural network. Using this data set, we will train few classification machine learning models to classify sonar signals those bounced off a metal cylinder and those bounced off a roughly cylindrical rock. We will perform the model fitting on scaled data. We will choose the best performing model by analyzing their accuracy.

## Analysis

### Data Exploration

Sonar data set is a data frame of 208 rows and 60 feature variables and a response variable.

```
head(Sonar)
```

```
##       V1     V2     V3     V4     V5     V6     V7     V8     V9    V10    V11
## 1 0.0200 0.0371 0.0428 0.0207 0.0954 0.0986 0.1539 0.1601 0.3109 0.2111 0.1609
## 2 0.0453 0.0523 0.0843 0.0689 0.1183 0.2583 0.2156 0.3481 0.3337 0.2872 0.4918
## 3 0.0262 0.0582 0.1099 0.1083 0.0974 0.2280 0.2431 0.3771 0.5598 0.6194 0.6333
## 4 0.0100 0.0171 0.0623 0.0205 0.0205 0.0368 0.1098 0.1276 0.0598 0.1264 0.0881
## 5 0.0762 0.0666 0.0481 0.0394 0.0590 0.0649 0.1209 0.2467 0.3564 0.4459 0.4152
## 6 0.0286 0.0453 0.0277 0.0174 0.0384 0.0990 0.1201 0.1833 0.2105 0.3039 0.2988
##      V12    V13    V14    V15    V16    V17    V18    V19    V20    V21    V22
## 1 0.1582 0.2238 0.0645 0.0660 0.2273 0.3100 0.2999 0.5078 0.4797 0.5783 0.5071
## 2 0.6552 0.6919 0.7797 0.7464 0.9444 1.0000 0.8874 0.8024 0.7818 0.5212 0.4052
## 3 0.7060 0.5544 0.5320 0.6479 0.6931 0.6759 0.7551 0.8929 0.8619 0.7974 0.6737
## 4 0.1992 0.0184 0.2261 0.1729 0.2131 0.0693 0.2281 0.4060 0.3973 0.2741 0.3690
## 5 0.3952 0.4256 0.4135 0.4528 0.5326 0.7306 0.6193 0.2032 0.4636 0.4148 0.4292
## 6 0.4250 0.6343 0.8198 1.0000 0.9988 0.9508 0.9025 0.7234 0.5122 0.2074 0.3985
##      V23    V24    V25    V26    V27    V28    V29    V30    V31    V32    V33
## 1 0.4328 0.5550 0.6711 0.6415 0.7104 0.8080 0.6791 0.3857 0.1307 0.2604 0.5121
## 2 0.3957 0.3914 0.3250 0.3200 0.3271 0.2767 0.4423 0.2028 0.3788 0.2947 0.1984
## 3 0.4293 0.3648 0.5331 0.2413 0.5070 0.8533 0.6036 0.8514 0.8512 0.5045 0.1862
## 4 0.5556 0.4846 0.3140 0.5334 0.5256 0.2520 0.2090 0.3559 0.6260 0.7340 0.6120
## 5 0.5730 0.5399 0.3161 0.2285 0.6995 1.0000 0.7262 0.4724 0.5103 0.5459 0.2881
## 6 0.5890 0.2872 0.2043 0.5782 0.5389 0.3750 0.3411 0.5067 0.5580 0.4778 0.3299
##      V34    V35    V36    V37    V38    V39    V40    V41    V42    V43    V44
## 1 0.7547 0.8537 0.8507 0.6692 0.6097 0.4943 0.2744 0.0510 0.2834 0.2825 0.4256
```

```
## 2 0.2341 0.1306 0.4182 0.3835 0.1057 0.1840 0.1970 0.1674 0.0583 0.1401 0.1628
## 3 0.2709 0.4232 0.3043 0.6116 0.6756 0.5375 0.4719 0.4647 0.2587 0.2129 0.2222
## 4 0.3497 0.3953 0.3012 0.5408 0.8814 0.9857 0.9167 0.6121 0.5006 0.3210 0.3202
## 5 0.0981 0.1951 0.4181 0.4604 0.3217 0.2828 0.2430 0.1979 0.2444 0.1847 0.0841
## 6 0.2198 0.1407 0.2856 0.3807 0.4158 0.4054 0.3296 0.2707 0.2650 0.0723 0.1238
##      V45    V46    V47    V48    V49    V50    V51    V52    V53    V54    V55
## 1 0.2641 0.1386 0.1051 0.1343 0.0383 0.0324 0.0232 0.0027 0.0065 0.0159 0.0072
## 2 0.0621 0.0203 0.0530 0.0742 0.0409 0.0061 0.0125 0.0084 0.0089 0.0048 0.0094
## 3 0.2111 0.0176 0.1348 0.0744 0.0130 0.0106 0.0033 0.0232 0.0166 0.0095 0.0180
## 4 0.4295 0.3654 0.2655 0.1576 0.0681 0.0294 0.0241 0.0121 0.0036 0.0150 0.0085
## 5 0.0692 0.0528 0.0357 0.0085 0.0230 0.0046 0.0156 0.0031 0.0054 0.0105 0.0110
## 6 0.1192 0.1089 0.0623 0.0494 0.0264 0.0081 0.0104 0.0045 0.0014 0.0038 0.0013
##      V56    V57    V58    V59    V60 Class
## 1 0.0167 0.0180 0.0084 0.0090 0.0032     R
## 2 0.0191 0.0140 0.0049 0.0052 0.0044     R
## 3 0.0244 0.0316 0.0164 0.0095 0.0078     R
## 4 0.0073 0.0050 0.0044 0.0040 0.0117     R
## 5 0.0015 0.0072 0.0048 0.0107 0.0094     R
## 6 0.0089 0.0057 0.0027 0.0051 0.0062     R
```

### Principal Component Analysis

As there are 60 predictors in this data set, principal component analysis seems appropriate for this data set.

```
dim(Sonar)
```

```
## [1] 208  61
```

This data set has two categories in response variable. The signals are to be classified into these two categories:

```
unique(Sonar$Class)
```

```
## [1] R M
## Levels: M R
```

This data set does not have any missing values.

```
any(is.na(Sonar))
```

```
## [1] FALSE
```

Approximately 47% of the sample are Rocks and 53% are Mines.

```
Sonar %>% group_by(Class) %>%
    summarise(n=n()) %>%
    mutate( prop=n/sum(n))
```

```
## # A tibble: 2 x 3
##   Class     n  prop
##   <fct> <int> <dbl>
## 1 M       111 0.534
## 2 R        97 0.466
```

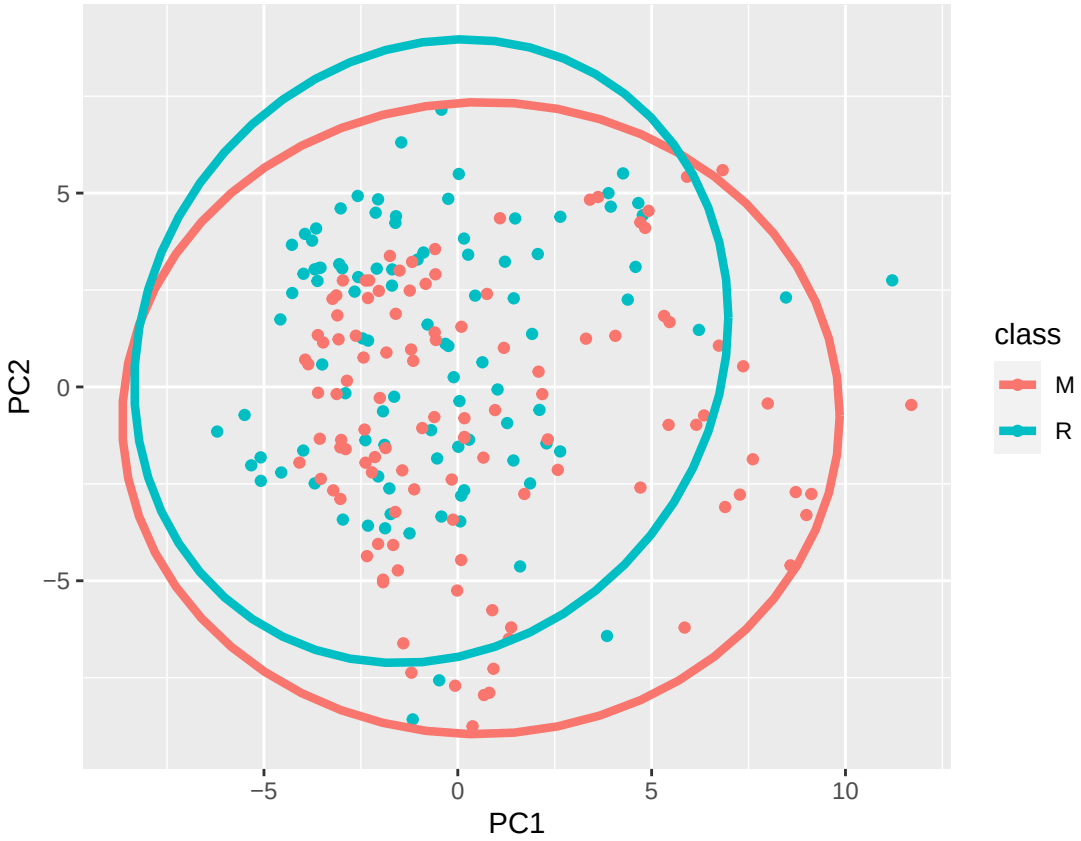
In order to do principal component analysis, first we need to scale the matrix. After Scaling, the column mean for the first column is: 1.025822e-17 and standard deviation: 1.

Then we perform the principal component analysis on the scaled matrix. In order to explain 95% of variance we need 30 PCs. Below is the summary of variable importance:

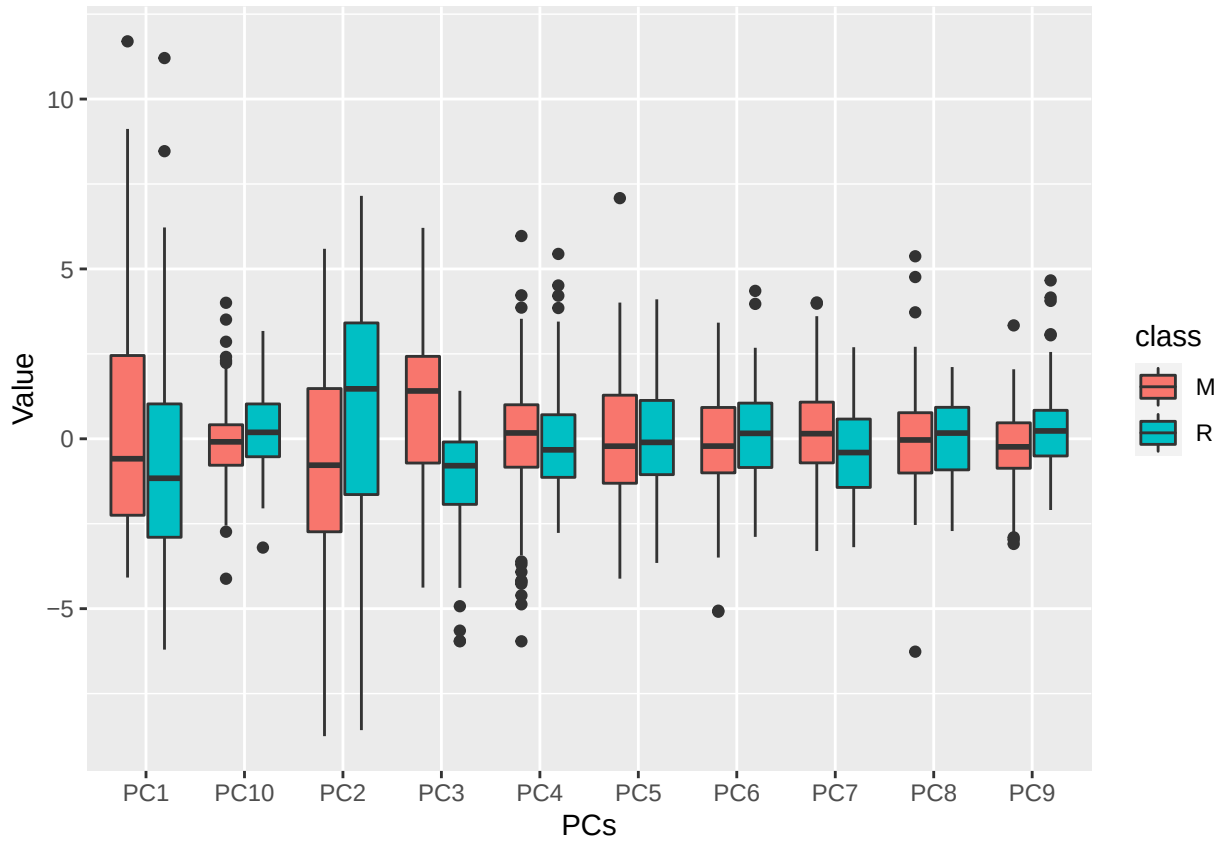
```
##                             PC1      PC2      PC3      PC4      PC5      PC6
## Standard deviation     3.493985 3.367244 2.264949 1.845945 1.733277 1.561729
## Proportion of Variance 0.203470 0.188970 0.085500 0.056790 0.050070 0.040650
## Cumulative Proportion  0.203470 0.392440 0.477940 0.534730 0.584800 0.625450
##                             PC7      PC8      PC9     PC10     PC11     PC12
## Standard deviation     1.402639 1.351991 1.240797 1.222563 1.115871 1.068267
## Proportion of Variance 0.032790 0.030460 0.025660 0.024910 0.020750 0.019020
## Cumulative Proportion  0.658240 0.688700 0.714360 0.739280 0.760030 0.779050
##                            PC13     PC14      PC15      PC16     PC17      PC18
## Standard deviation     1.02381 0.960776 0.9255691 0.9036455 0.860677 0.8373695
## Proportion of Variance 0.01747 0.015380 0.0142800 0.0136100 0.012350 0.0116900
## Cumulative Proportion  0.79652 0.811900 0.8261800 0.8397900 0.852140 0.8638200
##                             PC19     PC20      PC21      PC22      PC23
## Standard deviation     0.7864253 0.766421 0.7526271 0.7297509 0.7098764
## Proportion of Variance 0.0103100 0.009790 0.0094400 0.0088800 0.0084000
## Cumulative Proportion  0.8741300 0.883920 0.8933600 0.9022400 0.9106400
##                             PC24      PC25      PC26      PC27      PC28
## Standard deviation     0.6800756 0.6581472 0.6463919 0.6076369 0.5647656
## Proportion of Variance 0.0077100 0.0072200 0.0069600 0.0061500 0.0053200
## Cumulative Proportion  0.9183400 0.9255600 0.9325300 0.9386800 0.9440000
##                             PC29      PC30
## Standard deviation     0.5610961 0.5451957
## Proportion of Variance 0.0052500 0.0049500
## Cumulative Proportion  0.9492400 0.9542000
```

### Plots

Below is the plot for first 2 PCs to see how they explain the variance. Although PC1 and PC2 are not enough to distinguish between Mine and Rock but from the plot we can see Mines have higher PC1 value and Rocks have higher PC2 values.




Also plot for first 10 PCs:




In the plot above, we can see PCs are overlapping, but we can say PC1 ad PC2 explains most of the variability.

## Modeling

Now we will fit logistic, LDA, KNN and Random forest models to the scaled data set.

First we will split the scaled data set to 80% train set and 20% test set. 80/20 split has been made to be able to train the model with as much data as possible at the same time having a decent amount data for testing.

### Logistic Regression

Logistic regression is used to predict the categorical dependent variable with the help of independent variables. The output of Logistic Regression problem can be only between the 0 and 1. This is our first model to be used to classify the sonar signals.

```
train_glm <- train(train_x, train_y,method = "glm")
pred_glm <-  predict(train_glm,  test_x)
acc_glm <- confusionMatrix(pred_glm,test_y)$overall['Accuracy']
acc_glm
```

```
##  Accuracy
## 0.7674419
```

### LDA

LDA makes predictions by estimating the probability that a new set of inputs belongs to each class. The class that gets the highest probability is the output class and a prediction is made. LDA model is used in this data set as it can handle continuous independent variable and a categorical dependent variable.

```
train_lda <- train(train_x, train_y, method = "lda")
pred_lda <-  predict(train_lda,  test_x)
acc_lda <- confusionMatrix(pred_lda,test_y)$overall['Accuracy']
acc_lda
```

```
##  Accuracy
## 0.7906977
```

### KNN

KNN algorithm can be used for classification where input consists of the k closest training examples in data set and output is a class membership. An object is classified by the majority vote it gets by its neighbors. The object is assigned to the class most common among its k nearest neighbors. KNN algorithm is a good choice for this data set as the distance between signals from two rocks will be closer than distance between signals from a rock and a mine.

For KNN, I am using tuning parameter k from 3 to 21 and the default cross validation is performed by taking 25 bootstrap samples comprised of 25% of the observations

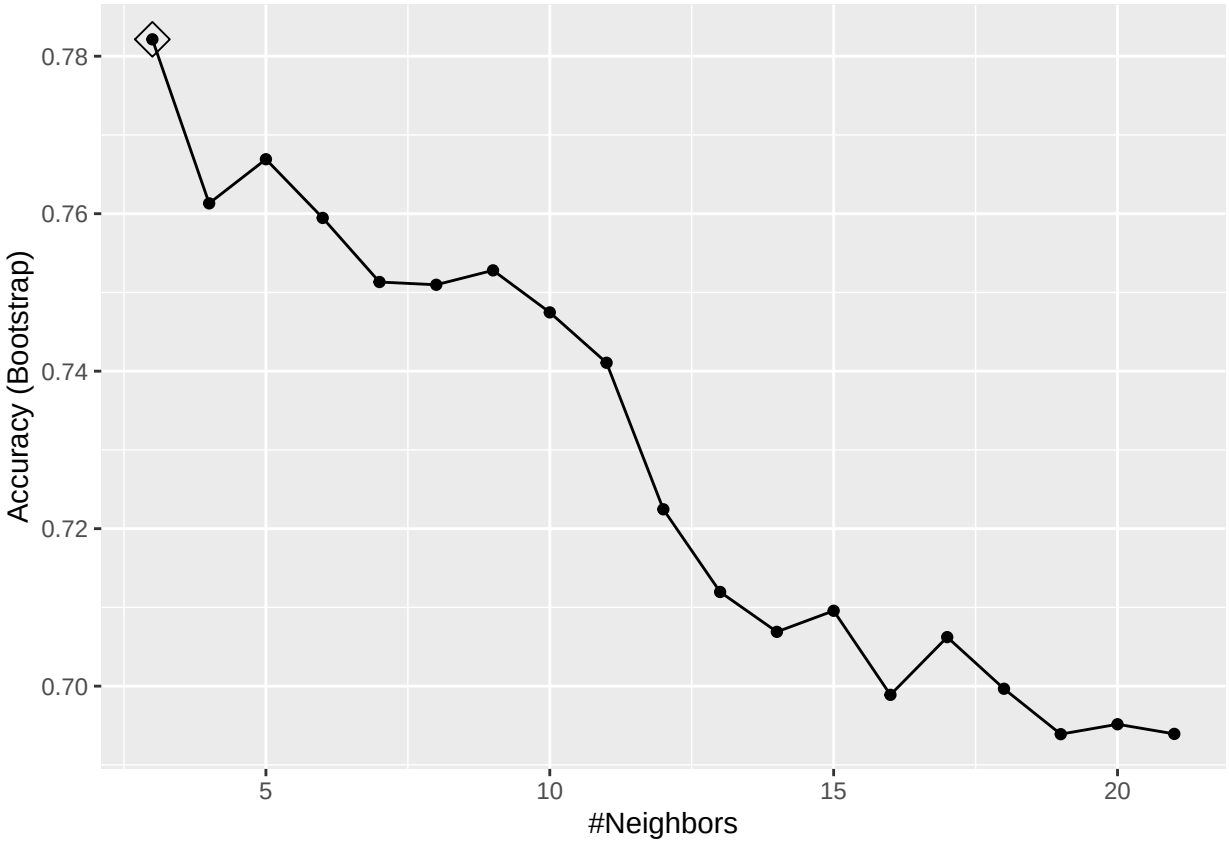
```
train_knn <- train(train_x, train_y, method = "knn", tuneGrid = data.frame(k=c(3:21)))
pred_knn <-  predict(train_knn,  test_x)
acc_knn <- confusionMatrix(pred_knn,test_y)$overall['Accuracy']

acc_knn
```

```
##  Accuracy
## 0.9069767
```

```
train_knn$bestTune
```

```
##   k
## 1 3
```

```
ggplot(train_knn, highlight = TRUE)
```




**Random Forest**

Random forest consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes the prediction of our model. Random forest allows each individual tree to randomly sample from the data set with replacement, resulting in different trees. This process is known as bagging. For this reason Random forest is also a good choice to make our predictions in this data set.

For Random forest, tune grid parameter is mtry (number of variables randomly sampled as candidates at each split) with values from 3 to 13. The best accuracy is achieved at mtry = 3 and most important variable is V12.

```
train_rf <- train(train_x, train_y, method = "rf",
                  tuneGrid = data.frame(mtry=c(3, 5, 7, 9, 11, 13)), importance=TRUE)
pred_rf <-  predict(train_rf,  test_x)
acc_rf <- confusionMatrix(pred_rf,test_y)$overall['Accuracy']

train_rf$bestTune
```

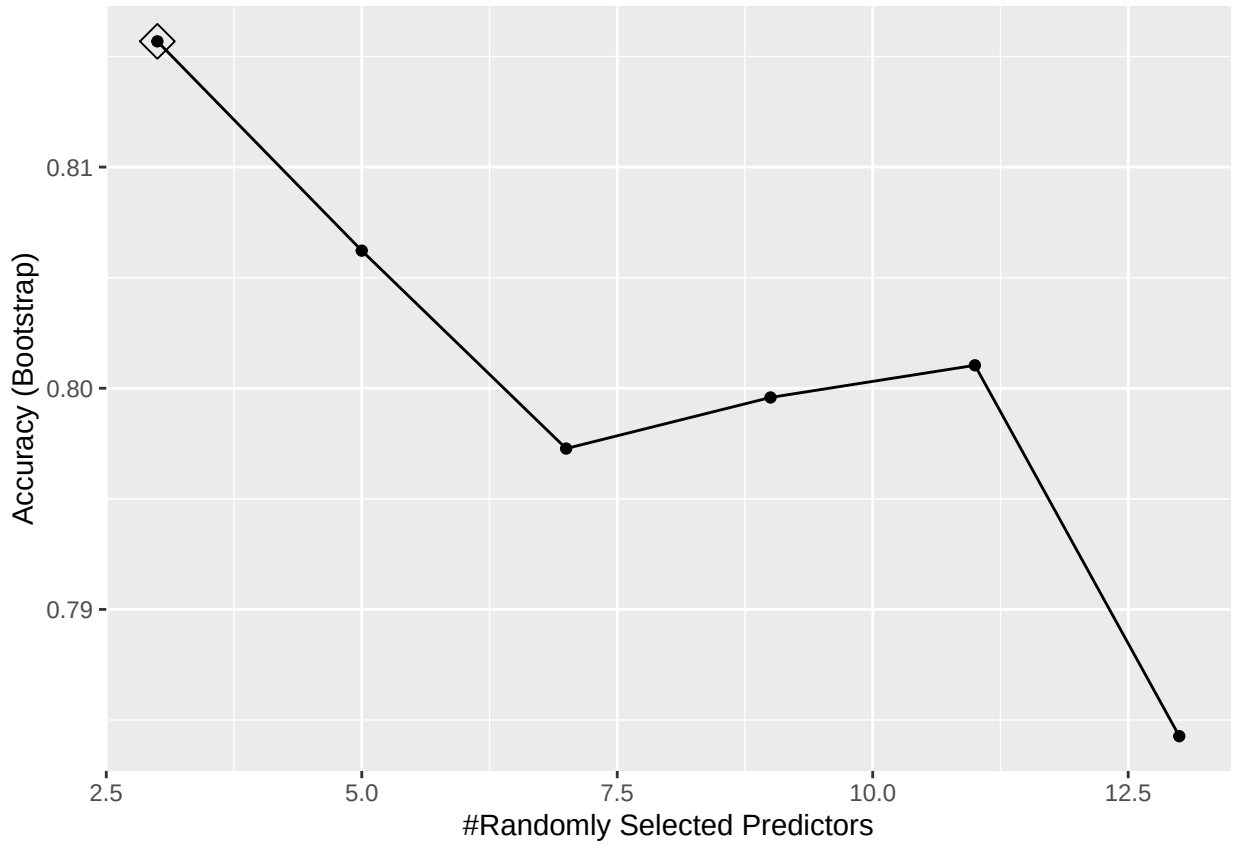
```
##   mtry
## 1    3
```

```
varImp(train_rf)
```

```
## rf variable importance
```

```
##
##   only 20 most important variables shown (out of 60)
##
##     Importance
## V12     100.00
## V11      90.72
## V9       74.97
## V10      68.84
## V37      66.43
## V36      64.45
## V49      63.10
## V48      61.86
## V45      61.26
## V28      57.73
## V27      51.71
## V23      49.80
## V43      49.51
## V52      49.50
## V16      49.47
## V46      45.99
## V20      44.67
## V47      43.60
## V21      43.48
## V33      42.82
```

```
ggplot(train_rf, highlight = TRUE)
```



**Ensemble**

We also combined all above model predictions to create an ensemble and calculate the accuracy of our prediction.

```
#ensemble
c_preds<- cbind(glm = pred_glm, lda =pred_lda, knn = pred_knn, rf = pred_rf)

ensem <- ifelse(rowMeans(c_preds == 1) > 0.5, 'M', 'R')

acc_ens <- mean(ensem == test_y)

acc_ens
```

```
## [1] 0.8372093
```

# Results

Now we can compare the results of different models and their accuracy.

Below is the accuracy table summary:

```
##          Method  Accuracy
## 1      Logistic 0.7674419
## 2           LDA 0.7906977
## 3           KNN 0.9069767
## 4 Random Forest 0.8139535
## 5      Ensemble 0.8372093
```

KNN has the highest accuracy in this analysis with k value 3, so it is the preferred model for this data set.

# Conclusion

In summary, this analysis shows it is possible to classify the sonar signals those bounce off a metal cylinder and those bounce off a roughly cylindrical rock. KNN is the highest performing model with accuracy around 90%. Future work can be done to improve the accuracy above 90%.