Ionoshere

Georgios Papadopoulos Vasileios Papadopoulos

Load Dataset

```r
df <- read.arff("ionoshere.arff")
head(df)
```

```
##   a01 a02     a03      a04      a05      a06      a07      a08     a09      a10
## 1   1   0 0.99539 -0.05889  0.85243  0.02306  0.83398 -0.37708 1.00000  0.03760
## 2   1   0 1.00000 -0.18829  0.93035 -0.36156 -0.10868 -0.93597 1.00000 -0.04549
## 3   1   0 1.00000 -0.03365  1.00000  0.00485  1.00000 -0.12062 0.88965  0.01198
## 4   1   0 1.00000 -0.45161  1.00000  1.00000  0.71216 -1.00000 0.00000  0.00000
## 5   1   0 1.00000 -0.02401  0.94140  0.06531  0.92106 -0.23255 0.77152 -0.16399
## 6   1   0 0.02337 -0.00592 -0.09924 -0.11949 -0.00763 -0.11824 0.14706  0.06637
##       a11      a12     a13      a14      a15      a16      a17      a18
## 1 0.85243 -0.17755 0.59755 -0.44945  0.60536 -0.38223  0.84356 -0.38542
## 2 0.50874 -0.67743 0.34432 -0.69707 -0.51685 -0.97515  0.05499 -0.62237
## 3 0.73082  0.05346 0.85443  0.00827  0.54591  0.00299  0.83775 -0.13644
## 4 0.00000  0.00000 0.00000  0.00000 -1.00000  0.14516  0.54094 -0.39330
## 5 0.52798 -0.20275 0.56409 -0.00712  0.34395 -0.27457  0.52940 -0.21780
## 6 0.03786 -0.06302 0.00000  0.00000 -0.04572 -0.15540 -0.00343 -0.10196
##        a19      a20      a21      a22      a23      a24      a25      a26
## 1  0.58212 -0.32192  0.56971 -0.29674  0.36946 -0.47357  0.56811 -0.51171
## 2  0.33109 -1.00000 -0.13151 -0.45300 -0.18056 -0.35734 -0.20332 -0.26569
## 3  0.75535 -0.08540  0.70887 -0.27502  0.43385 -0.12062  0.57528 -0.40220
## 4 -1.00000 -0.54467 -0.69975  1.00000  0.00000  0.00000  1.00000  0.90695
## 5  0.45107 -0.17813  0.05982 -0.35575  0.02309 -0.52879  0.03286 -0.65158
## 6 -0.11575 -0.05414  0.01838  0.03669  0.01519  0.00888  0.03513 -0.01535
##        a27      a28      a29      a30      a31      a32      a33      a34 class
## 1  0.41078 -0.46168  0.21266 -0.34090  0.42267 -0.54487  0.18641 -0.45300     g
## 2 -0.20468 -0.18401 -0.19040 -0.11593 -0.16626 -0.06288 -0.13738 -0.02447     b
## 3  0.58984 -0.22145  0.43100 -0.17365  0.60436 -0.24180  0.56045 -0.38238     g
## 4  0.51613  1.00000  1.00000 -0.20099  0.25682  1.00000 -0.32382  1.00000     b
## 5  0.13290 -0.53206  0.02431 -0.62197 -0.05707 -0.59573 -0.04608 -0.65697     g
## 6 -0.03240  0.09223 -0.07859  0.00732  0.00000  0.00000 -0.00039  0.12011     b
```

Methodology

- Define performance metrics
- Cross Validation
- Binomial Logistic Regression
- PCA

Cross Validation

The validation set approach consists of randomly splitting the data into two sets: one set is used to train the model and the remaining other set sis used to test the model. Steps:

1. Train a model on the training data set
2. Apply the model to the test data set to predict the outcome of new unseen observations
3. Quantify the prediction error, define performance metric

Performance metric

For our analysis the following performance metrics will be used.

- Accuracy
- Recall
- Precision
- F1 - Score

Exploratory data analysis

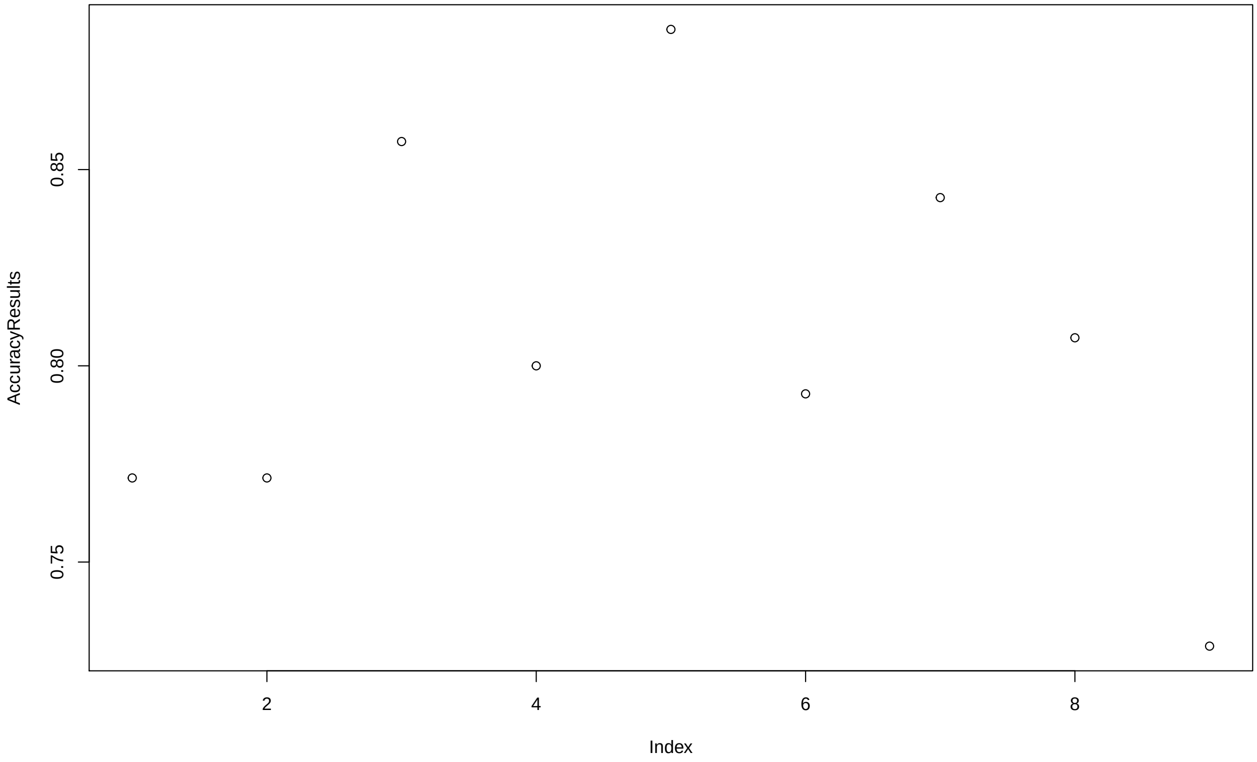
We define the helper function *logistic_regression*. The function splits the data into 60% train and 40% test sets.

```
logistic_regression <- function(data, var_range) {
  set.seed(1223)
  df_ibk <- data[var_range]
  df_ibk$class <- df_ibk$class == 'g'
  split=0.60
  trainIndex <- createDataPartition(df_ibk$class, p=split, list=FALSE)
  data_train <- df_ibk[ trainIndex,]
  data_test <- df_ibk[-trainIndex,]
  # train a binomial logistic regression model
  model <- glm(class~., data=data_train, binomial)
  #print(dim(data_train))
  # make predictions
  data_test$model_prob <- predict(model, data_test, type="response")
  data_test$pred_class <- data_test$model_prob > 0.5
  # summarize results
  res <- confusionMatrix(as.factor(data_test$pred_class), as.factor(data_test$class))
  accuracy <- res$overall['Accuracy']
  #precision <- posPredValue(as.factor(data_test$pred_class), as.factor(data_test$class), positive=TRUE
  #recall <- sensitivity(as.factor(data_test$pred_class), as.factor(data_test$class), positive=TRUE)
  #F1 <- (2 * precision * recall) / (precision + recall)
  #obj <- data.frame(acc = accuracy, f1 = F1)

  return (accuracy)
}

AccuracyResults <- c()
i = 1
ib_tests <- c(1,2,3,4,5,6,7,8,9)
for (val in ib_tests) {
  r <- logistic_regression(df, c(35, 1,val))
  AccuracyResults <- c(AccuracyResults,r)
  i=i+1
}

plot(AccuracyResults)
```



Checking the accuracy plots we see that *IBk* $=$ *5* achieves the higher accuracy level *0.88* on test set.

```
best_ibk <- which.max(AccuracyResults)
best_accu <- max(AccuracyResults)
```