```r
# General flow of Machine Learning programs

# 1. Import library

# 2. dataSet = Read dataset

# 3. Split dataSet into training and testing sets, 2/3 and 1/3 respectively

# 4. Obtain machine learning model by making function call with arguments
#          of training data and other parameters values for algorithm
#    model = functionCall ( labelInTrainingSet ~ featureNamesInTrainingSetSeparatedByPlus,
#                                  otherParameterValues )

# 5. Check prediction of model by passing model and test set to predict or compute
#    prediction = predictOrCompute( model , testSet )
#                         where testSet = testSet[ - label ] , that is testSet without label column

# 6. Build confusion matrix, cm
#    cm = table( actualTestLabel , prediction )
#                where actualTestLabel = testSet[ label ] , just the label column of testSet

#    cm can also be by बहुमत या गठबंधन
#                         ಬಹುಮತ ಅಥವಾ ಸಮಿಶ್ರ

# 7. Display efficiency
#    sum of diagonal elements of cm  /  sum of all elements of cm


# 1. Import library
library(neuralnet)  # neural network NN , credit risk
library(naivebayes) # Naïve-Bayes
library(kknn)       # k nearest neighbour KNN
library(e1071)      # Support Vector Machine SVM

# 2. dataSet = Read dataset, pre process
dataSet <- read.csv("dataSetName.csv")     #  NN - creditset.csv
dataSet <- read.csv("bc_data.csv")         #  Bayes - bc_data.csv

attach(iris)                                 #  KNN and SVM, save iris dataset as internal variable
dataSet <- iris[ sample(1:150, 150) , ]      #  reorder data

# Important: Check input size

# 3. Split dataSet into training and testing sets, 2/3 and 1/3 respectively, Check and assign inputSize
inputSize = 150
```

```r
trainingSet <- dataSet[ 1 : ( inputSize * 2/3 ) , ]
testSet <- dataSet[ ( inputSize * 2/3 + 1 ) : inputSize , ]

# 4. Obtain machine learning model by making function call with arguments
#          of training data and other parameters values for algorithm
#    model = functionCall ( labelInTrainingSet ~ featureNamesInTrainingSetSeparatedByPlus,
#                                otherParameterValues )

#    featureNamesInTrainingSet =  use + to separate feature names if using
#                                 only few features out of many else use dot . to use all

# Remember unique parameters of each algorithm, like hidden layers for NN, k for KNN

model <- neuralnet ( default10yr ~ LTI + age, trainingSet, hidden = 4)                # NN

model <- naive_bayes ( diagnosis ~ otherFeature1 + listAllOtherFeatureSeparatedByPlus , data=trainingSet ) # Bayes
                                     # diagnosis ~ radius_mean + texture_mean + perimeter_mean + area_mean
                                     # total 30 features separated by + other than id and diagnosis

model <- train.kknn ( Species ~ . , data = trainingSet , kmax = 12 )                  # KNN

model <- svm ( Species ~ . , data=iris )                                              # SVM


# 5. Check prediction of model by passing model and test set to predict or compute
#    prediction = computeOrPredict ( model , testSet )
#                        where testSet = testSet[ - label ] , that is testSet without label column

prediction <- compute ( model , subset ( testSet , select = c("LTI", "age") ) )              # NN

# Extract the diagonosis column of testSet before setting it to NULL, save it in actualDiagnosis
actualDiagnosis   = testSet$diagnosis                                                        # Bayes
testSet$diagnosis = NULL

prediction <- predict ( model , as.data.frame( testSet ) )                                   # Bayes

#                              column 5 is the Species, hence [ , -5 ] : without label column
prediction <- predict ( model , testSet [ , -5 ] )                                           # KNN

prediction <- predict ( model , subset ( iris ,  select = -Species ) )                       # SVM

# 6. Build confusion matrix
#    cm = table( actualLabelInTestSet , prediction )
#         where actualLabelInTestSet = testSet[ label ] , just the label column of testSet

cm <- table ( testSet$default10yr , round( prediction$net.result ) )             # NN
```

```
#              testSet$diagnosis was saved in actualDiagnosis
cm <- table ( actualDiagnosis , prediction )                                    # Bayes

cm <- table ( testSet[, 5] , prediction )                                       # KNN

cm <- table ( Species , prediction )                                            # SVM

# 7. Display efficiency
#    sum of diagonal elements of cm /  sum of all elements of cm
accuracy <- ( sum ( diag ( cm ) ) ) / sum ( cm )
accuracy
```