#Some libraries

library(car)

library(caret)

library(class)

library(devtools)

library(e1071)

library(ggord)

library(ggplot2)

library(Hmisc)

library(klaR)

library(klaR)

library(MASS)

library(nnet)

library(plyr)

library(pROC)

library(psych)

library(scatterplot3d)

library(SDMTools)

library(dplyr)

library(ElemStatLearn)

library(rpart)

library(rpart.plot)

library(randomForest)

library(neuralnet)

setwd( "/Users/bappa/Desktop/My.Folder/Machine Learning")

normalize<-function(x){

+return((x-min(x))/(max(x)-min(x)))}

#####################

data(iris)

iris

qplot(Sepal.Length, Petal.Length,color=Species, data=iris)

iris.data<-iris[,c(1,3,5)]

iris.data$Sepal.Length<-normalize(iris.data$Sepal.Length)

iris.data$Petal.Length<-normalize(iris.data$Petal.Length)

iris.data$Spec<-ifelse(iris.data$Species=="virginica","virginica","others")

iris.data$Spec.1<-ifelse(iris.data$Species=="virginica",1,0)

str(iris.data)

iris.data$Spec<-as.factor(iris.data$Spec)

#iris.data<-iris.data[,c(1,2,4,5)]

set.seed(1234)

pd<-sample(2,nrow(iris.data),replace=TRUE, prob=c(0.7,0.3))

train.iris<-iris.data[pd==1,]

val.iris<-iris.data[pd==2,]

####

#NAIVE BAYES

#Set the data frame

train.iris.NB<-train.iris[,c(1,2,4)]

val.iris.NB<-val.iris[,c(1,2,4)]

#

NB.iris<-naiveBayes(x=train.iris.NB[-3], y=train.iris$Spec)

#pedict

y\_pred.NB<-predict(NB.iris,newdata=val.iris.NB[-3])

y\_pred.NB

#Confusion matrix

cm.iris.NB=table(val.iris.NB[,3],y\_pred.NB)

cm.iris.NB

# Visualising the Test set results

library(ElemStatLearn)

set = val.iris.NB

X1 = seq(min(set[, 1])-1 , max(set[, 1]) +1, by = 0.01)

X2 = seq(min(set[, 2])-1 , max(set[, 2])+1 , by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Sepal.Length', 'Petal.Length')

y\_grid = predict(NB.iris, newdata = grid\_set)

plot(set[, -3],

main = 'Naive Bayes: Test.Iris ',

xlab = 'Sepal.Length', ylab = 'Petal.Length',

xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == "virginica", 'blue', 'red'))

points(set, pch = 21, bg = ifelse(set[, 3] == "virginica", 'black', 'green'))

#NB Multi Class

train.iris.NB.multi<-train.iris[,c(1,2,3)]

val.iris.NB.multi<-val.iris[,c(1,2,3)]

NB.iris.multi<-naiveBayes(x=train.iris.NB.multi[-3], y=train.iris.NB.multi$Species)

#pedict

y\_pred.NB.multi<-predict(NB.iris.multi,newdata=val.iris.NB.multi[-3])

y\_pred.NB.multi

#Confusion matrix

cm.iris.NB.multi=table(val.iris.NB.multi[,3],y\_pred.NB.multi)

cm.iris.NB.multi

# Visualising the Test set results

library(ElemStatLearn)

set = val.iris.NB.multi

X1 = seq(min(set[, 1]) -0.2, max(set[, 1]) +0.2, by = 0.0051)

X2 = seq(min(set[, 2])-0.2 , max(set[, 2]) +0.2, by = 0.0051)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Sepal.Length', 'Petal.Length')

y\_grid = predict(NB.iris.multi, newdata = grid\_set)

plot(set[, -3],

main = 'Naive Bayes: Test.Iris ',

xlab = 'Sepal.Length', ylab = 'Petal.Length',

xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == "virginica", 'blue', ifelse(y\_grid=="setosa",'red','tomato')))

points(set, pch = 21, bg = ifelse(set[, 3] == "virginica",'black' ,ifelse(set[,3]=="setosa", 'green','blue')))

####KNN

#Set the data frame

train.iris.knn<-train.iris[,c(1,2,4)]

val.iris.knn<-val.iris[,c(1,2,4)]

y\_pred.KNN<-knn(train.iris.knn[,-3],val.iris.knn[-3], cl=train.iris.knn[,3],k=3)

cm.knn<-table(val.iris.knn[,3],y\_pred.KNN)

cm.knn

#Visualizing Training set

set = val.iris.knn

X1 = seq(min(set[, 1])-1 , max(set[, 1]) +1, by = 0.01)

X2 = seq(min(set[, 2])-1 , max(set[, 2])+1 , by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Sepal.Length', 'Petal.Length')

y\_grid = knn(train=train.iris.knn[,-3],test=val.iris.knn[-3], cl=train.iris.knn[,3],k=3)

plot(set[, -3],

main = 'KNN: Test.Iris ',

xlab = 'Sepal.Length', ylab = 'Petal.Length',

xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == "virginica", 'blue', 'red'))

points(set, pch = 21, bg = ifelse(set[, 3] == "virginica", 'black', 'green'))

#######

#LDA

#DT

train.iris.LDA<-train.iris[,c(1,2,5)]

val.iris.LDA<-val.iris[,c(1,2,5)]

train.iris.LDA$Spec.1<-as.factor(train.iris.LDA$Spec.1)

val.iris.LDA$Spec.1<-as.factor(val.iris.LDA$Spec.1)

LDA<-Spec.1~Sepal.Length+Petal.Length

#

LDA.iris<-lda(LDA,data=train.iris.LDA)

LDA.iris

summary(LDA.iris)

#pedict

y\_pred.LDA<-predict(LDA.iris,newdata=val.iris.LDA[-3])

y\_pred.LDA<-y\_pred.LDA$class

#y\_pred.DT<-ifelse(y\_pred.DT>0.5,1,0)

#Confusion matrix

cm.iris.LDA=table(val.iris.LDA[,3],y\_pred.LDA)

cm.iris.LDA

# Visualising the Test set results

set = val.iris.LDA

X1 = seq(min(set[, 1])-1 , max(set[, 1]) +1, by = 0.01)

X2 = seq(min(set[, 2])-1 , max(set[, 2])+1 , by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Sepal.Length', 'Petal.Length')

y\_grid = predict(y\_pred.LDA$class, newdata=grid\_set)

plot(set[, -3],

main = 'LDA(Iris) ',

xlab = 'Sepal.Length', ylab = 'Petal.Length',

xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'blue', 'red'))

points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'black', 'green'))

#######

#SVM

train.iris.SVM<-train.iris[,c(1,2,5)]

val.iris.SVM<-val.iris[,c(1,2,5)]

str(val.iris.SVM)

train.iris.SVM$Spec.1<-as.factor(train.iris.SVM$Spec.1)

val.iris.SVM$Spec.1<-as.factor(val.iris.SVM$Spec.1)

SVM<-Spec.1~Sepal.Length+Petal.Length

SVM.iris<-svm(SVM, data=train.iris.SVM, type='C-classification', kernel = 'linear')

SVM.iris

#pedict

y\_pred.SVM<-predict(SVM.iris,newdata=val.iris.SVM)

y\_pred.SVM

#Confusion matrix

cm.iris.SVM=table(val.iris.SVM$Spec.1 ,y\_pred.SVM)

cm.iris.SVM

# Visualising the test set results

set = val.iris.SVM

X1 = seq(min(val.iris$Sepal.Length) - 1, max(val.iris$Sepal.Length) + 1, by = 0.01)

X2 = seq(min(val.iris$Petal.Length) - 1, max(set[val.iris$Petal.Length]) + 1, by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Sepal.Length', 'Petal.Length')

y\_grid = predict(SVM.iris, newdata = grid\_set)

plot(val.iris$Spec.1,

main = 'SVM (Test )',

xlab = 'Sepal.Length', ylab = 'Petal.Length',

xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'blue', 'red'))

points(set, pch = 21, bg = ifelse(val.iris$Spec.1 == 1, 'black', 'green'))

##########

#SVM Multi

train.iris.SVM.multi<-train.iris[,c(1,2,3)]

val.iris.SVM.multi<-val.iris[,c(1,2,3)]

SVM.multi<-Species~Sepal.Length+Petal.Length

SVM.iris.multi<-svm(SVM.multi, data=train.iris.SVM.multi, type='C-classification', kernel = 'linear')

SVM.iris.multi

#pedict

y\_pred.SVM.multi<-predict(SVM.iris.multi,newdata=val.iris.SVM.multi)

y\_pred.SVM.multi

#Confusion matrix

cm.iris.SVM.multi=table(val.iris.SVM.multi$Species,y\_pred.SVM.multi)

cm.iris.SVM.multi

# Visualising the test set results

set = val.iris.SVM.multi

X1 = seq(min(val.iris$Sepal.Length) - 1, max(val.iris$Sepal.Length) + 1, by = 0.01)

X2 = seq(min(val.iris$Petal.Length) - 1, max(set[val.iris$Petal.Length]) + 1, by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Sepal.Length', 'Petal.Length')

y\_grid = predict(SVM.iris.multi, newdata = grid\_set)

plot(val.iris$Spec.1,

main = 'SVM.Multi (Test )',

xlab = 'Sepal.Length', ylab = 'Petal.Length',

xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == "virginica", 'blue', ifelse(y\_grid=="setosa",'red','tomato')))

points(set, pch = 21, bg = ifelse(set[, 3] == "virginica",'black' ,ifelse(set[,3]=="setosa", 'green','blue')))

#####"LPM"

train.iris.LPM<-train.iris[,c(1,2,5)]

val.iris.LPM<-val.iris[,c(1,2,5)]

LPM<-Spec.1~Sepal.Length+Petal.Length

LPM.iris<-lm(LPM, data=train.iris.LPM)

summary(LPM.iris)

#pedict

y\_pred.LPM<-predict(LPM.iris,val.iris.LPM)

y\_pred.LPM

#Confusion matrix

cm.iris.LPM=table(val.iris.LPM$Spec.1 ,y\_pred.LPM>0.5)

cm.iris.LPM

# Visualising the test set results

set=val.iris.LPM

X1 = seq(min(set[, 1]) -1, max(set[, 1]) +1, by = 0.01)

X2 = seq(min(set[, 2]) -1, max(set[, 2])+1 , by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Sepal.Length', 'Petal.Length')

prob\_pred.LPM<-predict(LPM.iris,type='response',newdata=grid\_set)

y\_grid<-ifelse(prob\_pred.LPM>0.5,1,0)

plot(set[,-3],

main = 'LPM (Test )',

xlab = 'Sepal.Length', ylab = 'Petal.Length',

xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'blue', 'red'))

points(set, pch = 21, bg = ifelse(set[,3] == 1, 'black', 'green'))

#####

#logit

train.iris.Logit<-train.iris[,c(1,2,5)]

val.iris.Logit<-val.iris[,c(1,2,5)]

Logit<-Spec.1~Sepal.Length+Petal.Length

Logit.iris<-glm(Logit, family=binomial,data=train.iris.Logit)

summary(Logit.iris)

#pedict

prob\_pred.Logit<-predict(Logit.iris,type='response',newdata=val.iris.Logit[-3])

y\_pred.Logit<-ifelse(prob\_pred.Logit>0.5,1,0)

y\_pred.Logit

#Confusion matrix

cm.iris.Logit=table(val.iris.Logit$Spec.1 ,y\_pred.Logit)

cm.iris.Logit

summary(val.iris$Spec.1)

0.4211\*38

# Visualising the test set results

#set = val.iris

set=val.iris.Logit

X1 = seq(min(set[, 1]) -1, max(set[, 1]) +1, by = 0.01)

X2 = seq(min(set[, 2]) -1, max(set[, 2])+1 , by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Sepal.Length', 'Petal.Length')

prob\_pred.Logit<-predict(Logit.iris,type='response',newdata=grid\_set)

y\_grid<-ifelse(prob\_pred.Logit>0.5,1,0)

plot(set[,-3],

main = 'Logit (Test )',

xlab = 'Sepal.Length', ylab = 'Petal.Length',

xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'blue', 'red'))

points(set, pch = 21, bg = ifelse(set[,3] == 1, 'black', 'green'))

##########

#DT and RFM

#DT

train.iris.DT<-train.iris[,c(1,2,5)]

val.iris.DT<-val.iris[,c(1,2,5)]

train.iris.DT$Spec.1<-as.factor(train.iris.DT$Spec.1)

val.iris.DT$Spec.1<-as.factor(val.iris.DT$Spec.1)

str(train.iris.DT)

str(val.iris.DT)

DT<-Spec.1~Sepal.Length+Petal.Length

#

DT.iris<-rpart(DT,data=train.iris.DT)

DT.iris

rpart.plot(DT.iris)

rpart.plot(DT.iris, type=3,extra=101,fallen.leaves = T)

summary(DT.iris)

#pedict

y\_pred.DT<-predict(DT.iris,newdata=val.iris.DT[-3],type="class")

y\_pred.DT

#y\_pred.DT<-ifelse(y\_pred.DT>0.5,1,0)

#Confusion matrix

cm.iris.DT=table(val.iris.DT[,3],y\_pred.DT)

cm.iris.DT

summary(val.iris.DT$Spec.1)

# Visualising the Test set results

library(ElemStatLearn)

set = val.iris.DT

X1 = seq(min(set[, 1])-1 , max(set[, 1]) +1, by = 0.01)

X2 = seq(min(set[, 2])-1 , max(set[, 2])+1 , by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Sepal.Length', 'Petal.Length')

y\_grid = predict(DT.iris,newdata=grid\_set, type='class')

plot(set[, -3],

main = 'DT(Iris) ',

xlab = 'Sepal.Length', ylab = 'Petal.Length',

xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'blue', 'red'))

points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'black', 'green'))

#RFM

train.iris.RFM<-train.iris[,c(1,2,5)]

val.iris.RFM<-val.iris[,c(1,2,5)]

train.iris.RFM$Spec.1<-as.factor(train.iris.RFM$Spec.1)

val.iris.RFM$Spec.1<-as.factor(val.iris.RFM$Spec.1)

str(train.iris.RFM)

str(val.iris.RFM)

RFM<-Spec.1~Sepal.Length+Petal.Length

#

RFM.iris<-randomForest(RFM,data=train.iris.RFM)

RFM.iris

summary(RFM.iris)

#pedict

y\_pred.RFM<-predict(RFM.iris,newdata=val.iris.RFM[-3],type="class")

y\_pred.RFM

#y\_pred.RFM<-ifelse(y\_pred.RFM>0.5,1,0)

#Confusion matrix

cm.iris.RFM=table(val.iris.RFM[,3],y\_pred.RFM)

cm.iris.RFM

summary(val.iris.RFM$Spec.1)

# Visualising the Test set results

library(ElemStatLearn)

set = val.iris.RFM

X1 = seq(min(set[, 1])-1 , max(set[, 1]) +1, by = 0.01)

X2 = seq(min(set[, 2])-1 , max(set[, 2])+1 , by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Sepal.Length', 'Petal.Length')

y\_grid = predict(RFM.iris,newdata=grid\_set, type='class')

plot(set[, -3],

main = 'RFM(Iris) ',

xlab = 'Sepal.Length', ylab = 'Petal.Length',

xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'blue', 'red'))

points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'black', 'green'))

######

#Neural Net

train.iris.ANN<-train.iris[,c(1,2,5)]

val.iris.ANN<-val.iris[,c(1,2,5)]

ANN<-Spec.1~Sepal.Length+Petal.Length

ANN.iris <- neuralnet(ANN,

data = train.iris.ANN,

hidden = 1,

err.fct = "ce",

linear.output = FALSE)

plot(ANN.iris)

# Confusion Matrix & Misclassification Error - training data

output <- compute(ANN.iris, val.iris.ANN[,-3])

p1 <- output$net.result

pred1 <- ifelse(p1>0.5, 1, 0)

tab1 <- table(pred1, val.iris.ANN$Spec.1)

tab1

# Visualising the Test set results

set = val.iris.ANN

X1 = seq(min(set[, 1])-1 , max(set[, 1]) +1, by = 0.01)

X2 = seq(min(set[, 2])-1 , max(set[, 2])+1 , by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Sepal.Length', 'Petal.Length')

y\_grid = predict(p1,newdata=grid\_set)

plot(set[, -3],

main = 'ANN(Iris) ',

xlab = 'Sepal.Length', ylab = 'Petal.Length',

xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)

points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'blue', 'red'))

points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'black', 'green'))

#################

IRIS