Sample ANN Practice Exercise - Drug Reaction

PVS

Wednesday, February 22, 2017

## Problem statement

A Pharmaceutical firm that developed a particular drug for women wants to understand the characteristics that cause some of them to have an adverse reaction to a particular drug. They collect data on 15 women who had such a reaction and 15 women who did not. The variable measured are \* Systolic Blood Pressure \* Cholesterol level \* Age of the person \* Whether or not the woman was pregnant (1=yes) \* The dependent variable indicates if there was an adverse reaction (1=yes)

Using neural network classify and predict five cases.

## Solution

The sample data size is 30; we split that into training data and test data in the ration 25:5

## Load required packages

library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

library(nnet)# limited plot methods but very good to start learning ANN,Can use Neuralnet library  
## check "https://beckmw.wordpress.com/tag/nnet/" to learn more  
library(pROC)

## Type 'citation("pROC")' for a citation.

##   
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':  
##   
## cov, smooth, var

library(e1071)

## set working directory and read input data

setwd ("D:/NN")  
getwd()

## [1] "D:/NN"

data\_nn = read.table("D:/NN/DrugReaction.csv",sep = ",", header = T)

## check the data distribution & type

str(data\_nn)

## 'data.frame': 30 obs. of 5 variables:  
## $ BP : int 100 120 110 100 95 110 120 150 160 125 ...  
## $ Cholesterol : int 150 160 150 175 250 200 180 175 185 195 ...  
## $ Age : int 20 16 18 25 36 56 59 45 40 20 ...  
## $ Pregnant : int 0 0 0 0 0 0 0 0 0 1 ...  
## $ Drug.Reaction: int 0 0 0 0 0 0 0 0 0 0 ...

# Define a function to find the type of variable in the data frame, hrData  
  
findClass <- function (x) {  
 for (i in 1:length(x))  
 {  
 l1 <- names(x[i])  
 l2 <- class(x[[i]])  
 cat("\n ",l1,": ",l2)  
 }   
}  
  
#  
cat("\n Variables according to the class \n")

##   
## Variables according to the class

findClass(data\_nn) ## Get the list of type of variables

##   
## BP : integer  
## Cholesterol : integer  
## Age : integer  
## Pregnant : integer  
## Drug.Reaction : integer

**Observation**

There are 5 variables and 30 observations.

**A) Numerical variables:**

1. BP
2. Cholesterol
3. Age

**B) Integer variables -categorical**

1. Pregnant
2. Drug.Reaction

## Data cleaning

#### 1 Find if there are any missing values

#### 2 Remove Feature with constant values else will give error for Neural Network

#### 3 Remove columns with near zero variance as features should has variance in its distribution.

### 1 Find if there are any missing values  
  
cat("\n Variables with number of missing values \n")

##   
## Variables with number of missing values

sapply(data\_nn, function(x) sum(is.na(x))) # To report missing values

## BP Cholesterol Age Pregnant Drug.Reaction   
## 0 0 0 0 0

### 2 Find if there are any constant values for variables  
  
cat("\n Variables with constant values \n")

##   
## Variables with constant values

sapply(data\_nn, function(x) length(unique(x)))

## BP Cholesterol Age Pregnant Drug.Reaction   
## 15 14 22 2 2

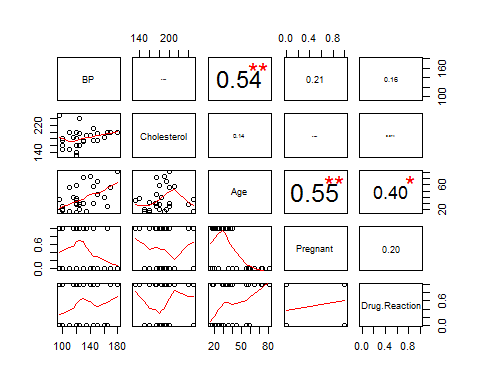
### 3 Find if any variables have near zero variance  
  
nzv <- nearZeroVar(data\_nn)  
nzv

## integer(0)

**Observation**

* There are **No variables** with missing values or constant values or near zero variance

## Exploratory Data Analysis

 **Observation**

1. We find there is no linear relationship between pair of variables except for

* BP and Age - with correlation coefficient is 0.54
* Age and Pregnant - with correlation coefficient is 0.55
* Age and Drug Reaction - with correlation coefficient is 0.40

1. Correlation coefficient is also very low for these variables - the highest being 0.20 (r for Drug.Reaction and Pregnant)

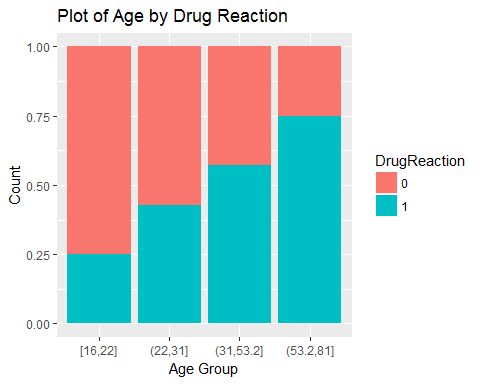
### Visual Representation

#### Drug Reaction % by Age

##   
## Attaching package: 'gtools'

## The following object is masked from 'package:e1071':  
##   
## permutations

##   
## Drug Reaction % by Age



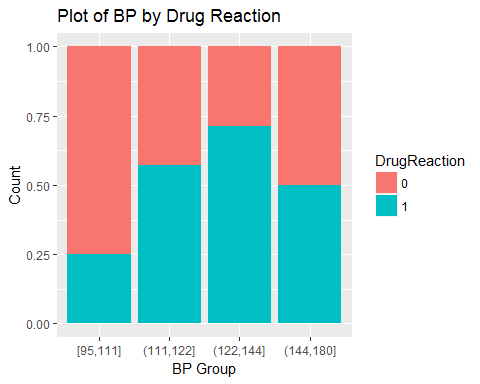
##   
##   
## In the age group [16,22], 75 % of participants have no Drug Reaction and   
## 25 % of participants have Drug Reaction

##   
##   
## In the age group [22,31], 57.14 % of participants have no Drug Reaction and   
## 42.86 % of participants have Drug Reaction

##   
##   
## In the age group [31,53.2], 42.86 % of participants have no Drug Reaction and   
## 57.14 % of participants have Drug Reaction

##   
##   
## In the age group [53.2,81], 25 % of participants have no Drug Reaction and   
## 75 % of participants have Drug Reaction

##   
## Drug Reaction % by BP



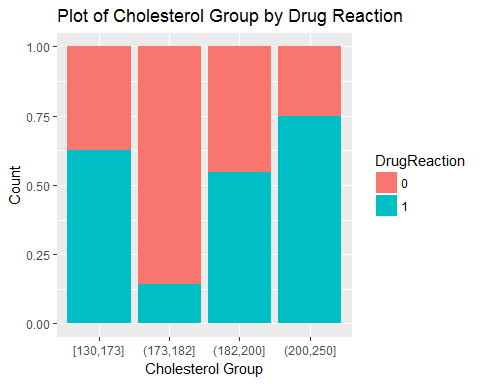
##   
##   
## In the BP group [95,111], 75 % of participants have no Drug Reaction and   
## 25 % of participants have Drug Reaction

##   
##   
## In the BP group [111,122], 42.86 % of participants have no Drug Reaction and   
## 57.14 % of participants have Drug Reaction

##   
##   
## In the BP group [122,144], 28.57 % of participants have no Drug Reaction and   
## 71.43 % of participants have Drug Reaction

##   
##   
## In the BP group [144,180], 50 % of participants have no Drug Reaction and   
## 50 % of participants have Drug Reaction

##   
## Drug Reaction % by Cholesterol Group



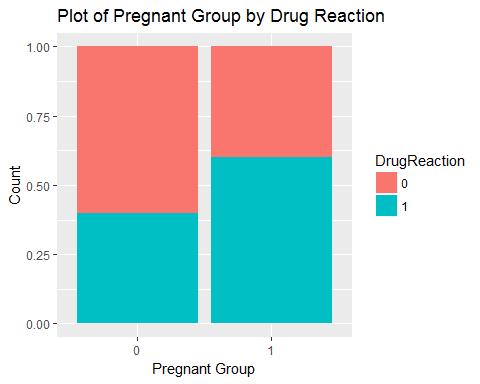
##   
##   
## In the Cholesterol Group [130,173], 37.5 % of participants have no Drug Reaction and   
## 62.5 % of participants have Drug Reaction

##   
##   
## In the Cholesterol Group[173,182], 85.71 % of participants have no Drug Reaction and   
## 14.29 % of participants have Drug Reaction

##   
##   
## In the Cholesterol Group [182,200], 45.45 % of participants have no Drug Reaction and   
## 54.55 % of participants have Drug Reaction

##   
##   
## In the Cholesterol Group [200,250], 25 % of participants have no Drug Reaction and   
## 75 % of participants have Drug Reaction

##   
## Drug Reaction % by Pregnant Group



##   
##   
## In the Non-Pregnant Group, 60 % of participants have no Drug Reaction and   
## 40 % of participants have Drug Reaction

##   
##   
## In the Pregnant Group, 40 % of participants have no Drug Reaction and   
## 60 % of participants have Drug Reaction

## Data distribution

### Understand dependent variable distribution split

reaction.ratio = table(data\_nn$Drug.Reaction)  
reaction.ratio

##   
## 0 1   
## 15 15

**Observation**

|  |  |  |
| --- | --- | --- |
| SNo | Description | Observation count |
| 1 | Observations with no Drug Reaction | 15 |
| 2 | Observations with Drug Reaction | 15 |

### understand Distribution in percentage terms

reaction.prop = prop.table(reaction.ratio)  
reaction.prop

##   
## 0 1   
## 0.5 0.5

|  |  |  |
| --- | --- | --- |
| SNo | Description | Percentage |
| 1 | Observations with no Drug Reaction | 50% |
| 2 | Observations with Drug Reaction | 50% |

## Split data into Training and Test dataset

set.seed(100)  
n <- 30; trainIndex = sample(1:n, size = round(5/6 \* n), replace=FALSE)  
train.data <- data\_nn[trainIndex,]  
test.data <- data\_nn[-trainIndex,]

## Check if distribution of partition data is correct

prop.table((table(train.data$Drug.Reaction)))

##   
## 0 1   
## 0.52 0.48

**Observations**

#### Distribution of Drug Reaction cases in training data set

|  |  |  |
| --- | --- | --- |
| SNo | Description | Percentage |
| 1 | Observations with no Drug Reaction | 52% |
| 2 | Observations with Drug Reaction | 48% |

prop.table((table(test.data$Drug.Reaction)))

##   
## 0 1   
## 0.4 0.6

**Observations**

#### Distribution of Drug Reaction cases in test data set

|  |  |  |
| --- | --- | --- |
| SNo | Description | Percentage |
| 1 | Observations with no Drug Reaction | 40% |
| 2 | Observations with Drug Reaction | 60% |

* So the data is almost well distributed in the training data and test data sets in between 40 - 52% to 60 to 48%

train.data$Drug.Reaction <- as.factor(train.data$Drug.Reaction)  
train.data$Pregnant <- as.factor(train.data$Pregnant)  
  
levels(train.data$Drug.Reaction) <- c("0","1")  
levels(train.data$Pregnant) <- c("0","1")  
levels(train.data$Drug.Reaction) <- make.names(levels(factor(train.data$Drug.Reaction)))  
  
test.data$Drug.Reaction <- as.factor(test.data$Drug.Reaction)  
test.data$Pregnant <- as.factor(test.data$Pregnant)  
  
levels(test.data$Drug.Reaction) <- c("0","1")  
levels(test.data$Pregnant) <- c("0","1")  
levels(test.data$Drug.Reaction) <- make.names(levels(factor(test.data$Drug.Reaction)))

### Create a control structure for traing NN also with required switches

### Set metric value as Accuracy or Kappa for neural Network model  
metric <- "Accuracy"  
  
# Create a control structure for traing NN also with required switches  
  
#Cross Validation  
cctrl <- trainControl(  
 method = 'cv',number = 10,classProbs = TRUE,  
 verboseIter = TRUE, summaryFunction = twoClassSummary,  
 preProcOptions = list(  
 thresh = 0.75,ICAcomp = 3, k = 5  
 )   
)  
  
# define decay and sizes of hidden neuron to train algorithm on   
my.grid <- expand.grid(.decay = c(0.1, 0.001, 0.0001), .size = c(5, 10, 15))  
# Train/create Neural Network Model  
nn\_model <-  
 train( Drug.Reaction ~ ., data = train.data,  
 method = 'nnet',preProcess = c('center', 'scale'),  
 trControl = cctrl,tuneGrid = my.grid )

## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" was  
## not in the result set. ROC will be used instead.

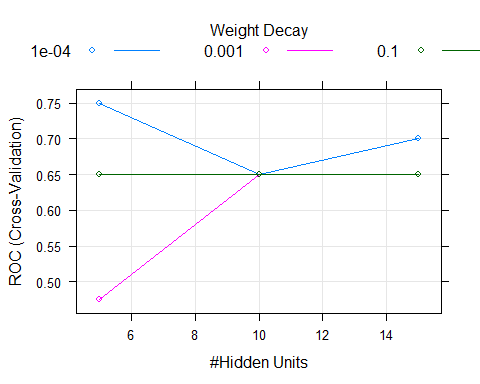
## + Fold01: decay=1e-01, size= 5   
## # weights: 31  
## initial value 17.734462   
## iter 10 value 12.257474  
## iter 20 value 11.879315  
## iter 30 value 11.841306  
## iter 40 value 11.840692  
## final value 11.840686   
## converged  
## - Fold01: decay=1e-01, size= 5   
## + Fold01: decay=1e-03, size= 5   
## # weights: 31  
## initial value 17.582875   
## iter 10 value 8.685935  
## iter 20 value 2.998318  
## iter 30 value 1.327791  
## iter 40 value 1.099472  
## iter 50 value 1.020457  
## iter 60 value 0.947710  
## iter 70 value 0.905868  
## iter 80 value 0.876504  
## iter 90 value 0.855019  
## iter 100 value 0.844691  
## final value 0.844691   
## stopped after 100 iterations  
## - Fold01: decay=1e-03, size= 5   
## + Fold01: decay=1e-04, size= 5   
## # weights: 31  
## initial value 16.841348   
## iter 10 value 3.467540  
## iter 20 value 1.960935  
## iter 30 value 1.680392  
## iter 40 value 0.577052  
## iter 50 value 0.363980  
## iter 60 value 0.315567  
## iter 70 value 0.285825  
## iter 80 value 0.247162  
## iter 90 value 0.227949  
## iter 100 value 0.217958  
## final value 0.217958   
## stopped after 100 iterations  
## - Fold01: decay=1e-04, size= 5   
## + Fold01: decay=1e-01, size=10   
## # weights: 61  
## initial value 20.063311   
## iter 10 value 12.105016  
## iter 20 value 11.777788  
## iter 30 value 11.738165  
## iter 40 value 11.732874  
## iter 50 value 11.732162  
## iter 60 value 11.732103  
## final value 11.732102   
## converged  
## - Fold01: decay=1e-01, size=10   
## + Fold01: decay=1e-03, size=10   
## # weights: 61  
## initial value 16.874041   
## iter 10 value 6.397300  
## iter 20 value 2.247031  
## iter 30 value 1.166522  
## iter 40 value 0.997714  
## iter 50 value 0.896968  
## iter 60 value 0.867058  
## iter 70 value 0.840597  
## iter 80 value 0.806885  
## iter 90 value 0.793568  
## iter 100 value 0.785067  
## final value 0.785067   
## stopped after 100 iterations  
## - Fold01: decay=1e-03, size=10   
## + Fold01: decay=1e-04, size=10   
## # weights: 61  
## initial value 16.692532   
## iter 10 value 3.739956  
## iter 20 value 0.648586  
## iter 30 value 0.346860  
## iter 40 value 0.226330  
## iter 50 value 0.197043  
## iter 60 value 0.177542  
## iter 70 value 0.163355  
## iter 80 value 0.147759  
## iter 90 value 0.143611  
## iter 100 value 0.137764  
## final value 0.137764   
## stopped after 100 iterations  
## - Fold01: decay=1e-04, size=10   
## + Fold01: decay=1e-01, size=15   
## # weights: 91  
## initial value 17.917441   
## iter 10 value 11.795976  
## iter 20 value 11.711960  
## iter 30 value 11.704650  
## iter 40 value 11.699774  
## iter 50 value 11.694433  
## iter 60 value 11.694282  
## iter 70 value 11.690840  
## iter 80 value 11.688654  
## final value 11.688587   
## converged  
## - Fold01: decay=1e-01, size=15   
## + Fold01: decay=1e-03, size=15   
## # weights: 91  
## initial value 19.558047   
## iter 10 value 3.959064  
## iter 20 value 1.231273  
## iter 30 value 0.948864  
## iter 40 value 0.881384  
## iter 50 value 0.846886  
## iter 60 value 0.825309  
## iter 70 value 0.800551  
## iter 80 value 0.789782  
## iter 90 value 0.784095  
## iter 100 value 0.780974  
## final value 0.780974   
## stopped after 100 iterations  
## - Fold01: decay=1e-03, size=15   
## + Fold01: decay=1e-04, size=15   
## # weights: 91  
## initial value 19.859344   
## iter 10 value 3.066035  
## iter 20 value 0.358575  
## iter 30 value 0.240872  
## iter 40 value 0.201259  
## iter 50 value 0.183751  
## iter 60 value 0.157672  
## iter 70 value 0.145826  
## iter 80 value 0.134916  
## iter 90 value 0.127930  
## iter 100 value 0.125976  
## final value 0.125976   
## stopped after 100 iterations  
## - Fold01: decay=1e-04, size=15   
## + Fold02: decay=1e-01, size= 5   
## # weights: 31  
## initial value 16.738916   
## iter 10 value 9.997051  
## iter 20 value 9.820076  
## iter 30 value 9.816019  
## final value 9.815955   
## converged  
## - Fold02: decay=1e-01, size= 5   
## + Fold02: decay=1e-03, size= 5   
## # weights: 31  
## initial value 15.704266   
## iter 10 value 1.043389  
## iter 20 value 0.625920  
## iter 30 value 0.563440  
## iter 40 value 0.545855  
## iter 50 value 0.536105  
## iter 60 value 0.527314  
## iter 70 value 0.517959  
## iter 80 value 0.510463  
## iter 90 value 0.499066  
## iter 100 value 0.491269  
## final value 0.491269   
## stopped after 100 iterations  
## - Fold02: decay=1e-03, size= 5   
## + Fold02: decay=1e-04, size= 5   
## # weights: 31  
## initial value 16.750614   
## iter 10 value 2.198630  
## iter 20 value 0.201886  
## iter 30 value 0.140144  
## iter 40 value 0.109390  
## iter 50 value 0.100816  
## iter 60 value 0.097670  
## iter 70 value 0.093629  
## iter 80 value 0.092125  
## iter 90 value 0.089992  
## iter 100 value 0.088528  
## final value 0.088528   
## stopped after 100 iterations  
## - Fold02: decay=1e-04, size= 5   
## + Fold02: decay=1e-01, size=10   
## # weights: 61  
## initial value 15.767813   
## iter 10 value 9.796514  
## iter 20 value 9.724756  
## iter 30 value 9.689875  
## iter 40 value 9.677688  
## iter 50 value 9.676768  
## final value 9.676762   
## converged  
## - Fold02: decay=1e-01, size=10   
## + Fold02: decay=1e-03, size=10   
## # weights: 61  
## initial value 20.411895   
## iter 10 value 1.352829  
## iter 20 value 0.611306  
## iter 30 value 0.518301  
## iter 40 value 0.500768  
## iter 50 value 0.485357  
## iter 60 value 0.473439  
## iter 70 value 0.467603  
## iter 80 value 0.464038  
## iter 90 value 0.460843  
## iter 100 value 0.458175  
## final value 0.458175   
## stopped after 100 iterations  
## - Fold02: decay=1e-03, size=10   
## + Fold02: decay=1e-04, size=10   
## # weights: 61  
## initial value 16.371409   
## iter 10 value 0.185001  
## iter 20 value 0.108898  
## iter 30 value 0.083203  
## iter 40 value 0.078677  
## iter 50 value 0.075557  
## iter 60 value 0.073289  
## iter 70 value 0.069341  
## iter 80 value 0.067325  
## iter 90 value 0.065878  
## iter 100 value 0.064766  
## final value 0.064766   
## stopped after 100 iterations  
## - Fold02: decay=1e-04, size=10   
## + Fold02: decay=1e-01, size=15   
## # weights: 91  
## initial value 20.950542   
## iter 10 value 9.769776  
## iter 20 value 9.665014  
## iter 30 value 9.653353  
## iter 40 value 9.649865  
## iter 50 value 9.649368  
## iter 60 value 9.649317  
## final value 9.649312   
## converged  
## - Fold02: decay=1e-01, size=15   
## + Fold02: decay=1e-03, size=15   
## # weights: 91  
## initial value 16.496644   
## iter 10 value 0.852936  
## iter 20 value 0.541162  
## iter 30 value 0.484373  
## iter 40 value 0.456951  
## iter 50 value 0.448534  
## iter 60 value 0.444022  
## iter 70 value 0.440657  
## iter 80 value 0.438502  
## iter 90 value 0.435906  
## iter 100 value 0.434503  
## final value 0.434503   
## stopped after 100 iterations  
## - Fold02: decay=1e-03, size=15   
## + Fold02: decay=1e-04, size=15   
## # weights: 91  
## initial value 19.791124   
## iter 10 value 0.965697  
## iter 20 value 0.100412  
## iter 30 value 0.083281  
## iter 40 value 0.077523  
## iter 50 value 0.074007  
## iter 60 value 0.071181  
## iter 70 value 0.069067  
## iter 80 value 0.067971  
## iter 90 value 0.066818  
## iter 100 value 0.065939  
## final value 0.065939   
## stopped after 100 iterations  
## - Fold02: decay=1e-04, size=15   
## + Fold03: decay=1e-01, size= 5   
## # weights: 31  
## initial value 17.204244   
## iter 10 value 12.533720  
## iter 20 value 12.154209  
## iter 30 value 12.131177  
## iter 40 value 12.130522  
## final value 12.130521   
## converged  
## - Fold03: decay=1e-01, size= 5   
## + Fold03: decay=1e-03, size= 5   
## # weights: 31  
## initial value 16.375053   
## iter 10 value 3.517273  
## iter 20 value 1.426287  
## iter 30 value 0.967192  
## iter 40 value 0.931328  
## iter 50 value 0.916248  
## iter 60 value 0.908387  
## iter 70 value 0.903471  
## iter 80 value 0.898696  
## iter 90 value 0.889807  
## iter 100 value 0.881946  
## final value 0.881946   
## stopped after 100 iterations  
## - Fold03: decay=1e-03, size= 5   
## + Fold03: decay=1e-04, size= 5   
## # weights: 31  
## initial value 18.185791   
## iter 10 value 6.787791  
## iter 20 value 2.154124  
## iter 30 value 0.411826  
## iter 40 value 0.235245  
## iter 50 value 0.228735  
## iter 60 value 0.217311  
## iter 70 value 0.205258  
## iter 80 value 0.189725  
## iter 90 value 0.184743  
## iter 100 value 0.181043  
## final value 0.181043   
## stopped after 100 iterations  
## - Fold03: decay=1e-04, size= 5   
## + Fold03: decay=1e-01, size=10   
## # weights: 61  
## initial value 16.033815   
## iter 10 value 12.148761  
## iter 20 value 12.036791  
## iter 30 value 12.014105  
## iter 40 value 12.012080  
## iter 50 value 12.011960  
## iter 50 value 12.011960  
## iter 50 value 12.011960  
## final value 12.011960   
## converged  
## - Fold03: decay=1e-01, size=10   
## + Fold03: decay=1e-03, size=10   
## # weights: 61  
## initial value 16.604652   
## iter 10 value 4.951552  
## iter 20 value 1.442444  
## iter 30 value 1.010795  
## iter 40 value 0.907118  
## iter 50 value 0.861925  
## iter 60 value 0.829776  
## iter 70 value 0.813163  
## iter 80 value 0.800021  
## iter 90 value 0.790634  
## iter 100 value 0.783442  
## final value 0.783442   
## stopped after 100 iterations  
## - Fold03: decay=1e-03, size=10   
## + Fold03: decay=1e-04, size=10   
## # weights: 61  
## initial value 16.763950   
## iter 10 value 3.641409  
## iter 20 value 0.882955  
## iter 30 value 0.335559  
## iter 40 value 0.260679  
## iter 50 value 0.225287  
## iter 60 value 0.194642  
## iter 70 value 0.169002  
## iter 80 value 0.156565  
## iter 90 value 0.145718  
## iter 100 value 0.139340  
## final value 0.139340   
## stopped after 100 iterations  
## - Fold03: decay=1e-04, size=10   
## + Fold03: decay=1e-01, size=15   
## # weights: 91  
## initial value 17.524514   
## iter 10 value 12.071022  
## iter 20 value 12.003161  
## iter 30 value 11.986284  
## iter 40 value 11.980484  
## iter 50 value 11.978917  
## iter 60 value 11.978806  
## final value 11.978803   
## converged  
## - Fold03: decay=1e-01, size=15   
## + Fold03: decay=1e-03, size=15   
## # weights: 91  
## initial value 17.690970   
## iter 10 value 3.559608  
## iter 20 value 1.544281  
## iter 30 value 1.029011  
## iter 40 value 0.866856  
## iter 50 value 0.810831  
## iter 60 value 0.786903  
## iter 70 value 0.772712  
## iter 80 value 0.765810  
## iter 90 value 0.757306  
## iter 100 value 0.751094  
## final value 0.751094   
## stopped after 100 iterations  
## - Fold03: decay=1e-03, size=15   
## + Fold03: decay=1e-04, size=15   
## # weights: 91  
## initial value 18.279751   
## iter 10 value 2.289525  
## iter 20 value 0.265858  
## iter 30 value 0.207439  
## iter 40 value 0.160854  
## iter 50 value 0.143456  
## iter 60 value 0.131620  
## iter 70 value 0.124970  
## iter 80 value 0.120085  
## iter 90 value 0.117294  
## iter 100 value 0.113885  
## final value 0.113885   
## stopped after 100 iterations  
## - Fold03: decay=1e-04, size=15   
## + Fold04: decay=1e-01, size= 5   
## # weights: 31  
## initial value 18.029418   
## iter 10 value 12.307010  
## iter 20 value 12.201733  
## iter 30 value 12.200269  
## iter 40 value 12.200107  
## final value 12.200107   
## converged  
## - Fold04: decay=1e-01, size= 5   
## + Fold04: decay=1e-03, size= 5   
## # weights: 31  
## initial value 16.234364   
## iter 10 value 4.094477  
## iter 20 value 1.204006  
## iter 30 value 0.977376  
## iter 40 value 0.885546  
## iter 50 value 0.863646  
## iter 60 value 0.853873  
## iter 70 value 0.846304  
## iter 80 value 0.840612  
## iter 90 value 0.832974  
## iter 100 value 0.825469  
## final value 0.825469   
## stopped after 100 iterations  
## - Fold04: decay=1e-03, size= 5   
## + Fold04: decay=1e-04, size= 5   
## # weights: 31  
## initial value 18.209322   
## iter 10 value 2.872874  
## iter 20 value 0.754854  
## iter 30 value 0.348799  
## iter 40 value 0.260900  
## iter 50 value 0.214024  
## iter 60 value 0.187369  
## iter 70 value 0.177228  
## iter 80 value 0.170450  
## iter 90 value 0.162969  
## iter 100 value 0.154685  
## final value 0.154685   
## stopped after 100 iterations  
## - Fold04: decay=1e-04, size= 5   
## + Fold04: decay=1e-01, size=10   
## # weights: 61  
## initial value 19.320934   
## iter 10 value 11.978128  
## iter 20 value 11.920770  
## iter 30 value 11.906028  
## iter 40 value 11.904952  
## final value 11.904938   
## converged  
## - Fold04: decay=1e-01, size=10   
## + Fold04: decay=1e-03, size=10   
## # weights: 61  
## initial value 23.754133   
## iter 10 value 2.218118  
## iter 20 value 1.046580  
## iter 30 value 0.930082  
## iter 40 value 0.840490  
## iter 50 value 0.798288  
## iter 60 value 0.768251  
## iter 70 value 0.746008  
## iter 80 value 0.737171  
## iter 90 value 0.727384  
## iter 100 value 0.723638  
## final value 0.723638   
## stopped after 100 iterations  
## - Fold04: decay=1e-03, size=10   
## + Fold04: decay=1e-04, size=10   
## # weights: 61  
## initial value 19.395163   
## iter 10 value 3.283880  
## iter 20 value 0.396410  
## iter 30 value 0.275943  
## iter 40 value 0.211151  
## iter 50 value 0.182605  
## iter 60 value 0.173789  
## iter 70 value 0.157234  
## iter 80 value 0.141999  
## iter 90 value 0.131497  
## iter 100 value 0.125614  
## final value 0.125614   
## stopped after 100 iterations  
## - Fold04: decay=1e-04, size=10   
## + Fold04: decay=1e-01, size=15   
## # weights: 91  
## initial value 20.707902   
## iter 10 value 11.970450  
## iter 20 value 11.892826  
## iter 30 value 11.879706  
## iter 40 value 11.875339  
## iter 50 value 11.874481  
## iter 60 value 11.872455  
## iter 70 value 11.872336  
## final value 11.872335   
## converged  
## - Fold04: decay=1e-01, size=15   
## + Fold04: decay=1e-03, size=15   
## # weights: 91  
## initial value 29.211092   
## iter 10 value 4.198141  
## iter 20 value 1.336140  
## iter 30 value 0.894158  
## iter 40 value 0.790358  
## iter 50 value 0.750543  
## iter 60 value 0.725712  
## iter 70 value 0.705667  
## iter 80 value 0.688345  
## iter 90 value 0.681201  
## iter 100 value 0.675086  
## final value 0.675086   
## stopped after 100 iterations  
## - Fold04: decay=1e-03, size=15   
## + Fold04: decay=1e-04, size=15   
## # weights: 91  
## initial value 17.499521   
## iter 10 value 2.461887  
## iter 20 value 0.347857  
## iter 30 value 0.257730  
## iter 40 value 0.200856  
## iter 50 value 0.169556  
## iter 60 value 0.150805  
## iter 70 value 0.134460  
## iter 80 value 0.122432  
## iter 90 value 0.115917  
## iter 100 value 0.111534  
## final value 0.111534   
## stopped after 100 iterations  
## - Fold04: decay=1e-04, size=15   
## + Fold05: decay=1e-01, size= 5   
## # weights: 31  
## initial value 17.039884   
## iter 10 value 10.999131  
## iter 20 value 10.788669  
## iter 30 value 10.750761  
## iter 40 value 10.747132  
## final value 10.747128   
## converged  
## - Fold05: decay=1e-01, size= 5   
## + Fold05: decay=1e-03, size= 5   
## # weights: 31  
## initial value 17.700042   
## iter 10 value 2.234273  
## iter 20 value 0.901292  
## iter 30 value 0.816520  
## iter 40 value 0.792819  
## iter 50 value 0.775967  
## iter 60 value 0.768891  
## iter 70 value 0.759924  
## iter 80 value 0.755072  
## iter 90 value 0.744119  
## iter 100 value 0.732311  
## final value 0.732311   
## stopped after 100 iterations  
## - Fold05: decay=1e-03, size= 5   
## + Fold05: decay=1e-04, size= 5   
## # weights: 31  
## initial value 19.154671   
## iter 10 value 3.070223  
## iter 20 value 0.204572  
## iter 30 value 0.187859  
## iter 40 value 0.174553  
## iter 50 value 0.155987  
## iter 60 value 0.137344  
## iter 70 value 0.131472  
## iter 80 value 0.128216  
## iter 90 value 0.124989  
## iter 100 value 0.120705  
## final value 0.120705   
## stopped after 100 iterations  
## - Fold05: decay=1e-04, size= 5   
## + Fold05: decay=1e-01, size=10   
## # weights: 61  
## initial value 19.412402   
## iter 10 value 10.765707  
## iter 20 value 10.672766  
## iter 30 value 10.648889  
## iter 40 value 10.632903  
## iter 50 value 10.632297  
## iter 60 value 10.632287  
## iter 60 value 10.632287  
## iter 60 value 10.632287  
## final value 10.632287   
## converged  
## - Fold05: decay=1e-01, size=10   
## + Fold05: decay=1e-03, size=10   
## # weights: 61  
## initial value 19.478569   
## iter 10 value 2.515636  
## iter 20 value 0.665167  
## iter 30 value 0.606211  
## iter 40 value 0.585826  
## iter 50 value 0.574238  
## iter 60 value 0.569713  
## iter 70 value 0.565759  
## iter 80 value 0.558186  
## iter 90 value 0.552282  
## iter 100 value 0.548385  
## final value 0.548385   
## stopped after 100 iterations  
## - Fold05: decay=1e-03, size=10   
## + Fold05: decay=1e-04, size=10   
## # weights: 61  
## initial value 16.782457   
## iter 10 value 1.745175  
## iter 20 value 0.177421  
## iter 30 value 0.139943  
## iter 40 value 0.127238  
## iter 50 value 0.113122  
## iter 60 value 0.105598  
## iter 70 value 0.100548  
## iter 80 value 0.094372  
## iter 90 value 0.089923  
## iter 100 value 0.088204  
## final value 0.088204   
## stopped after 100 iterations  
## - Fold05: decay=1e-04, size=10   
## + Fold05: decay=1e-01, size=15   
## # weights: 91  
## initial value 18.130202   
## iter 10 value 10.822084  
## iter 20 value 10.619097  
## iter 30 value 10.593974  
## iter 40 value 10.586568  
## iter 50 value 10.585965  
## iter 60 value 10.585954  
## final value 10.585954   
## converged  
## - Fold05: decay=1e-01, size=15   
## + Fold05: decay=1e-03, size=15   
## # weights: 91  
## initial value 16.767144   
## iter 10 value 1.801729  
## iter 20 value 0.725507  
## iter 30 value 0.600005  
## iter 40 value 0.564732  
## iter 50 value 0.547633  
## iter 60 value 0.540211  
## iter 70 value 0.535234  
## iter 80 value 0.531738  
## iter 90 value 0.530321  
## iter 100 value 0.528990  
## final value 0.528990   
## stopped after 100 iterations  
## - Fold05: decay=1e-03, size=15   
## + Fold05: decay=1e-04, size=15   
## # weights: 91  
## initial value 15.956872   
## iter 10 value 0.787741  
## iter 20 value 0.131281  
## iter 30 value 0.116642  
## iter 40 value 0.105632  
## iter 50 value 0.098389  
## iter 60 value 0.091555  
## iter 70 value 0.086156  
## iter 80 value 0.084077  
## iter 90 value 0.082084  
## iter 100 value 0.080701  
## final value 0.080701   
## stopped after 100 iterations  
## - Fold05: decay=1e-04, size=15   
## + Fold06: decay=1e-01, size= 5   
## # weights: 31  
## initial value 16.674396   
## iter 10 value 12.111441  
## iter 20 value 11.602973  
## iter 30 value 11.553205  
## iter 40 value 11.549600  
## final value 11.549593   
## converged  
## - Fold06: decay=1e-01, size= 5   
## + Fold06: decay=1e-03, size= 5   
## # weights: 31  
## initial value 17.698234   
## iter 10 value 4.131339  
## iter 20 value 1.305941  
## iter 30 value 0.911504  
## iter 40 value 0.843969  
## iter 50 value 0.813268  
## iter 60 value 0.798498  
## iter 70 value 0.788456  
## iter 80 value 0.783521  
## iter 90 value 0.773879  
## iter 100 value 0.762750  
## final value 0.762750   
## stopped after 100 iterations  
## - Fold06: decay=1e-03, size= 5   
## + Fold06: decay=1e-04, size= 5   
## # weights: 31  
## initial value 16.190691   
## iter 10 value 2.570646  
## iter 20 value 0.380978  
## iter 30 value 0.290712  
## iter 40 value 0.233478  
## iter 50 value 0.205911  
## iter 60 value 0.198827  
## iter 70 value 0.187752  
## iter 80 value 0.181682  
## iter 90 value 0.172888  
## iter 100 value 0.163772  
## final value 0.163772   
## stopped after 100 iterations  
## - Fold06: decay=1e-04, size= 5   
## + Fold06: decay=1e-01, size=10   
## # weights: 61  
## initial value 17.399902   
## iter 10 value 11.949950  
## iter 20 value 11.494446  
## iter 30 value 11.447084  
## iter 40 value 11.441024  
## iter 50 value 11.440570  
## iter 60 value 11.440561  
## final value 11.440560   
## converged  
## - Fold06: decay=1e-01, size=10   
## + Fold06: decay=1e-03, size=10   
## # weights: 61  
## initial value 17.406273   
## iter 10 value 2.942861  
## iter 20 value 1.295939  
## iter 30 value 0.995213  
## iter 40 value 0.916771  
## iter 50 value 0.872536  
## iter 60 value 0.837505  
## iter 70 value 0.808409  
## iter 80 value 0.796525  
## iter 90 value 0.786495  
## iter 100 value 0.780161  
## final value 0.780161   
## stopped after 100 iterations  
## - Fold06: decay=1e-03, size=10   
## + Fold06: decay=1e-04, size=10   
## # weights: 61  
## initial value 23.596965   
## iter 10 value 5.065447  
## iter 20 value 2.099138  
## iter 30 value 0.507256  
## iter 40 value 0.218186  
## iter 50 value 0.183728  
## iter 60 value 0.164109  
## iter 70 value 0.147400  
## iter 80 value 0.138383  
## iter 90 value 0.133899  
## iter 100 value 0.128986  
## final value 0.128986   
## stopped after 100 iterations  
## - Fold06: decay=1e-04, size=10   
## + Fold06: decay=1e-01, size=15   
## # weights: 91  
## initial value 16.732744   
## iter 10 value 11.488236  
## iter 20 value 11.418556  
## iter 30 value 11.404435  
## iter 40 value 11.401314  
## iter 50 value 11.401061  
## iter 60 value 11.401056  
## final value 11.401056   
## converged  
## - Fold06: decay=1e-01, size=15   
## + Fold06: decay=1e-03, size=15   
## # weights: 91  
## initial value 19.068557   
## iter 10 value 3.511483  
## iter 20 value 1.148323  
## iter 30 value 0.897754  
## iter 40 value 0.817538  
## iter 50 value 0.776170  
## iter 60 value 0.761249  
## iter 70 value 0.750789  
## iter 80 value 0.743365  
## iter 90 value 0.740256  
## iter 100 value 0.738096  
## final value 0.738096   
## stopped after 100 iterations  
## - Fold06: decay=1e-03, size=15   
## + Fold06: decay=1e-04, size=15   
## # weights: 91  
## initial value 16.506762   
## iter 10 value 2.697816  
## iter 20 value 0.254852  
## iter 30 value 0.215479  
## iter 40 value 0.181074  
## iter 50 value 0.158042  
## iter 60 value 0.144989  
## iter 70 value 0.137217  
## iter 80 value 0.131284  
## iter 90 value 0.125651  
## iter 100 value 0.121511  
## final value 0.121511   
## stopped after 100 iterations  
## - Fold06: decay=1e-04, size=15   
## + Fold07: decay=1e-01, size= 5   
## # weights: 31  
## initial value 15.488742   
## iter 10 value 11.489800  
## iter 20 value 11.332739  
## iter 30 value 11.331958  
## final value 11.331950   
## converged  
## - Fold07: decay=1e-01, size= 5   
## + Fold07: decay=1e-03, size= 5   
## # weights: 31  
## initial value 15.675773   
## iter 10 value 3.747285  
## iter 20 value 3.033283  
## iter 30 value 1.486557  
## iter 40 value 1.224692  
## iter 50 value 1.149555  
## iter 60 value 1.110574  
## iter 70 value 1.078456  
## iter 80 value 1.040815  
## iter 90 value 0.987346  
## iter 100 value 0.962649  
## final value 0.962649   
## stopped after 100 iterations  
## - Fold07: decay=1e-03, size= 5   
## + Fold07: decay=1e-04, size= 5   
## # weights: 31  
## initial value 15.625937   
## iter 10 value 2.029061  
## iter 20 value 0.230457  
## iter 30 value 0.213679  
## iter 40 value 0.176517  
## iter 50 value 0.160177  
## iter 60 value 0.149270  
## iter 70 value 0.140738  
## iter 80 value 0.139127  
## iter 90 value 0.137261  
## iter 100 value 0.135852  
## final value 0.135852   
## stopped after 100 iterations  
## - Fold07: decay=1e-04, size= 5   
## + Fold07: decay=1e-01, size=10   
## # weights: 61  
## initial value 14.776149   
## iter 10 value 11.232136  
## iter 20 value 11.215861  
## iter 30 value 11.215457  
## final value 11.215446   
## converged  
## - Fold07: decay=1e-01, size=10   
## + Fold07: decay=1e-03, size=10   
## # weights: 61  
## initial value 15.523901   
## iter 10 value 2.755752  
## iter 20 value 1.239921  
## iter 30 value 0.964696  
## iter 40 value 0.879687  
## iter 50 value 0.845969  
## iter 60 value 0.821767  
## iter 70 value 0.809604  
## iter 80 value 0.802020  
## iter 90 value 0.797201  
## iter 100 value 0.794525  
## final value 0.794525   
## stopped after 100 iterations  
## - Fold07: decay=1e-03, size=10   
## + Fold07: decay=1e-04, size=10   
## # weights: 61  
## initial value 14.481328   
## iter 10 value 3.764301  
## iter 20 value 0.499812  
## iter 30 value 0.269778  
## iter 40 value 0.212519  
## iter 50 value 0.177032  
## iter 60 value 0.152580  
## iter 70 value 0.140460  
## iter 80 value 0.133793  
## iter 90 value 0.129730  
## iter 100 value 0.126414  
## final value 0.126414   
## stopped after 100 iterations  
## - Fold07: decay=1e-04, size=10   
## + Fold07: decay=1e-01, size=15   
## # weights: 91  
## initial value 16.996118   
## iter 10 value 11.254491  
## iter 20 value 11.196883  
## iter 30 value 11.190047  
## iter 40 value 11.183018  
## iter 50 value 11.181960  
## iter 60 value 11.181901  
## final value 11.181900   
## converged  
## - Fold07: decay=1e-01, size=15   
## + Fold07: decay=1e-03, size=15   
## # weights: 91  
## initial value 15.805018   
## iter 10 value 3.423320  
## iter 20 value 1.358095  
## iter 30 value 0.890079  
## iter 40 value 0.805147  
## iter 50 value 0.764819  
## iter 60 value 0.750808  
## iter 70 value 0.743157  
## iter 80 value 0.736656  
## iter 90 value 0.732456  
## iter 100 value 0.727914  
## final value 0.727914   
## stopped after 100 iterations  
## - Fold07: decay=1e-03, size=15   
## + Fold07: decay=1e-04, size=15   
## # weights: 91  
## initial value 14.278178   
## iter 10 value 5.449570  
## iter 20 value 0.481268  
## iter 30 value 0.367627  
## iter 40 value 0.250346  
## iter 50 value 0.203403  
## iter 60 value 0.170369  
## iter 70 value 0.152151  
## iter 80 value 0.138554  
## iter 90 value 0.128446  
## iter 100 value 0.124841  
## final value 0.124841   
## stopped after 100 iterations  
## - Fold07: decay=1e-04, size=15   
## + Fold08: decay=1e-01, size= 5   
## # weights: 31  
## initial value 15.757042   
## iter 10 value 11.679111  
## iter 20 value 11.514636  
## iter 30 value 11.512870  
## iter 40 value 11.512201  
## final value 11.512201   
## converged  
## - Fold08: decay=1e-01, size= 5   
## + Fold08: decay=1e-03, size= 5   
## # weights: 31  
## initial value 18.518144   
## iter 10 value 5.944565  
## iter 20 value 1.950217  
## iter 30 value 1.104293  
## iter 40 value 0.976695  
## iter 50 value 0.925076  
## iter 60 value 0.890824  
## iter 70 value 0.868601  
## iter 80 value 0.850730  
## iter 90 value 0.844656  
## iter 100 value 0.843981  
## final value 0.843981   
## stopped after 100 iterations  
## - Fold08: decay=1e-03, size= 5   
## + Fold08: decay=1e-04, size= 5   
## # weights: 31  
## initial value 18.589263   
## iter 10 value 4.234316  
## iter 20 value 2.575029  
## iter 30 value 1.747812  
## iter 40 value 0.196672  
## iter 50 value 0.155729  
## iter 60 value 0.149724  
## iter 70 value 0.145876  
## iter 80 value 0.139316  
## iter 90 value 0.136614  
## iter 100 value 0.135654  
## final value 0.135654   
## stopped after 100 iterations  
## - Fold08: decay=1e-04, size= 5   
## + Fold08: decay=1e-01, size=10   
## # weights: 61  
## initial value 19.233719   
## iter 10 value 11.442056  
## iter 20 value 11.311903  
## iter 30 value 11.270905  
## iter 40 value 11.256452  
## iter 50 value 11.255942  
## final value 11.255934   
## converged  
## - Fold08: decay=1e-01, size=10   
## + Fold08: decay=1e-03, size=10   
## # weights: 61  
## initial value 18.036078   
## iter 10 value 3.547239  
## iter 20 value 1.630805  
## iter 30 value 1.180236  
## iter 40 value 1.033050  
## iter 50 value 0.922597  
## iter 60 value 0.871013  
## iter 70 value 0.836119  
## iter 80 value 0.817728  
## iter 90 value 0.807132  
## iter 100 value 0.800182  
## final value 0.800182   
## stopped after 100 iterations  
## - Fold08: decay=1e-03, size=10   
## + Fold08: decay=1e-04, size=10   
## # weights: 61  
## initial value 16.503272   
## iter 10 value 4.056321  
## iter 20 value 0.586806  
## iter 30 value 0.263269  
## iter 40 value 0.199142  
## iter 50 value 0.174598  
## iter 60 value 0.161237  
## iter 70 value 0.146061  
## iter 80 value 0.137133  
## iter 90 value 0.131132  
## iter 100 value 0.128199  
## final value 0.128199   
## stopped after 100 iterations  
## - Fold08: decay=1e-04, size=10   
## + Fold08: decay=1e-01, size=15   
## # weights: 91  
## initial value 17.515440   
## iter 10 value 11.286421  
## iter 20 value 11.244651  
## iter 30 value 11.235014  
## iter 40 value 11.233456  
## iter 50 value 11.230333  
## iter 60 value 11.229812  
## iter 70 value 11.229745  
## final value 11.229743   
## converged  
## - Fold08: decay=1e-01, size=15   
## + Fold08: decay=1e-03, size=15   
## # weights: 91  
## initial value 17.631164   
## iter 10 value 4.966939  
## iter 20 value 2.120413  
## iter 30 value 1.200804  
## iter 40 value 1.018922  
## iter 50 value 0.898676  
## iter 60 value 0.853488  
## iter 70 value 0.830508  
## iter 80 value 0.808460  
## iter 90 value 0.792359  
## iter 100 value 0.782730  
## final value 0.782730   
## stopped after 100 iterations  
## - Fold08: decay=1e-03, size=15   
## + Fold08: decay=1e-04, size=15   
## # weights: 91  
## initial value 17.671170   
## iter 10 value 3.778227  
## iter 20 value 1.201100  
## iter 30 value 0.490504  
## iter 40 value 0.333493  
## iter 50 value 0.239204  
## iter 60 value 0.178794  
## iter 70 value 0.165791  
## iter 80 value 0.154450  
## iter 90 value 0.143900  
## iter 100 value 0.136600  
## final value 0.136600   
## stopped after 100 iterations  
## - Fold08: decay=1e-04, size=15   
## + Fold09: decay=1e-01, size= 5   
## # weights: 31  
## initial value 17.763316   
## iter 10 value 11.477736  
## iter 20 value 11.275114  
## iter 30 value 11.271241  
## iter 40 value 11.270661  
## final value 11.270659   
## converged  
## - Fold09: decay=1e-01, size= 5   
## + Fold09: decay=1e-03, size= 5   
## # weights: 31  
## initial value 15.857032   
## iter 10 value 2.898958  
## iter 20 value 1.690786  
## iter 30 value 1.167766  
## iter 40 value 1.063476  
## iter 50 value 1.008026  
## iter 60 value 0.975809  
## iter 70 value 0.963316  
## iter 80 value 0.950064  
## iter 90 value 0.936898  
## iter 100 value 0.928623  
## final value 0.928623   
## stopped after 100 iterations  
## - Fold09: decay=1e-03, size= 5   
## + Fold09: decay=1e-04, size= 5   
## # weights: 31  
## initial value 15.651540   
## iter 10 value 2.849263  
## iter 20 value 1.319801  
## iter 30 value 0.341788  
## iter 40 value 0.213087  
## iter 50 value 0.172353  
## iter 60 value 0.149680  
## iter 70 value 0.143903  
## iter 80 value 0.140476  
## iter 90 value 0.135649  
## iter 100 value 0.134071  
## final value 0.134071   
## stopped after 100 iterations  
## - Fold09: decay=1e-04, size= 5   
## + Fold09: decay=1e-01, size=10   
## # weights: 61  
## initial value 17.925257   
## iter 10 value 11.385843  
## iter 20 value 11.326150  
## iter 30 value 11.320414  
## iter 40 value 11.319681  
## final value 11.319665   
## converged  
## - Fold09: decay=1e-01, size=10   
## + Fold09: decay=1e-03, size=10   
## # weights: 61  
## initial value 15.987634   
## iter 10 value 2.680122  
## iter 20 value 1.147870  
## iter 30 value 0.859135  
## iter 40 value 0.793098  
## iter 50 value 0.772797  
## iter 60 value 0.762012  
## iter 70 value 0.754710  
## iter 80 value 0.751058  
## iter 90 value 0.748077  
## iter 100 value 0.745256  
## final value 0.745256   
## stopped after 100 iterations  
## - Fold09: decay=1e-03, size=10   
## + Fold09: decay=1e-04, size=10   
## # weights: 61  
## initial value 16.463084   
## iter 10 value 3.012553  
## iter 20 value 0.721318  
## iter 30 value 0.362394  
## iter 40 value 0.207916  
## iter 50 value 0.172416  
## iter 60 value 0.153227  
## iter 70 value 0.144286  
## iter 80 value 0.138946  
## iter 90 value 0.132959  
## iter 100 value 0.128673  
## final value 0.128673   
## stopped after 100 iterations  
## - Fold09: decay=1e-04, size=10   
## + Fold09: decay=1e-01, size=15   
## # weights: 91  
## initial value 16.921787   
## iter 10 value 11.341938  
## iter 20 value 11.179672  
## iter 30 value 11.170251  
## iter 40 value 11.165696  
## iter 50 value 11.164723  
## iter 60 value 11.164477  
## final value 11.164456   
## converged  
## - Fold09: decay=1e-01, size=15   
## + Fold09: decay=1e-03, size=15   
## # weights: 91  
## initial value 14.655588   
## iter 10 value 2.203875  
## iter 20 value 1.043305  
## iter 30 value 0.875375  
## iter 40 value 0.818290  
## iter 50 value 0.798920  
## iter 60 value 0.776868  
## iter 70 value 0.767700  
## iter 80 value 0.762264  
## iter 90 value 0.754748  
## iter 100 value 0.747408  
## final value 0.747408   
## stopped after 100 iterations  
## - Fold09: decay=1e-03, size=15   
## + Fold09: decay=1e-04, size=15   
## # weights: 91  
## initial value 14.607209   
## iter 10 value 2.995785  
## iter 20 value 0.455217  
## iter 30 value 0.269821  
## iter 40 value 0.189206  
## iter 50 value 0.159700  
## iter 60 value 0.142118  
## iter 70 value 0.132737  
## iter 80 value 0.128885  
## iter 90 value 0.123281  
## iter 100 value 0.119992  
## final value 0.119992   
## stopped after 100 iterations  
## - Fold09: decay=1e-04, size=15   
## + Fold10: decay=1e-01, size= 5   
## # weights: 31  
## initial value 16.526972   
## iter 10 value 10.523035  
## iter 20 value 10.411866  
## iter 30 value 10.390973  
## iter 40 value 10.368453  
## final value 10.368411   
## converged  
## - Fold10: decay=1e-01, size= 5   
## + Fold10: decay=1e-03, size= 5   
## # weights: 31  
## initial value 16.057709   
## iter 10 value 2.081071  
## iter 20 value 1.117097  
## iter 30 value 0.868387  
## iter 40 value 0.703801  
## iter 50 value 0.649799  
## iter 60 value 0.638187  
## iter 70 value 0.630520  
## iter 80 value 0.628882  
## iter 90 value 0.627819  
## iter 100 value 0.627231  
## final value 0.627231   
## stopped after 100 iterations  
## - Fold10: decay=1e-03, size= 5   
## + Fold10: decay=1e-04, size= 5   
## # weights: 31  
## initial value 16.775608   
## iter 10 value 2.219360  
## iter 20 value 0.187636  
## iter 30 value 0.170001  
## iter 40 value 0.142231  
## iter 50 value 0.134139  
## iter 60 value 0.125638  
## iter 70 value 0.118861  
## iter 80 value 0.115809  
## iter 90 value 0.113946  
## iter 100 value 0.112120  
## final value 0.112120   
## stopped after 100 iterations  
## - Fold10: decay=1e-04, size= 5   
## + Fold10: decay=1e-01, size=10   
## # weights: 61  
## initial value 22.043211   
## iter 10 value 10.362508  
## iter 20 value 10.282476  
## iter 30 value 10.255972  
## iter 40 value 10.253316  
## final value 10.253269   
## converged  
## - Fold10: decay=1e-01, size=10   
## + Fold10: decay=1e-03, size=10   
## # weights: 61  
## initial value 16.667757   
## iter 10 value 1.387056  
## iter 20 value 0.954871  
## iter 30 value 0.675464  
## iter 40 value 0.602962  
## iter 50 value 0.581909  
## iter 60 value 0.575585  
## iter 70 value 0.572440  
## iter 80 value 0.570677  
## iter 90 value 0.569375  
## iter 100 value 0.568678  
## final value 0.568678   
## stopped after 100 iterations  
## - Fold10: decay=1e-03, size=10   
## + Fold10: decay=1e-04, size=10   
## # weights: 61  
## initial value 20.495746   
## iter 10 value 0.765135  
## iter 20 value 0.145943  
## iter 30 value 0.132415  
## iter 40 value 0.123581  
## iter 50 value 0.113326  
## iter 60 value 0.102084  
## iter 70 value 0.097436  
## iter 80 value 0.094227  
## iter 90 value 0.092362  
## iter 100 value 0.088970  
## final value 0.088970   
## stopped after 100 iterations  
## - Fold10: decay=1e-04, size=10   
## + Fold10: decay=1e-01, size=15   
## # weights: 91  
## initial value 19.788989   
## iter 10 value 10.387478  
## iter 20 value 10.252185  
## iter 30 value 10.231982  
## iter 40 value 10.221094  
## iter 50 value 10.219644  
## iter 60 value 10.219382  
## iter 70 value 10.219371  
## final value 10.219370   
## converged  
## - Fold10: decay=1e-01, size=15   
## + Fold10: decay=1e-03, size=15   
## # weights: 91  
## initial value 16.644731   
## iter 10 value 1.556302  
## iter 20 value 0.798928  
## iter 30 value 0.650339  
## iter 40 value 0.619168  
## iter 50 value 0.597837  
## iter 60 value 0.586383  
## iter 70 value 0.580026  
## iter 80 value 0.571855  
## iter 90 value 0.566335  
## iter 100 value 0.562989  
## final value 0.562989   
## stopped after 100 iterations  
## - Fold10: decay=1e-03, size=15   
## + Fold10: decay=1e-04, size=15   
## # weights: 91  
## initial value 16.277162   
## iter 10 value 1.367870  
## iter 20 value 0.166497  
## iter 30 value 0.143880  
## iter 40 value 0.124156  
## iter 50 value 0.114076  
## iter 60 value 0.101780  
## iter 70 value 0.097438  
## iter 80 value 0.093220  
## iter 90 value 0.090319  
## iter 100 value 0.088525  
## final value 0.088525   
## stopped after 100 iterations  
## - Fold10: decay=1e-04, size=15   
## Aggregating results  
## Selecting tuning parameters  
## Fitting size = 5, decay = 1e-04 on full training set  
## # weights: 31  
## initial value 17.454377   
## iter 10 value 1.836314  
## iter 20 value 0.394756  
## iter 30 value 0.314563  
## iter 40 value 0.256429  
## iter 50 value 0.209456  
## iter 60 value 0.178045  
## iter 70 value 0.167361  
## iter 80 value 0.160537  
## iter 90 value 0.153551  
## iter 100 value 0.146776  
## final value 0.146776   
## stopped after 100 iterations

#### view model results

print(nn\_model)

## Neural Network   
##   
## 25 samples  
## 4 predictor  
## 2 classes: 'X0', 'X1'   
##   
## Pre-processing: centered (4), scaled (4)   
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 23, 22, 23, 23, 23, 23, ...   
## Resampling results across tuning parameters:  
##   
## decay size ROC Sens Spec  
## 1e-04 5 0.750 0.75 0.45  
## 1e-04 10 0.650 0.75 0.50  
## 1e-04 15 0.700 0.75 0.45  
## 1e-03 5 0.475 0.65 0.45  
## 1e-03 10 0.650 0.75 0.50  
## 1e-03 15 0.650 0.75 0.50  
## 1e-01 5 0.650 0.55 0.65  
## 1e-01 10 0.650 0.55 0.65  
## 1e-01 15 0.650 0.55 0.65  
##   
## ROC was used to select the optimal model using the largest value.  
## The final values used for the model were size = 5 and decay = 1e-04.

plot(nn\_model)



#### print variable importance

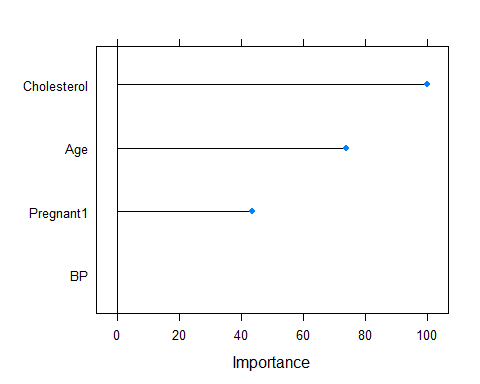
**varImp** is a generic method for calculating variable importance for objects produced by train and method specific methods.

In Neural Network models, the method used is based on Gevrey et al (2003), which uses combinations of the absolute values of the weights. For classification models, the class-specific importances will be the same.

varImp(nn\_model)

## nnet variable importance  
##   
## Overall  
## Cholesterol 100.00  
## Age 73.95  
## Pregnant1 43.60  
## BP 0.00

plot(varImp(nn\_model))



**Observation**

* We observe that the variables are important in classifying Durg.Reaction:

1. Cholesterol
2. Age
3. Pregnant women

### Measuring Model Performance

#### check prediction on Test data

test.pred <- predict(nn\_model, newdata=test.data)  
test.confusion.m <- confusionMatrix(test.pred, test.data$Drug.Reaction)  
test.confusion.m

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction X0 X1  
## X0 2 1  
## X1 0 2  
##   
## Accuracy : 0.8   
## 95% CI : (0.2836, 0.9949)  
## No Information Rate : 0.6   
## P-Value [Acc > NIR] : 0.337   
##   
## Kappa : 0.6154   
## Mcnemar's Test P-Value : 1.000   
##   
## Sensitivity : 1.0000   
## Specificity : 0.6667   
## Pos Pred Value : 0.6667   
## Neg Pred Value : 1.0000   
## Prevalence : 0.4000   
## Detection Rate : 0.4000   
## Detection Prevalence : 0.6000   
## Balanced Accuracy : 0.8333   
##   
## 'Positive' Class : X0   
##

**Observation**

|  |  |
| --- | --- |
| Measure | Value |
| Accuracy of the model | 0.8 |
| Kappa | 0.6154 |
| Sensitivity | 1 |
| Specificity | 0.6667 |

### Hence, the Model performance is good.

### Add scoring data on the test data set

test.data$predict.class <-  
 predict(nn\_model, test.data, type = "raw")  
test.data$predict.score <-  
 predict(nn\_model, test.data, type = "prob")

#### Deciling function to create rank order matrix and to understand the accuracy of the nn model

#### make deciling function

decile <- function(x) {  
 deciles <- vector(length = 10)  
 for (i in seq(0.1, 1, .1)) {  
 deciles[i \* 10] <- quantile(x, i, na.rm = T)  
 }  
   
 return (ifelse(x < deciles[1], 1,  
 ifelse(  
 x < deciles[2], 2,  
 ifelse(x < deciles[3], 3,  
 ifelse(  
 x < deciles[4], 4,  
 ifelse(x < deciles[5], 5,  
 ifelse(  
 x < deciles[6], 6,  
 ifelse(x < deciles[7], 7,  
 ifelse(x < deciles[8], 8,  
 ifelse(x < deciles[9], 9, 10)))  
 ))  
 ))  
 )))  
}

### deciling

test.data$deciles <- decile(test.data$predict.score[, 2])

#### Ranking code

library(data.table)  
# make a copy of final data for any unseen error and calculate KS statistics  
tmp\_DT <- data.table(test.data)  
rank <- tmp\_DT[, list(  
 cnt = length(Drug.Reaction),  
 cnt\_resp = sum(ifelse(Drug.Reaction == 'X1', 1, 0)),  
 cnt\_non\_resp = sum(ifelse(Drug.Reaction == 'X0', 1, 0))  
) ,  
by = deciles][order(-deciles)]  
rank$rrate <- round(rank$cnt\_resp \* 100 / rank$cnt, 2)  
  
rank$cum\_resp <- cumsum(rank$cnt\_resp)  
rank$cum\_non\_resp <- cumsum(rank$cnt\_non\_resp)  
rank$cum\_rel\_resp <- round(rank$cum\_resp / sum(rank$cnt\_resp), 2)  
  
rank$cum\_rel\_non\_resp <-  
 round(rank$cum\_non\_resp / sum(rank$cnt\_non\_resp), 2)  
  
rank$ks <- abs(rank$cum\_rel\_resp - rank$cum\_rel\_non\_resp)  
  
View(rank)

#### Plot ROC and AUC Curve

library(ROCR)

## Loading required package: gplots

##   
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':  
##   
## lowess

library(pROC)

### Plot ROC and Compute AUC for predicting Class with the model

prob <- predict(nn\_model, newdata = test.data, prob = TRUE)  
  
rocCurve<-roc(response=test.data$Drug.Reaction,predictor=test.data$predict.score[,2]  
 ,levels = rev(levels(test.data$Drug.Reaction)))  
auc(rocCurve)

## Area under the curve: 1

#### Area under the curve: 1

#### Confidence Intervals

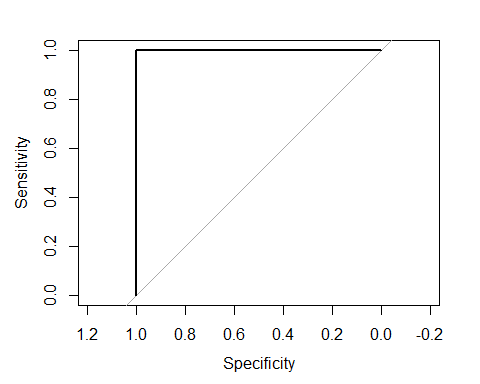
ci.auc(rocCurve)

## Warning in ci.auc.roc(rocCurve): ci.auc() of a ROC curve with AUC == 1 is  
## always 1-1 and can be misleading.

## 95% CI: 1-1 (DeLong)

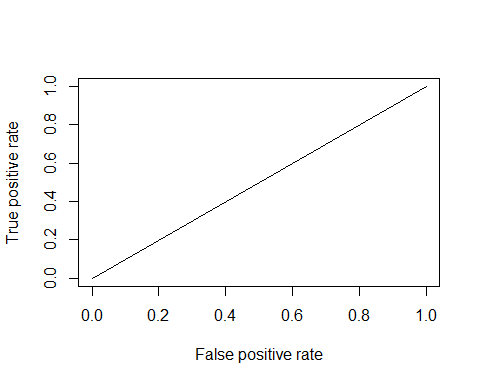
#### 95% CI: 1 - 1 (DeLong)

plot.roc(rocCurve)



### Plot True/False positive rates

predvec <- ifelse(prob == "Yes", 1, 0)  
realvec <- ifelse(test.data$Drug.Reaction == "X1", 1, 0)  
pr <- prediction(predvec, realvec)  
  
pref <- performance(pr, "tpr", "fpr")  
plot(pref)



KS <- max(attr(pref, 'y.values')[[1]] - attr(pref, 'x.values')[[1]])  
KS

## [1] 0

## [1] 0  
# AUC calculation  
auc <- performance(pr, measure = "auc")  
auc <- auc@y.values[[1]]  
auc

## [1] 0.5

### Observation

Area under the curve is 0.5

### Conclusion

### Problem statement

To understand the characteristics that cause some of women to have an adverse reaction to a particular drug

### Conclusion

* We observe that the characteristics that cause some of women to have an adverse reaction to a particular drug are:

1. Cholesterol
2. Age
3. Pregnant women