Основи прикладного математичного моделювання в R. Нейронні мережі. Частина 2

Юрій Клебан, IntelSoft Tech

# Вступ

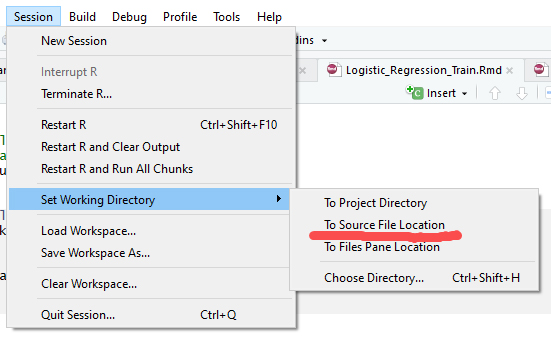
Даний матеріал містить інформацію у межах курсу “Основи прикладного математичного моделювання в R”. Курс створено для працівників банків. У даному навчальному матеріалі використано дані про відтік клієнтів банку Churn Modeling. Увага, дані не є оригінальними і є зміненими для потреб навчального процесу.

Джерела:

* <https://www.kaggle.com/barelydedicated/bank-customer-churn-modeling>

Увага! Частина значень у Вашому дослідженні може відрізнятися, адже це залежить від сформованих тренувальної та тестової вибірок.

# Етап 1. Попередній огляд даних

Для початку варто встановити робочу папку. Це можна зробити за допомогою команди setwd(шлях\_до\_каталогу) або через меню *R Studio*: 

Для очистки сесії від непотрібних даних використайте команду rm:

rm(list = ls()) #видаляє усі змінні

Завантажимо файли з даними:

customers <- read.csv("data/customers.csv")  
products <- read.csv("data/products.csv")  
cards <- read.csv("data/cards.csv")

Цей датасет містить такі стовпці:

* *RowNumber* - номер рядка.
* *CustomerId* - ідентифікатор клієнта.
* *Surname* - прізвище клієнта.
* *CreditScore* - кредитний рейтинг клієнта.
* *Geography* - регіон.
* *Gender* - стать.
* *Age* - вік.
* *Tenure* - час обслуговування цього клієнта в банку.
* *IsActiveMember* - активний клієнт, виконує операції.
* *EstimatedSalary* - заробітна плата.
* *Exited* - залишив/не залишив банк.

Інформація про клієнтів:

str(customers)

## 'data.frame': 10000 obs. of 11 variables:  
## $ RowNumber : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ CustomerId : int 15634602 15647311 15619304 15701354 15737888 15574012 15592531 15656148 15792365 15592389 ...  
## $ Surname : Factor w/ 2932 levels "Abazu","Abbie",..: 1116 1178 2041 290 1823 538 178 2001 1147 1082 ...  
## $ CreditScore : int 619 608 502 699 850 645 822 376 501 684 ...  
## $ Geography : Factor w/ 3 levels "France","Germany",..: 1 3 1 1 3 3 1 2 1 1 ...  
## $ Gender : Factor w/ 2 levels "Female","Male": 1 1 1 1 1 2 2 1 2 2 ...  
## $ Age : int 42 41 42 39 43 44 50 29 44 27 ...  
## $ Tenure : int 2 1 8 1 2 8 7 4 4 2 ...  
## $ IsActiveMember : int 1 1 0 0 1 0 1 0 1 1 ...  
## $ EstimatedSalary: num 101349 112543 113932 93827 79084 ...  
## $ Exited : int 1 0 1 0 0 1 0 1 0 0 ...

head(customers)

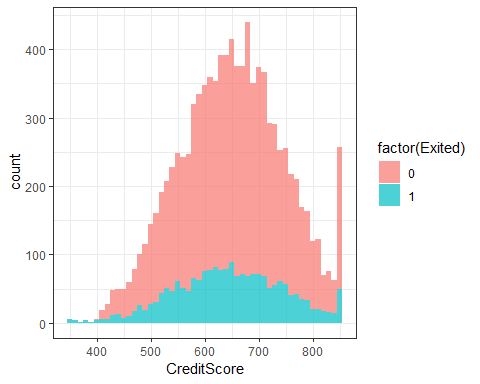
## RowNumber CustomerId Surname CreditScore Geography Gender Age Tenure  
## 1 1 15634602 Hargrave 619 France Female 42 2  
## 2 2 15647311 Hill 608 Spain Female 41 1  
## 3 3 15619304 Onio 502 France Female 42 8  
## 4 4 15701354 Boni 699 France Female 39 1  
## 5 5 15737888 Mitchell 850 Spain Female 43 2  
## 6 6 15574012 Chu 645 Spain Male 44 8  
## IsActiveMember EstimatedSalary Exited  
## 1 1 101348.88 1  
## 2 1 112542.58 0  
## 3 0 113931.57 1  
## 4 0 93826.63 0  
## 5 1 79084.10 0  
## 6 0 149756.71 1

Оглянемо дані колекції customers візуально та за допомогою крос-таблиць:

library(gmodels)  
library(ggplot2)

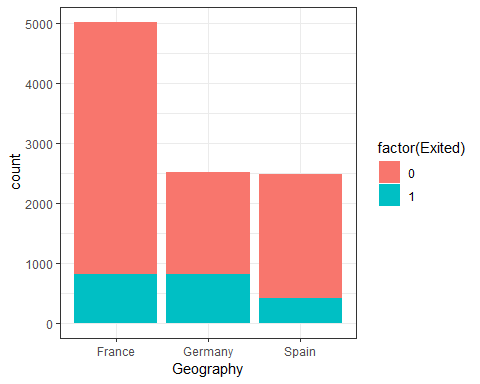
Кредитний скоринг (CreditScore):

ggplot(customers, aes(x=CreditScore, fill=factor(Exited))) +   
 geom\_histogram(binwidth = 10, alpha=0.7) + theme\_bw()



Регіон/країна (Geography):

ggplot(customers, aes(x=Geography, fill=factor(Exited))) +   
 geom\_bar(position = "stack") + theme\_bw()

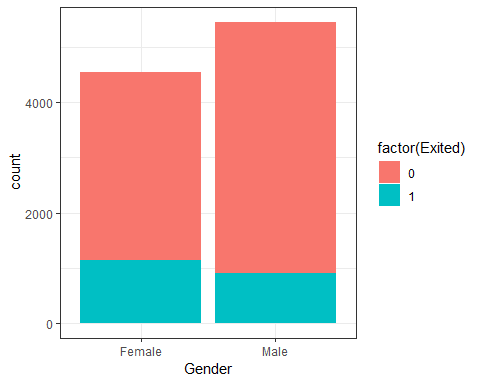


CrossTable(customers$Geography, customers$Exited)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | Chi-square contribution |  
## | N / Row Total |  
## | N / Col Total |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 10000   
##   
##   
## | customers$Exited   
## customers$Geography | 0 | 1 | Row Total |   
## --------------------|-----------|-----------|-----------|  
## France | 4204 | 810 | 5014 |   
## | 11.188 | 43.736 | |   
## | 0.838 | 0.162 | 0.501 |   
## | 0.528 | 0.398 | |   
## | 0.420 | 0.081 | |   
## --------------------|-----------|-----------|-----------|  
## Germany | 1695 | 814 | 2509 |   
## | 45.927 | 179.537 | |   
## | 0.676 | 0.324 | 0.251 |   
## | 0.213 | 0.400 | |   
## | 0.170 | 0.081 | |   
## --------------------|-----------|-----------|-----------|  
## Spain | 2064 | 413 | 2477 |   
## | 4.251 | 16.617 | |   
## | 0.833 | 0.167 | 0.248 |   
## | 0.259 | 0.203 | |   
## | 0.206 | 0.041 | |   
## --------------------|-----------|-----------|-----------|  
## Column Total | 7963 | 2037 | 10000 |   
## | 0.796 | 0.204 | |   
## --------------------|-----------|-----------|-----------|  
##   
##

Стать (Gender):

ggplot(customers, aes(x=Gender, fill=factor(Exited))) +   
 geom\_bar(position = "stack") + theme\_bw()

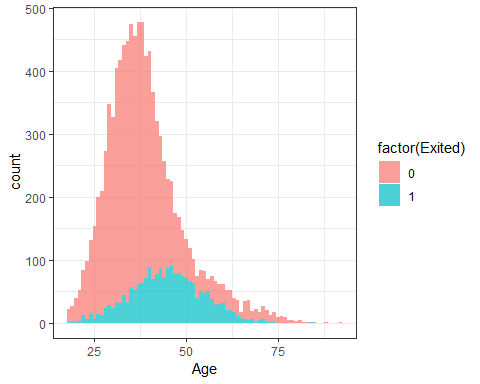


CrossTable(customers$Gender, customers$Exited)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | Chi-square contribution |  
## | N / Row Total |  
## | N / Col Total |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 10000   
##   
##   
## | customers$Exited   
## customers$Gender | 0 | 1 | Row Total |   
## -----------------|-----------|-----------|-----------|  
## Female | 3404 | 1139 | 4543 |   
## | 12.611 | 49.298 | |   
## | 0.749 | 0.251 | 0.454 |   
## | 0.427 | 0.559 | |   
## | 0.340 | 0.114 | |   
## -----------------|-----------|-----------|-----------|  
## Male | 4559 | 898 | 5457 |   
## | 10.499 | 41.041 | |   
## | 0.835 | 0.165 | 0.546 |   
## | 0.573 | 0.441 | |   
## | 0.456 | 0.090 | |   
## -----------------|-----------|-----------|-----------|  
## Column Total | 7963 | 2037 | 10000 |   
## | 0.796 | 0.204 | |   
## -----------------|-----------|-----------|-----------|  
##   
##

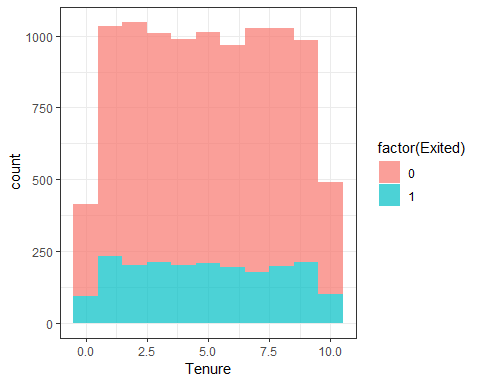
Вік (Age):

ggplot(customers, aes(x=Age, fill=factor(Exited))) +   
 geom\_histogram(binwidth = 1, alpha=0.7) + theme\_bw()



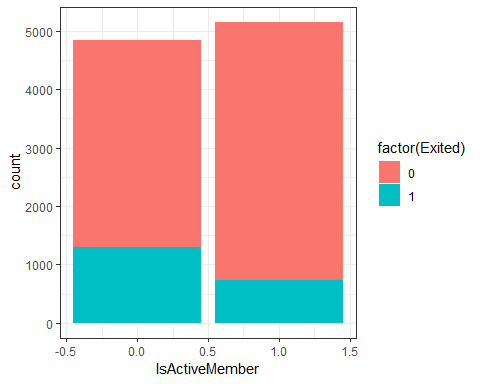
Час обслуговування клієнта (Tenure):

ggplot(customers, aes(x=Tenure, fill=factor(Exited))) +   
 geom\_histogram(binwidth = 1, alpha=0.7) + theme\_bw()



Активність (IsActiveMember):

ggplot(customers, aes(x=IsActiveMember, fill=factor(Exited))) +   
 geom\_bar(position = "stack") + theme\_bw()

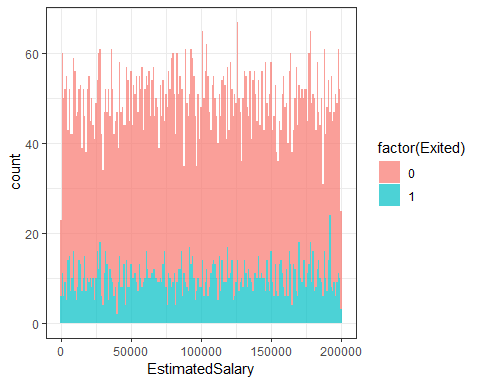


CrossTable(customers$IsActiveMember, customers$Exited)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | Chi-square contribution |  
## | N / Row Total |  
## | N / Col Total |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 10000   
##   
##   
## | customers$Exited   
## customers$IsActiveMember | 0 | 1 | Row Total |   
## -------------------------|-----------|-----------|-----------|  
## 0 | 3547 | 1302 | 4849 |   
## | 25.577 | 99.984 | |   
## | 0.731 | 0.269 | 0.485 |   
## | 0.445 | 0.639 | |   
## | 0.355 | 0.130 | |   
## -------------------------|-----------|-----------|-----------|  
## 1 | 4416 | 735 | 5151 |   
## | 24.077 | 94.122 | |   
## | 0.857 | 0.143 | 0.515 |   
## | 0.555 | 0.361 | |   
## | 0.442 | 0.073 | |   
## -------------------------|-----------|-----------|-----------|  
## Column Total | 7963 | 2037 | 10000 |   
## | 0.796 | 0.204 | |   
## -------------------------|-----------|-----------|-----------|  
##   
##

Заробітна плата (EstimatedSalary):

ggplot(customers, aes(x=EstimatedSalary, fill=factor(Exited))) +   
 geom\_histogram(binwidth = 1000, alpha=0.7) + theme\_bw()



Exited:

CrossTable(customers$Exited)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 10000   
##   
##   
## | 0 | 1 |   
## |-----------|-----------|  
## | 7963 | 2037 |   
## | 0.796 | 0.204 |   
## |-----------|-----------|  
##   
##   
##   
##

Інформація про карти клієнта:

str(cards)

## 'data.frame': 13545 obs. of 4 variables:  
## $ CustomerId : int 15634602 15647311 15647311 15619304 15619304 15701354 15701354 15737888 15737888 15574012 ...  
## $ CardNo : int 684618 357092 802678 888594 987103 507476 928960 370210 935036 581042 ...  
## $ IsCreditCard: int 1 0 0 1 0 0 0 1 0 1 ...  
## $ Balance : num 5 41922 41913 79846 79859 ...

head(cards)

## CustomerId CardNo IsCreditCard Balance  
## 1 15634602 684618 1 5.00  
## 2 15647311 357092 0 41921.93  
## 3 15647311 802678 0 41912.93  
## 4 15619304 888594 1 79846.40  
## 5 15619304 987103 0 79859.40  
## 6 15701354 507476 0 28.00

summary(cards)

## CustomerId CardNo IsCreditCard Balance   
## Min. :15565701 Min. :100231 Min. :0.0000 Min. : 0   
## 1st Qu.:15628272 1st Qu.:323587 1st Qu.:0.0000 1st Qu.: 24   
## Median :15691011 Median :545073 Median :0.0000 Median : 52991   
## Mean :15690848 Mean :548414 Mean :0.4688 Mean : 51096   
## 3rd Qu.:15752816 3rd Qu.:774943 3rd Qu.:1.0000 3rd Qu.: 77276   
## Max. :15815690 Max. :999985 Max. :1.0000 Max. :221549

Інформація про продукти клієнта:

str(products)

## 'data.frame': 13862 obs. of 2 variables:  
## $ CustomerId : int 15634602 15647311 15619304 15619304 15619304 15701354 15701354 15737888 15574012 15574012 ...  
## $ ProductName: Factor w/ 4 levels "PROD\_1","PROD\_2",..: 1 1 1 2 3 1 2 1 1 2 ...

head(products)

## CustomerId ProductName  
## 1 15634602 PROD\_1  
## 2 15647311 PROD\_1  
## 3 15619304 PROD\_1  
## 4 15619304 PROD\_2  
## 5 15619304 PROD\_3  
## 6 15701354 PROD\_1

# Етап 2. Формування набору додаткових змінних

Сформуємо додаткові змінні на основі наявних даних:

* *Balance* - сума по усіх картах клієнат.
* *NumOfProducts* - кількість продуктів банку, які використовує клієнт.
* *HasCreditCard* - dummy-змінна, наявність кредитної карти у клієнта.

### Обчислимо значення NumOfProducts

customers$NumOfProducts <- c(0)  
  
for(i in 1:nrow(customers))  
{  
 id <- customers$CustomerId[i]  
 prods <- subset(products, CustomerId == id)  
 customers$NumOfProducts[i] <- nrow(prods)  
}

### Обчислимо значення HasCreditCard

customers$HasCreditCard <- c(0)  
  
for(i in 1:nrow(customers))  
{  
 id <- customers$CustomerId[i]  
 crds <- subset(cards, CustomerId == id & IsCreditCard == 1)  
 customers$HasCreditCard[i] <- nrow(crds)  
}

### Обчислимо значення Balance

customers$Balance <- c(0)  
  
for(i in 1:nrow(customers))  
{  
 id <- customers$CustomerId[i]  
 crds <- subset(cards, CustomerId == id)  
 customers$Balance[i] <- round(sum(crds$Balance))  
}

Для того аби прискорити процес обчислень можна об’єднати все в 1 цикл:

customers$NumOfProducts <- c(0)  
customers$HasCreditCard <- c(0)  
customers$Balance <- c(0)  
  
for(i in 1:nrow(customers))  
{  
 # get customer id  
 id <- customers$CustomerId[i]  
   
 #get products of current customer  
 prods <- subset(products, CustomerId == id)  
   
 # set products count. DONE!  
 customers$NumOfProducts[i] <- nrow(prods)  
   
 # get customer cards  
 crds <- subset(cards, CustomerId == id)  
   
 # set customer balance as sum card balances  
 customers$Balance[i] <- round(sum(crds$Balance))  
   
 customers$HasCreditCard[i] <- nrow(crds[crds$IsCreditCard == 1, ])  
}

Переглянемо інформацію про клієнтів (customers):

str(customers)

## 'data.frame': 10000 obs. of 14 variables:  
## $ RowNumber : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ CustomerId : int 15634602 15647311 15619304 15701354 15737888 15574012 15592531 15656148 15792365 15592389 ...  
## $ Surname : Factor w/ 2932 levels "Abazu","Abbie",..: 1116 1178 2041 290 1823 538 178 2001 1147 1082 ...  
## $ CreditScore : int 619 608 502 699 850 645 822 376 501 684 ...  
## $ Geography : Factor w/ 3 levels "France","Germany",..: 1 3 1 1 3 3 1 2 1 1 ...  
## $ Gender : Factor w/ 2 levels "Female","Male": 1 1 1 1 1 2 2 1 2 2 ...  
## $ Age : int 42 41 42 39 43 44 50 29 44 27 ...  
## $ Tenure : int 2 1 8 1 2 8 7 4 4 2 ...  
## $ IsActiveMember : int 1 1 0 0 1 0 1 0 1 1 ...  
## $ EstimatedSalary: num 101349 112543 113932 93827 79084 ...  
## $ Exited : int 1 0 1 0 0 1 0 1 0 0 ...  
## $ NumOfProducts : num 1 1 3 2 1 2 2 4 2 1 ...  
## $ HasCreditCard : num 1 0 1 0 1 1 1 1 0 1 ...  
## $ Balance : num 5 83835 159706 47 125571 ...

Видалимо зайві дані з пам’яті:

rm(cards, crds, prods, products, i, id)

Видалимо зайві стовпці з датасету:

customers$RowNumber <- NULL  
customers$CustomerId <- NULL  
customers$Surname <- NULL  
  
# OR  
#customers <- customers[,4:ncol(customers)]

Для нормалізації усієї вибірки за 1 раз перетворимо фактори до числового типу:

customers$Geography <- as.numeric(customers$Geography)  
customers$Gender <- as.numeric(customers$Gender)

# Етап 3. Нормалізація даних та генерація вибірок

Використовуючи функцію scale() нормалізуємо дані:

scaled <- scale(customers[-8], center = TRUE)  
scaled <- cbind(scaled, customers[8])  
View(scaled)

Розділимо вибірку на тестову та тренувальну за допомогою пакету caTools та функції sample.split():

library(caTools)  
set.seed(2019)  
split <- sample.split(scaled$Exited, SplitRatio = 0.7)  
train\_data <- subset(scaled, split == TRUE)  
test\_data <- subset(scaled, split == FALSE)

# Етап 4. Побудова Deep Learning моделі

Викосритаємо можливості пакету h2o для побудови моделі на основі deep learning. Підключимо пакет.

#install.packages("h2o")  
library(h2o)

##   
## ----------------------------------------------------------------------  
##   
## Your next step is to start H2O:  
## > h2o.init()  
##   
## For H2O package documentation, ask for help:  
## > ??h2o  
##   
## After starting H2O, you can use the Web UI at http://localhost:54321  
## For more information visit http://docs.h2o.ai  
##   
## ----------------------------------------------------------------------

##   
## Attaching package: 'h2o'

## The following objects are masked from 'package:stats':  
##   
## cor, sd, var

## The following objects are masked from 'package:base':  
##   
## %\*%, %in%, &&, ||, apply, as.factor, as.numeric, colnames,  
## colnames<-, ifelse, is.character, is.factor, is.numeric, log,  
## log10, log1p, log2, round, signif, trunc

*Увага! Для запуску пакету потрібна віртуальна машина Java на ПК (JVM)*. Завантажити актуальну версію Java можна з сайту <https://www.java.com/en/download/>.

Запустимо двигун h2o:

h2o.init(nthreads = -1)

##   
## H2O is not running yet, starting it now...  
##   
## Note: In case of errors look at the following log files:  
## C:\Users\Kleban\AppData\Local\Temp\RtmpwTuJde/h2o\_Kleban\_started\_from\_r.out  
## C:\Users\Kleban\AppData\Local\Temp\RtmpwTuJde/h2o\_Kleban\_started\_from\_r.err  
##   
##   
## Starting H2O JVM and connecting: . Connection successful!  
##   
## R is connected to the H2O cluster:   
## H2O cluster uptime: 3 seconds 509 milliseconds   
## H2O cluster timezone: Europe/Helsinki   
## H2O data parsing timezone: UTC   
## H2O cluster version: 3.20.0.8   
## H2O cluster version age: 2 months and 29 days   
## H2O cluster name: H2O\_started\_from\_R\_Kleban\_ndx012   
## H2O cluster total nodes: 1   
## H2O cluster total memory: 0.87 GB   
## H2O cluster total cores: 4   
## H2O cluster allowed cores: 4   
## H2O cluster healthy: TRUE   
## H2O Connection ip: localhost   
## H2O Connection port: 54321   
## H2O Connection proxy: NA   
## H2O Internal Security: FALSE   
## H2O API Extensions: Algos, AutoML, Core V3, Core V4   
## R Version: R version 3.5.1 (2018-07-02)

Побудуємо математичну модель:

h2o\_model <- h2o.deeplearning(y = 'Exited',   
 training\_frame = as.h2o(train\_data),  
 activation = "Rectifier",  
 hidden = c(6,6),  
 epochs = 100)

##   
 |   
 | | 0%  
 |   
 |=================================================================| 100%  
##   
 |   
 | | 0%  
 |   
 |=================================================================| 100%

Варто перегляднути набір параметрів, що може приймати функція h2o.deeplearning(), адже вона досить складна:

help(h2o.deeplearning)

## starting httpd help server ... done

Здійснимо прогноз на тестовій вибірці (h2o.predict()), а також класифікуємо значення за cutOff = 0.5:

h2o\_predict\_prob <- h2o.predict(h2o\_model, newdata = as.h2o(test\_data[-(ncol(test\_data))]))

##   
 |   
 | | 0%  
 |   
 |=================================================================| 100%  
##   
 |   
 | | 0%  
 |   
 |=================================================================| 100%

h2o\_predict\_class <- ifelse(h2o\_predict\_prob > 0.5, 1 , 0)  
h2o\_predict\_class <- as.vector(h2o\_predict\_class)

Побудуємо матрицю неточностей:

library(caret)

## Loading required package: lattice

caret::confusionMatrix(factor(test\_data$Exited), factor(h2o\_predict\_class), positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 2316 73  
## 1 383 228  
##   
## Accuracy : 0.848   
## 95% CI : (0.8346, 0.8607)  
## No Information Rate : 0.8997   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.4223   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.7575   
## Specificity : 0.8581   
## Pos Pred Value : 0.3732   
## Neg Pred Value : 0.9694   
## Prevalence : 0.1003   
## Detection Rate : 0.0760   
## Detection Prevalence : 0.2037   
## Balanced Accuracy : 0.8078   
##   
## 'Positive' Class : 1   
##

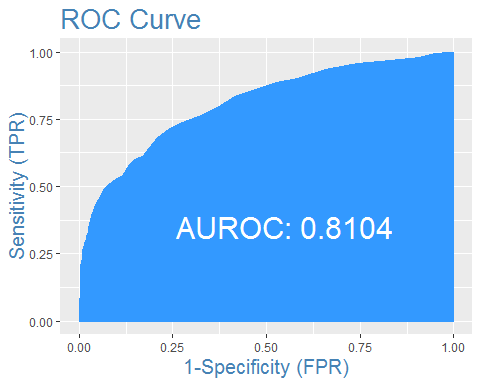
Побудуємо ROC-криву:

library(InformationValue)

##   
## Attaching package: 'InformationValue'

## The following objects are masked from 'package:caret':  
##   
## confusionMatrix, precision, sensitivity, specificity

InformationValue::plotROC(test\_data$Exited, as.vector(h2o\_predict\_prob))



Побудуємо модель з більшою кількістю прихованих шарів та нейронів:

h2o\_model2 <- h2o.deeplearning(y = 'Exited',   
 training\_frame = as.h2o(train\_data),  
 activation = "Rectifier",  
 hidden = c(10,10),  
 epochs = 100)

##   
 |   
 | | 0%  
 |   
 |=================================================================| 100%  
##   
 |   
 | | 0%  
 |   
 |==================== | 30%  
 |   
 |========================================================== | 90%  
 |   
 |=================================================================| 100%

Здійснимо прогноз:

h2o\_predict\_prob2 <- h2o.predict(h2o\_model2, newdata = as.h2o(test\_data[-(ncol(test\_data))]))

##   
 |   
 | | 0%  
 |   
 |=================================================================| 100%  
##   
 |   
 | | 0%  
 |   
 |=================================================================| 100%

h2o\_predict\_class2 <- ifelse(h2o\_predict\_prob2 > 0.5, 1 , 0)  
h2o\_predict\_class2 <- as.vector(h2o\_predict\_class2)

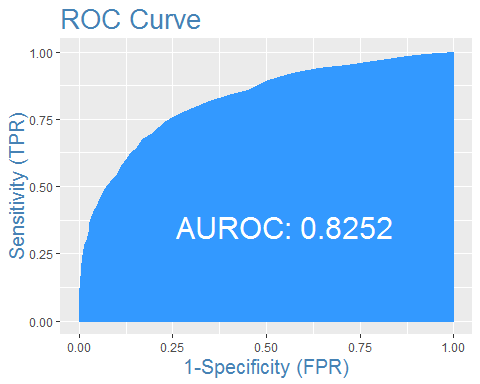
Матриця неточностей:

caret::confusionMatrix(factor(test\_data$Exited), factor(h2o\_predict\_class2), positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 2294 95  
## 1 361 250  
##   
## Accuracy : 0.848   
## 95% CI : (0.8346, 0.8607)  
## No Information Rate : 0.885   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.4408   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.72464   
## Specificity : 0.86403   
## Pos Pred Value : 0.40917   
## Neg Pred Value : 0.96023   
## Prevalence : 0.11500   
## Detection Rate : 0.08333   
## Detection Prevalence : 0.20367   
## Balanced Accuracy : 0.79433   
##   
## 'Positive' Class : 1   
##

ROC-крива:

InformationValue::plotROC(test\_data$Exited, as.vector(h2o\_predict\_prob2))



Зупинимо двигун h2o:

h2o.shutdown()

## Are you sure you want to shutdown the H2O instance running at http://localhost:54321/ (Y/N)?

## [1] TRUE

Для порівняння побудуємо ще одну нейронну мережу nnet:

library(nnet)  
nnet\_model <- nnet(Exited ~ ., data = train\_data, size = 3, maxit = 100)

## # weights: 37  
## initial value 1240.576390   
## iter 10 value 921.204811  
## iter 20 value 856.867281  
## iter 30 value 836.720484  
## iter 40 value 827.990229  
## iter 50 value 822.390720  
## iter 60 value 818.982938  
## iter 70 value 809.448375  
## iter 80 value 806.571019  
## iter 90 value 805.975565  
## iter 100 value 805.609374  
## final value 805.609374   
## stopped after 100 iterations

summary(nnet\_model)

## a 10-3-1 network with 37 weights  
## options were -  
## b->h1 i1->h1 i2->h1 i3->h1 i4->h1 i5->h1 i6->h1 i7->h1 i8->h1   
## -25.93 0.10 -3.46 0.04 0.01 0.07 -28.92 -0.14 15.85   
## i9->h1 i10->h1   
## -0.27 0.55   
## b->h2 i1->h2 i2->h2 i3->h2 i4->h2 i5->h2 i6->h2 i7->h2 i8->h2   
## -2.13 -0.01 0.10 0.11 -0.87 0.04 -3.27 -0.04 -0.94   
## i9->h2 i10->h2   
## 0.02 -0.07   
## b->h3 i1->h3 i2->h3 i3->h3 i4->h3 i5->h3 i6->h3 i7->h3 i8->h3   
## 4.91 -1.38 2.24 -1.89 0.92 -1.41 -34.64 1.55 14.18   
## i9->h3 i10->h3   
## -0.56 4.57   
## b->o h1->o h2->o h3->o   
## -1.85 -5.41 -12.15 11.81

Переглянемо модель візуально:

library(devtools)  
library(clusterGeneration)

## Loading required package: MASS

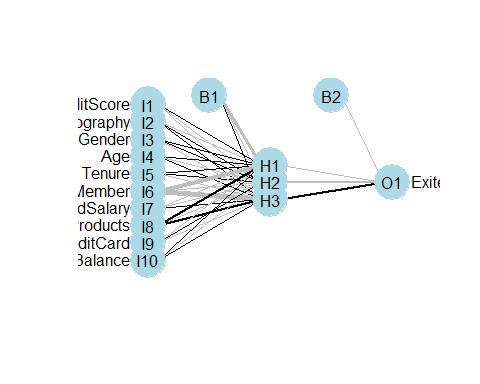
source\_url('https://gist.githubusercontent.com/fawda123/7471137/raw/466c1474d0a505ff044412703516c34f1a4684a5/nnet\_plot\_update.r')

## SHA-1 hash of file is 74c80bd5ddbc17ab3ae5ece9c0ed9beb612e87ef

plot(nnet\_model)

## Loading required package: scales

## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape  
## Loading required package: reshape



Здійснимо прогноз на тестовій вибірці:

nnet\_predicted\_prob <- predict(nnet\_model, test\_data)  
nnet\_predicted\_class <- ifelse(nnet\_predicted\_prob > 0.5, 1, 0)

Матриця неточностей та ROC-крива:

caret::confusionMatrix(factor(test\_data$Exited), factor(nnet\_predicted\_class), positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 2323 66  
## 1 383 228  
##   
## Accuracy : 0.8503   
## 95% CI : (0.8371, 0.8629)  
## No Information Rate : 0.902   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.4282   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.7755   
## Specificity : 0.8585   
## Pos Pred Value : 0.3732   
## Neg Pred Value : 0.9724   
## Prevalence : 0.0980   
## Detection Rate : 0.0760   
## Detection Prevalence : 0.2037   
## Balanced Accuracy : 0.8170   
##   
## 'Positive' Class : 1   
##

InformationValue::plotROC(test\_data$Exited, as.vector(nnet\_predicted\_prob))

