Building a Neural Network Model using R

Contents

Case Study – Predicting Loan Defaulters

Background

- The bank possesses demographic and transactional data of its loan customers. If the bank has a robust model to predict defaulters it can undertake better resource allocation.

Objective

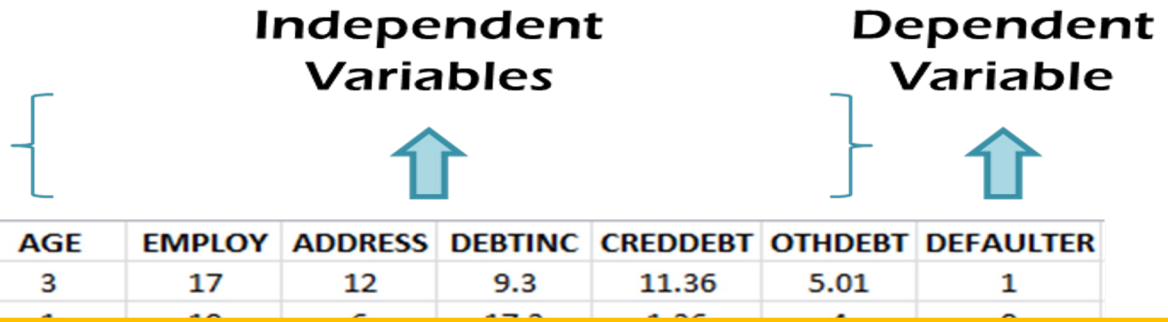
- To predict whether the customer applying for the loan will be a defaulter

Available Information

- Sample size is 700
- Age group, Years at current address, Years at current employer, Debt to Income Ratio, Credit Card Debts, Other Debts are the independent variables
- Defaulter (=1 if defaulter, 0 otherwise) is the dependent variable

Neural Network in R...Data Snapshot

BANK LOAN



SN	AGE	EMPLOY	ADDRESS	DEBTINC	CREDDEBT	OTHDEBT	DEFAULTER
1	3	17	12	9.3	11.36	5.01	1

Column	Description	Type	Measurement	Possible Values
SN	Serial Number	-	-	-
AGE	Age Groups	Integer	1(<28 years),2(28-40 years),3(>40 years)	3
EMPLOY	Number of years customer working at current employer	Integer	-	Positive value
ADDRESS	Number of years customer staying at current address	Integer	-	Positive value
DEBTINC	Debt to Income Ratio	Continuous	-	Positive value
CREDDEBT	Credit to Debit Ratio	Continuous	-	Positive value
OTHDEBT	Other Debt	Continuous	-	Positive value

Data Pre-Processing

- Since AGE is categorical variable, we create dummy variables before proceeding to neural network model.
- To set up a neural network to a dataset it is very important that we ensure a proper scaling of data. The scaling of data is essential because otherwise, a variable may have a large impact on the prediction variable only because of its scale.
- The common techniques to scale data are min-max normalization and Z-score normalization
- The min-max normalization transforms the data into a common range, thus removing the scaling effect from all the variables. Here we are using min-max normalization for scaling data.

Neural Network in R - Classifying Loan Defaulters...

```
bankloan<-read.csv("BANK LOAN.csv")

library(fastDummies)
bankloan <- dummy_cols(bankloan,select_columns =
"AGE",remove_first_dummy = T)

normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}

bankloan$EMPLOY<-normalize(bankloan$EMPLOY)
bankloan$ADDRESS<-normalize(bankloan$ADDRESS)
bankloan$DEBTINC<-normalize(bankloan$DEBTINC)
bankloan$CREDDEBT<-normalize(bankloan$CREDDEBT)
bankloan$OTHDEBT<-normalize(bankloan$OTHDEBT)
```

Neural Network in R - Classifying Loan Defaulters…

```
library(neuralnet)
set.seed(09092022)
nn_bank <- neuralnet(DEFAULTER~AGE_2+AGE_3+EMPLOY+ADDRESS+
                       DEBTINC+CREDDEBT+OTHDEBT, data=bankloan,
                     hidden=3,err.fct="ce",linear.output=FALSE)

out_bank <- cbind(nn_bank$covariate,nn_bank$net.result[[1]])
dimnames(out_bank) <- list(NULL, c("AGE_2","AGE_3",
"EMPLOY","ADDRESS","DEBTINC","CREDDEBT","OTHDEBT","nn-output"))
head(out_bank)
```

- **neuralnet()** takes formula interface. Here, case (status) is the variable of interest.
- **hidden=** Number of hidden neurons in each layer.
- **err.fct=** a differentiable function that is used for the calculation of the error. ‘ce’ stands for cross entropy.
- **linear.output=** FALSE ensures that the output is mapped by the activation function to the interval [0, 1].
- By default the command uses Resilient Backpropagation with Weight Backtracking

Neural Network in R - Classifying Loan Defaulters...

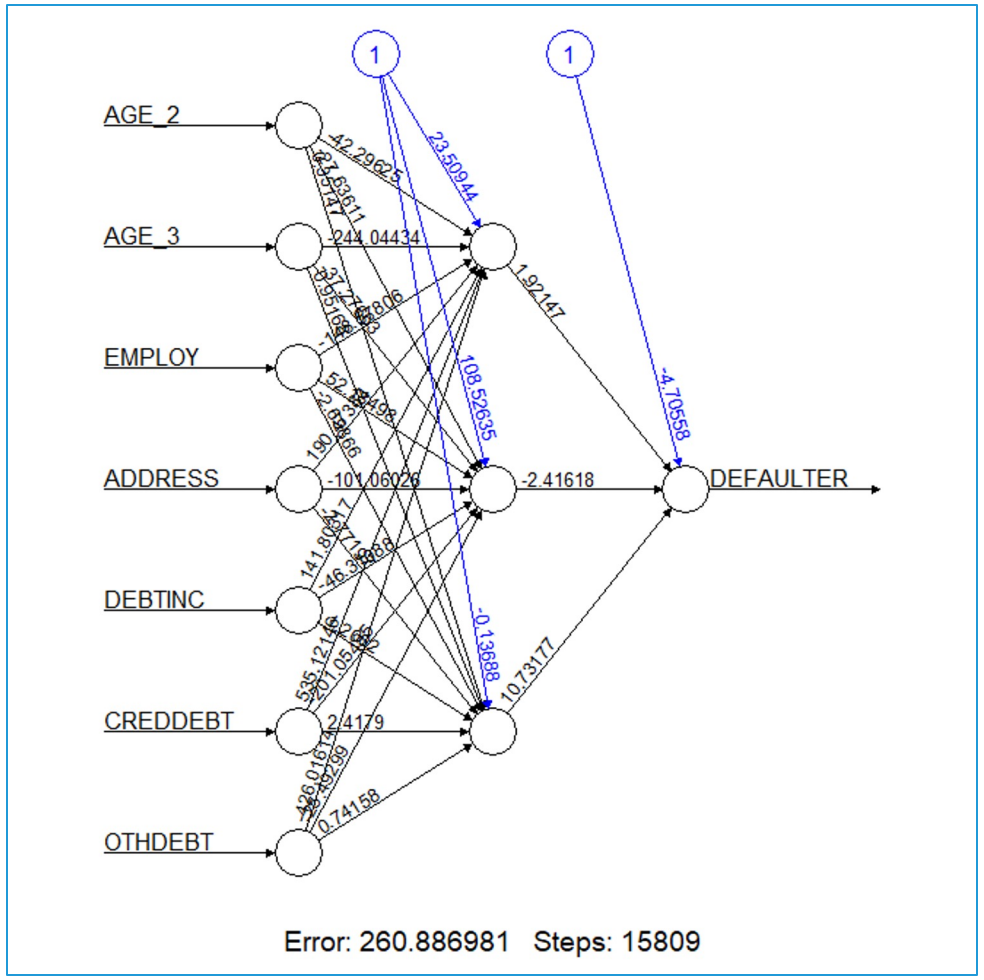
```
# Output
```

	AGE_2	AGE_3	EMPLOY	ADDRESS	DEBTINC	CREDDEBT	OTHDEBT	nn-output
1,]	0	1	0.54838710	0.3529412	0.21760391	0.55231144	0.18383988	0.943837210
2,]	0	0	0.32258065	0.1764706	0.41320293	0.06569343	0.14640474	0.152127464
3,]	1	0	0.48387097	0.4117647	0.12469438	0.04136253	0.07857672	0.003172850
4,]	0	1	0.48387097	0.4117647	0.06112469	0.12895377	0.02853966	0.009129689
5,]	0	0	0.06451613	0.0000000	0.41320293	0.08661800	0.11156412	0.805638180
6,]	0	1	0.16129032	0.1470588	0.23960880	0.01849148	0.07820608	0.311169470

Neural Network in R - Classifying Loan Defaulters...

```
plot(nn_bank)

# Output
```

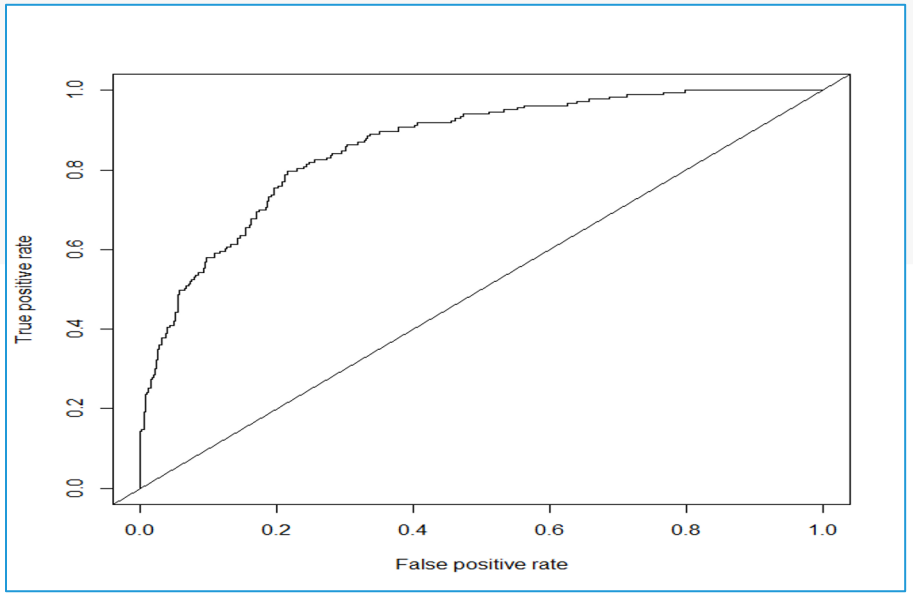


Neural Network in R - Classifying Loan Defaulters...

```
# ROC Curve for NN output
```

```
library(ROCR)
# Convert 'out' to dataframe
outdf<-as.data.frame(out_bank)
pred<-prediction(outdf$`nn-output`,bankloan$DEFAULTER)
perf<-performance(pred,"tpr","fpr")
plot(perf)
abline(0,1)

auc<-performance(pred,"auc")
auc@y.values
[1] 0.8606505
```



More About ANN Work

- Applications in Artificial Intelligence - Handwriting or Face Recognition, Voice Analysis
- The “building blocks” of neural networks are the neurons. In technical systems, we also refer to them as units or nodes.
- Basically, each neuron receives input from many other neurons, changes its internal state (activation) based on the current input, sends one output signal to many other neurons, possibly including its input neurons (recurrent network)
- Information is transmitted as a series of electric impulses, so-called spikes.
- The frequency and phase of these spikes encodes the information.
- In biological systems, one neuron can be connected to as many as 10,000 other neurons.
- Neurons of similar functionality are usually organized in separate areas (or layers).
- Often, there is a hierarchy of interconnected layers with the lowest layer receiving sensory input and neurons in higher layers computing more complex functions.

Quick Recap

Neural Networks in R	• Package **“neuralnet"** has **neuralnet()** that trains a neural network model