AMITY UNIVERSITY
UTTAR PRADESH

Summer Internship Report on

Prediction Analysis of Diabetes Dataset In Health-care Sector

Submitted to

Amity Institute of Information Technology In partial fulfilment of the requirements for the award of the degree of

Master of Computer Application

Submitted To

Dr Sarika Jain

Associate professor

Submitted By

Priyaank Sinha

A1000718022

2018 – 2021

DECLARATION BY STUDENT

I Priyaank Sinha, student of MCA hereby declare that the Seminar titled "**Prediction Analysis of Diabetes Dataset In Health-care Sector**" which is submitted by me to Dr Sarika Jain, Amity Institute of Information Technology, Amity University Uttar Pradesh, Noida, in partial fulfilment of the requirement for the award of the degree of master of computer applications, has not been previously formed the basis for the award of any degree, diploma or other similar title or recognition.

The author attests that permission has been obtained for the use of any copyrighted material appearing in the Dissertation / Project report other than brief excerpts requiring only proper acknowledgement in scholarly writing and all such use is acknowledged.

Place: Noida

Priyaank Sinha

A1000718022

Semester: MCA 5th

GUIDE CERTIFICATE

I hereby certify that the Seminar Report by Priyaank Sinha, student of MCA 5th (Enrolment No: A1000718022) with the title "**Prediction Analysis of Diabetes Dataset In Health-care Sector**" which is submitted to Amity Institute of Information Technology, Amity University Uttar Pradesh, Noida in partial fulfilment of the requirement for the award of the degree of master of computer applications is an original contribution with existing knowledge and faithful record of work carried out by him/her under my guidance and supervision and to the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Noida

Date:

Dr Sarika Jain

Associate Professor

Amity Institute of Information Technology
AUUP, Noida

ACKNOWLEDGEMENT

It is a high privilege for me to express my deep sense of gratitude to those entire faculty Members who helped me in the completion of the project, especially my internal guide Dr Sarika Jain who was always there at the hour of need.

My special thanks to particular faculty members and Batch mate of Amity Institute of Information Technology, Amity University, Uttar Pradesh for helping me in the completion of project work and its report submission.

Priyaank Sinha

A1000718022

AMITY UNIVERSITY

-----UTTAR PRADESH-----

Amity Institute of Information Technology

Summer Internship

Student Name	PRIYAANK SINHA
Enrollment No	A1000718022
Programme	Master of Computer Applications
Company's Name and Address	Amity Institute of Information Technology Sector - 125, Amity University, Noida 243005
Industry Guide	
Name	
Designation	
Contact Number	
	Ph.(O) : **(R) :**
	Mobile :
	Fax :
	E-mail : sjain@amity.edu

Project Information

1) Project Duration : **(28 Days)**

a) Date of Summer Internship commencement **(15/04/2020)**

a) Date of Summer Internship Completion **(16/05/2020)**

2) Topic

Prediction Analysis of Diabetes Dataset In Health-care Sector

3) Project Objective

This project provides a thorough walk-through of the steps required for the prediction analysis of the data including two different Machine Learning

algorithms and also brief about the accuracy check.

4) Methodology to be adopted

In this project, Data exploration -> Data extraction -> Data Cleaning -> Data partitioning -> Data Normalization -> Logistic and Neural Network Model -> Prediction Analysis For Accuracy

5) **Brief Summery of project(*to be duly certified by the industry guide*)**

This project provides a thorough walk-through of the steps required for the prediction analysis of the data including two different Machine Learning algorithms to design our model that are: a) Neural Network and b) Logistic regression on the same dataset. Additionally, we have discussed the Confusion Matrix, which predicts the accuracy of the designed models.

Signature (Student)	Signature (Industry Guide)	Signature (Faculty Guide)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The overwhelming increase in the Data in today's world demands more experts to handle this massive amount of data in all industries. It stands out that the companies need more Data Scientists and Data Engineers to design and code the algorithms. Having said that, we often confuse ourselves in the responsibilities shared between these two groups of experts. "*Data Scientists*" are the ones who primarily design algorithms that can serve as the foundation for different prediction models such as *Neural Network, Linear and Logistic regression,* etc. However, "*Data Engineers*" are the ones who code using the predefined prediction models and try to increase the accuracy in the results. This clearly shows that these two groups of experts i.e. Data Scientists and Data Engineers works in perfect harmony side-by-side to create impressive solutions to handle the ginormous amount of data produced each day.

This project provides a thorough walk-through of the steps required for the prediction analysis of the data including two different Machine Learning algorithms to design our model that are: a) *Neural Network* and b) *Logistic regression* on the same dataset. Additionally, we have discussed the *Confusion Matrix,* which predicts the accuracy of the designed models.

CHAPTER 1: PROBLEM STATEMENT

Problem: To generate the prediction analysis model of diabetes data set of the health care sector with two different Machine Learning algorithms and check the accuracy for the same model.

Explanation: The demand for Machine Learning in health care sectors is remarkably high. This project is an attempt to analyse the diabetes dataset and try to create the Machine Learning prediction models using *neural network* and *logistic regression* algorithms. We endeavour to create models that are more accurate as per the problem statement. For exploring the dataset, we implemented the visualisation techniques.

A) Pre-requisites:

- ✓ **Tools:** R Studios
- ✓ **Language:** R Programming

B) Source of Dataset:

- https://www.kaggle.com

CHAPTER 2: INTRODUCTION

In today's era, one must have experienced Data Science in several forms. Have anyone ever thought, from search engines to Global Positioning System (GPS); everyone is communicating with data science tools? It is one of the powerful sciences in the world of technologies. At this point, one might be suffering from a question that, what a *Data Science* is? The fusion of multiple tools, machine learning algorithms and principles that are required for investigating meaningful patterns in the raw data, is *Data Science*.

With time, Big Data is becoming a critical tool for businesses and companies of all sizes. All the business models are modifying based on big data analytics. For that reason, data science plays a very important role in breaking and extracting useful data from big data. To understand these concepts more deeply, let us first go through the basic ideas and technical aspects of data science used in this project.

2.1 Data Exploration

It is the inaugural step in data science, where the data engineer explores the data set in an unstructured way to divulge the initial patterns, characteristics and points of interest. Moreover, we can utilize data visualization techniques like *histograms, bar graphs, pie charts, etc.* to scrutinize the data with more accuracy.

2.2 Data Cleaning

The process of preparing the data for analysis by eliminating or modifying irrelevant, duplicated, incorrect, or improperly formatted (i.e.; NULL or VOID spaces) in data. Data cleaning is not all about removing data but it is a way to maximize the accuracy of the dataset without deleting crucial information.

2.3 Partitioning of the dataset

Once the dataset is ready for the analysis, we divide it into *training* and *testing* sets. The training set is the first part of a dataset on which our Machine Learning models get training and then we apply *prediction analysis* on the same ML model utilizing the testing set. This helps to predict the accuracy of the data.

2.4 Logistic Regression

Logistic regression is the type of supervised classification algorithm of Machine Learning. It predicts the probability of outcome when the dependent variable is dichotomous (i.e.; binary either 0 or 1). It also explains the relationship between the one dependent variable with the other one or more ordinal independent variables.

2.5 Data Normalization

Data normalization is one of the most important steps, which has to apply on the dataset before the Neural Network prediction analysis model. The normalization of the data means that every observation in the dataset should be in the range of **"-1 to +1".**

2.6 Neural Network

The most widely used prediction models of Machine Learning is a neural network. The Neural Network is comprised of neurons that are connected by the weighted links. It consists of three layers:

a) *Input Layer:* The number of neurons generated depends on the number of input variables.
b) *Hidden Layers:* The neurons generated in this layer was defined by the user according to the dataset or the usage of the model in the sequence like "*n, (n-1), (n-2)… 3, 2, 1*".
c) *Output Layer:* It comprises of only one neuron that generates the result of the whole model.

2.7 Confusion Matrix

A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions summarized with count values and broken by each class. The interpretation of a confusion matrix is in the form of:

Expected / **Predicted**	**Event**	**No Event**
Event	True Positive	False Positive
No Event	False Negative	True Negative

Table No. 1. Confusion Matrix

CHAPTER 3: BACKGROUND

This chapter comprises of the detailed information of all the important technical aspects, which are the pillars of this project.

3.1 Machine Learning

Machine Learning is a science that allows computers to proceed without explicitly programmed. The major applications of ML are *supervised learning* and *Un-supervised learning*. In unsupervised learning, unlabelled data was fed into a training algorithm to discover relationships and patterns between the data attributes. While supervised learning is human-labelled data.

ML consists of multiple classification algorithms such as:

- Linear and Logistic Regression
- Neural Networks
- Support Vector Machine (SVM)
- K-Nearest Neighbour (KNN)
- Random Forest and many more.

In this project, we are using only *Logistic Regression* and *Neural Networks* for prediction analysis.

3.1.1 Logistic Regression

To predict the outcome of a dependent variable based on prior information can easily be executed by *Logistic Regression.* This statistical analysis algorithm is in the use to predict different real-life problems such as:

- *Health care sector*, to predict whether the patient is suffering from a particular disease or not.
- *Forecasting*, for snowfall prediction.
- *The banking sector*, to predict the eligibility of the customer for a loan.

Logistic regression can be prorated into three types:

- *Binary Logistic Regression:* The responses have only two possible outcomes.
- *Multinomial Logistic Regression:* The responses have three or more possible outcomes without ordering.
- *Ordinal Logistic Regression:* The responses have three or more possible outcomes with ordering.

In this chapter, we will only focus on the binary logistic regression, because in this project we have only two outcomes whether the patient is *diabetic positive* or *diabetic negative*.

Binary Logistic Regression:

As we have discussed earlier that this type of logistic regression will work only on two possible outcomes either ‘1’ or ‘0’. To understand this please refer to figure.1.

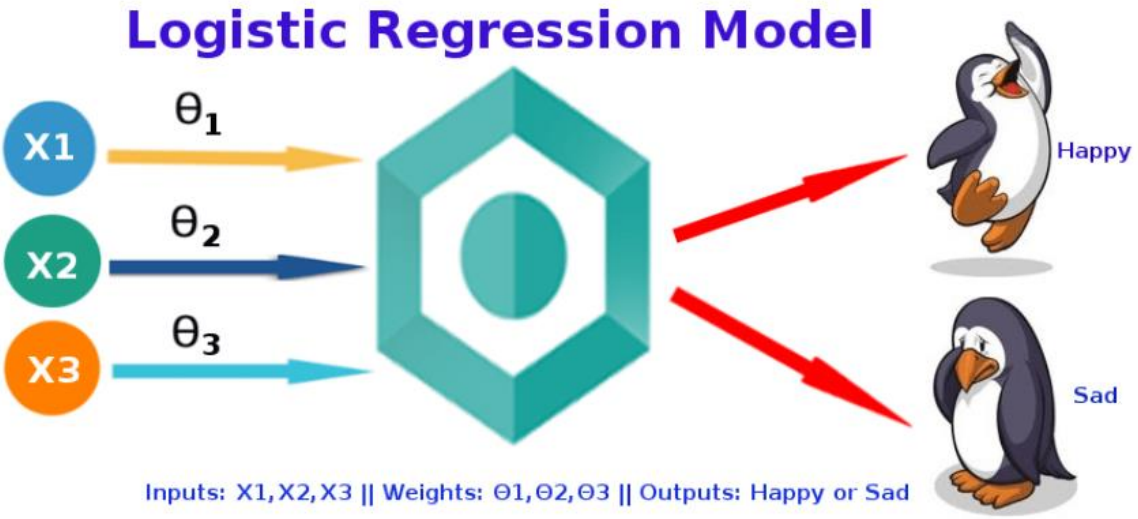

Fig.1. Binary Logistic Regression

According to fig.1, X1, X2, X3 are the inputs, Θ1, Θ2, Θ3 are the weights and Happy & Sad are the outputs.

The formula used in the Logistic Regression algorithm:

Estimate value of **Intercept** is the value of coefficient **'a'** and other variables **Estimate values** are the value of coefficient **'b'**, in the equation of logistic regression.

- **EQUATION:**

$$\mathbf{y = a * e^{bx}}$$

Now, a question arises, that ***what is the estimated value?*** Well, estimate value is the numerical value generated, when the data trained in the ***generalized linear model (glm)*** whose family is of *binomial type*, which consists of logistic regression algorithm.

To apply logistic regression, we first need to install the following packages in R Studios:

- ***GGally:*** It will help us to calculate the correlations between every variable with each other of the diabetes dataset.
  Command: **install.packages("GGally")**
- ***Caret:*** This package will help to partition the dataset into two sets that are, *Training and testing* dataset.
  Command: **install.packages("caret")**
- ***E1071:*** This is misc. function, which is used to calculate the statistics and probability, and thus, it will help in predicting the accuracy of our model.
  Command: **install.packages("e1071")**

3.1.2 Neural Network

It named as *Neural* because the designing of the algorithm is based on the basic functionality of the *neurons.* Every neuron is connected with the weighted links. According to fig.2, the

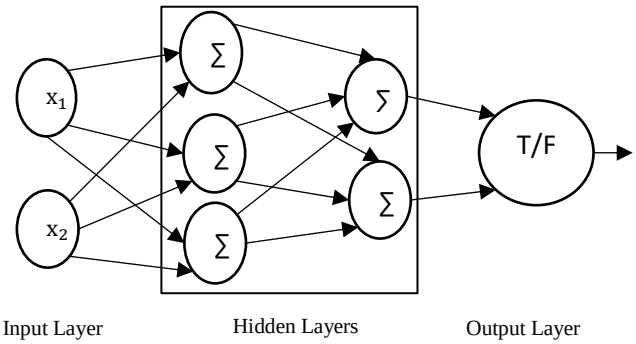

Fig.2. The architecture of the Neural Network

The neural network comprises of three layers:

a) *Input Layer:* The number of neurons generated depends on the number of input variables.
b) *Hidden Layers:* The neurons generated in this layer was defined by the user according to the dataset or the usage of the model in the sequence like "*n, (n-1), (n-2)… 3, 2, 1*". For instance, if user-defined sequence **"c(2, 1)" (R command)** then we can say that the hidden layer consists of two layers, one is having two neurons and the other is having one neuron.
c) *Output Layer:* It comprises of only one neuron that generates the result of the whole model.

The formula used in Neural Network:

$$\boldsymbol{Output} = \sum_{\boldsymbol{i=0,a=0}}^{\boldsymbol{n,\infty}} \boldsymbol{w_i} * \boldsymbol{v_{i_a}}$$

Where $\boldsymbol{n}$ = Total number of variables in the dataset

$\boldsymbol{w_i}$ = Weight on the links generated by the neural network

$\boldsymbol{v_{i_a}}$ = Input values of the variable of the normalized dataset, such that

$\boldsymbol{i}$ = Number of variables & $\boldsymbol{a}$ = Number of observation of that particular variable.

The most important step before the neural network is to normalize the data, now again the question arises that, ***what is normalized data*?** Well, data normalization is the process in which we will apply the normalize formula and convert all the values of the dataset between "-1 to +1".

Formula Used:

$$\boldsymbol{Norm = \frac{X - Xmax}{Xmax - Xmin}}$$

Prerequisite packages in R Studios for the neural network are:

- ***Caret and e1071:*** These packages were discussed in **3.1.1** logistic regression part.
- ***Neuralnet:*** This package is used for the implementation of the *Neural Network algorithm*. The algorithm was predefined in the package, this project is just using the algorithm and creating a more accurate model for prediction analysis.
  Command: install.packages("neuralnet")

CHAPTER 4: WORKING MODEL

Data Flow Diagram:

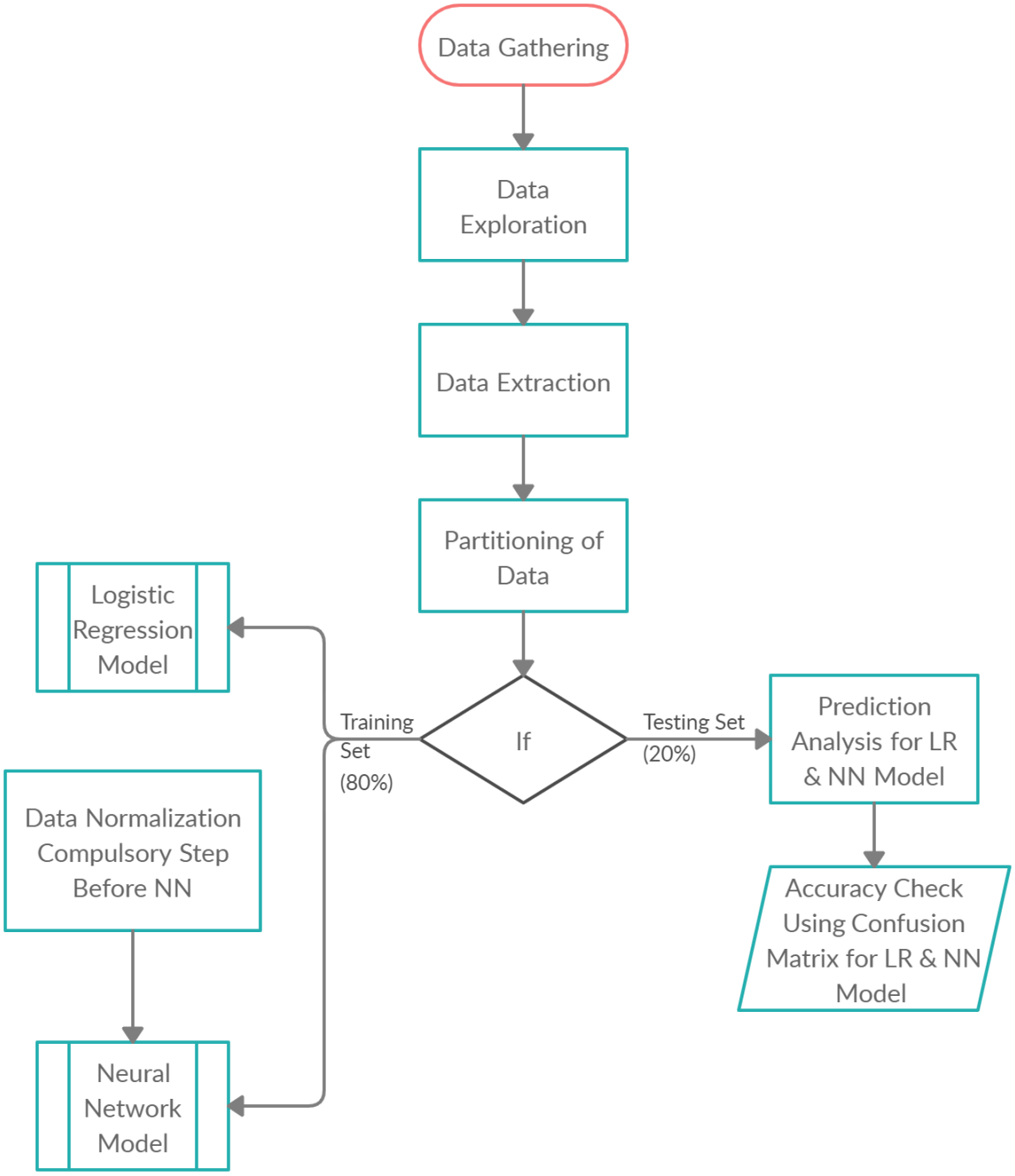

Fig. 3. DFD of Working Model

Code Snippet with explanation:

Data Exploration:

The command used is to explore the datatypes of complete dataset and to check the abnormality if any.

```
str(diabetes)
'data.frame':	2000 obs. of 9 variables:
 $ Pregnancies              : int 2 0 0 0 1 0 4 8 2 2...
 $ Glucose                  : int 138 84 145 135 139 173 99 194 83 89...
 $ BloodPressure            : int 62 82 0 68 62 78 72 80 65 90...
 $ SkinThickness            : int 35 31 0 42 41 32 17 0 28 30...
 $ Insulin                  : int 0 125 0 250 480 265 0 0 66 0...
 $ BMI                      : num 33.6 38.2 44.2 42.3 40.7 46.5 25.6 26.1 36.8 33.5...
 $ DiabetesPedigreeFunction: num 0.127 0.233 0.63 0.365 0.536...
 $ Age                      : int 47 23 31 24 21 58 28 67 24 42...
 $ Outcome                  : int 1 0 1 1 0 0 0 0 0 0...
```

Data Extraction:

The given command will extract the valuable data to train our created model of logistic regression and Neural Network.

```
diabetes_2 <- diabetes_1[!diabetes_1$Pregnancies < 1,]

view(diabetes_2)
```

dataset[!dataset$x < a,]: This command is used to remove all the entries less than 'a' row-wise from the column x.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
5	1	139	62	41	480	40.7	0.536	21	0
26	1	130	60	23	170	28.6	0.692	21	0
47	1	146	56	0	0	29.7	0.564	29	0
51	1	103	80	11	82	19.4	0.491	22	0
52	1	101	50	15	36	24.2	0.526	26	0
56	1	73	50	10	0	23.0	0.248	21	0
69	1	95	66	13	38	19.6	0.334	25	0
75	1	79	75	30	0	32.0	0.396	22	0

Fig. 4. A glimpse of Extracted Data

From fig.3, we can see that we have successfully removed the zero entries of 'pregnancies.'

NOTE: For diabetes dataset, we will avoid data cleaning because while exploring the data we did not found any NA value or unwanted value as none of the datatypes is of ***"Factor or Character Type"***.

Correlation Analysis:

To calculate the correlations of each variable with every variable we will be using the following "ggcorr" command (part of GGally package).

```
library(GGally)

ggcorr(diabetes_2, label = TRUE, label_alpha = TRUE, method = c('everything', 'spearman'))
```

where, method = c('everything', 'spearman'), means that we will calculate the relation between every variable using **spearman rank method.**

Curve Fitting for Logistic Regression:

We are now going to fit the curve between the strongly correlated variables concerning the dependent variable that is *Output.* For plotting the curve, the first thing is to import "***ggplot2***" library (part of GGally package).

```
library(ggplot2)

ggplot(diabetes_2, aes(x = BloodPressure, y = BMI) + facet_grid(~Outcome) + geom_point()
+ geom_smooth())
```

Where **aes():** This is known as an aesthetic function where we will declare the variables

facet_grid(): It forms a matrix of panels defined by row and column faceting variables. It is most useful when you have two discrete variables, and all combinations of the variables exist in the data. Here, **"Outcome"** is the faceting variable.

geom_point(): This function is used to plot the geometric points on the graph surface.

geom_smooth(): This function is used to plot the smooth curve on the graph surface.

Partitioning of Dataset:

The commands for partitioning the dataset will remain same for the both *Logistic Regression model* and *Neural Network model.* For this, first, we import the "*caret*" library.

```
library(caret)

index <- createDataPartition(diabetes_2$Outcome, p = 0.80, list = FALSE)

train_set <- diabetes_2[index,]

test_set <- diabetes_2[-index,]
```

createDataPartition(): This function is used to divide the dataset according to the percentage value set in the parameter. Here, **"p = 0.80",** means that the dataset is divided into two parts **80%** as the training set and **20%** as the testing set.

To verify that the partition is proper or not we use *dimension function.*

```
dim(train_set); dim(test_set)
```

Logistic Regression Model:

For applying logistic model, we have to use "*glm function (Generalized Linear Model)*". In this method, the most important parameters are *formula creation* and select the *family* type. For better understanding, see the following command:

```
diabetes_model_glm <- glm(Outcome~., data = train_set, family = 'binomial')
```

glm(): Generalized Linear Model function is used to apply logistic regression,

Where, **Outcome~. :** means that after solving the logistic binomial equation, the system will train to give proper outcome based on all the variables of the dataset.

family: is used to providing the type of link to the model. Here, the link is of **binomial** that means, the binomial family links: **logit, probit, cauchit,** (corresponding to logistic, normal and Cauchy CDFs respectively). We need to choose the family link according to the prediction technique we are applying to the data.

Here, we are using **logistic regression** which comes under **logit()** function which is the part of the **binomial family.**

Similarly, the different family objects are:

```
"binomial(link = "logit")
gaussian(link = "identity")
Gamma(link = "inverse")
inverse.gaussian(link = "1/mu^2")
poisson(link = "log")
quasi(link = "identity", variance = "constant")
quasibinomial(link = "logit")
quasipoisson(link = "log")"
```

Prediction Analysis:

In this, we will use test set to predict the accuracy of our trained model. The commands used are same for both logistic regression and neural network models. Before the implementation of the commands, we first need to import the "*e1071*" package.

```
library(e1071)
```

CASE1: For Logistic Regression

```
test_set$Outcome <- factor(test_set$Outcome, levels = unique(test_set$Outcome))

predict1 <- predict(diabetes_model_glm, newdata = test_set, type = 'response')

predict1_class <- factor(ifelse(predict1 > 0.5, "1", "0"))

confusion_matrix1 <- ConfusionMatrix(predict1_class, test_set$Outcome, positive = "1")
```

CASE2: For Neural Network

```
test_set_nn$diabetes_2.Outcome <- factor(test_set_nn$diabetes_2.Outcome, levels = unique(test_set_nn$diabestes_2.Outcome))
```

predict_nn <- predict(nn, newdata = test_set_nn, type = 'response')

predict_class_nn <- factor(ifelse(predict_nn > 0.5, "1", "0"))

confusion_matrix_nn <- ConfusionMatrix(predict1_class, test_set_nn$diabetes_2.Outcome, positive = "1")

Where, **factor():** This function is used to factorise the complete testing dataset using **unique()** outcomes.

predict(): This function will try to predict the outcomes of the dataset with more accuracy.

type = "response": option tells **R** to output probabilities of the form **P(Y = 1|X),** as opposed to other information such as the **logit**.

factor(ifelse(predict1 > 0.5,"1","0")): This statement means that if the prediction is more t han 50% the outcome should be "1" (i.e.; the person is diabetic) else the outcome should be " 0" (i.e.; the person is non-diabetic).

Data Normalization:

Before we apply NN, we have to take care that the values of complete dataset should be between -1 to +1. The following command is required to normalize the data:

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

normalize_diabetes_2 <- data.frame(lapply(diabetes_2[1:8], normalize))

normalize_diabetes_2 <- data.frame(normalize_diabetes_2, diabetes_2$Outcome)

view(normalize_diabetes_2)

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	diabetes_2.Outcome
1	-1	-0.30150754	-0.4918033	-0.6272727	-0.3548387	-0.4950372	-0.7986637	-1.0000000	0
2	-1	-0.34673367	-0.5081967	-0.7909091	-0.7715054	-0.6451613	-0.7291759	-1.0000000	0
3	-1	-0.26633166	-0.5409836	-1.0000000	-1.0000000	-0.6315136	-0.7861915	-0.8666667	0
4	-1	-0.48241206	-0.3442623	-0.9000000	-0.8897849	-0.7593052	-0.8187082	-0.9833333	0
5	-1	-0.49246231	-0.5901639	-0.8636364	-0.9516129	-0.6997519	-0.8031180	-0.9166667	0
6	-1	-0.63316583	-0.5901639	-0.9090909	-1.0000000	-0.7146402	-0.9269488	-1.0000000	0
7	-1	-0.52261307	-0.4590164	-0.8818182	-0.9489247	-0.7568238	-0.8886414	-0.9333333	0
8	-1	-0.60301508	-0.3852459	-0.7272727	-1.0000000	-0.6029777	-0.8610245	-0.9833333	0
9	-1	-1.00000000	-0.6065574	-0.8181818	-1.0000000	-0.6935484	-0.9750557	-0.9833333	0
10	-1	-0.46231156	-0.4426230	-0.8272727	-1.0000000	-0.6712159	-0.9639198	-0.9500000	0
11	-1	-0.59798995	-0.5491803	-1.0000000	-1.0000000	-0.7630273	-0.9224944	-1.0000000	0
12	-1	-0.64321608	-0.6065574	-0.8363636	-0.8978495	-0.7468983	-0.8935412	-0.9833333	0
13	-1	-0.38693467	-0.2622951	-0.5363636	-0.7043011	-0.3833747	-0.8926503	-0.8333333	1
14	-1	-0.18090452	-0.4098361	-1.0000000	-1.0000000	-0.5161290	-0.4930958	-0.8000000	1
15	-1	-0.24120603	-0.5081967	-1.0000000	-1.0000000	-0.6761787	-0.9576837	-0.9833333	0
16	-1	-0.59296482	-0.4098361	-0.8363636	-0.9462366	-0.6699752	-0.9113586	-0.9500000	0
17	-1	-0.36683417	-0.5409836	-0.7363636	-0.7956989	-0.6439206	-0.6806236	-1.0000000	0

Fig. 5. Glimpse of Normalize Dataset

Form fig.4, we can easily say that every value of dataset is now between -1 to +1.

Neural Network:

To implement NN, first we need to import *neuralnet* package. The most important parameters of neuralnet method are the number of hidden layers, formula and threshold value. The following command is to set the neural network model:

```
library(neuralnet)

nn <- neuralnet(diabetes_2.Outcome~., data = train_set_nn, hidden = c(2,1), linear.output = FALSE, threshold = 0.01)
```

Where, **formula** = diabetes_2.Outcome~.

data = train set, because we need to train our model according to the formula.

hidden = c(2,1), which means that in our model we need two hidden layers out of which one layer has two neurons and another layer has one neuron.

The threshold value, determine the value of auto-generated weights of the connected links.

CHAPTER 5: RESULTS

ScreenShots with explanation (wherever required):

Correlation Analysis:

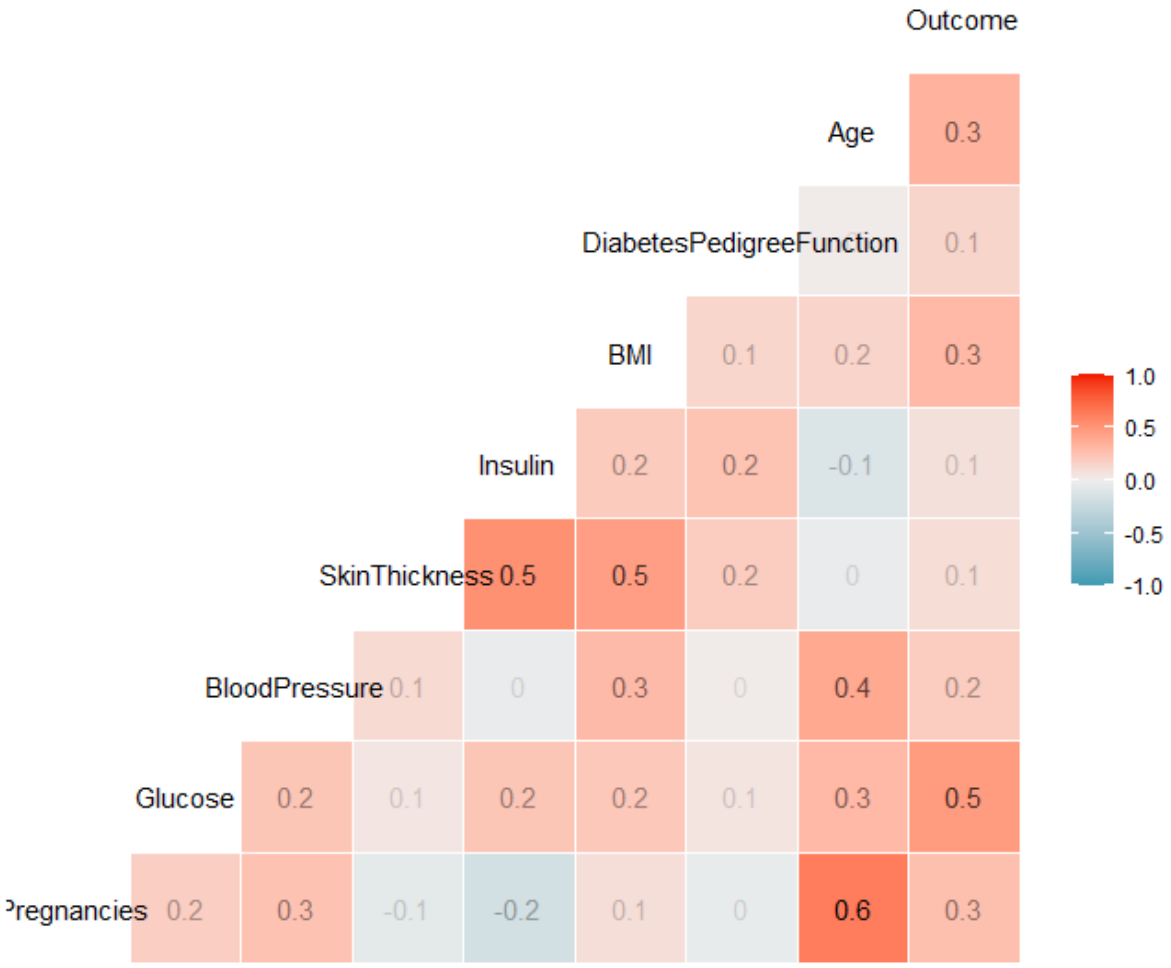

Fig. 6. Correlation between dataset variables

Curve Fitting:

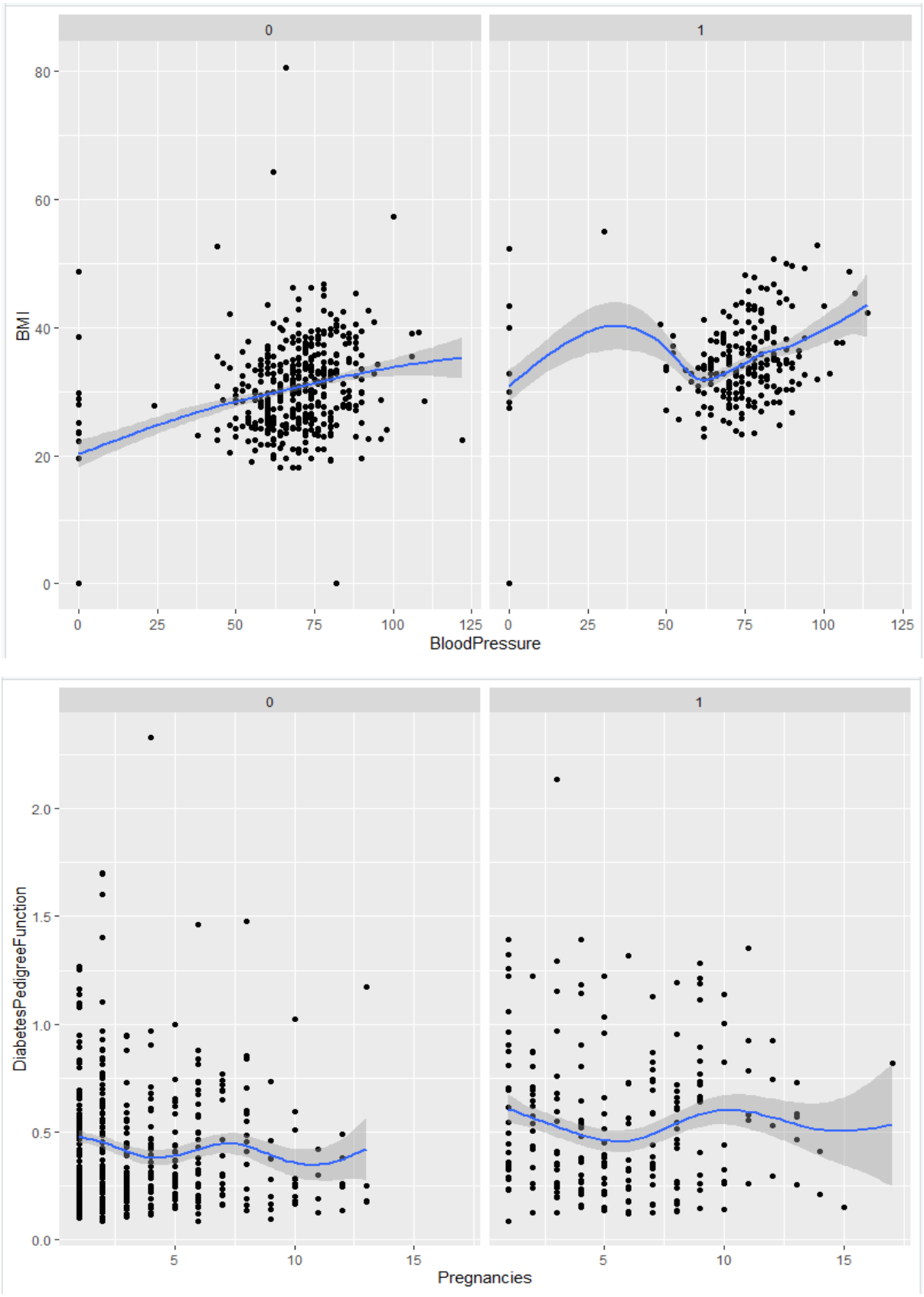

Fig. 7. Curve fittings for Logistic Regression

Accuracy Check by Prediction Analysis for the Logistic Regression Model:

```
> confusion_matrix1
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 203  52
         1  19  65

               Accuracy : 0.7906
                 95% CI : (0.7433, 0.8326)
    No Information Rate : 0.6549
    P-Value [Acc > NIR] : 3.196e-08

                  Kappa : 0.5036

 Mcnemar's Test P-Value : 0.000146

            Sensitivity : 0.5556
            Specificity : 0.9144
         Pos Pred Value : 0.7738
         Neg Pred Value : 0.7961
             Prevalence : 0.3451
         Detection Rate : 0.1917
   Detection Prevalence : 0.2478
      Balanced Accuracy : 0.7350

       'Positive' Class : 1

>
```

Fig. 8. Accuracy Check for Logistic Regression Model

Accuracy: This system predicts with the accuracy of 0.7906 means **79.06%.**

No Information Rate: As the accuracy is higher than no information rate, therefore we can say that the model is ***Good Predictor.***

Sensitivity: value is 0.5556, means **55.56%** system is capable of predicting correct decision, that a person has diabetes.

Specificity: value is 0.9144, means **91.44%** system is capable of predicting correct decision, that a person is non-diabetic.

Positive Prediction Value: is 0.7738, means **77.38%** probability of having diabetes among those who predicted to be diabetics.

Negative Prediction Value: is 0.7961, means **79.61%** probability of being non-diabetics among those who predicted to be non-diabetics.

Neural Network Model:

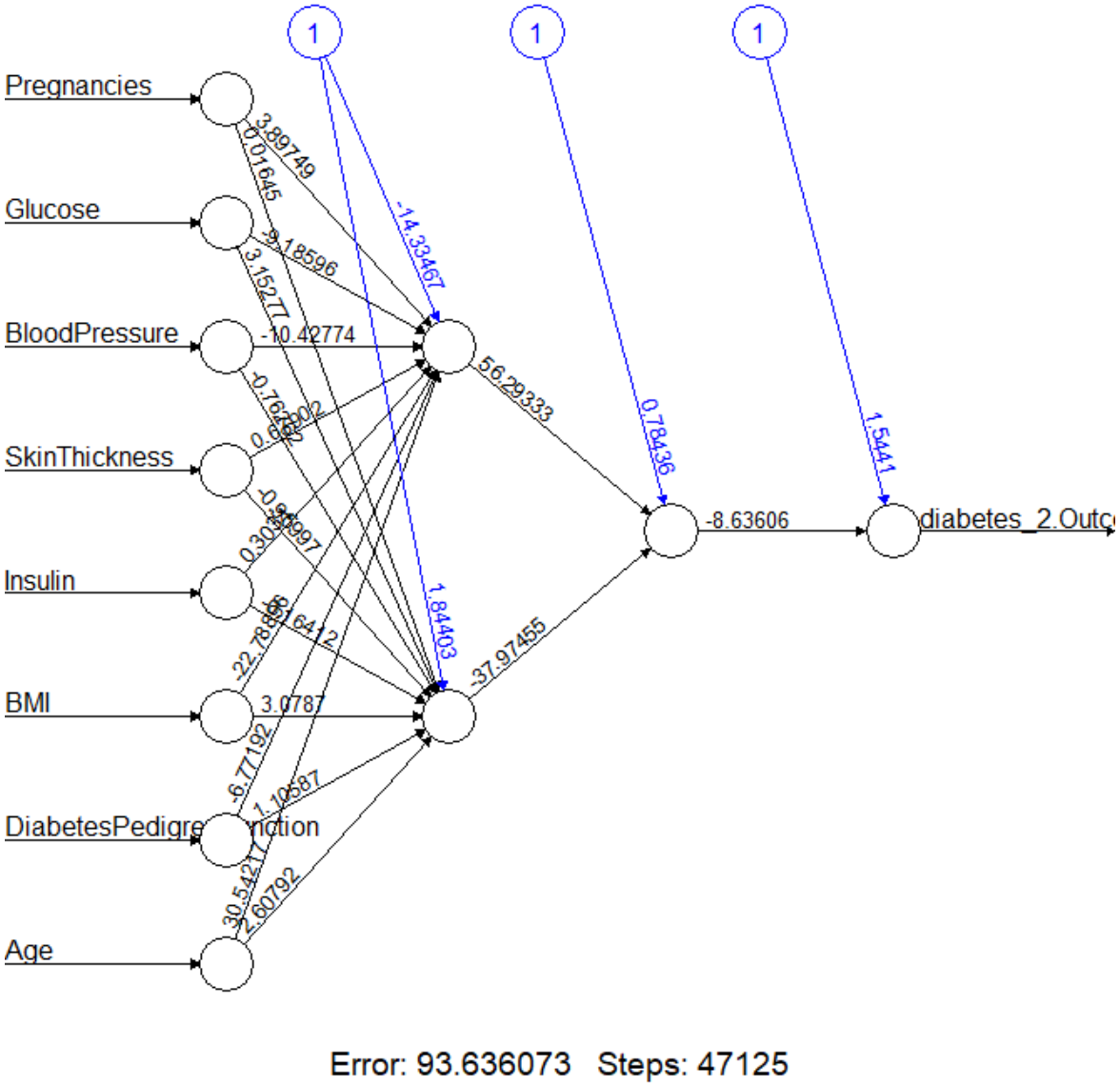

Fig. 9. Neuron diagram of Neural Network Model

Here, the system generates the **weights** randomly that displayed on the links between **one neuron to the other**.

Accuracy Check by prediction analysis for NN:

```
> confusion_matrix_nn
Confusion Matrix and Statistics

          Reference
Prediction   1   0
         1  78  27
         0  50 184

               Accuracy : 0.7729
                 95% CI : (0.7245, 0.8164)
    No Information Rate : 0.6224
    P-Value [Acc > NIR] : 2.269e-09

                  Kappa : 0.4991

 Mcnemar's Test P-Value : 0.01217

            Sensitivity : 0.6094
            Specificity : 0.8720
         Pos Pred Value : 0.7429
         Neg Pred Value : 0.7863
             Prevalence : 0.3776
         Detection Rate : 0.2301
   Detection Prevalence : 0.3097
      Balanced Accuracy : 0.7407

       'Positive' Class : 1
```

Fig. 10. Accuracy Check for NN Model

Since, **Accuracy = 0.7729 (i.e.; 77.29%)** which is more than **No Information Rate (NIR) = 0.6224 (i.e.; 62.24%),** therefore, the model is capable of predicting the outcome with the accuracy of **77.29%,** hence, we can say that the model is a **good predictor.**

CHAPTER 6: CONCLUSION & FUTURE SCOPE

Conclusion:

From small to large industries, data flow is everywhere; hence, the data analysis is the deciding factor for any industry success rate. This project is a part of data analysis in which we have successfully explained all the concepts i.e.; from *data exploration* to *accuracy check using confusion matrix* in detail. From the experimental point of view, we have successfully create the prediction models using *Logistic Regression* and *Neural Networks* with the accuracy of approximately **79%** and **77%** respectively. Hence, the demonstrated model in the project can successfully predict whether the patient is diabetic or not.

Future Work:

As per the future concern, one can convert the model into recommender system for predicting proper diet plan for diabetic positive or negative patients. One can also apply remaining *Machine Learning classification algorithms:*

- Random Forest
- Support Vector Machine
- K-Nearest Neighbour
- Naïve Bayes
- Stochastic Gradient Descent and many more.

REFERENCES

[1] https://www.analyticsindiamag.com/7-types-classification-algorithms/

[2] https://app.creately.com/

[3] https://briatte.github.io/ggcorr/

[4] http://r-statistics.co/ggplot2-Tutorial-With-R.html

[5] http://www.authorstream.com/Presentation/NIKKUTANU-1431448-pradeep-correlation-and-regression/

[6] https://datascienceplus.com/neuralnet-train-and-test-neural-networks-using-r/

[7] https://www.sas.com/en_gb/insights/articles/analytics/a-guide-to-predictive-analytics-and-machine-learning.html

[8] https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc

[9] https://www.freecodecamp.org/news/an-introduction-to-web-scraping-using-r-40284110c848/

[10] https://stats.idre.ucla.edu/r/dae/logit-regression/

[11] http://r-statistics.co/Logistic-Regression-With-R.html

[12] https://www.datacamp.com/community/tutorials/logistic-regression-R

[13] https://www.machinelearningplus.com/machine-learning/logistic-regression-tutorial-examples-r/

[14] https://www.tutorialspoint.com/r/r_logistic_regression.htm

[15] https://www.datacamp.com/community/tutorials/neural-network-models-r

[16] https://cran.r-project.org/web/packages/neuralnet/neuralnet.pdf

[17] https://www.kdnuggets.com/2016/08/begineers-guide-neural-networks-r.html

[18] http://uc-r.github.io/ann_classification

[19] https://towardsdatascience.com/build-your-own-neural-network-classifier-in-r-b7f1f183261d

[20] http://www.learnbymarketing.com/tutorials/neural-networks-in-r-tutorial/