EX.NO: 9	REGRESSION MODEL- LINEAR MODEL,LOGISTIC MODEL.
DATE: 20-10-2022	

Aim:

To implement linear regression and logistic regression model using R .

Description:

Linear regression :

Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables they are considering, and the number of independent variables getting used.

Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression.

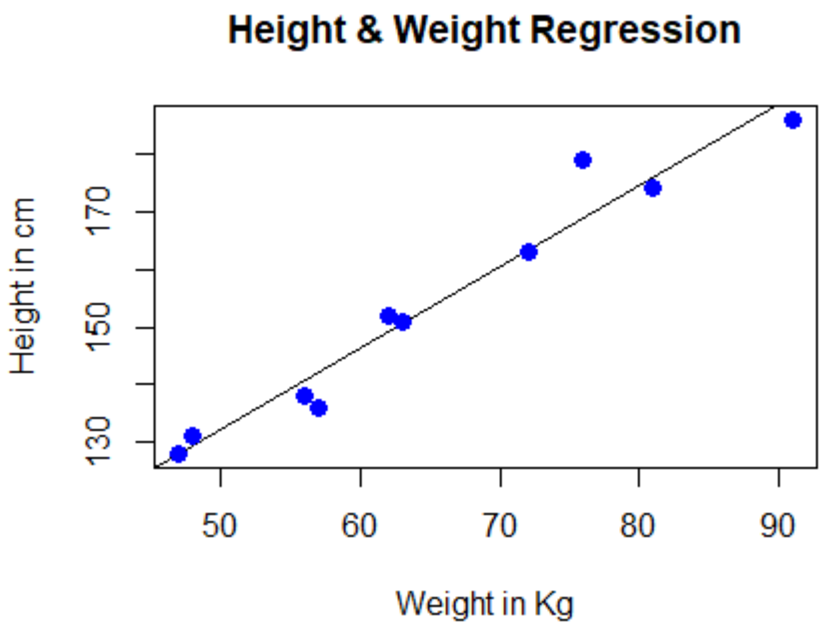
$$y = \theta_1 + \theta_2 . x$$

Program:

Input:

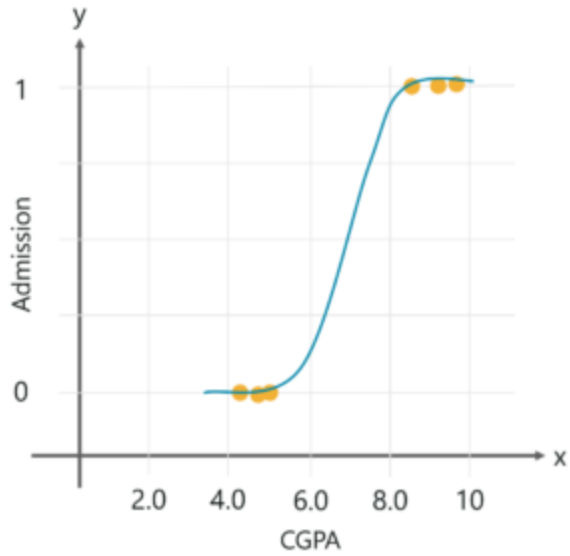
```
# Create the predictor and response variable.

x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)

y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

relation <- lm(y~x)

# Plot the chart.

plot(y,x,col = "blue",main = "Height & Weight Regression",

    abline(lm(x~y)),cex = 1.3,pch = 16,xlab = "Weight in Kg",ylab = "Height in cm")
```

output:



Logistic regression :

Logistic regression in [R Programming]() is a classification algorithm used to find the probability of event success and event failure. Logistic regression is used when the dependent variable is binary(0/1, True/False, Yes/No) in nature. Logit function is used as a link function in a binomial distribution.



Logistic Regression Example – Logistic Regression In R

Program:

Input:

```
#loading Packages
library(tidyverse)
library(modelr)
library(broom)
#Install ISLR Package
install.packages('ISLR')
#Load ISLR Package
library('ISLR')
mydata <- as_tibble(ISLR::Default)
#Creating the Training and Testing data set
sample <- sample(c(TRUE, FALSE), nrow(mydata), replace = T, prob = c(0.6,0.4))
train <- mydata[sample, ]
test <- mydata[!sample, ]
#Fitting a logistic regression model
logmodel <- glm(default ~ balance, family = "binomial", data = train)
#Plotting a graph: Probability of default Vs Balance
mydata %>%
  mutate(prob = ifelse(default == "Yes", 1, 0)) %>%
  ggplot(aes(balance, prob)) +
  geom_point(alpha = .15) +
  geom_smooth(method = "glm", method.args = list(family = "binomial")) +
  ggtitle("Logistic regression model fit") +
  xlab("Balance") +
  ylab("Probability of Default")
```

#Summary of the Logistic Regression Model

summary(logmodel)

output:


```
Call:
glm(formula = default ~ balance, family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2605  -0.1546  -0.0634  -0.0245   3.6323

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.035e+01  4.403e-01  -23.50   <2e-16 ***
balance      5.368e-03  2.702e-04   19.87   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1875.7  on 6031  degrees of freedom
Residual deviance: 1035.5  on 6030  degrees of freedom
AIC: 1039.5

Number of Fisher Scoring iterations: 8
```

Result:

Thus, Successfully completed implementation of regression using R.