

**EXPERIMENT NO.**

Aim of the experiment: - To implement Logistic Regression.

Course Outcome: - To implement an appropriate machine learning model for

Date of Conduction: - 23/08/2022

Date of Submission: - 20/09/2022

02

the given application.

Implementation

(05)

Understanding

(05)

Punctuality and Discipline

(05)

Total Marks (15)

Practical In charge

**Experiment: 02**

**Aim:** To implement Logistic Regression

**Theory:** Regression analysis is a statistical method to model the relationship between a dependent (target) and independent (predictor) variables with one or more independent variables. More specifically, Regression analysis helps us to understand how the value of the dependent variable is changing corresponding to an independent variable when other independent variables are held fixed. It predicts continuous/real values such as temperature, age, salary, price, etc.

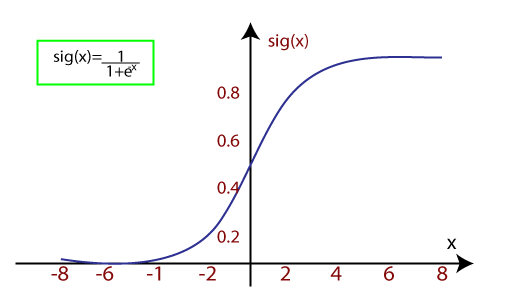
## Types of Regression

## There are various types of regressions which are used in data science and machine learning. Each type has its own importance on different scenarios, but at the core, all the regression methods analyze the effect of the independent variable on dependent variables. Here we are discussing some important types of regression which are given below:

* Linear Regression
* Logistic Regression
* Polynomial Regression
* Support Vector Regression
* Decision Tree Regression
* Random Forest Regression
* Ridge Regression
* **Lasso Regression**
* **Logistic Regression:** Logistic regression is another supervised learning algorithm which is used to solve the classification problems. In **classification problems**, we have dependent variables in a binary or discrete format such as 0 or 1.
* Logistic regression algorithm works with the categorical variable such as 0 or 1, Yes or No, True or False, Spam or not spam, etc.
* It is a predictive analysis algorithm which works on the concept of probability.
* Logistic regression is a type of regression, but it is different from the linear regression algorithm in the term how they are used.
* Logistic regression uses **sigmoid function** or logistic function which is a complex cost function. This sigmoid function is used to model the data in logistic regression. The function can be represented as:



* f(x)= Output between the 0 and 1 value.
* x= input to the function
* e= base of natural logarithm



* It uses the concept of threshold levels, values above the threshold level are rounded up to 1, and values below the threshold level are rounded up to 0.

**Algorithm:**

Training data assumptions for logistic regression:

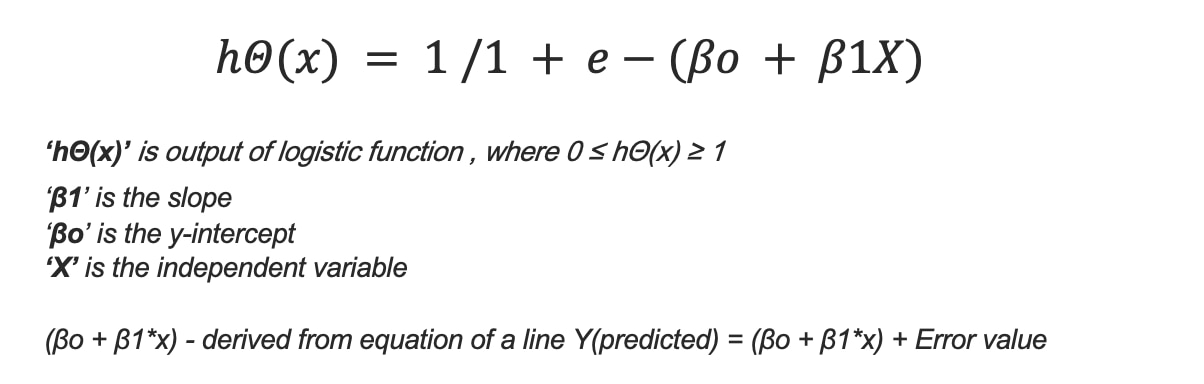
Training data that satisfies the below assumptions is usually a good fit for logistic regression.

* The predicted outcome is strictly binary or dichotomous. (This applies to binary logistic regression).
* The factors, or the independent variables, that influence the outcome are independent of each other. In other words there is little or no multicollinearity among the independent variables.
* The independent variables can be linearly related to the log odds.
* Fairly large sample sizes.

If your training data does not satisfy the above assumptions, logistic regression may not work for your use case.

Mathematics behind logistic regression:

Probability always ranges between 0 (does not happen) and 1 (happens). Using our Covid-19 example, in the case of binary classification, the probability of testing positive and not testing positive will sum up to 1. We use [logistic function or sigmoid function](https://en.wikipedia.org/wiki/Logistic_function) to calculate probability in logistic regression. The logistic function is a simple S-shaped curve used to convert data into a value between 0 and 1.



**Program:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

dataset = pd.read\_csv("purchase\_history.csv")

# input

x = dataset.iloc[:, [2, 3]].values

# output

y = dataset.iloc[:, 4].values

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.25, random\_state = 0)

from sklearn.preprocessing import StandardScaler

sc\_x = StandardScaler()

xtrain = sc\_x.fit\_transform(X\_train)

xtest = sc\_x.transform(X\_test)

print (xtrain[0:10, :])

from sklearn.linear\_model import LogisticRegression

classifier = LogisticRegression(random\_state = 0)

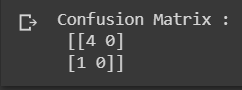
classifier.fit(X\_train, y\_train)

y\_pred = classifier.predict(X\_test)

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(y\_test, y\_pred)

print ("Confusion Matrix : \n", cm)

****

from sklearn.metrics import accuracy\_score

print ("Accuracy : ", accuracy\_score(y\_test, y\_pred))

****

from matplotlib.colors import ListedColormap

X\_set, y\_set = X\_test , y\_test

X1, X2 = np.meshgrid(np.arange(start = X\_set[:, 0].min() - 1,

                               stop = X\_set[:, 0].max() + 1, step = 0.01),

                     np.arange(start = X\_set[:, 1].min() - 1,

                               stop = X\_set[:, 1].max() + 1, step = 0.01))

plt.contourf(X1, X2, classifier.predict(

             np.array([X1.ravel(), X2.ravel()]).T).reshape(

             X1.shape), alpha = 0.75, cmap = ListedColormap(('red', 'green')))

plt.xlim(X1.min(), X1.max())

plt.ylim(X2.min(), X2.max())

for i, j in enumerate(np.unique(y\_set)):

    plt.scatter(X\_set[y\_set == j, 0], X\_set[y\_set == j, 1],

                c = ListedColormap(('red', 'green'))(i), label = j)

plt.title('Classifier (Test set)')

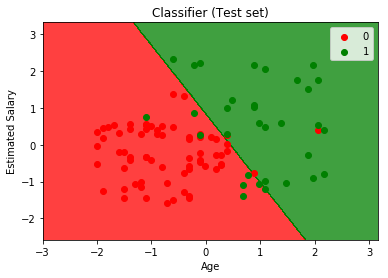
plt.xlabel('Age')

plt.ylabel('Estimated Salary')

plt.legend()

plt.show()

**Output:**



**Conclusion:**

Thus we have successfully implemented Logistic regression.

**Notebook Link:**

[**https://colab.research.google.com/drive/1heFS5St3AG6NW2gzQ-mGX5uuYyj-joVa#scrollTo=J2sLOfp5trXw**](https://colab.research.google.com/drive/1heFS5St3AG6NW2gzQ-mGX5uuYyj-joVa#scrollTo=J2sLOfp5trXw)