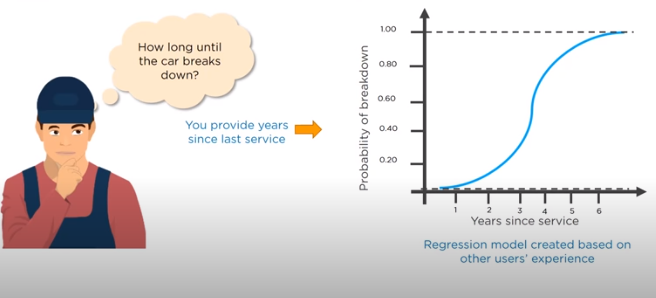
## 

## What is Logistic Regression?



Logistic Regression is a classification algorithm. It is used to predict a **binary outcome (1 / 0, Yes / No, True / False)** given a set of independent variables. To represent binary / **categorical outcome**, we use dummy variables.

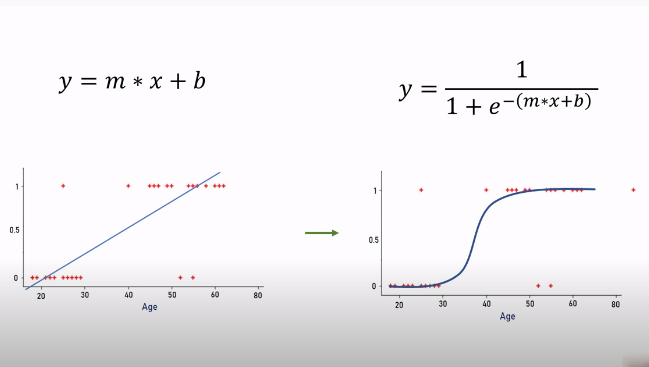
In [statistics](https://en.wikipedia.org/wiki/Statistics), logistic regression, or logit regression, or **logit mode** is a [regression](https://en.wikipedia.org/wiki/Regression_analysis) model where the [dependent variable (DV)](https://en.wikipedia.org/wiki/Dependent_and_independent_variables) is [categorical](https://en.wikipedia.org/wiki/Categorical_variable).

This covers the case of a [binary dependent variable](https://en.wikipedia.org/wiki/Binary_variable)—that is, where the output can take only two values, "0" and "1", which represent outcomes such as pass/fail, win/lose, alive/dead or healthy/sick. Cases where the dependent variable has more than two outcome categories may be analysed in [multinomial logistic regression](https://en.wikipedia.org/wiki/Multinomial_logistic_regression), or, if the multiple categories are [ordered](https://en.wikipedia.org/wiki/Level_of_measurement#Ordinal_type), in [ordinal logistic regression](https://en.wikipedia.org/wiki/Ordinal_logistic_regression).

Logistic regression may be used to predict the risk of developing a given disease (e.g. [diabetes](https://en.wikipedia.org/wiki/Diabetes_mellitus); [coronary heart disease](https://en.wikipedia.org/wiki/Coronary_artery_disease)), based on observed characteristics of the patient (age, sex, [body mass index](https://en.wikipedia.org/wiki/Body_mass_index), results of various [blood tests](https://en.wikipedia.org/wiki/Blood_test), etc.).

Another example might be to predict whether an American voter will vote **Democratic** or **Republican**, based on age, income, sex, race, state of residence, votes in previous elections, etc.

## Derivation of Logistic Regression Equation/Model Theory





Logistic Regression is part of a larger class of algorithms known as **Generalized Linear Model (GLM)**. In 1972, Nelder and Wedderburn proposed this model with an effort to provide a means of using linear regression to the problems which were not directly suited for application of linear regression. Infact, they proposed a class of different models (linear regression, ANOVA, Poisson Regression etc) which included logistic regression as a special case.

The fundamental equation of generalized linear model is:

g(E(y)) = α + βx1 + γx2

Here, g() is the link function, E(y) is the expectation of target variable and α + βx1 + γx2 is the linear predictor ( α,β,γ to be predicted). The role of link function is to ‘link’ the expectation of y to linear predictor.

Important Points

1. GLM **does not assume a linear relationship** between dependent and independent variables. However, it assumes a linear relationship between link function and independent variables in logit model.
2. The dependent variable **need not to be normally distributed**.
3. It **does not uses OLS** (Ordinary Least Square) for parameter estimation. Instead, it uses **maximum likelihood estimation (MLE).**
4. Errors need to be independent but not normally distributed.

 Let’s understand it further using an example:

We are provided a sample of 1000 customers. We need to predict the probability whether a customer will buy (y) a **particular magazine or not**. As you can see, we’ve a categorical outcome variable, we’ll use logistic regression.

To start with logistic regression, I’ll first write the simple linear regression equation with dependent variable enclosed in a link function:

       g(y) = βo + β(Age)         ---- (a)

Note: For ease of understanding, I’ve considered ‘Age’ as independent variable.

In logistic regression, we are only concerned about the **probability of outcome dependent variable** ( success or failure). As described above, g() is the link function. This function is established using two things: Probability of Success(p) and Probability of Failure(1-p). p should meet following criteria:

1. It must always be positive (since p >= 0)
2. It must always be less than equals to 1 (since p <= 1)

Now, we’ll simply satisfy these 2 conditions and get to the core of logistic regression. To establish link function, we’ll denote g() with ‘p’ initially and eventually end up deriving this function.

Since **probability must always be positive**, we’ll put the linear equation in exponential form. For any value of slope and dependent variable, exponent of this equation **will never be negative.**

p = exp(βo + β(Age)) = e^(βo + β(Age))    ------- (b)

To make the probability less than 1, we must divide p by a number greater than p. This can simply be done by:

p  =  exp(βo + β(Age)) / exp(βo + β(Age)) + 1   =   e^(βo + β(Age)) / e^(βo + β(Age)) + 1    ----- (c)

Using (a), (b) and (c), we can redefine the probability as:

              p = e^y/ 1 + e^y           --- (d)

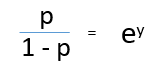
where p is the probability of success. This (d) is the Logit Function

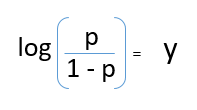
If p is the probability of success, 1-p will be the probability of failure which can be written as:

q = 1 - p = 1 - (e^y/ 1 + e^y)    --- (e)

where q is the probability of failure

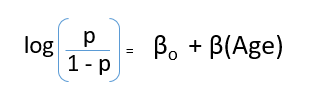
On dividing, (d) / (e), we get,



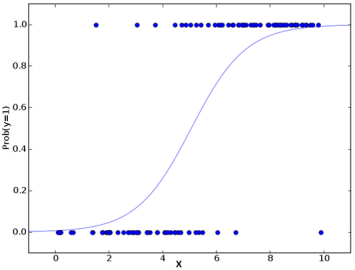
After taking log on both side, we get,  


**log(p/1-p) is the link function**. Logarithmic transformation on the outcome variable allows us to model a non-linear association in a linear way.

After substituting value of y, we’ll get:



This is the equation used in Logistic Regression. Here **(p/1-p) is the odd ratio**. Whenever the log of odd ratio is found **to be positive**, the probability of success is always **more than 50%.** A typical logistic model plot is shown below. You can see probability never goes below 0 and above 1.



import pandas as pd

import matplotlib.pyplot as plt

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

from sklearn.linear\_model import LogisticRegression

from sklearn import metrics

df = pd.read\_csv("/content/logit.csv")

df.info()

df.head(5)

plt.scatter(df.age,df.charges,color= 'red', marker='+')

plt.xlabel("Age of person")

plt.ylabel("Bought Insurance 1=Bought 0=Did not Buy")

x = df.iloc[:, 0:1].values

y = df.iloc[:, -1].values

#Split the dataset into train and test sets (70:30)

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x,y,test\_size=0.3,random\_state=42)

print(x\_test)

reg =LogisticRegression()

reg.fit(x\_train, y\_train)

yPrediction = reg.predict(x\_test) #Predict the test set

print()

print(y\_test)

plt.scatter(x\_test,y\_test,color='green', marker='\*')

plt.scatter(x\_test,yPrediction,color='blue', marker='.')

ins\_accuracy=accuracy\_score(y\_test,yPrediction)

print('insurance score:',ins\_accuracy\*100)

print(confusion\_matrix(y\_test,yPrediction))

print(classification\_report(y\_test,yPrediction))

o=reg.predict\_proba(x\_test)

print(o)

yPrediction1=reg.predict([[96.9]])

print()

print(yPrediction1)