LOGISTIC REGRESSION

Classification algorithm used to predict the probabilities that an observation belongs to one of two categories (a binary outcome, e.g yes/no, 1/0, spam/not spam)

#PROBLEM WITH LINEAR REGRESSION IN CLASSIFICATION:

Y = B0 + B1X1 + B2X2 ………… BpXp

The o/p Y is continuous value that ranges from -infinity to +infinity

However, in classification, we want o/p to be probability P(event) belongs to [0,1]. If we use Linear model, we could easily predict probability less than 0 or greater than 1, which is nonsensical.

# THE SOLUTION: We must transform linear outputs XB so that it is always constrained to [0,1] range. This transformation is achieved through **SIGMOID FUNCTION.**

#THE SIGMOID FUNCTION (LOGISTIC FUNCTION):

Activation function that converts the linear combination of inputs into a probability.

FORMULA:

The probability of an event occurring, pie, given by predictors X is defined as:

P(X) = pie = 1 / 1 + e^ (B0 + B1X1+……. BpXp)

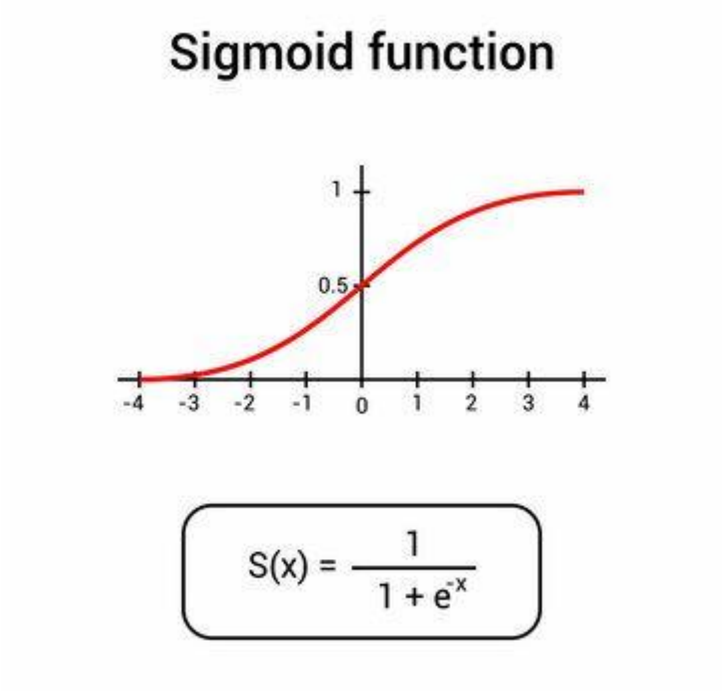
Where pie: probability of +ve class

E: Euler's number (base of logarithm, approx. 2.718)

Z = B0 + B1X1 + B2X2 ………… BpXp : linear combination of i/p variables and coeff. Also called **logit or score.**

Sigma(Z) = 1 / 1 + e^-Z

1. **INPUT (Z):** I/P is linear regression style score which can be any real number from -infinity to +infinity
2. **Exponentiation (e^-Z):** if Z is too large +ve no. -> e^-z **= e^-100** approx. 0
   If z is large -ve number -> e^-z = **e^100 ->** very large number
   If z = 0, **e^0 = 1**
3. **Denominator (1+e^-z):** This ensures denominator is always >=1
4. **Fraction (1/…)** : reciprocal of a number that is always >=1, final output pie is always constrained to a range (0,1).



If output of sigmoid function is less than 0.5 -> classify as 0 or No.
If output is 0.75 -> 75% chances of target value to be 1, yes or true
If more than 0.5(threshold) -> classified as yes or 1.

#Terminologies involved in Logistic Regression

Here are some common terms involved in logistic regression:

1. **Independent Variables:** These are the input features or predictor variables used to make predictions about the dependent variable.
2. **Dependent Variable**: This is the target variable that we aim to predict. In logistic regression, the dependent variable is categorical.
3. **Logistic Function**: This function transforms the independent variables into a probability between 0 and 1 which represents the likelihood that the dependent variable is either 0 or 1.
4. **Odds**: This is the ratio of the probability of an event happening to the probability of it not happening. It differs from probability because probability is the ratio of occurrences to total possibilities.
5. **Log-Odds (Logit)**: The natural logarithm of the odds. In logistic regression, the log-odds are modeled as a linear combination of the independent variables and the intercept.
6. **Coefficient**: These are the parameters estimated by the logistic regression model which shows how strongly the independent variables affect the dependent variable.
7. **Intercept**: The constant term in the logistic regression model which represents the log-odds when all independent variables are equal to zero.
8. **Maximum Likelihood Estimation (MLE)**: This method is used to estimate the coefficients of the logistic regression model by maximizing the likelihood of observing the given data.

Types of Logistic Regression

Logistic regression can be classified into three main types based on the nature of the dependent variable:

1. **Binomial Logistic Regression**: This type is used when the dependent variable has only two possible categories. Examples include Yes/No, Pass/Fail or 0/1. It is the most common form of logistic regression and is used for binary classification problems.
2. **Multinomial Logistic Regression**: This is used when the dependent variable has three or more possible categories that are not ordered. For example, classifying animals into categories like "cat," "dog" or "sheep." It extends the binary logistic regression to handle multiple classes.
3. **Ordinal Logistic Regression**: This type applies when the dependent variable has three or more categories with a natural order or ranking. Examples include ratings like "low," "medium" and "high." It takes the order of the categories into account when modelling.

#Implementation for Logistic Regression

Now, let's see the implementation of logistic regression in Python. Here we will be implementing two main types of Logistic Regression:

1. Binomial Logistic regression:

In binomial logistic regression, the target variable can only have two possible values such as "0" or "1", "pass" or "fail". The sigmoid function is used for prediction.

We will be using sckit-learn library for this and shows how to use the breast cancer dataset to implement a Logistic Regression model for classification.

```
from sklearn.datasets import load_breast_cancer
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

```
X, y = load_breast_cancer(return_X_y=True)

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.20, random_state=23)

clf = LogisticRegression(max_iter=10000, random_state=0)
clf.fit(X_train, y_train)

acc = accuracy_score(y_test, clf.predict(X_test)) * 100
print(f"Logistic Regression model accuracy: {acc:.2f}%")
```

Output:

Logistic Regression model accuracy (in %): 96.49%

This code uses logistic regression to classify whether a sample from the breast cancer dataset is malignant or benign.

2. Multinomial Logistic Regression:

Target variable can have 3 or more possible types which are not ordered i.e types have no quantitative significance like “disease A” vs “disease B” vs “disease C”.

In this case, the softmax function is used in place of the sigmoid function. Softmax function for K classes will be:

Below is an example of implementing multinomial logistic regression using the Digits dataset from scikit-learn:

```
from sklearn.model_selection import train_test_split
from sklearn import datasets, linear_model, metrics

digits = datasets.load_digits()
```

```
X = digits.data
y = digits.target

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.4, random_state=1)

reg = linear_model.LogisticRegression(max_iter=10000,
random_state=0)
reg.fit(X_train, y_train)

y_pred = reg.predict(X_test)

print(f"Logistic Regression model accuracy:
{metrics.accuracy_score(y_test, y_pred) * 100:.2f}%")
```

Output:

Logistic Regression model accuracy: 96.66%

This model is used to predict one of 10 digits (0-9) based on the image features.

How to Evaluate Logistic Regression Model?

Evaluating the logistic regression model helps assess its performance and ensure it generalizes well to new, unseen data. The following metrics are commonly used:

1. Accuracy: Accuracy provides the proportion of correctly classified instances.

Accuracy=TruePositives+TrueNegatives/Total

2. Precision: Precision focuses on the accuracy of positive predictions.

Precision=TruePositives/TruePositives+FalsePositivesPrecision

3. Recall (Sensitivity or True Positive Rate): Recall measures the proportion of correctly predicted positive instances among all actual positive instances.

Recall=TruePositives/TruePositives+FalseNegativesRecall

4. F1 Score: F1 score is the harmonic mean of precision and recall.
F1Score=2∗Precision∗Recall/Precision+Recall
5. Area Under the Receiver Operating Characteristic Curve (AUC-ROC): The ROC curve plots the true positive rate against the false positive rate at various thresholds. AUC-ROC measures the area under this curve which provides an aggregate measure of a model's performance across different classification thresholds.
6. Area Under the Precision-Recall Curve (AUC-PR): Similar to AUC-ROC, AUC-PR measures the area under the precision-recall curve helps in providing a summary of a model's performance across different precision-recall trade-offs.