**Logistic Regression in Python**

Classification techniques are an essential part of machine learning and data mining applications. Approximately 70% of problems in Data Science are classification problems. There are lots of classification problems that are available, but the logistics regression is common and is a useful regression method for solving the binary classification problem. Another category of classification is Multinomial classification, which handles the issues where multiple classes are present in the target variable. For example, IRIS dataset a very famous example of multi-class classification.

Logistic Regression can be used for various classification problems such as spam detection. Diabetes prediction, if a given customer will purchase a particular product or will they churn another competitor, whether the user will click on a given advertisement link or not, and many more examples are in the bucket.

Logistic Regression is one of the simplest and commonly used Machine Learning algorithms for two-class classification. It is easy to implement and can be used as the baseline for any binary classification problem. Its basic fundamental concepts are also constructive in deep learning. Logistic regression describes and estimates the relationship between one dependent binary variable and independent variables.

In this example, you will learn the following things in Logistic Regression:

* Introduction to Logistic Regression
* Linear Regression Vs. Logistic Regression
* Maximum Likelihood Estimation Vs. Ordinary Least Square Method
* How do Logistic Regression work?
* Model building in Scikit-learn
* Model Evaluation using Confusion Matrix.
* Advantages and Disadvantages of Logistic Regression

### Logistic Regression

Logistic regression is a statistical method for predicting binary classes. The outcome or target variable is dichotomous in nature. Dichotomous means there are only two possible classes. For example, it can be used for cancer detection problems. It computes the probability of an event occurrence.

It is a special case of linear regression where the target variable is categorical in nature. It uses a log of odds as the dependent variable. Logistic Regression predicts the probability of occurrence of a binary event utilizing a logit function.

Linear Regression Equation:



Where, y is dependent variable and x1, x2 ... and Xn are explanatory variables.

### Linear Regression Vs. Logistic Regression

Linear regression gives you a continuous output, but logistic regression provides a constant output. An example of the continuous output is house price and stock price. Examples of the discrete output is predicting whether a patient has cancer or not, predicting whether the customer will churn. Linear regression is estimated using Ordinary Least Squares (OLS) while logistic regression is estimated using Maximum Likelihood Estimation (MLE) approach.



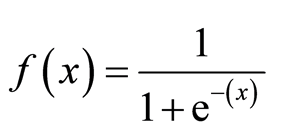
### Maximum Likelihood Estimation Vs. Least Square Method

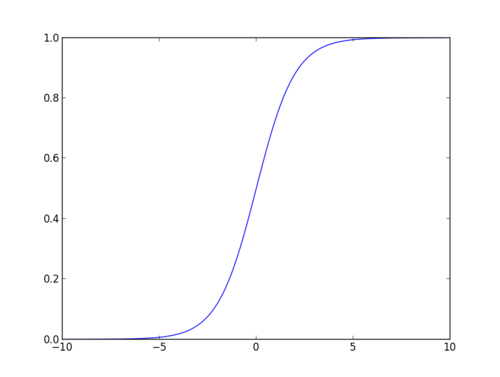
The MLE is a "likelihood" maximization method, while OLS is a distance-minimizing approximation method. Maximizing the likelihood function determines the parameters that are most likely to produce the observed data. From a statistical point of view, MLE sets the mean and variance as parameters in determining the specific parametric values for a given model. This set of parameters can be used for predicting the data needed in a normal distribution.

Ordinary Least squares estimates are computed by fitting a regression line on given data points that has the minimum sum of the squared deviations (least square error). Both are used to estimate the parameters of a linear regression model. MLE assumes a joint probability mass function, while OLS doesn't require any stochastic assumptions for minimizing distance.

### Sigmoid Function

The sigmoid function, also called logistic function gives an ‘S’ shaped curve that can take any real-valued number and map it into a value between 0 and 1. If the curve goes to positive infinity, y predicted will become 1, and if the curve goes to negative infinity, y predicted will become 0. If the output of the sigmoid function is more than 0.5, we can classify the outcome as 1 or YES, and if it is less than 0.5, we can classify it as 0 or NO. For example: If the output is 0.75, we can say in terms of probability as: There is a 75 percent chance that patient will suffer from cancer.





### Types of Logistic Regression

Types of Logistic Regression:

* Binary Logistic Regression: The target variable has only two possible outcomes such as Spam or Not Spam, Cancer or No Cancer.
* Multinomial Logistic Regression: The target variable has three or more nominal categories such as predicting the type of Wine.
* Ordinal Logistic Regression: the target variable has three or more ordinal categories such as restaurant or product rating from 1 to 5.

### We’ll be using Machine Learning to predict whether a person has diabetes or not, based on information about the patient such as blood pressure, body mass index (BMI), age, etc.

The following features have been provided to help us predict whether a person is diabetic or not:

* **Pregnancies:**Number of times pregnant
* **Glucose:** Plasma glucose concentration over 2 hours in an oral glucose tolerance test
* **BloodPressure:**Diastolic blood pressure (mm Hg)
* **SkinThickness:** Triceps skin fold thickness (mm)
* **Insulin:** 2-Hour serum insulin (mu U/ml)
* **BMI:** Body mass index (weight in kg/(height in m)2)
* **DiabetesPedigreeFunction:** Diabetes pedigree function (a function which scores likelihood of diabetes based on family history)
* **Age:** Age (years)
* **Outcome:** Class variable (0 if non-diabetic, 1 if diabetic)

### **Model building in Scikit-learn**

Let's build the diabetes prediction model.

Here, you are going to predict diabetes using Logistic Regression Classifier.

#### Loading Data

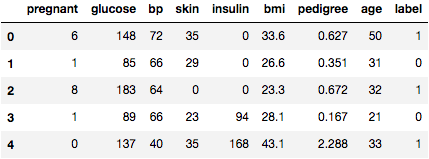
#import pandas

import pandas as pd

# load dataset

diab = pd.read\_csv(r"E:\Backup 14.03.2021\Desktop\Imarticus Post\Online DSP\Python\diabetes.csv")

diab.head()



#### Selecting Feature

Here, you need to divide the given columns into two types of variables dependent(or target variable) and independent variable(or feature variables).

#split dataset in features and target variable

feature\_cols = ['Pregnancies', 'Insulin', 'BMI', ‘Age’, ‘Glucose’, ’BloodPressure’, 'DiabetesPedigreeFunction']

X = diab[feature\_cols] # Features

y = diab.Outcome # Target variable

#### Splitting Data

To understand model performance, dividing the dataset into a training set and a test set is a good strategy.

Let's split dataset by using function train\_test\_split(). You need to pass 3 parameters features, target, and test\_set size. Additionally, you can use random\_state to select records randomly.

# split X and y into training and testing sets

from sklearn.cross\_validation import train\_test\_split

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

Here, the Dataset is broken into two parts in a ratio of 75:25. It means 75% data will be used for model training and 25% for model testing.

#### Model Development and Prediction

First, import the Logistic Regression module and create a Logistic Regression classifier object using LogisticRegression() function.

Then, fit your model on the train set using fit() and perform prediction on the test set using predict().

# import the class

from sklearn.linear\_model import LogisticRegression

# instantiate the model (using the default parameters)

logreg = LogisticRegression(solver='lbfgs', max\_iter=400)

#solver='lbfgs' - Algorithm to use in the optimization problem

#max\_iter=400 - Maximum number of iterations taken for the solvers to converge

# fit the model with data

logreg.fit(X\_train,y\_train)

y\_pred=logreg.predict(X\_test)

### Model Evaluation using Confusion Matrix

A confusion matrix is a table that is used to evaluate the performance of a classification model. You can also visualize the performance of an algorithm. The fundamental of a confusion matrix is the number of correct and incorrect predictions are summed up class-wise.

# import the metrics class

from sklearn import metrics

cnf\_matrix = metrics.confusion\_matrix(y\_test, y\_pred)

cnf\_matrix

array([[119, 11],

[ 26, 36]])

Here, you can see the confusion matrix in the form of the array object. The dimension of this matrix is 2\*2 because this model is binary classification. You have two classes 0 and 1. Diagonal values represent accurate predictions, while non-diagonal elements are inaccurate predictions. In the output, 118 and 36 are actual predictions, and 26 and 12 are incorrect predictions.

#### Visualizing Confusion Matrix using Heatmap

Let's visualize the results of the model in the form of a confusion matrix using matplotlib and seaborn.

Here, you will visualize the confusion matrix using Heatmap.

# import required modules

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

%matplotlib inline

class\_names=[0,1] # name of classes

fig, ax = plt.subplots()

tick\_marks = np.arange(len(class\_names))

#The arange() function is used to get evenly spaced values within a given interval

plt.xticks(tick\_marks, class\_names)

plt.yticks(tick\_marks, class\_names)

# create heatmap

sns.heatmap(pd.DataFrame(cnf\_matrix), annot=True, cmap="YlGnBu" ,fmt='g')

#heatmap() fmt parameter – add text on each cell. The annot only help to add #numeric value on python heatmap cell but fmt parameter allows to add string (text) #values on the cell.

#'g' General format

ax.xaxis.set\_label\_position("top")

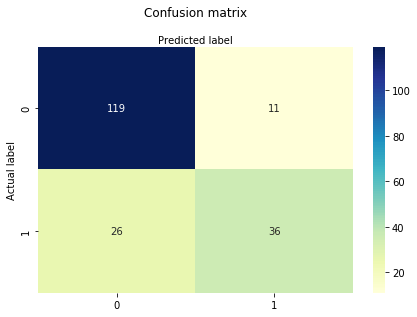
plt.tight\_layout()

plt.title('Confusion matrix', y=1.1)

plt.ylabel('Actual label')

plt.xlabel('Predicted label')

Text(0.5,257.44,'Predicted label')



#### Confusion Matrix Evaluation Metrics

Let's evaluate the model using model evaluation metrics such as accuracy, precision, and recall.

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

print("Precision:",metrics.precision\_score(y\_test, y\_pred))

print("Recall:",metrics.recall\_score(y\_test, y\_pred))

Accuracy: 0.8072916666666666

Precision: 0.7659574468085106

Recall: 0.5806451612903226

We got a classification rate of 80%, considered as good accuracy.

Precision: Precision is about being precise, i.e., how accurate your model is. In other words, you can say, when a model makes a prediction, how often it is correct. In your prediction case, when your Logistic Regression model predicted patients are going to suffer from diabetes, that patients have 76% of the time.

Recall: If there are patients who have diabetes in the test set and your Logistic Regression model can identify it 58% of the time.

#### ROC Curve

Receiver Operating Characteristic(ROC) curve is a plot of the true positive rate against the false positive rate. It shows the tradeoff between sensitivity and specificity.

y\_pred\_proba = logreg.predict\_proba(X\_test)[::,1]

#The function predict\_proba() returns a numpy array of two columns. The first column #is the probability that target=0 and the second column is the probability that target=1 #That is why we add [:,1] after predict\_proba() in order to get the probabilities of #target=1 .

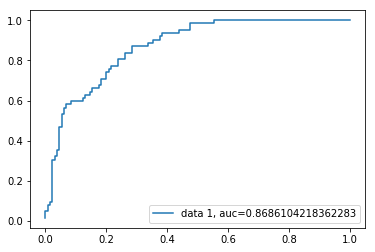
fpr, tpr, \_ = metrics.roc\_curve(y\_test, y\_pred\_proba)

auc = metrics.roc\_auc\_score(y\_test, y\_pred\_proba)

plt.plot(fpr,tpr,label="data 1, auc="+str(auc))

plt.legend(loc=4)

plt.show()



#The TPR defines how many correct positive results occur among all positive #samples available during the test

#FPR, on the other hand, defines how many incorrect positive results occur #among all negative samples available during the test

#Threshold corresponds to a point on the ROC curve that is colinear with #adjacent points

AUC score for the case is 0.86. AUC score 1 represents perfect classifier, and 0.5 represents a worthless classifier.

### Advantages

Because of its efficient and straightforward nature, doesn't require high computation power, easy to implement, easily interpretable, used widely by data analyst and scientist. Also, it doesn't require scaling of features. Logistic regression provides a probability score for observations.

### Disadvantages

Logistic regression is not able to handle a large number of categorical features/variables. It is vulnerable to overfitting. Also, can't solve the non-linear problem with the logistic regression that is why it requires a transformation of non-linear features. Logistic regression will not perform well with independent variables that are not correlated to the target variable and are very similar or correlated to each other.