## 1.1. Mô tả bài toán

Nhấp chuột là một số liệu tiếp thị tính số lượng khách hàng đã tương tác với quảng cáo của bạn. Nhấp chuột thể hiện phần trăm nhấp chuột trên tổng số lần xem quảng cáo. Nó xác định tỉ lệ thành công của chiến dịch tiếp thị của bạn. Số lượng nhấp chuột cao hơn là tín hiệu cho thấy đối tượng của bạn đang nhận được thương mại / quảng cáo thích hợp và lợi tức đầu tư (ROI) cao hơn cho công ty của bạn. Vì vậy, mục đích của phân tích này là để dự đoán ai và tại sao người dùng Internet sẽ nhấp vào quảng cáo của bạn.

Trong báo cáo này, em sử dụng kỹ thuật học máy sử dụng là Logistic Regression để giải bài toán phân lớp nhị phân, dự đoán người dùng có khả năng click vào quảng cáo hay không dựa trên các đặc điểm của họ. Phương pháp đánh giá mô hình học máy được sử dụng là đo độ chính xác Accuracy và Confusion Matrix.

## 1.2 Thu thập và mô tả dữ liệu

Trong bài toán dự đoán quảng cáo này, em đã tìm hiểu và thu thập đươc tập dữ liệu các đặc điểm của người dùng Internet. Tập dữ liệu nói trên được em thu thập từ **https://www.kaggle.com/fayomi/advertising.** Tập dữ liệu bao gồm 1001 dòng mẫu với 10 cột đặc điểm của người dùng.

**1.2.1. Mô tả thuộc tính**

|  |  |  |
| --- | --- | --- |
| **STT** | **Thuộc tính** | **Mô tả** |
| 1 | Daily Time Spent on Site | Thời gian người dùng sử dụng trang web tính bằng phút |
| 2 | Age | Tuổi của người dùng |
| 3 | Area Income | Thu thập trung bình theo khu vực địa lý của người dùng |
| 4 | Daily Internet Usage | Trung bình thời gian người dùng truy cập Internet mỗi ngày tính bằng phút |
| 5 | Ad Topic Line | Tiêu đề của bài quảng cáo |
| 6 | City | Thành phố của người dùng |
| 7 | Male | Người dùng có phải nam hay không? 1 là đúng, 0 là sai |
| 8 | Country | Đất nước của người dùng |
| 9 | Timestamp | Thời gian người tiêu dùng nhấp vào Quảng cáo hoặc đóng cửa sổ |
| 10 | Clicked on Ad | 1 là click vào quảng cáo, 0 là không click vào quảng cáo |

Với 10 đặc điểm của người dùng Internet như trên, để phù hợp với yêu cầu là dự đoán người dùng nào sẽ click vào quảng cáo, chúng em chọn thuộc tính **Y là Clicked on Ad** ở đây làm biến mục tiêu cần tìm, các thuộc tính còn lại làm biến giải thích.

**1.2.2. Số lượng mẫu**

Tập dữ liệu của em bao gồm 1001 dòng tương ứng với số lượng mẫu. Dưới đây là một số mẫu có trong tập dữ liệu:

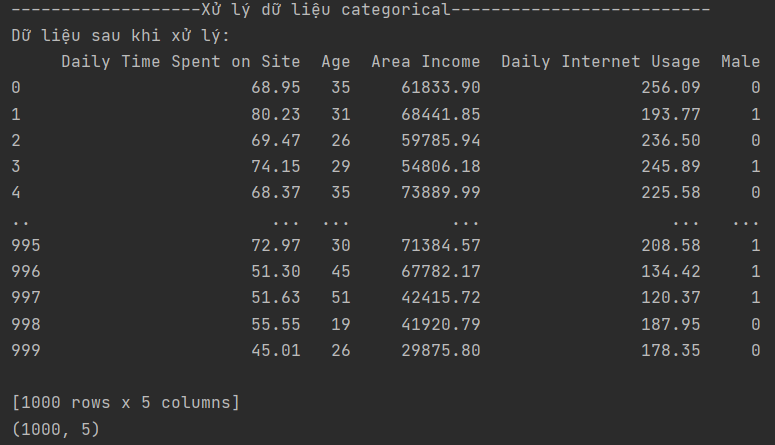
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Daily Time Spent on Site** | **Age** | **Area Income** | **Daily Internet Usage** | **Ad Topic Line** | **City** | **Male** | **Country** | **Timestamp** | **Clicked on Ad** |
| 68.95 | 35 | 61833.9 | 256.09 | Cloned 5thgeneration orchestration | Wrightburgh | 0 | Tunisia | 27/03/2016 0:53 | 0 |
| 80.23 | 31 | 68441.85 | 193.77 | Monitored national standardization | West Jodi | 1 | Nauru | 04/04/2016 1:39 | 0 |
| 69.47 | 26 | 59785.94 | 236.5 | Organic bottom-line service-desk | Davidton | 0 | San Marino | 13/03/2016 20:35 | 0 |
| 74.15 | 29 | 54806.18 | 245.89 | Triple-buffered reciprocal time-frame | West Terrifurt | 1 | Italy | 10/01/2016 2:31 | 0 |
| 68.37 | 35 | 73889.99 | 225.58 | Robust logistical utilization | South Manuel | 0 | Iceland | 03/06/2016 3:36 | 0 |
| 59.99 | 23 | 59761.56 | 226.74 | Sharable client-driven software | Jamieberg | 1 | Norway | 19/05/2016 14:30 | 0 |
| 88.91 | 33 | 53852.85 | 208.36 | Enhanced dedicated support | Brandonstad | 0 | Myanmar | 28/01/2016 20:59 | 0 |

## 1.3. Xử lí dữ liệu

**Bước 1:** đọc file dữ liệu ở dạng csv, xử lí các cột dữ liệu ở dạng object: “Ad Topic Line”, “City”, “Country”, “Timestamp”. Chọn biến mục tiêu y = “Clicked on Ads”

**Code và kết quả:**

from sklearn import preprocessing  
df = pd.read\_csv("advertising.csv")  
print('-------------------Xử lý dữ liệu categorical--------------------------')  
  
X= df.drop(["Clicked on Ad","Timestamp","City","Country","Ad Topic Line"], axis = 1)y = df["Clicked on Ad"]  
print('Dữ liệu sau khi xử lý:')  
print(X)  
print (X.shape)

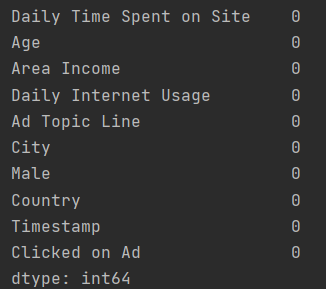
****

**Bước 2:** Kiểm tra dữ liệu thiếu

**Code và kết quả:**

print ('---------------------Kiểm tra missing value---------------------------')

print(df.isnull().sum()) #Kiểm tra missing value

****

**Bước 3:** Chia tập dữ liệu thành 2 tập train và test với train = 0.7, test = 0.3

**Code và kết quả:**

print ('-------------------------Chia dữ liệu train-test----------------------')

from sklearn.model\_selection import train\_test\_split #chia tập dữ liêu train-test

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=35)

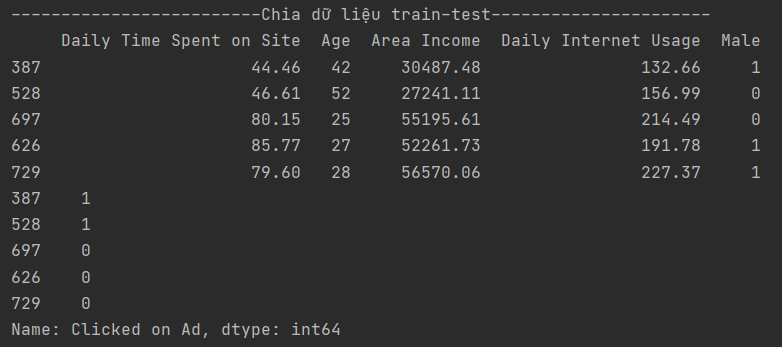
print (X\_train.head(5))

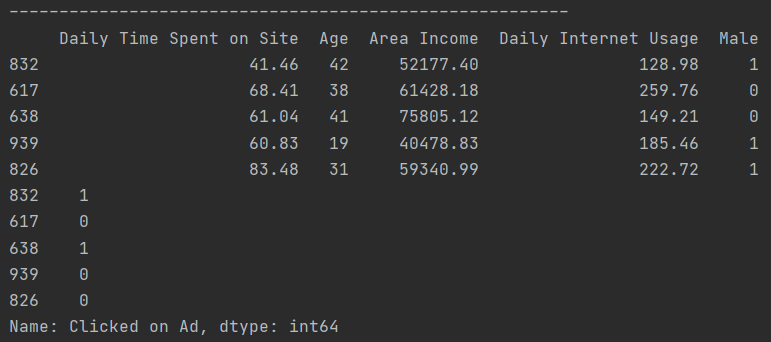
print (y\_train.head(5))

print ('--------------------------------------------------------')

print (X\_test.head(5))

print (y\_test.head(5))

****

****

**Bước 4:** Kiểm tra dữ liệu có bị mất cân bằng hay không?

**Code và kết quả:**

print ('---------------------Cân bằng SMOTE---------------------')

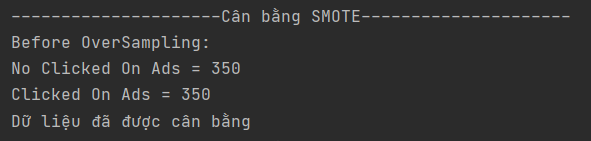
#Cân bằng smote

print ('Before OverSampling:')

print ('no =', sum (y\_train == 0))

print ('yes =', sum (y\_train == 1))

print('Dữ liệu đã được cân bằng')



## 1.4. Kỹ thuật học máy sử dụng

Với bài toán trên, em sử dụng kỹ thuật học máy Logistic Regression. Logistic Regression là 1 thuật toán phân loại được dùng để gán các đối tượng cho 1 tập hợp giá trị rời rạc (như 0, 1, ...). Một ví dụ điển hình là phân loại Email, gồm có email công việc, email gia đình, email spam,... Giao dịch trực tuyến có là an toàn hay không an toàn, khối u lành tính hay ác tình. Thuật toán trên dùng hàm sigmoid logistic để đưa ra đánh giá theo xác suất.

Kỹ thuật Logistic Regression sử dụng hàm Sigmoid làm hàm dự đoán. Hàm này có dạng:

https://images.viblo.asia/43965a36-f47e-4f23-8d05-695a315f7ce5.png

**Code và kết quả:**

print ('--------------------Chạy kỹ thuật học máy----------------')

#Kỹ thuật học máy Logistic Regression

from sklearn.linear\_model import LogisticRegression

#Bước 2. Huấn luyện (với tập dữ liệu train X\_train, y\_train)

model = LogisticRegression(max\_iter = 10000).fit(X\_train,y\_train)

#Bước 3. Dự đoán (với tập dữ liệu train X\_test, y\_test)

y\_pred = model.predict(X\_test) #y\_prediction là y dự đoán được

print ("Hệ số w:", model.coef\_)

print (model.coef\_.shape)

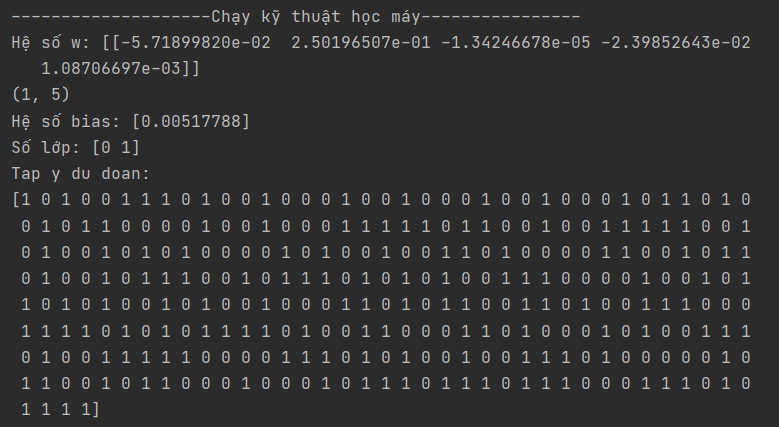
print ("Hệ số bias:", model.intercept\_)

print ("Số lớp:", model.classes\_)

#print ("Số vòng lặp", log\_model.n\_iter\_)

print ("Tap y du doan:")

print (y\_pred)

****

**1.5. Phương pháp đánh giá độ chính xác**

Phương pháp đánh giá độ chính xác được sử dụng trong bài toán này là Accuracy và Confusion Matrix.

**Accuracy:** Cách đánh giá này tính tỉ lệ giữa số điểm được dự đoán đúng và tổng số điểm trong tập dữ liệu kiểm thử. Giả sử, ta đếm được 7 điểm dữ liệu được dự đoán đúng trên tổng số 10 điểm. Như vậy, Accuracy Score của bài toán sẽ là 0.7 (hay 70%).

**Confusion Matrix:** Cách tính sử dụng Accuracy như ở trên chỉ cho chúng ta biết được bao nhiêu phần trăm lượng dữ liệu được phân loại đúng mà không chỉ ra được cụ thể mỗi loại được phân loại như thế nào, lớp nào được phân loại đúng nhiều nhất, và dữ liệu thuộc lớp nào thường bị phân loại nhầm vào lớp khác. Để có thể đánh giá được các giá trị này, chúng ta sử dụng một ma trận được gọi là confusion matrix. Về cơ bản, confusion matrix thể hiện có bao nhiêu điểm dữ liệu thực sự thuộc vào một class, và được dự đoán là rơi vào một class.

**Code và kết quả:**

print ('------------------Đánh giá mô hình học máy----------------')

#Bước 4: Đánh giá mô hình học dựa trên kết quả dự đoán

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

accuracy = accuracy\_score(y\_test, y\_pred)

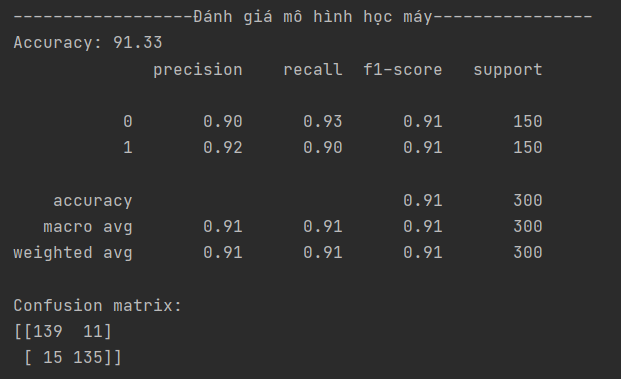
print ('Accuracy: %.2f' % (accuracy \* 100))

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

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

print ('Confusion matrix:')

print(cnf\_matrix)



**1.7. Kết luận**

Logistic Regression chạy khá nhanh và chuẩn xác, tuy nhiên nhược điểm của nó cũng như các thuật toán phân lớp khác như SVM, DecisionTreeClassifier, Random Forest là rất nhạy cảm với dữ liệu mất cân bằng. Kết quả khi chạy có thể rất đẹp nhưng mô hình dự đoán thường không đáng tin cậy. Chính vì thế, để đảm bảo dữ liệu được huấn luyện tốt và cho ra dự đoán chính xác, cần tiến hành xử lý mất cân bằng ngay từ khâu ban đầu trước khi chạy mô hình dự đoán kết quả.