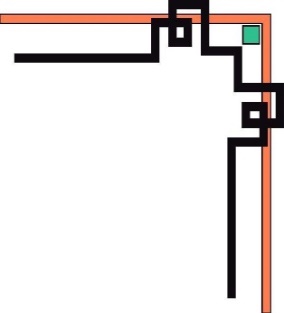
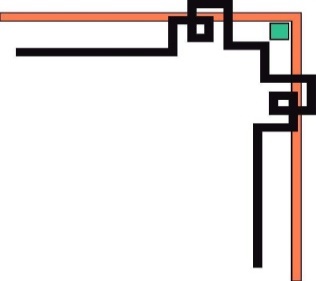
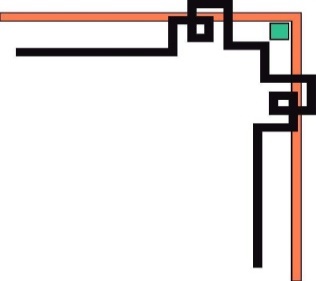
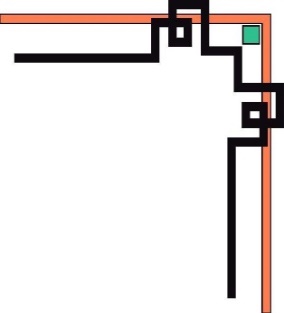
**BỘ GIÁO DỤC VÀ ĐÀO TẠO**



**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN MÔN NHẬP MÔN PYTHON**

**CHO PHÂN TÍCH**

*Phân tích dữ liệu theo*

*sklearn.linear\_model.LogisticRegression*

**SINH VIÊN THỰC HIỆN:**

**1. TRẦN TIẾN ĐỨC 18133007**

**2. LƯƠNG UY LONG 18133026**

**3. TRẦN NHƯ THUẬN 18133054**

**GIẢNG VIÊN HƯỚNG DẪN:**

**ThS.TỪ TUYẾT HỒNG**

**TP. HỒ CHÍ MINH – Tháng 6/2020**

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

Giảng viên hướng dẫn

Từ Tuyết Hồng

**LỜI CẢM ƠN**

Để hoàn thành tốt bài tập lớn này, nhóm em xin chân thành gửi lời cảm ơn sâu sắc tới Cô Từ Tuyết Hồng, giáo viên hướng dẫn môn Nhập môn lập trình Python cho phân tích. Cô đã tận tâm hướng dẫn nhóm em qua từng buổi học trên lớp cũng như những buổi nói chuyện, thảo luận về lĩnh vực sáng tạo trong nghiên cứu khoa học.Nếu không có những lời hướng dẫn, chỉ bảo của cô thì bài tập lớn môn học này của nhóm em rất khó có thể hoàn thiện được.

Trong quá trình làm bài tập lớn cuối kỳ này, do trình độ cũng như kinh nghiệm thực tiễn còn hạn chế nên không thể tránh khỏi những thiếu sót, nhóm em rất mong nhận được ý kiến đóng góp của cô để nhóm em có thể học hỏi được thêm nhiều bài học và kinh nghiệm trên con đường học vấn.

**Mục lục**

[**Chương 1: Tổng quan đề tài 6**](#_Toc44355780)

[*1.* *Giới thiệu chung* 6](#_Toc44355781)

[**1.1. Yêu cầu đề tài 6**](#_Toc44355782)

[**1.2. Phân tích đề tài 6**](#_Toc44355783)

[*2.* *Hướng nghiên cứu* 6](#_Toc44355784)

[**2.1. Tìm hiểu Logistic Regression 6**](#_Toc44355785)

[**2.2. Dữ liệu đầu vào 6**](#_Toc44355786)

[**2.3. Dữ liệu đầu ra 7**](#_Toc44355787)

[**2.4. Mục đích nghiên cứu 7**](#_Toc44355788)

[*3.* *Tài liệu tham khảo* 8](#_Toc44355789)

[**Chương 2: Phân công công việc 9**](#_Toc44355790)

[**Chương 3: Nội dung chi tiết 10**](#_Toc44355791)

[*1.* *Các thư viện sử dụng* 10](#_Toc44355792)

[*2.* *Data exploration* 11](#_Toc44355793)

[**2.1. Kiểm tra thông tin dữ liệu 12**](#_Toc44355794)

[**2.2. Tiến hành lọc dữ liệu có giá trị null (nếu có) 13**](#_Toc44355795)

[**2.3. Biến đổi trường Gender thành biến nhị phân (Chỉ giá trị 0 hoặc 1) 13**](#_Toc44355796)

[**2.4. Lọc và kiểm tra đối tượng theo độ tuổi 15**](#_Toc44355797)

[**2.5. Nhóm dữ liệu theo trường "Age" 17**](#_Toc44355798)

[**2.6. Nhóm dữ liệu theo trường "Purchased" 19**](#_Toc44355799)

[*3.* *Visualizations* 22](#_Toc44355800)

[**3.1. Trực quan số người theo độ tuổi bằng Matplotlib 23**](#_Toc44355801)

[**3.2. Trực quan hoá dữ liệu của các biến được tính toán bằng Seaborn 23**](#_Toc44355802)

[**3.3. Số người mua hàng xếp theo giới tính 24**](#_Toc44355803)

[**3.4. Số người mua hàng xếp theo độ tuổi 26**](#_Toc44355804)

[*4.* *Over-sampling using SMOTE* 29](#_Toc44355805)

[*5.* *Implementing the model* 52](#_Toc44355806)

[*6.* *Logistic Regression Model Fitting* 53](#_Toc44355807)

[*7.* *Confusion Matrix* 53](#_Toc44355808)

[*9.* *Compute precision, recall, F-measure and support* 55](#_Toc44355809)

[*10.* *ROC Curve* 56](#_Toc44355810)

[**Chương 4: Kết luận 58**](#_Toc44355811)

[*1.* *Nhận xét, đánh giá.* 58](#_Toc44355812)

[*2.* *Kì vọng* 58](#_Toc44355813)

1. Tổng quan đề tài
   1. Giới thiệu chung
      1. Yêu cầu đề tài

- Tìm hiểu sklearn.linear\_model.LogisticRegression và áp dụng phân loại dữ liệu trong tập dữ liệu xã hội: http://www.mediafire.com/file/ymu5r0baapzhevj

* + 1. Phân tích đề tài

Phân tích và tính toán các dữ liệu của bảng Xahoi.csv theo mô hình hồi quy Logictis

* 1. Hướng nghiên cứu
     1. Tìm hiểu Logistic Regression

Logistic Regression (hay còn gọi là Logit Regression) được sử dụng phổ biến để ước lượng xác suất 1 điểm dữ liệu có thể thuộc về 1 lớp nào đó (ví dụ tính xác suất để 1 email là spam). Nếu xác suất > 50% thì mô hình dự đoán có thể khẳng định được điểm dữ liệu đó thuộc về lớp 1(nhãn là 1) hoặc ngược lại (nhãn là 0). Việc này được gọi là phân loại nhị phân (chỉ có 0 với 1).

* + 1. Dữ liệu đầu vào

Một tập dữ liệu xã hội gồm 400 dòng của các thuộc tính User ID, Gender, Age, Estimated, Salary, Purchased có dạng file Xahoi.csv.

|  |  | **User ID** | **Gender** | **Age** | **EstimatedSalary** | **Purchased** |
| --- | --- | --- | --- | --- | --- | --- |
|  | **0** | 15624510 | 0 | 19 | 19000 | 0 |
|  | **1** | 15810944 | 0 | 35 | 20000 | 0 |
|  | **2** | 15668575 | 1 | 26 | 43000 | 0 |
|  | **3** | 15603246 | 1 | 27 | 57000 | 0 |
|  | **4** | 15804002 | 0 | 19 | 76000 | 0 |
|  | **...** | ... | ... | ... | ... | ... |
|  | **395** | 15691863 | 1 | 46 | 41000 | 1 |
|  | **396** | 15706071 | 0 | 51 | 23000 | 1 |
|  | **397** | 15654296 | 1 | 50 | 20000 | 1 |
|  | **398** | 15755018 | 0 | 36 | 33000 | 0 |
|  | **399** | 15594041 | 1 | 49 | 36000 | 1 |

* + 1. Dữ liệu đầu ra

Dữ liệu đã được phân tích theo hướng hồi quy logictis

* + 1. Mục đích nghiên cứu

Tìm hiểu và thực hành những kiến thức cơ bản về phân tích dữ liệu trong Python

Phân tích dữ liệu cơ bản từ bảng đã được cho trước

Sử dụng được Sklearn

* 1. Tài liệu tham khảo

https://towardsdatascience.com/building-a-logistic-regression-in-python-step-by-step-becd4d56c9c8?gi=52dbb8da7819

1. **Phân công công việc**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Họ và tên | MSSV | Công việc | Phần trăm đóng góp |
| 1 | Trần Tiến Đức | 18133007 | Viết code chạy mô hình hồi quy Logistic và hiệu chỉnh dữ liệu | 35% |
| 2 | Lương Uy Long | 18133026 | Đánh giá, nhận xét dữ liệu, output và viết báo cáo. | 30% |
| 3 | Trần Như Thuận | 18133054 | Viết code phân tích và trực quan hoá và dữ liệu. | 35% |

1. Nội dung chi tiết
   1. *Các thư viện sử dụng*

**import** **numpy** **as** **np**

**import** **pandas** **as** **pd**

**import** **seaborn** **as** **sb**

sb.set(style="white")

sb.set(style="whitegrid", color\_codes=**True**)

**import** **matplotlib.pyplot** **as** **plt**

plt.rc("font",size=14)

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

**from** **sklearn.linear\_model** **import** LogisticRegression

**from** **sklearn.preprocessing** **import** StandardScaler

**from** **sklearn.metrics** **import** confusion\_matrix, accuracy\_score

1. NumPy (thư viện nền tảng chuyên xử lý toán học)
2. Pandas (Thư viện tương tác dữ liệu sử dụng numpy làm nền tảng tính toán đại số)
3. Seaborn (Thư viện trực quan hoá dữ liệu sử dụng thư viện đồ hoạ matplotlib làm nền tảng)
4. Matplotlib (Thư viện đồ hoạ được sử dụng để vẽ đồ thị và lưu hình ảnh)
5. Sklearn (Thư viện machine learning mã nguồn mở của python)

* Model được sử dụng ở đây là train\_test\_split (Tách dữ liệu biến độc lập và biến phụ thuộc thành hai phần: phần dữ liệu để huấn luyện và phần dữ liệu để kiểm tra (test)
* linear\_model được sử dụng là hồi quy Logistic
* Tiền xử lý là hiệu chuẩn độ chênh lệch giá trị của các biến để tăng độ chính xác cho mô hình hồi quy Logistic
* Đơn vị đo để kiểm tra mức độ chính xác của mô hình
  1. *Data exploration*

- Nạp dữ liệu từ file ‘Xahoi.csv’ vào DataFrame

raw\_data = pd.read\_csv('Xahoi.csv')

raw\_data

| **User ID** | **Gender** | **Age** | **EstimatedSalary** | **Purchased** |
| --- | --- | --- | --- | --- |
| **0** | 15624510 | Male | 19 | 19000 | 0 |
| **1** | 15810944 | Male | 35 | 20000 | 0 |
| **2** | 15668575 | Female | 26 | 43000 | 0 |
| **3** | 15603246 | Female | 27 | 57000 | 0 |
| **4** | 15804002 | Male | 19 | 76000 | 0 |
| **...** | ... | ... | ... | ... | ... |
| **395** | 15691863 | Female | 46 | 41000 | 1 |
| **396** | 15706071 | Male | 51 | 23000 | 1 |
| **397** | 15654296 | Female | 50 | 20000 | 1 |
| **398** | 15755018 | Male | 36 | 33000 | 0 |
| **399** | 15594041 | Female | 49 | 36000 | 1 |

400 rows × 5 columns

* + 1. **Kiểm tra thông tin dữ liệu**

Như trong bài tập này, dữ liệu được nạp vào có 5 thuộc tính: User ID, Gender, Age, EstimatedSalary (Lương ước tính), Purchased và 400 bản ghi có kiểu là int64.

raw\_data.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 400 entries, 0 to 399

Data columns (total 5 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 User ID 400 non-null int64

1 Gender 400 non-null object

2 Age 400 non-null int64

3 EstimatedSalary 400 non-null int64

4 Purchased 400 non-null int64

dtypes: int64(4), object(1)

memory usage: 15.8+ KB

* + 1. **Tiến hành lọc dữ liệu có giá trị null (nếu có)**

raw\_data.isnull().sum()

User ID 0

Gender 0

Age 0

EstimatedSalary 0

Purchased 0

dtype: int64

raw\_data.dropna(inplace=True)

* + 1. **Biến đổi trường Gender thành biến nhị phân (Chỉ giá trị 0 hoặc 1)**

raw\_data['Gender']=pd.get\_dummies(raw\_data['Gender'])

raw\_data

| **User ID** | **Gender** | **Age** | **EstimatedSalary** | **Purchased** |
| --- | --- | --- | --- | --- |
| **0** | 15624510 | 0 | 19 | 19000 | 0 |
| **1** | 15810944 | 0 | 35 | 20000 | 0 |
| **2** | 15668575 | 1 | 26 | 43000 | 0 |
| **3** | 15603246 | 1 | 27 | 57000 | 0 |
| **4** | 15804002 | 0 | 19 | 76000 | 0 |
| **...** | ... | ... | ... | ... | ... |
| **395** | 15691863 | 1 | 46 | 41000 | 1 |
| **396** | 15706071 | 0 | 51 | 23000 | 1 |
| **397** | 15654296 | 1 | 50 | 20000 | 1 |
| **398** | 15755018 | 0 | 36 | 33000 | 0 |
| **399** | 15594041 | 1 | 49 | 36000 | 1 |

400 rows × 5 columns

* + 1. **Lọc và kiểm tra đối tượng theo độ tuổi**

raw\_data['Age'].unique()

array([19, 35, 26, 27, 32, 25, 20, 18, 29, 47, 45, 46, 48, 49, 31, 21, 28,

33, 30, 23, 24, 22, 59, 34, 39, 38, 37, 42, 40, 36, 41, 58, 55, 52,

60, 56, 53, 50, 51, 57, 44, 43, 54], dtype=int64)

raw\_data['Age'].value\_counts()

35 32

37 20

41 16

42 16

26 16

39 15

40 15

47 14

48 14

38 13

27 13

28 12

46 12

36 12

31 11

30 11

29 10

49 10

32 9

33 9

24 9

20 7

19 7

60 7

59 7

45 7

25 6

34 6

58 6

52 6

23 6

57 5

22 5

18 5

53 5

21 4

50 4

54 4

55 3

43 3

56 3

51 3

44 2

Name: Age, dtype: int64

* + 1. **Nhóm dữ liệu theo trường "Age"**

raw\_data.groupby('Age').mean()

User ID Gender EstimatedSalary Purchased

Age

18 1.567696e+07 0.600000 66400.000000 0.000000

19 1.571065e+07 0.285714 46000.000000 0.000000

20 1.569884e+07 0.571429 61714.285714 0.000000

21 1.565027e+07 0.500000 61000.000000 0.000000

22 1.568825e+07 0.600000 48800.000000 0.000000

23 1.564283e+07 0.666667 51166.666667 0.000000

24 1.569152e+07 0.444444 49111.111111 0.000000

25 1.568253e+07 0.000000 65166.666667 0.000000

26 1.571145e+07 0.562500 53500.000000 0.000000

27 1.572950e+07 0.615385 67615.384615 0.153846

28 1.569555e+07 0.583333 69416.666667 0.083333

29 1.572048e+07 0.400000 69100.000000 0.100000

30 1.571796e+07 0.272727 76000.000000 0.181818

31 1.568649e+07 0.545455 62454.545455 0.090909

32 1.566527e+07 0.555556 95666.666667 0.555556

33 1.568587e+07 0.777778 65000.000000 0.111111

34 1.562324e+07 0.500000 68333.333333 0.166667

35 1.568546e+07 0.406250 61812.500000 0.093750

36 1.569138e+07 0.416667 83250.000000 0.416667

37 1.569091e+07 0.500000 79400.000000 0.350000

38 1.571674e+07 0.538462 69153.846154 0.076923

39 1.565741e+07 0.466667 82800.000000 0.400000

40 1.567124e+07 0.266667 72200.000000 0.200000

41 1.568317e+07 0.562500 63562.500000 0.062500

42 1.571945e+07 0.437500 78937.500000 0.375000

43 1.565452e+07 0.666667 124666.666667 0.666667

44 1.576285e+07 1.000000 89000.000000 0.500000

45 1.569255e+07 0.428571 51000.000000 0.857143

46 1.568186e+07 0.500000 61750.000000 0.583333

47 1.568834e+07 0.571429 60071.428571 0.857143

48 1.567615e+07 0.571429 80285.714286 0.928571

49 1.568920e+07 0.300000 67400.000000 0.800000

50 1.572968e+07 1.000000 47000.000000 0.750000

51 1.567736e+07 0.666667 101000.000000 0.666667

52 1.572194e+07 0.833333 91833.333333 0.833333

53 1.567942e+07 0.800000 87000.000000 1.000000

54 1.570967e+07 0.750000 77000.000000 1.000000

55 1.565082e+07 0.666667 98000.000000 1.000000

56 1.574343e+07 0.333333 99000.000000 1.000000

57 1.576347e+07 0.600000 63000.000000 1.000000

58 1.565926e+07 0.833333 74666.666667 1.000000

59 1.569425e+07 0.714286 84428.571429 0.714286

60 1.568474e+07 0.285714 65285.714286 1.000000

* + 1. **Nhóm dữ liệu theo trường "Purchased"**

raw\_data.groupby('Purchased').mean()

User ID Gender Age EstimatedSalary

Purchased

0 1.569116e+07 0.494163 32.793774 60544.747082

1 1.569222e+07 0.538462 46.391608 86272.727273

raw\_data.columns.values.tolist()

['User ID', 'Gender', 'Age', 'EstimatedSalary', 'Purchased']

pd.get\_dummies(raw\_data)

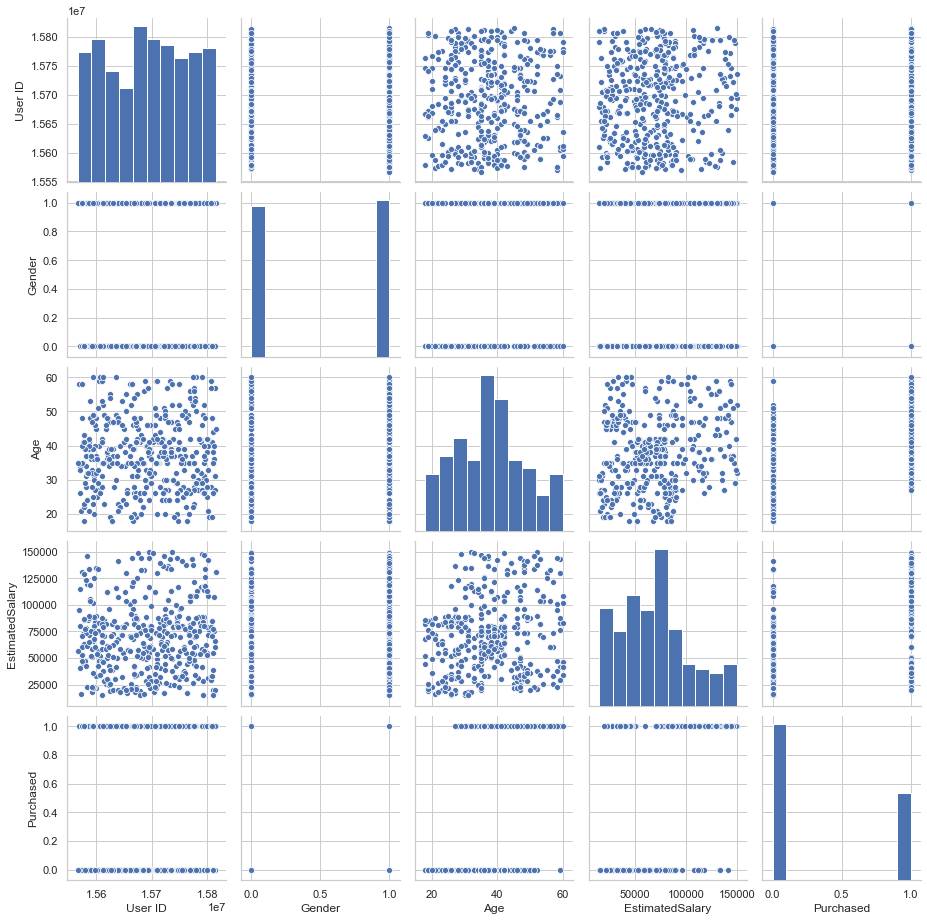
| **User ID** | **Gender** | **Age** | **EstimatedSalary** | **Purchased** |
| --- | --- | --- | --- | --- |
| **0** | 15624510 | 0 | 19 | 19000 | 0 |
| **1** | 15810944 | 0 | 35 | 20000 | 0 |
| **2** | 15668575 | 1 | 26 | 43000 | 0 |
| **3** | 15603246 | 1 | 27 | 57000 | 0 |
| **4** | 15804002 | 0 | 19 | 76000 | 0 |
| **...** | ... | ... | ... | ... | ... |
| **395** | 15691863 | 1 | 46 | 41000 | 1 |
| **396** | 15706071 | 0 | 51 | 23000 | 1 |
| **397** | 15654296 | 1 | 50 | 20000 | 1 |
| **398** | 15755018 | 0 | 36 | 33000 | 0 |
| **399** | 15594041 | 1 | 49 | 36000 | 1 |

400 rows × 5 columns

* 1. *Visualizations*

Trực quan hoá dữ liệu của các biến được tính toán

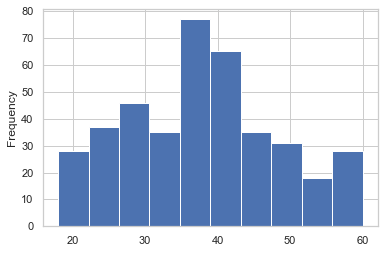
sb.pairplot(raw\_data)

**

* + 1. *Trực quan số người theo độ tuổi bằng Matplotlib*

raw\_data['Age'].plot.hist()

<matplotlib.axes.\_subplots.AxesSubplot at 0x23c3a86da08>

**

* + 1. *Trực quan hoá dữ liệu của các biến được tính toán bằng Seaborn*

sb.countplot(x='Purchased',data=raw\_data,palette='hls')

plt.show()

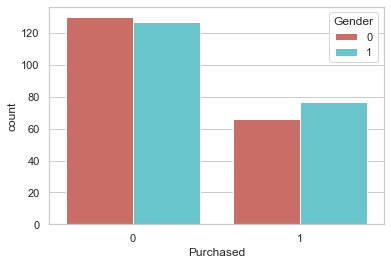
plt.savefig('count\_plot')

**

* + 1. *Số người mua hàng xếp theo giới tính*

sb.countplot(x='Purchased',hue="Gender",data=raw\_data,palette='hls')

plt.show()

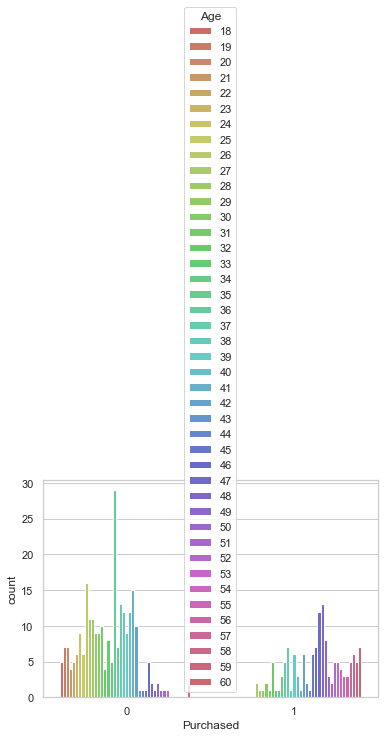
**plt.savefig('count\_plot\_purchased\_Gender’)

* + 1. *Số người mua hàng xếp theo độ tuổi*

sb.countplot(x='Purchased',hue="Age",data=raw\_data,palette='hls')

plt.show()

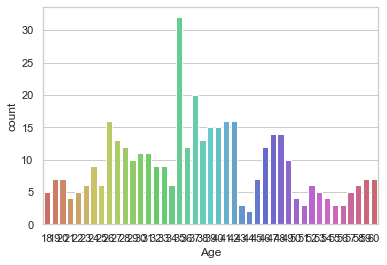
plt.savefig('count\_plot\_purchased')

**

sb.countplot(x='Age',data=raw\_data,palette='hls')

plt.show()

plt.savefig('count\_plot\_age\_purchased')

**

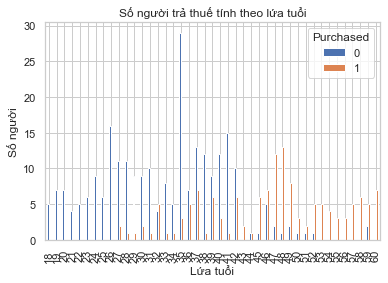
%matplotlib inline

pd.crosstab(raw\_data.Age,raw\_data.Purchased).plot(kind='bar')

plt.title('Số người trả thuế tính theo lứa tuổi')

plt.xlabel('Lứa tuổi')

plt.ylabel('Số người')

plt.savefig('purchased\_age')

* 1. *Over-sampling using SMOTE*

*Nhập module SMOTE (Synthetic Minority Oversampling Technique) để tổng hợp và phân tích dữ liệu quá khổ*

SMOTE hoạt động bằng cách lấy các mẫu gần không gian tính năng (feature space), vẽ 1 đường thẳng giữa các mẫu trong không gian tính năng và vẽ 1 điểm mới dọc theo đường đó.

— Page 47, Imbalanced Learning: Foundations, Algorithms, and Applications, 2013.

X = raw\_data.loc[:,['Age','EstimatedSalary']]

y = raw\_data['Purchased']

from imblearn.over\_sampling import SMOTE

os = SMOTE(random\_state=0)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X,y,test\_size=0.3,random\_state=0)

columns = X\_train.columns

os\_data\_X,os\_data\_y=os.fit\_sample(X\_train, y\_train)

os\_data\_X = pd.DataFrame(data=os\_data\_X,columns=columns )

os\_data\_y= pd.DataFrame(data=os\_data\_y,columns=['Purchased'])

os\_data\_X

|  |  |  |
| --- | --- | --- |
|  | Age | EstimatedSalary |
| 0 | 23 | 63000 |
| 1 | 48 | 33000 |
| 2 | 48 | 90000 |
| 3 | 42 | 104000 |
| 4 | 44 | 39000 |
| ... | ... | ... |
| 375 | 49 | 88199 |
| 376 | 49 | 73295 |
| 377 | 52 | 33000 |
| 378 | 45 | 118682 |
| 379 | 38 | 95713 |

380 rows × 2 columns

os\_data\_y

|  |
| --- |
| Purchased |
| 0 | 0 |
| 1 | 1 |
| 2 | 1 |
| 3 | 1 |
| 4 | 0 |
| ... | ... |
| 375 | 1 |
| 376 | 1 |
| 377 | 1 |
| 378 | 1 |
| 379 | 1 |

380 rows × 1 columns

print("Độ dài của dữ liệu quá khổ:",len(os\_data\_X))

print("Số lượng bản ghi có giá trị 0 trong dữ liệu quá khổ:",len(os\_data\_y[os\_data\_y['Purchased']==0]))

print("Số lượng bản ghi có giá trị 1:",len(os\_data\_y[os\_data\_y['Purchased']==1]))

print("Tỷ lệ dữ liệu có giá trị 0 trong dữ liệu quá khổ:",len(os\_data\_y[os\_data\_y['Purchased']==0])/len(os\_data\_X))

print("Tỷ lệ dữ liệu có giá trị 1 trong dữ liệu quá khổ:",len(os\_data\_y[os\_data\_y['Purchased']==1])/len(os\_data\_X))

Độ dài của dữ liệu quá khổ: 380

Số lượng bản ghi có giá trị 0 trong dữ liệu quá khổ: 190

Số lượng bản ghi có giá trị 1: 190

Tỷ lệ dữ liệu có giá trị 0 trong dữ liệu quá khổ: 0.5

Tỷ lệ dữ liệu có giá trị 1 trong dữ liệu quá khổ: 0.5

X\_test

|  |  |  |
| --- | --- | --- |
|  | Age | EstimatedSalary |
| 132 | 30 | 87000 |
| 309 | 38 | 50000 |
| 341 | 35 | 75000 |
| 196 | 30 | 79000 |
| 246 | 35 | 50000 |
| ... | ... | ... |
| 253 | 37 | 146000 |
| 56 | 23 | 48000 |
| 8 | 25 | 33000 |
| 190 | 24 | 84000 |
| 146 | 27 | 96000 |

96 rows × 2 columns

X\_train

|  |  |  |
| --- | --- | --- |
|  | Age | EstimatedSalary |
| 135 | 23 | 63000 |
| 390 | 48 | 33000 |
| 264 | 48 | 90000 |
| 364 | 42 | 104000 |
| 250 | 44 | 39000 |
| ... | ... | ... |
| 323 | 48 | 30000 |
| 192 | 29 | 43000 |
| 117 | 36 | 52000 |
| 47 | 27 | 54000 |
| 172 | 26 | 118000 |

304 rows × 2 columns

sc = StandardScaler()

(Chuẩn hoá bằng cách loại bỏ giá trị trung bình và và tỷ lệ theo phương sai đơn vị

Điểm chuẩn của một mẫu x được tính như sau:

z = (x - u) / s

Trong đó, u là giá trị trung bình của mẫu huấn luyện hoặc bằng 0 nếu`with\_mean=False`, và s là độ lệch chuẩn của mẫu huấn luyện hoặc bằng 1 nếu `with\_std=False`)

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

X\_test = sc.fit\_transform(X\_test)

X\_train

array([[-1.50005227, -0.19245733],

[ 0.97452929, -1.06439518],

[ 0.97452929, 0.59228673],

[ 0.38062972, 0.99919106],

[ 0.57859624, -0.89000761],

[-0.60920291, 1.46422457],

[-0.01530333, -0.57029707],

[-0.60920291, 1.9001935 ],

[ 1.37046234, -1.41317032],

[ 1.46944561, 0.99919106],

[ 0.08367993, -0.80281383],

[-0.01530333, -0.25058652],

[-0.21326986, -0.57029707],

[-0.21326986, -0.19245733],

[-0.31225312, -1.29691194],

[-0.31225312, -0.57029707],

[ 0.38062972, 0.09818861],

[ 0.87554603, -0.59936166],

[ 2.06334518, -1.18065356],

[ 1.07351256, -0.13432814],

[ 0.67757951, 1.78393512],

[-0.70818617, 0.56322213],

[ 0.77656277, 0.35976997],

[ 0.87554603, -0.54123247],

[-1.20310249, -1.58755789],

[ 2.16232845, 0.94106187],

[-0.01530333, 1.23170781],

[ 0.18266319, 1.08638484],

[ 0.38062972, -0.48310328],

[-0.31225312, -0.30871571],

[ 0.97452929, -0.83187842],

[ 0.97452929, 1.8711289 ],

[-0.01530333, 1.26077241],

[-0.9061527 , 2.27803323],

[-1.20310249, -1.58755789],

[ 2.16232845, -0.80281383],

[-1.40106901, -1.47129951],

[ 0.38062972, 2.30709782],

[ 0.77656277, 0.7666743 ],

[-1.00513596, -0.30871571],

[ 0.08367993, 0.7666743 ],

[-1.00513596, 0.56322213],

[ 0.28164645, 0.06912402],

[ 0.67757951, -1.26784734],

[-0.51021965, -0.01806976],

[-1.79700206, 0.35976997],

[-0.70818617, 0.12725321],

[ 0.38062972, 0.30164078],

[-0.31225312, 0.06912402],

[-0.51021965, 2.30709782],

[ 0.18266319, 0.04005943],

[ 1.27147908, 2.21990404],

[ 0.77656277, 0.27257618],

[-0.31225312, 0.1563178 ],

[-0.01530333, -0.54123247],

[-0.21326986, 0.1563178 ],

[-0.1142866 , 0.24351159],

[-0.01530333, -0.25058652],

[ 2.16232845, 1.11544943],

[-1.79700206, 0.35976997],

[ 1.86537866, 0.12725321],

[ 0.38062972, -0.13432814],

[-1.20310249, 0.30164078],

[ 0.77656277, 1.37703079],

[-0.31225312, -0.25058652],

[-1.6980188 , -0.04713436],

[-1.00513596, -0.74468464],

[ 0.28164645, 0.50509294],

[-0.1142866 , -1.06439518],

[-1.10411922, 0.59228673],

[ 0.08367993, -0.80281383],

[-1.00513596, 1.55141836],

[-0.70818617, 1.40609538],

[-1.30208575, 0.50509294],

[-0.31225312, 0.04005943],

[-0.1142866 , 0.01099483],

[-0.31225312, -0.89000761],

[ 0.87554603, -1.35504113],

[-0.31225312, 2.24896863],

[ 0.97452929, 1.98738728],

[-1.20310249, 0.47602835],

[-1.30208575, 0.27257618],

[ 1.37046234, 1.98738728],

[ 1.27147908, -1.35504113],

[-0.31225312, -0.27965112],

[-0.51021965, 1.26077241],

[-0.80716944, 1.08638484],

[ 0.97452929, -1.06439518],

[ 0.28164645, 0.30164078],

[ 0.97452929, 0.7666743 ],

[-0.70818617, -1.5003641 ],

[-0.70818617, 0.04005943],

[ 0.47961298, 1.72580593],

[ 2.06334518, 0.1853824 ],

[-1.99496859, -0.74468464],

[-0.21326986, 1.40609538],

[ 0.38062972, 0.59228673],

[ 0.87554603, -1.15158896],

[-1.20310249, -0.77374923],

[ 0.18266319, 0.24351159],

[ 0.77656277, -0.30871571],

[ 2.06334518, -0.80281383],

[ 0.77656277, 0.12725321],

[-0.31225312, 0.62135132],

[-1.00513596, -0.30871571],

[ 0.18266319, -0.3668449 ],

[ 2.06334518, 2.13271025],

[ 1.86537866, -1.26784734],

[ 1.37046234, -0.91907221],

[ 0.87554603, 1.26077241],

[ 1.46944561, 2.13271025],

[-0.31225312, -1.23878275],

[ 1.96436192, 0.91199727],

[ 0.67757951, -0.71562004],

[-1.50005227, 0.35976997],

[ 0.77656277, -1.35504113],

[ 0.38062972, -0.13432814],

[-1.00513596, 0.41789916],

[-0.01530333, -0.30871571],

[-1.20310249, 0.41789916],

[-0.9061527 , -1.20971815],

[-0.1142866 , 0.04005943],

[-1.59903554, -0.42497409],

[ 0.97452929, -1.00626599],

[ 1.07351256, -1.20971815],

[-0.01530333, -0.13432814],

[-1.10411922, -1.5294287 ],

[ 0.77656277, -1.20971815],

[ 0.97452929, 2.07458107],

[-1.20310249, -1.5294287 ],

[-0.31225312, 0.79573889],

[ 0.08367993, -0.30871571],

[-1.40106901, -1.23878275],

[-0.60920291, -1.5003641 ],

[ 0.77656277, 0.53415754],

[-0.31225312, -0.33778031],

[ 1.7663954 , -0.27965112],

[ 0.87554603, -1.03533058],

[ 0.18266319, 0.06912402],

[-0.60920291, 0.88293268],

[-1.89598533, -1.41317032],

[-1.30208575, 0.59228673],

[-0.31225312, 0.53415754],

[-1.00513596, -1.09345977],

[ 1.17249582, -1.44223491],

[ 0.18266319, -0.30871571],

[ 1.17249582, -0.74468464],

[-0.31225312, 0.06912402],

[ 0.18266319, 2.10364566],

[ 0.77656277, -1.09345977],

[ 0.08367993, 0.04005943],

[-1.79700206, 0.12725321],

[-0.9061527 , 0.1563178 ],

[-0.70818617, 0.1853824 ],

[ 0.87554603, -1.29691194],

[ 0.18266319, -0.25058652],

[-0.41123638, 1.23170781],

[-0.01530333, 0.30164078],

[ 0.38062972, 0.1563178 ],

[ 0.87554603, -0.65749085],

[ 0.08367993, 0.1563178 ],

[-1.89598533, -1.29691194],

[-0.1142866 , 0.30164078],

[-0.21326986, -0.27965112],

[ 0.28164645, -0.51216788],

[-0.21326986, 1.60954755],

[ 0.97452929, -1.18065356],

[-0.21326986, 1.63861214],

[ 1.27147908, 1.8711289 ],

[-1.10411922, -0.3668449 ],

[-0.01530333, 0.04005943],

[ 0.08367993, -0.25058652],

[-1.59903554, -1.23878275],

[-0.51021965, -0.27965112],

[ 0.97452929, 0.12725321],

[ 1.96436192, -1.35504113],

[ 1.46944561, 0.06912402],

[-0.60920291, 1.37703079],

[ 1.56842887, 0.01099483],

[-0.80716944, 0.30164078],

[ 1.96436192, 0.7376097 ],

[-1.20310249, -0.51216788],

[ 0.67757951, 0.27257618],

[-1.40106901, -0.42497409],

[ 0.18266319, 0.1563178 ],

[-0.51021965, -1.20971815],

[ 0.57859624, 2.01645188],

[-1.59903554, -1.5003641 ],

[-0.51021965, -0.54123247],

[ 0.47961298, 1.84206431],

[-1.40106901, -1.09345977],

[ 0.77656277, -1.38410572],

[-0.31225312, -0.42497409],

[ 1.56842887, 0.99919106],

[ 0.97452929, 1.43515998],

[-0.31225312, -0.48310328],

[-0.1142866 , 2.16177485],

[-1.50005227, -0.10526355],

[-0.1142866 , 1.95832269],

[-0.70818617, -0.33778031],

[-0.51021965, -0.83187842],

[ 0.67757951, -1.38410572],

[-0.80716944, -1.58755789],

[-1.89598533, -1.47129951],

[ 1.07351256, 0.12725321],

[ 0.08367993, 1.52235376],

[-0.31225312, 0.09818861],

[ 0.08367993, 0.04005943],

[-1.40106901, -1.35504113],

[ 0.28164645, 0.06912402],

[-0.9061527 , 0.38883456],

[ 1.56842887, -1.26784734],

[-0.31225312, -0.74468464],

[-0.1142866 , 0.1563178 ],

[-0.9061527 , -0.65749085],

[-0.70818617, -0.04713436],

[ 0.38062972, -0.45403869],

[-0.80716944, 1.9001935 ],

[ 1.37046234, 1.289837 ],

[ 1.17249582, -0.97720139],

[ 1.7663954 , 1.84206431],

[-0.9061527 , -0.25058652],

[-0.80716944, 0.56322213],

[-1.20310249, -1.55849329],

[-0.51021965, -1.12252437],

[ 0.28164645, 0.06912402],

[-0.21326986, -1.06439518],

[ 1.66741213, 1.60954755],

[ 0.97452929, 1.78393512],

[ 0.28164645, 0.04005943],

[-0.80716944, -0.22152193],

[-0.1142866 , 0.06912402],

[ 0.28164645, -0.19245733],

[ 1.96436192, -0.65749085],

[-0.80716944, 1.34796619],

[-1.79700206, -0.59936166],

[-0.1142866 , 0.12725321],

[ 0.28164645, -0.30871571],

[ 1.07351256, 0.56322213],

[-1.00513596, 0.27257618],

[ 1.46944561, 0.35976997],

[ 0.18266319, -0.3668449 ],

[ 2.16232845, -1.03533058],

[-0.31225312, 1.11544943],

[-1.6980188 , 0.06912402],

[-0.01530333, 0.04005943],

[ 0.08367993, 1.05732025],

[-0.1142866 , -0.3668449 ],

[-1.20310249, 0.06912402],

[-0.31225312, -1.35504113],

[ 1.56842887, 1.11544943],

[-0.80716944, -1.5294287 ],

[ 0.08367993, 1.8711289 ],

[-0.9061527 , -0.77374923],

[-0.51021965, -0.77374923],

[-0.31225312, -0.91907221],

[ 0.28164645, -0.71562004],

[ 0.28164645, 0.06912402],

[ 0.08367993, 1.8711289 ],

[-1.10411922, 1.95832269],

[-1.6980188 , -1.55849329],

[-1.20310249, -1.09345977],

[-0.70818617, -0.10526355],

[ 0.08367993, 0.09818861],

[ 0.28164645, 0.27257618],

[ 0.87554603, -0.57029707],

[ 0.28164645, -1.15158896],

[-0.1142866 , 0.67948051],

[ 2.16232845, -0.68655545],

[-1.30208575, -1.38410572],

[-1.00513596, -0.9481368 ],

[-0.01530333, -0.42497409],

[-0.21326986, -0.45403869],

[-1.79700206, -0.97720139],

[ 1.7663954 , 0.99919106],

[ 0.18266319, -0.3668449 ],

[ 0.38062972, 1.11544943],

[-1.79700206, -1.35504113],

[ 0.18266319, -0.13432814],

[ 0.87554603, -1.44223491],

[-1.99496859, 0.47602835],

[-0.31225312, 0.27257618],

[ 1.86537866, -1.06439518],

[-0.41123638, 0.06912402],

[ 1.07351256, -0.89000761],

[-1.10411922, -1.12252437],

[-1.89598533, 0.01099483],

[ 0.08367993, 0.27257618],

[-1.20310249, 0.33070537],

[-1.30208575, 0.30164078],

[-1.00513596, 0.44696375],

[ 1.66741213, -0.89000761],

[ 1.17249582, 0.53415754],

[ 1.07351256, 0.53415754],

[ 1.37046234, 2.33616242],

[-0.31225312, -0.13432814],

[ 0.38062972, -0.45403869],

[-0.41123638, -0.77374923],

[-0.1142866 , -0.51216788],

[ 0.97452929, -1.15158896],

[-0.9061527 , -0.77374923],

[-0.21326986, -0.51216788],

[-1.10411922, -0.45403869],

[-1.20310249, 1.40609538]])

X\_test

array([[-0.53215076, 0.51272272],

[ 0.16886431, -0.61146932],

[-0.09401634, 0.1481199 ],

[-0.53215076, 0.26965417],

[-0.09401634, -0.61146932],

[-0.79503141, -1.52297638],

[-0.44452388, -1.67489423],

[-0.00638946, 2.24458614],

[-1.58367336, -0.06456508],

[ 0.95750626, -0.8241543 ],

[-0.53215076, -0.64185289],

[-0.70740453, -0.45955148],

[ 0.08123742, -0.45955148],

[ 0.25649119, 0.20888704],

[-1.4084196 , 0.48233915],

[-0.35689699, 1.42422979],

[ 0.08123742, 0.20888704],

[-1.49604648, 0.45195559],

[ 1.65852133, 1.81921618],

[-0.09401634, -1.46220925],

[-0.09401634, -0.70262003],

[ 0.95750626, 2.24458614],

[ 0.43174496, -0.58108575],

[ 0.95750626, 1.05962696],

[-1.14553895, -1.27990783],

[ 1.13276002, 2.15343544],

[-0.70740453, 0.51272272],

[-0.61977764, 0.30003774],

[ 0.08123742, -0.2468665 ],

[-0.35689699, 0.48233915],

[-1.32079271, 0.54310629],

[ 0.08123742, 0.26965417],

[ 1.83377509, -0.30763363],

[ 0.08123742, -0.52031861],

[-1.05791206, -0.36840077],

[-1.58367336, -0.55070218],

[-1.23316583, 0.33042131],

[-0.18164323, -0.8241543 ],

[-0.44452388, -1.09760642],

[ 1.13276002, -1.03683928],

[-0.79503141, 0.54310629],

[ 0.43174496, -0.55070218],

[-0.79503141, 0.42157202],

[-0.09401634, -1.52297638],

[ 0.60699872, 1.27231194],

[-0.79503141, -0.36840077],

[ 0.08123742, 0.30003774],

[ 1.39564068, 0.60387343],

[-0.88265829, -1.2191407 ],

[ 1.13276002, 0.48233915],

[ 1.83377509, 1.57614763],

[-0.18164323, -1.37105854],

[-0.09401634, -0.39878434],

[-0.18164323, 1.36346265],

[ 2.00902886, 0.54310629],

[ 0.78225249, -1.15837356],

[-0.61977764, 0.39118845],

[-0.88265829, 0.30003774],

[ 1.13276002, -1.27990783],

[-1.14553895, -1.52297638],

[-0.35689699, -1.58374352],

[ 2.09665574, -0.85453787],

[-1.49604648, 0.17850347],

[-0.00638946, 0.87732555],

[-1.49604648, -1.34067497],

[ 2.09665574, 0.39118845],

[-1.05791206, 0.57348986],

[-0.79503141, -0.36840077],

[ 0.34411807, -0.70262003],

[ 0.51937184, -0.00379795],

[-0.35689699, 2.42688755],

[-0.09401634, 0.20888704],

[-1.23316583, -0.21648293],

[ 0.78225249, -1.46220925],

[-0.79503141, 0.57348986],

[-1.58367336, 0.36080488],

[ 0.51937184, 0.26965417],

[ 0.34411807, -0.30763363],

[ 1.48326756, -1.09760642],

[ 0.95750626, 1.1203941 ],

[ 1.92140198, 2.24458614],

[ 2.00902886, 0.39118845],

[-1.05791206, -0.45955148],

[-0.88265829, -1.06722285],

[ 1.92140198, -0.97607215],

[ 0.51937184, 0.30003774],

[ 0.34411807, 0.1481199 ],

[ 2.00902886, 1.81921618],

[ 0.86987937, -0.88492144],

[ 0.43174496, -0.30763363],

[ 0.51937184, -0.18609936],

[ 0.08123742, 2.30535328],

[-1.14553895, -0.67223646],

[-0.97028518, -1.12798999],

[-1.05791206, 0.42157202],

[-0.79503141, 0.78617484]])

* 1. *Implementing the model*

import statsmodels.api as sm

logit\_model=sm.Logit(y,X)

result=logit\_model.fit()

print(result.summary2())

Optimization terminated successfully.

Current function value: 0.686532

Iterations 3

Results: Logit

===============================================================

Model: Logit Pseudo R-squared: -0.053

Dependent Variable: Purchased AIC: 553.2253

Date: 2020-06-29 20:25 BIC: 561.2082

No. Observations: 400 Log-Likelihood: -274.61

Df Model: 1 LL-Null: -260.79

Df Residuals: 398 LLR p-value: 1.0000

Converged: 1.0000 Scale: 1.0000

No. Iterations: 3.0000

---------------------------------------------------------------

Coef. Std.Err. z P>|z| [0.025 0.975]

---------------------------------------------------------------

Age -0.0054 0.0055 -0.9815 0.3263 -0.0162 0.0054

EstimatedSalary -0.0000 0.0000 -0.1002 0.9202 -0.0000 0.0000

===============================================================

* 1. *Logistic Regression Model Fitting*

model = LogisticRegression(random\_state=0)

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

LogisticRegression(C=1.0, class\_weight=None, dual=False, fit\_intercept=True,

intercept\_scaling=1, l1\_ratio=None, max\_iter=100,

multi\_class='auto', n\_jobs=None, penalty='l2',

random\_state=0, solver='lbfgs', tol=0.0001, verbose=0,

warm\_start=False)

* 1. *Confusion Matrix*

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

cfs

array([[62, 5],

[ 5, 24]], dtype=int64)

* 1. *Predicting the test set results and calculating the accuracy*

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

model.predict(X\_test)

array([0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,

0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,

1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1,

0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1,

0, 0, 0, 1, 0, 0, 0, 0], dtype=int64)

model.score(X\_test,y\_test)

0.8958333333333334

* 1. *Compute precision, recall, F-measure and support*

Tính toán độ chính xác, thu hồi, đo F và hỗ trợ

The precision là tỷ lệ tp / (tp + fp) trong đó tp là số lượng dương thực sự và fp số lượng dương tính giả. The precision là trực giác khả năng của bộ phân loại không gắn nhãn một mẫu là dương tính nếu nó là âm tính.

The recall là tỷ lệ tp / (tp + fn) trong đó tp là số lượng dương thực sự và fn số lượng âm tính giả. The recall là trực giác khả năng của bộ phân loại để tìm tất cả các mẫu dương tính.

F-beta score có thể được hiểu là trung bình hài hòa của độ chính xác và thu hồi, trong đó một F-beta score đạt giá trị tốt nhất ở mức 1 và điểm kém nhất là 0.

F-beta score có trọng số thu hồi cao hơn độ chính xác theo hệ số beta. beta = 1.0 có nghĩa là thu hồi và độ chính xác đều quan trọng như nhau.

The support là số lần xuất hiện của mỗi lớp trong y\_test.

**from** **sklearn.metrics** **import** classification\_report

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

precision recall f1-score support

0 0.93 0.93 0.93 67

1 0.83 0.83 0.83 29

accuracy 0.90 96

macro avg 0.88 0.88 0.88 96

weighted avg 0.90 0.90 0.90 96

* 1. *ROC Curve*

Trực quan tỉ lệ đúng sai bằng Matplotlib

**from** **sklearn.metrics** **import** roc\_auc\_score

**from** **sklearn.metrics** **import** roc\_curve

logit\_roc\_auc = roc\_auc\_score(y\_test, model.predict(X\_test))

fpr, tpr, thresholds = roc\_curve(y\_test, model.predict\_proba(X\_test)[:,1])

plt.figure()

plt.plot(fpr, tpr, label='Logistic Regression (area = **%0.2f**)' % logit\_roc\_auc)

plt.plot([0, 1], [0, 1],'r--')

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.05])

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver operating characteristic')

plt.legend(loc="lower right")

plt.savefig('Log\_ROC')

plt.show()



The receiver operating characteristic (ROC) curve là một công cụ phổ biến được sử dụng với các phân loại nhị phân. Đường chấm biểu thị đường cong ROC của một phân loại hoàn toàn ngẫu nhiên; một bộ phân loại tốt tức càng xa đường đó càng tốt (về phía góc trên bên trái).

1. Kết luận
   1. *Nhận xét, đánh giá*

Chúng em tự nhận thấy rằng phần bài tập của mình vẫn chưa được hoàn thiện nhưng vẫn phân tích cơ bản dữ liệu từ bảng cho sẵn. Nhóm chúng em vẫn cần nhiều cố gắng hơn để đạt được mức điểm tốt hơn.Bài của chúng em chưa phân tích dữ liệu được sâu và rộng, chỉ mới là những phần cơ bản của Sklearn cho việc phân tích dữ liệu.

* 1. *Kì vọng*

Nếu có thời gian tiếp tục thực hiện đề tài này, nhóm chúng em kì vọng rằng bài tập lớn của mình sẽ được tính toán tăng độ chính xác và hiển thị hoàn thiện hơn. Ngoài ra, chúng em còn mong muốn phân chia các thành phần ra từng bảng để phân tích được sâu rộng. Ví dụ, nhóm em sẽ phân tích riêng các bảng User ID, Gender, Age, EstimatedSalary, Purchased ra.Sau đó, mới gộp các bảng lại rồi phân tích tổng hợp dữ liệu của bài.