**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**--------------------oOo--------------------**

**KHAI THÁC DỮ LIỆU**

**(Data Mining)**

**Bài thực hành số 1(LAB1)**

Giảng viên hướng dẫn:

**ThS. Lê Hà Hoài Trung**

Sinh viên thực hiện:

**Hoàng Lê Nam – MSSV: 18521120**

🙡🙢 Tp. Hồ Chí Minh, 3/2021 🙠🙣

# **Bài 1**

* Yêu cầu: Thực hiện lại bài tập ở phần hướng dẫn.
* Import các thư viện:

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

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

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

**import** seaborn **as** sns

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

%matplotlib inline

* Đọc dữ liệu từ file .csv vào trong biến df:

**with** open('titanic.csv') **as** f:

df = pd.read\_csv(f, dtype={'Age': np.float64, 'Sibsp': np.int64, 'Parch': np.int64})

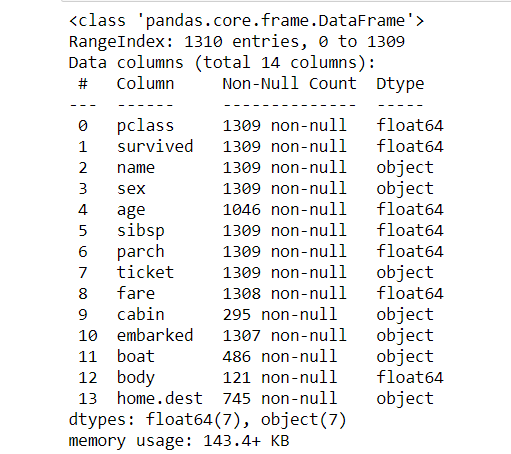
* Hiển thị thông tin của 12 dòng đầu của data vừa được đọc vào biến df:

df.head(12)



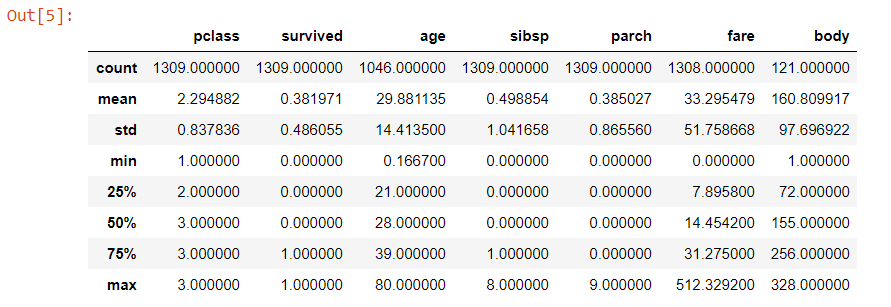
* Hiển thị thông tin về tổng quan kiểu dữ liệu của tập dữ liệu.

df.info()



* Thống kê các chỉ số thống kê cho từng thuộc tính định lượng:

df.decribe()



* Thống kê 5 thuộc tính có số lượng giá trị rỗng nhiều nhất:

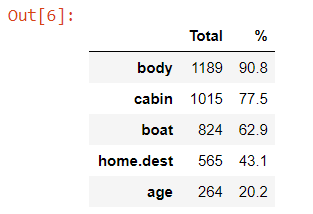
total = df.isnull().sum().sort\_values(ascending=**False**)

percent\_1 = df.isnull().sum()/df.isnull().count()\*100

percent\_2 = (round(percent\_1,1)).sort\_values(ascending=**False**)

missing\_data = pd.concat([total, percent\_2], axis=1, keys=['Total', '%'] )

missing\_data.head(5)



* Trực quan hóa mối quan hệ giữa các thuộc tính Độ tuổi(age), Giới tính(sex) và khả năng sống sót(survived):

survived = 'survived'

not\_survived = 'not survived'

fig,axes = plt.subplots(nrows=1, ncols=2,figsize=(10,4))

women = df[df['sex']=='female']

men = df[df['sex']=='male']

ax = sns.distplot(women[women['survived']==1].age.dropna(),bins=18,label=survived,ax = axes[0],kde=False)

ax = sns.distplot(women[women['survived']==0].age.dropna(),bins=40,label=not\_survived,ax = axes[0],kde=False)

ax.legend()

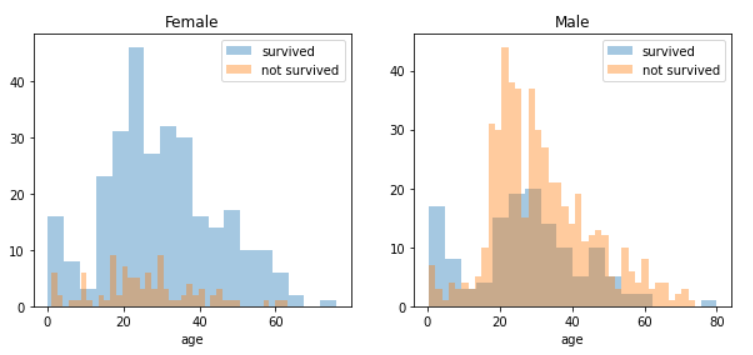
ax.set\_title('Female')

ax = sns.distplot(men[men['survived']==1].age.dropna(),bins=18,label=survived,ax = axes[1],kde=False)

ax = sns.distplot(men[men['survived']==0].age.dropna(),bins=40,label=not\_survived,ax = axes[1],kde=False)

ax.legend()

\_=ax.set\_title('Male')

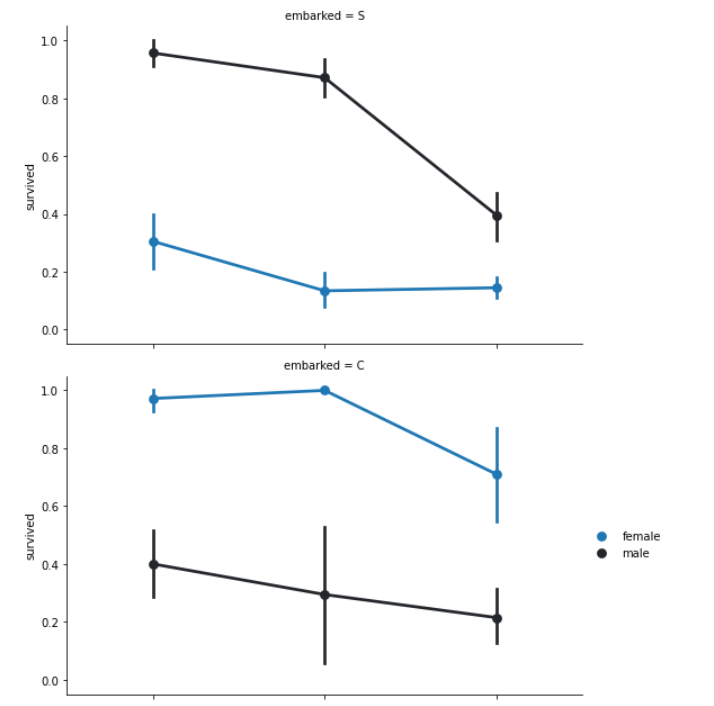


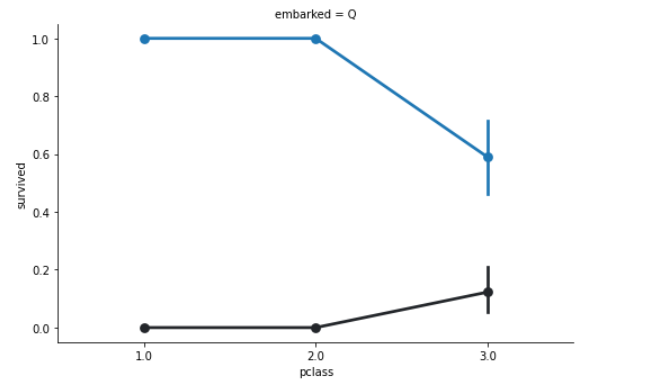
* Trực quan hóa mối quan hệ giữa các thuộc tính Cổng lên tàu(embarked) và loại vé(pclass) với khả năng sống sót(survived), phân biệt theo Giới tính(sex):

FacetGrid = sns.FacetGrid(df,row='embarked',height=4.5,aspect=1.6)

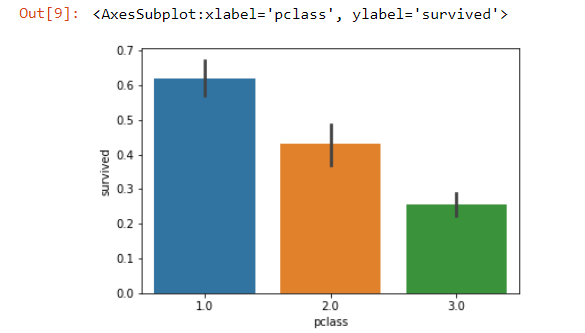
FacetGrid.map(sns.pointplot, 'pclass','survived','sex', palette=**None**,order=**None**,hue\_order=**None**)

FacetGrid.add\_legend()





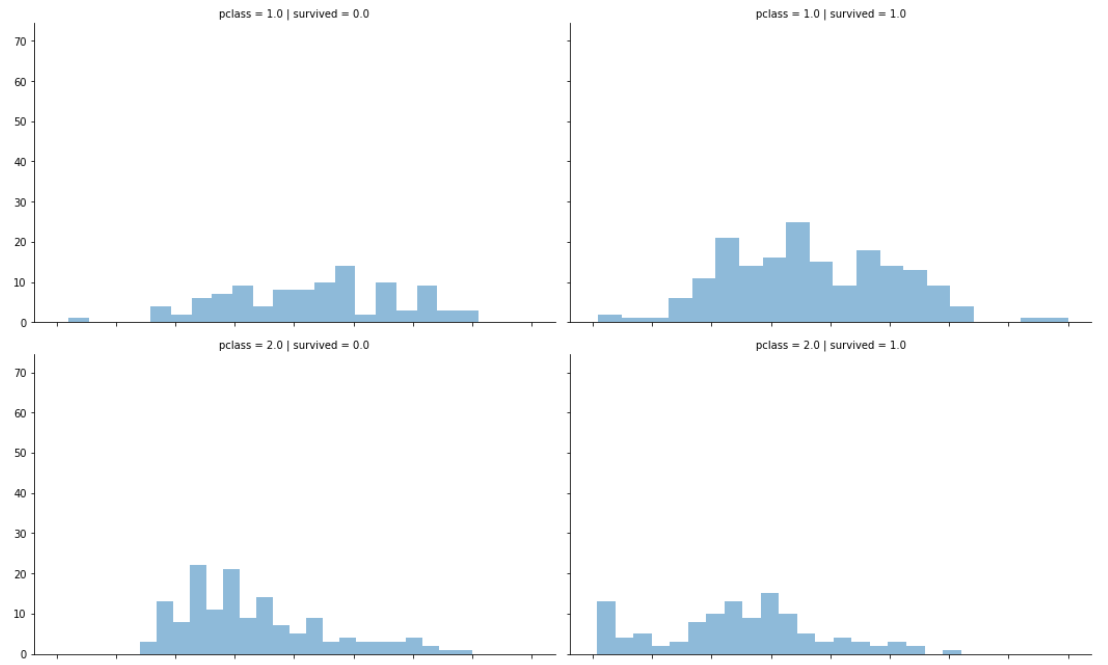
sns.barplot(x='pclass',y='survived',data=df)

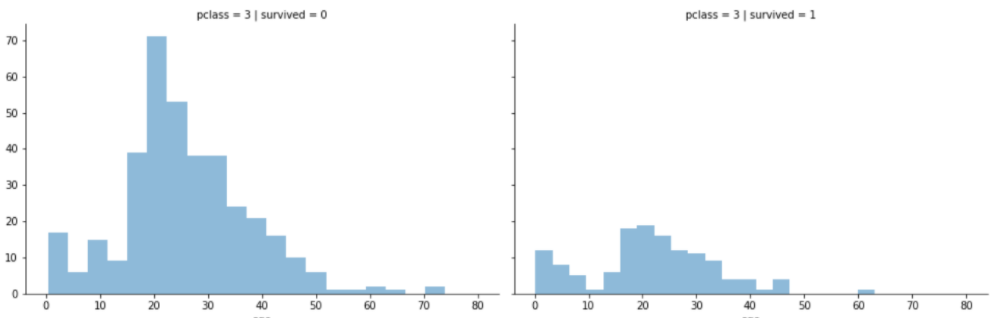


grid =sns.FacetGrid(df, col='survived',row='pclass',height=4.6,aspect=1.6)

grid.map(plt.hist,'age',alpha=.5,bins=20)

grid.add\_legend()





* Thống kê số người đi cùng (anh chị em/vợ chồng hoặc cha mẹ/con cái):

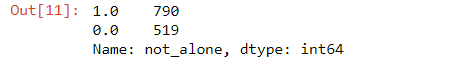
**for** dataset **in** [df]:

dataset['relatives'] = dataset['sibsp'] + dataset['parch']

dataset.loc[dataset['relatives'] > 0, 'not\_alone'] = 0

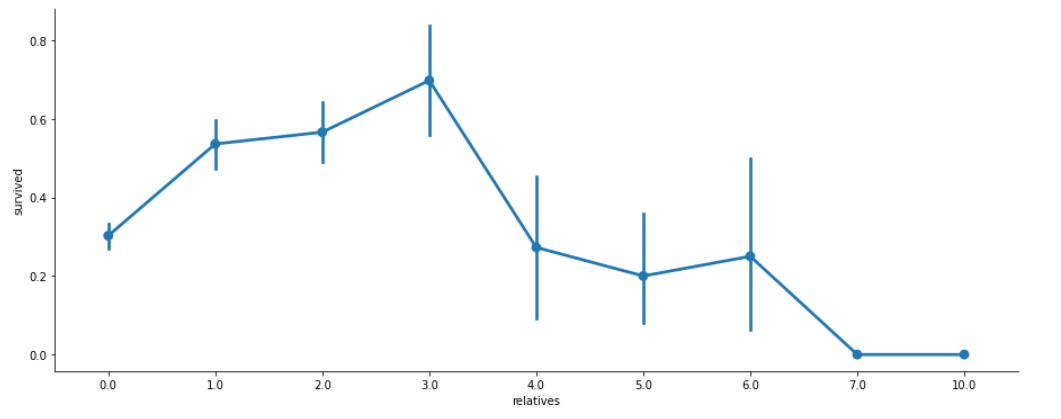
dataset.loc[dataset['relatives'] == 0, 'not\_alone'] = 1

df['not\_alone'].value\_counts()



* Trực quan hóa mối quan hệ số người đi cùng và khả năng sống sót:

axes **=** sns.factorplot('relatives','survived', data**=**df, aspect **=** 2.5, )



* Tiền xử lý dữ liệu: Lọc bỏ thuộc tính 'cabin', 'ticket', 'boat', 'body', 'home.dest':

**import** re

deck **=** {"A": 1, "B": 2, "C": 3, "D": 4, "E": 5, "F": 6, "G": 7, "U": 8}

for dataset in [df]:

dataset['cabin'] **=** dataset['cabin'].fillna("U0")

dataset['deck'] **=** dataset['cabin'].map(**lambda** x: re.compile("([a-zA-Z]+)").search(x).group())

dataset['deck'] **=** dataset['deck'].map(deck)

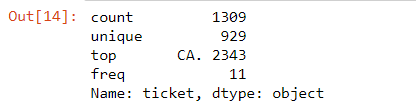
dataset['deck'] **=** dataset['deck'].fillna(0)

dataset['deck'] **=** dataset['deck'].astype(int)

df **=** df.drop(['cabin'], axis**=**1)

* Xem thuộc tính 'ticket'

df['deck'].describe()



df **=** df.drop(['ticket'],axis**=**1)

df **=** df.drop(['boat'],axis**=**1)

df **=** df.drop(['body'],axis**=**1)

df **=** df.drop(['home.dest'],axis**=**1)

* Điền giá trị còn thiếu vào thuộc tính tuổi 'age' và kiểm tra kết quả.

**for** dataset **in** [df]:

mean **=** df["age"].mean()

std **=** df["age"].std()

is\_null **=** dataset["age"].isnull().sum()

rand\_age **=** np.random.randint(mean **-** std, mean **+** std, size **=** is\_null)

age\_slice **=** dataset["age"].copy()

age\_slice[np.isnan(age\_slice)] **=** rand\_age

dataset["age"] **=** age\_slice

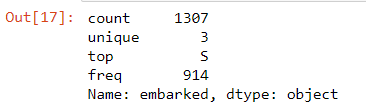
dataset["age"] **=** df["age"].astype(int)

df["age"].isnull().sum())



* Xem thông tin và điền giá trị còn thiếu vào thuộc tính Cổng lên tàu 'embarked'.

df['embarked'].describe()



.

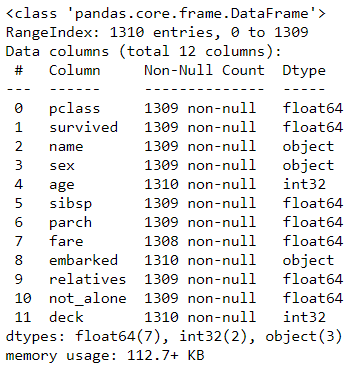
common\_value **=** 'S'

**for** dataset **in** [df]:

dataset['embarked'] **=** dataset['embarked'].fillna(common\_value)

* Xem lại thông tin của bộ dữ liệu.

df.info()



+ Xử lý thuộc tính Giá vé 'fare':

**for** dataset **in** [df]:

dataset['fare'] **=** dataset['fare'].fillna(0)

dataset['fare'] **=** dataset['fare'].astype(int)

+ Xử lý thuộc tính Tên 'name':

titles **=** {"Mr": 1, "Miss": 2, "Mrs": 3, "Master": 4, "Rare": 5}

**for** dataset **in** [df]:

dataset['title'] **=** dataset.name.str.extract(' ([A-Za-z]+)\.', expand**=False**)

dataset['title'] **=** dataset['title'].replace(['Lady', 'Countess','Capt', 'Col','Don', 'Dr',\

'Major', 'Rev', 'Sir', 'Jonkheer', 'Dona'], 'Rare')

dataset['title'] **=** dataset['title'].replace('Mlle', 'Miss')

dataset['title'] **=** dataset['title'].replace('Ms', 'Miss')

dataset['title'] **=** dataset['title'].replace('Mme', 'Mrs')

dataset['title'] **=** dataset['title'].map(titles)

dataset['title'] **=** dataset['title'].fillna(0)

df **=** df.drop(['name'], axis**=**1)

+ Xử lý thuộc tính Giới tính 'sex':

genders **=** {"male": 0, "female": 1}

**for** dataset **in** [df]:

dataset['sex'] **=** dataset['sex'].map(genders)

+ Xử lý thuộc tính Cổng lên tàu 'embarked':

ports **=** {"S": 0, "C": 1, "Q": 2}

**for** dataset **in** [df]:

dataset['embarked'] **=** dataset['embarked'].map(ports)

+ Chia nhóm thuộc tính Tuổi 'age' và xem xét kết quả:

**for** dataset **in** [df]:

dataset['Age'] **=** dataset['age'].astype(int)

dataset.loc[ dataset['age'] **<=** 11, 'age'] **=** 0

dataset.loc[(dataset['age'] **>** 11) **&** (dataset['age'] **<=** 18), 'age'] **=** 1

dataset.loc[(dataset['age'] **>** 18) **&** (dataset['age'] **<=** 22), 'age'] **=** 2

dataset.loc[(dataset['age'] **>** 22) **&** (dataset['age'] **<=** 27), 'age'] **=** 3

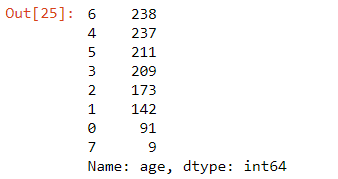
dataset.loc[(dataset['age'] **>** 27) **&** (dataset['age'] **<=** 33), 'age'] **=** 4

dataset.loc[(dataset['age'] **>** 33) **&** (dataset['age'] **<=** 40), 'age'] **=** 5

dataset.loc[(dataset['age'] **>** 40) **&** (dataset['age'] **<=** 66), 'age'] **=** 6

dataset.loc[ dataset['age'] **>** 66, 'age'] **=** 7

df['age'].value\_counts()



+ Chia nhóm thuộc tính Giá vé 'fare' và xem xét kết quả:

**for** dataset **in** [df]:

dataset.loc[ dataset['fare'] **<=** 7.91, 'fare'] **=** 0

dataset.loc[ (dataset['fare'] **>** 7.91) **&** (dataset['fare'] **<=** 14.454), 'fare'] **=** 1

dataset.loc[ (dataset['fare'] **>** 14.454) **&** (dataset['fare'] **<=** 31), 'fare'] **=** 2

dataset.loc[ (dataset['fare'] **>** 31) **&** dataset['fare'] **<=** 99, 'fare'] **=** 3

dataset.loc[ (dataset['fare'] **>** 99) **&** dataset['fare'] **<=** 250, 'fare'] **=** 4

dataset.loc[ dataset['fare'] **>** 250, 'fare'] **=** 5

dataset['fare'] **=** dataset['fare'].astype(int)

df['fare'].value\_counts()

.

+ Tạo thuộc tính Tuổi 'age' \* Loại vé'age\_pclass':

**for** dataset **in** [df]:

dataset['age\_pclass']**=** dataset['age'] **\*** dataset['pclass']

+ Tạo thuộc tính Giá vé trên mỗi người 'fare\_per\_person':

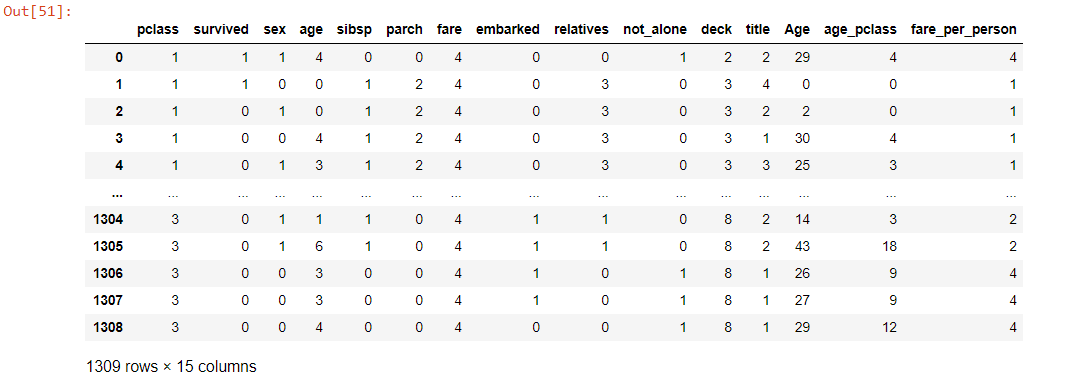
**for** dataset **in** [df]:

dataset['fare\_per\_person'] **=** round(dataset['fare']**/**(dataset['relatives'] **+** 1))

df **=** df.astype(int)

+ Xem xét kết quả:

df

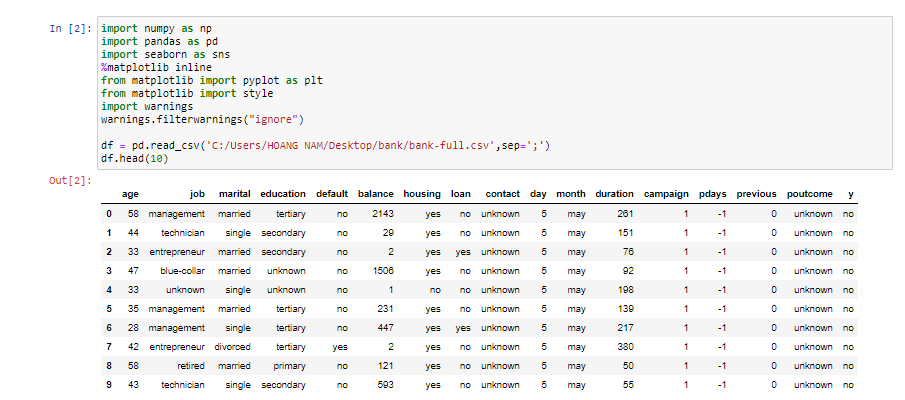


# **Bài 2**

Yêu cầu: Thực hiện khảo sát và tiền xử lý dữ liệu cho dữ liệu Bank marketing.

1. **Data Exploration**

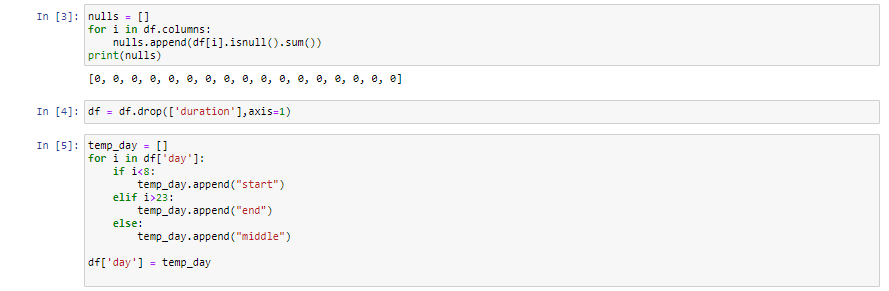
Đọc dữ liệu từ file excel



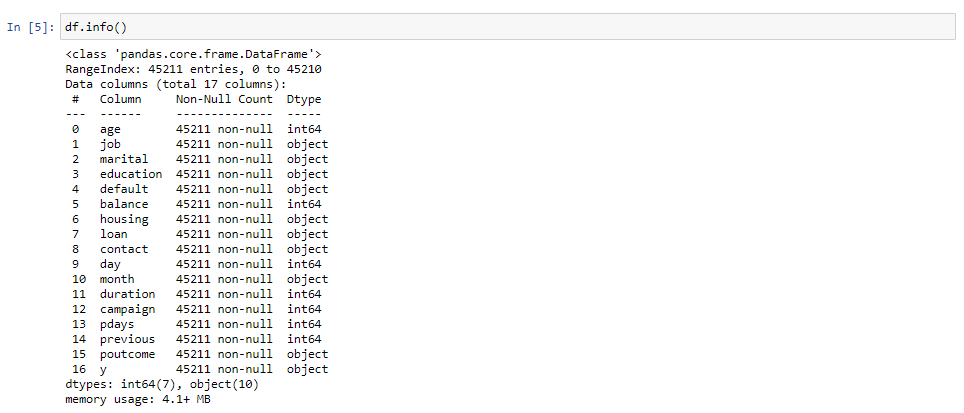
Kiểm tra file data có bao nhiêu dòng và cột



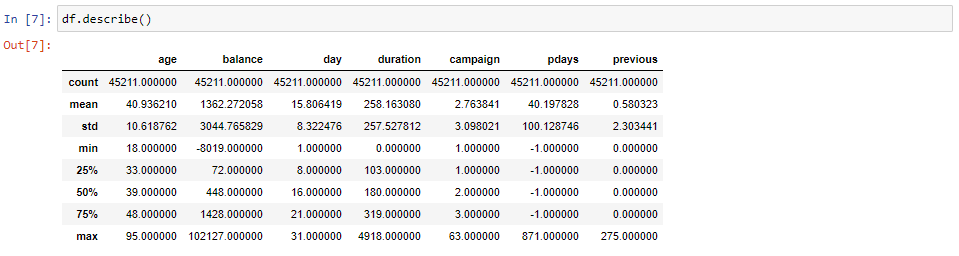
Kiểm tra số dòng dữ liệu rỗng của từng cột



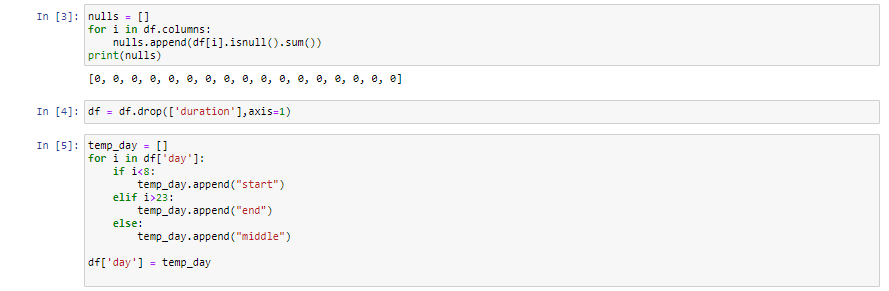
Kiểm tra kiểu dữ liệu của từng biến



Coi nhưng giá trị mô tả cơ bản của từng cột

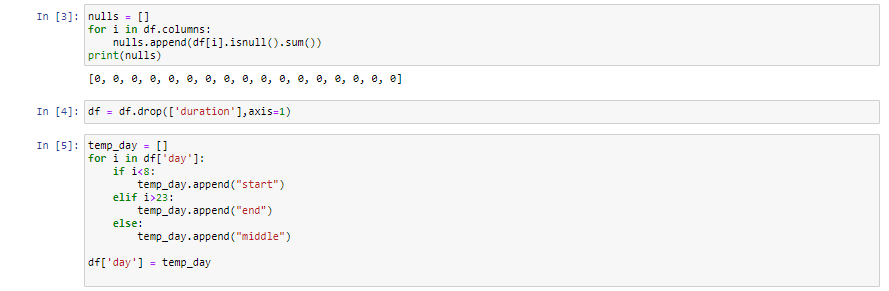


Trước khi tiếp tục, chúng ta sẽ xóa cột "duration" khỏi tập dữ liệu của mình vì đây là thông tin mà lẽ ra chúng ta chưa có. Trường này ảnh hưởng nhiều đến kết quả đầu ra vì nếu một người có thời lượng cuộc gọi bằng 0, chúng ta có thể cho rằng người gọi không nhấc máy và do đó không có cơ hội bán được hàng. Ngược lại, nếu thời lượng cuộc gọi lớn đồng nghĩa khách hàng thật sự rất quan tâm đến sản phẩm và cơ hội bán được sẽ gần như 100%.



## Data Cleaning

Cột ‘day’ thể hiện theo ngày trong tuần, chúng ta thực sự nên coi đây là một biến số chứ không phải một số. Ngoài ra, chúng ta sẽ kết hợp các giá trị này thành 3 nhóm: đầu tháng, giữa tháng và cuối tháng. Điều này sẽ ngăn chúng ta có 31 biến được mã hóa khác nhau và tổng quát hóa biến để nó có ý nghĩa đối với vấn đề kinh doanh hiện tại. Sau khi chuyển đổi "day" thành biến số, chúng ta sẽ mã hóa một biến này bằng các biến phân loại khác .



Biến "pdays" đại diện cho số ngày đã trôi qua sau khi khách hàng được liên hệ lần cuối từ một chiến dịch trước đó. Chúng ta cũng cần lưu ý rằng nếu khách hàng chưa từng được liên hệ, giá trị của trường này là -1. Trở thành giá trị -1 không có ý nghĩa trong trường này và sẽ không nắm bắt chính xác rằng khách hàng không được liên hệ Ta thường sẽ tạo một biến nhị phân mới nhưng "poutcome" = nonexistant nắm bắt điều này cho chúng ta.

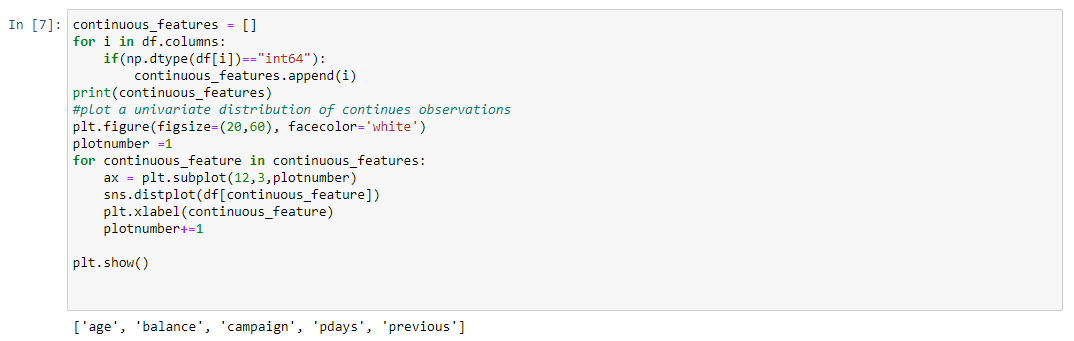
Như chúng ta đã thấy từ các thống kê ở trên về các biến số, có những danh sách tiềm năng. Hãy nói về một số giá trị tối đa cho một số biến của chúng ta:

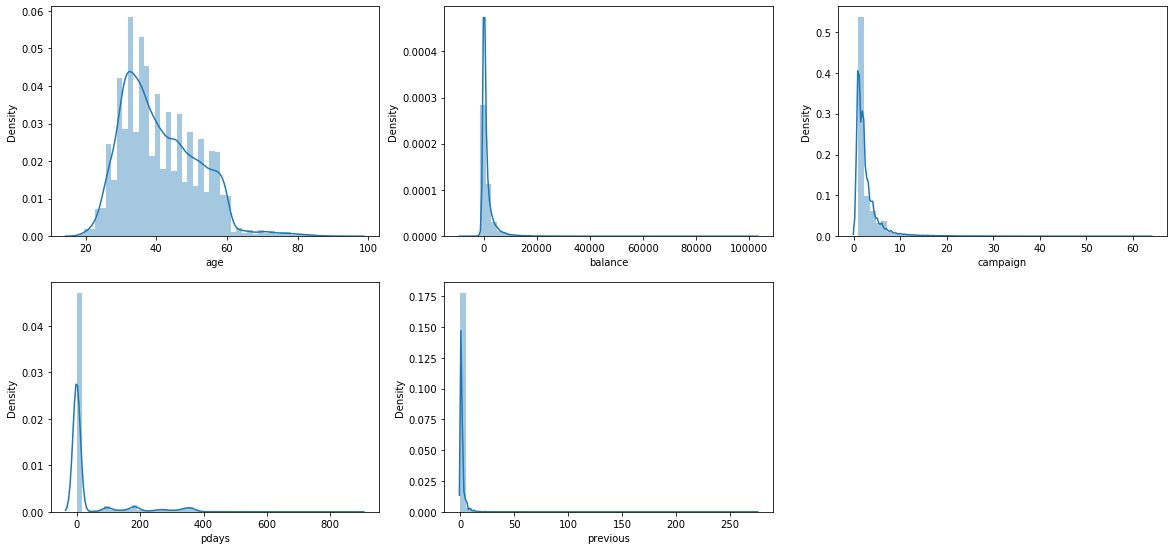
1 - **Balance**: Việc có 81.000 đô la trong tài khoản séc của bạn không hẳn là chưa từng thấy. Ví dụ, điều này sẽ có ý nghĩa nếu một người có mức lương cao, xen kẽ với việc tiết kiệm.

2 - **pdays**: Hơi bất thường khi liên hệ với khách hàng tiềm năng sau 854 ngày (gần 2,5 năm) mặc dù điều đó là hợp lý. Ví dụ: có thể sau 100 ngày, họ đã bị xóa khỏi danh sách tiếp thị nhưng một lá cờ đã được kéo lên có thể cho biết rằng người đó thích thú sẽ mở tài khoản và do đó được đưa trở lại danh sách tiếp thị. Mặc dù chúng đã bị xóa sau đó được thêm lại, nhưng có thể chúng không ngừng đếm số ngày ở giữa.

3 - **previous**: Giống như các biến trước mà chúng ta đã thảo luận, biến này cũng hợp lý. Nếu một khách hàng ở trong nhiều chiến dịch kéo dài trong nhiều quý hoặc thậm chí nhiều năm thì chắc chắn có thể thêm tới hơn 50 địa chỉ liên hệ.

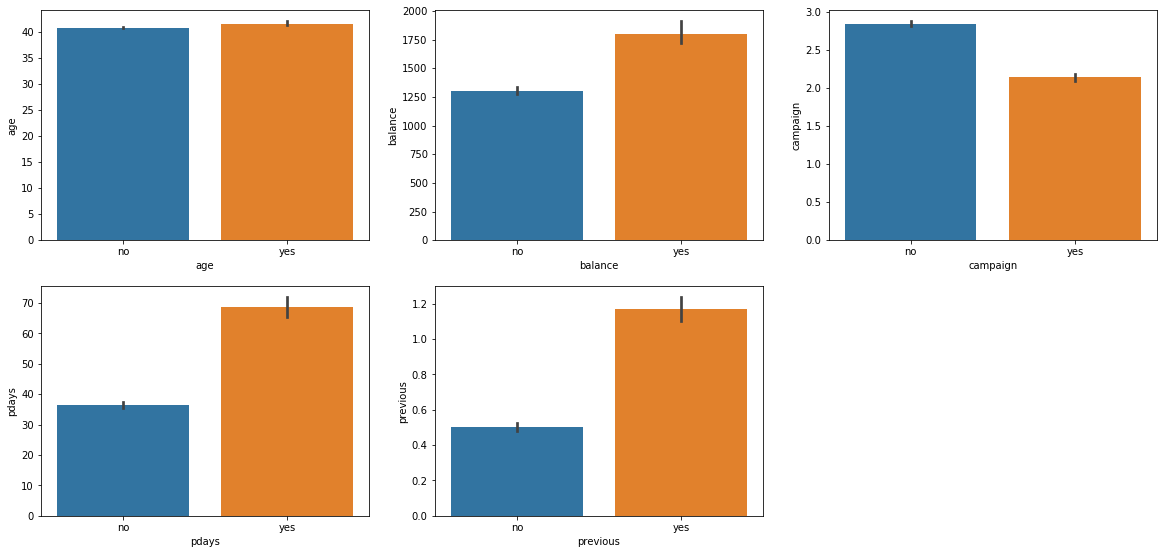
4 -  **Campaign**: Việc tranh giành hơn 60 lần cho một chiến dịch tiếp thị là điều hơi chưa từng xảy ra ngay cả đối với một chiến dịch kéo dài một năm. Hãy thử nghĩ xem, sau khi tiếp cận với một người 20 lần, bạn có nghĩ rằng họ có thể không quan tâm? Ngoài ra, hầu hết các công ty sẽ không lãng phí tiền như vậy. Điều này sẽ cần điều tra thêm.



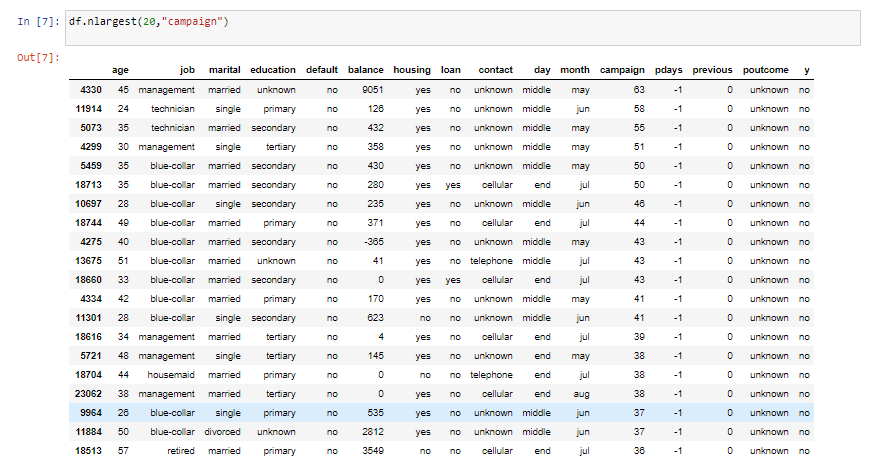


Ta thấy, có vẻ như age được phân phối bình thường

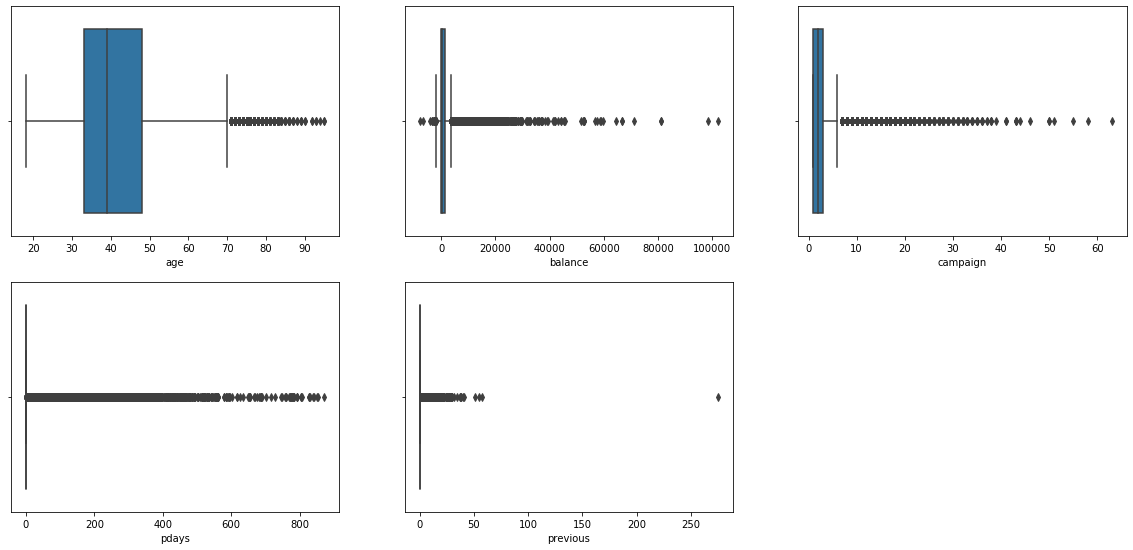
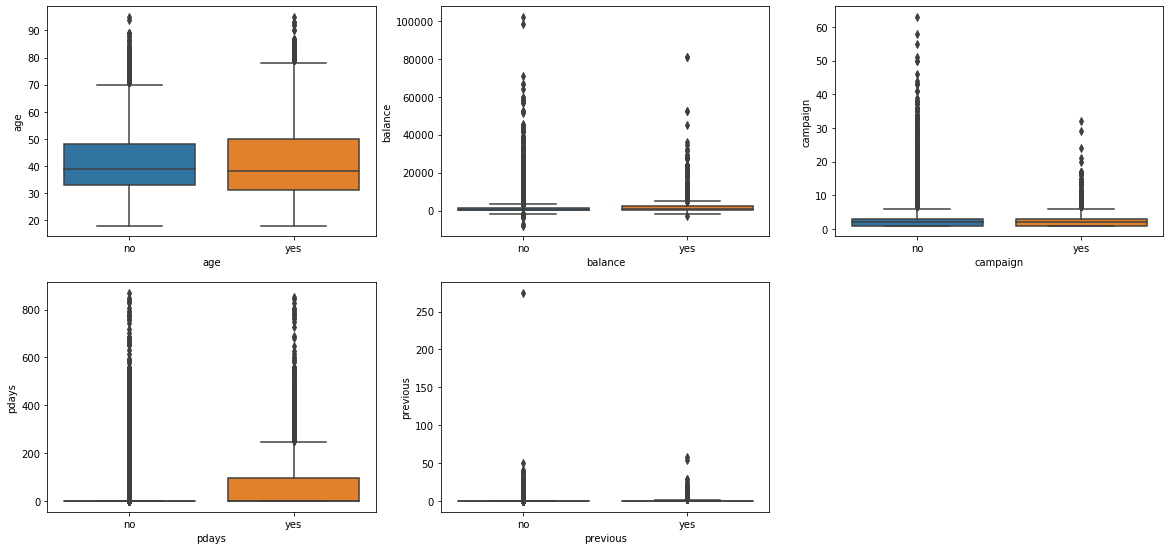
Balance, compaign, pdays and previous  bị lệch về bên trái và dường như có một số ngoại lệ.



Sắp xếp giảm “**campaign”**



Dựa trên dữ liệu trên, không có dấu hiệu cho thấy có sai sót trong dữ liệu trên. Có vẻ như tất cả các dữ liệu trên đều có mà trước đây chúng không được liên hệ với nhau. Ta có thể cho rằng công ty muốn thực hiện một cách tiếp cận linh hoạt để tiếp thị cho các khách hàng tiềm năng mới. Vì lý do này, ta sẽ không đưa ra bất kỳ quan sát nào mặc dù ta sẽ đảm bảo ghi nhớ điều này. Mặc dù, ta quyết định không loại bỏ bất kỳ biến nào, nhưng chắc chắn có một trường hợp mạnh mẽ được thực hiện để chuyển các hàng hoặc chuyển đổi các cột để điều chỉnh cho các giá trị ngoại lệ.

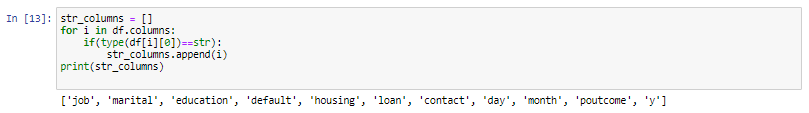


age, balance, compaign, pdays and previous có một số ngoại lệ

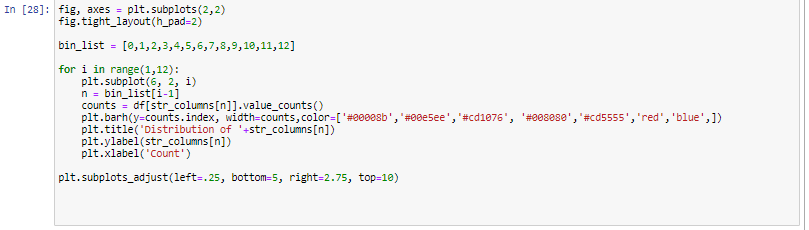
Chọn những cột có kiểu dữ liệu String

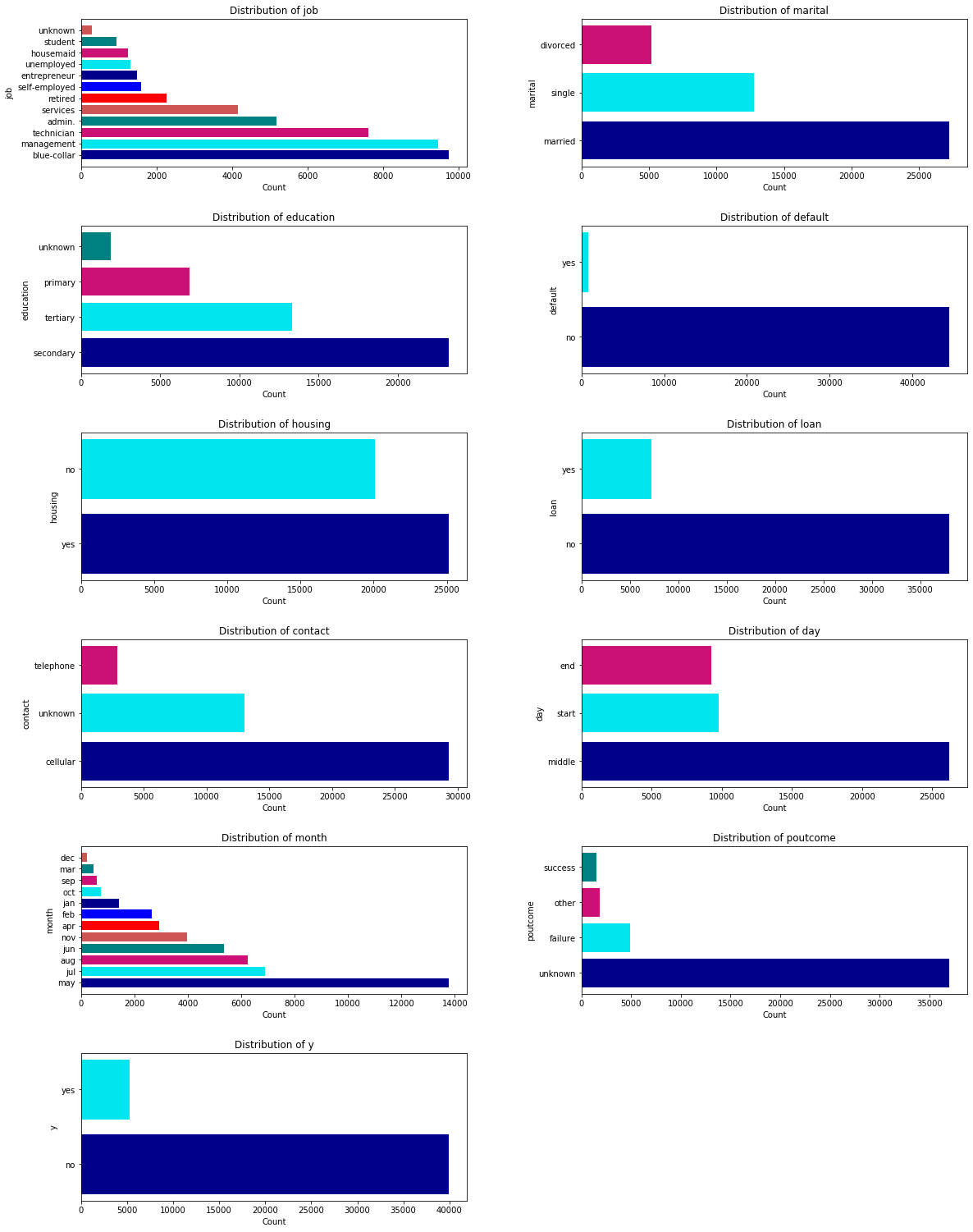


Lấy ra các biến số kiểu string

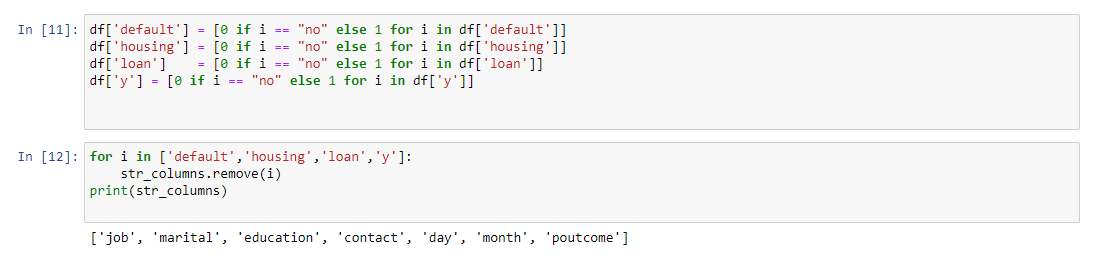


Đồ thị phân tán của từng biến

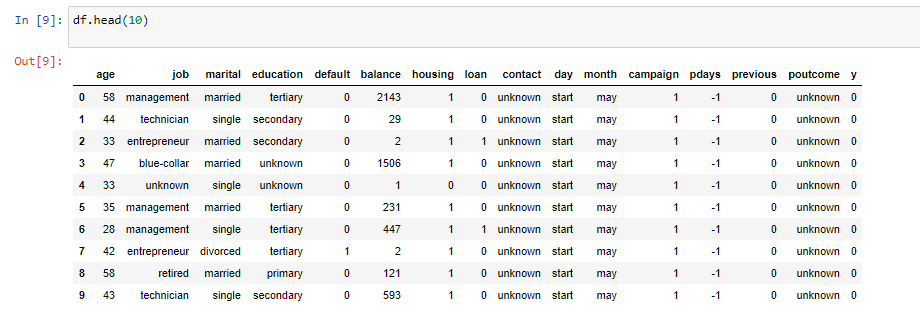




Thay đổi giá trị ký tự trong các cột trên thành 0/1 giá trị nhị phân 0 = "không" và 1 = "có"



Bảng dữ liệu cuối cùng:



Ở trên, chúng ta đã điều chỉnh các biến phân loại nhị phân và biến chúng thành các biến số mà chúng ta có thể làm việc với. Bây giờ chúng ta có thể biến đổi các biến phân loại có chứa nhiều hơn một giá trị duy nhất bằng cách sử dụng mã hóa. Trước khi chúng ta chuyển đổi các biến, chúng ta phải xem phân phối để có được ý tưởng về hành vi cho mỗi biến

Một số lưu ý từ các bản phân phối trên:

Thật thú vị khi lượng cuộc gọi lớn nhất xảy ra vào những tháng mùa hè và yếu nhất trong những tháng lạnh hơn.

Hãy nhớ rằng "poutcome" phản ánh cách họ phản ứng với chiến dịch trước đó. Chúng ta cần lưu ý rằng khoảng 75% trong danh mục này được gắn nhãn là "không xác định", phù hợp với những gì chúng ta đã tìm thấy trước đó.

Dựa trên các biến nhân khẩu học, có vẻ như chiến dịch này nhắm mục tiêu đến các cá nhân thuộc tầng lớp trung lưu hỗ trợ thống kê từ biến "balance" của chúng ta.