

## Lab 01: Trực quan hóa dữ liệu với Python | nhóm 22

### Employee Attrition and Factorsy

#### Các thư viện sử dụng

```
"""
!pip install matplotlib
!pip install seaborn
!pip install plotly
!pip install wordcloud
!pip install altair
!pip install missingno
!pip install imbalanced-learn
!pip install sklearn
"""

'\n!pip install matplotlib\n!pip install seaborn\n!pip install plotly\n!pip install wordcloud\n!pip install altair\n!pip install missingno\n!pip install imbalanced-learn\n!pip install sklearn\n'

import pandas as pd
import numpy as np

#Thư viện cho trực quan hóa
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
from wordcloud import WordCloud
from plotly.offline import iplot
from plotly.subplots import make_subplots
import missingno as msno
from pandas.plotting import parallel_coordinates
import altair as alt

#Thư viện cho xây dựng mô hình học máy
import sklearn
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report, accuracy_score
```

## Mục lục

- A. Thu thập dữ liệu
  - I. Giới thiệu chủ đề và thông tin tập dữ liệu
    - 1. Chủ đề
    - 2. Ngữ cảnh tìm kiếm dữ liệu
    - 3. Thông tin bộ dữ liệu
  - II. Cấu trúc bộ dữ liệu
- B. Khám phá dữ liệu
  - I. Mục tiêu
  - II. Nội dung
    - 1. Đọc dữ liệu và tính số dòng và cột
    - 2. Mỗi dòng có ý nghĩa gì? Có vấn đề các dòng có ý nghĩa khác nhau không?
    - 3. Dữ liệu có các dòng bị lặp không?
    - 4. Tỷ lệ giá trị thiếu và thống kê mô tả của từng cột
    - 5. Kiểu dữ liệu của mỗi cột
    - 6. Xem xét tập giá trị của các thuộc tính phân loại
    - 7. Xem xét sự phân bố giá trị của các cột dữ liệu dạng số
    - 8. Xem xét sự phân bố giá trị của các cột dữ liệu không phải dạng số
- C. Khám phá mối quan hệ trong dữ liệu
  - Biểu đồ 1
  - Biểu đồ 2
  - Biểu đồ 3
  - Biểu đồ 4
  - Biểu đồ 5
  - Biểu đồ 6
  - Biểu đồ 7
  - Biểu đồ 8
  - Biểu đồ 9
  - Biểu đồ 10
  - Biểu đồ 11
  - Biểu đồ 12
  - Biểu đồ 13
  - Biểu đồ 14
  - Biểu đồ 15
- D. Mô hình học máy
  - I. Bài toán đặt ra
  - II. Tiền xử lý dữ liệu
    - 1. Mã hóa các thuộc tính dạng danh mục về dạng số
    - 2. Loại những thuộc tính không có ý nghĩa cho bài toán

- 3. Xử lý các giá trị thiếu
- 4. Feature Scaling
- III. Xây dựng mô hình học máy
  - 1. Logistic Regression cho Phân loại nhị phân
  - 2. Sử dụng Pipeline và Các độ đo được dùng để đánh giá mô hình
  - 3. Cài đặt
- V. Tổng kết

## A. Thu thập dữ liệu

I. Giới thiệu chủ đề & thông tin tập dữ liệu

1. Chủ đề

**Tên chủ đề:** Employee Attrition and Factors.

**Tạm dịch:** Các yếu tố ảnh hưởng đến sự tiêu hao nhân viên.

2. Ngữ cảnh tìm kiếm dữ liệu

Trong môi trường làm việc, Employee Attrition diễn tả sự tiêu hao lực lượng lao động không được dự báo trước. Nguyên nhân của sự sụt giảm này đều là những lý do không thể tránh được như nghỉ hưu, từ chức, nhân viên mất sức lao động hay đột ngột qua đời. Những công ty có tỷ lệ tiêu hao lực lượng lao động cao thường phải đối mặt với nguy cơ lạm dụng nguồn lực nội bộ.

Nhận thấy, có rất nhiều nguyên nhân khác nhau khiến cho tỷ lệ tiêu hao lực lượng lao động cao ở các doanh nghiệp. Ví dụ như:

- Điều kiện làm việc không đảm bảo.
- Mức lương quá thấp.
- Công việc không phù hợp với sở thích.
- Không có tương lai phát triển sự nghiệp.
- Không thể cân bằng giữa công việc và cuộc sống.
- Thiếu sự nhìn nhận và đánh giá đúng mực đối với nhân viên từ phía những người quản lý.

Tỷ lệ tiêu hao lực lượng lao động là một chỉ số quan trọng trong quản trị nguồn nhân lực, có thể cho thấy các vấn đề còn tồn đọng cần phải được giải quyết. Attrition rate thấp cho thấy công ty đang đi đúng hướng. Ngược lại, attrition rate cao là điều không công ty nào mong muốn.

Do đó, nhóm chọn chủ đề và bộ dữ liệu này nhằm mục đích:

- Khám phá ra các yếu tố ảnh hưởng đến sự tiêu hao nhân viên và khám phá các đặc trưng của tổ chức này.
- Xây dựng mô hình học máy dựa trên các yếu tố của nhân viên để dự đoán xem liệu nhân viên đó có khả năng tiêu hao hay không?

### 3. Thông tin bộ dữ liệu

#### Nguồn gốc:

Bộ dữ liệu được tìm thấy trên Kaggle:

<https://www.kaggle.com/datasets/thedevastator/employee-attrition-and-factors>

**Người đăng tải:** The Devastator

#### Phương pháp thu thập:

Đây là một tập dữ liệu hư cấu được tạo bởi các nhà khoa học dữ liệu IBM. Bộ dữ liệu này nhằm phục vụ cho mục đích nghiên cứu và khám phá ra các yếu tố dẫn đến sự tiêu hao của nhân viên hoặc xây dựng các mô hình học máy để dự đoán sự tiêu hao nhân viên.

#### Giấy phép:

- CO 1.0 Universal (CC0 1.0) - Public Domain Dedication
- No Copyright - Bạn có thể sao chép, sửa đổi, phân phối và thực hiện công việc, ngay cả cho mục đích thương mại, tất cả mà không cần xin phép.

[⬆️ Back to Table of Contents](#) [⬆️](#)

## II. Cấu trúc bộ dữ liệu

Bộ dữ liệu này cung cấp một phân tích toàn diện và đa dạng về nhân viên của một tổ chức, tập trung vào các lĩnh vực như sự tiêu hao của nhân viên, các yếu tố cá nhân và liên quan đến công việc cũng như tài chính. Bao gồm nhiều 35 thuộc tính và ý nghĩa các thuộc tính như sau:

STT	Tên thuộc tính	Ý nghĩa
1	Age	Tuổi của nhân viên
2	Gender	Giới tính của nhân viên
3	BusinessTravel	Tần suất đi công tác của nhân viên
4	DailyRate	Tỷ lệ lương hàng ngày cho nhân viên
5	Department	Phòng ban làm việc của nhân viên
6	DistanceFromHome	Khoảng cách từ nhà theo dặm đến nơi làm việc
7	Education	Mức độ giáo dục đạt được

STT	Tên thuộc tính	Ý nghĩa
		bởi nhân viên
8	EducationField	Lĩnh vực học tập của nhân viên
9	EmployeeCount	Tổng số nhân viên trong tổ chức
10	EmployeeNumber	Một định danh duy nhất cho mỗi hồ sơ nhân viên
11	EnvironmentSatisfaction	Sự hài lòng của nhân viên với môi trường làm việc của họ
12	HourlyRate	Tỷ lệ lương hàng giờ cho nhân viên
13	JobInvolvement	Mức độ tham gia cần thiết cho công việc của nhân viên
14	JobLevel	Mức độ công việc của nhân viên
15	JobRole	Vai trò của nhân viên trong tổ chức
16	JobSatisfaction	Sự hài lòng của nhân viên với công việc của họ
17	MaritalStatus	Tình trạng hôn nhân của nhân viên
18	MonthlyIncome	Thu nhập hàng tháng của nhân viên
19	MonthlyRate	Tỷ lệ lương hàng tháng cho nhân viên
20	NumCompaniesWorked	Số lượng công ty mà nhân viên đã làm việc cho
21	Over18	Nhân viên có trên 18 tuổi hay không
22	Overtime	Nhân viên có tăng ca làm việc hay không
23	PercentSalaryHike	Tỷ lệ tăng lương cho nhân viên
24	PerformanceRating	The performance rating of the employee
25	RelationshipSatisfaction	Sự hài lòng của nhân viên với các mối quan hệ của họ
26	StandardHours	Giờ làm việc tiêu chuẩn cho

STT	Tên thuộc tính	Ý nghĩa
		nhân viên
27	StockOptionLevel	Mức tùy chọn cổ phiếu của nhân viên
28	TotalWorkingYears	Tổng số năm mà nhân viên đã làm việc
29	TrainingTimesLastYear	Số lần nhân viên được thực hiện để đào tạo trong năm ngoái
30	WorkLifeBalance	Nhận thức của nhân viên về cân bằng cuộc sống công việc của họ
31	YearsAtCompany	Số năm nhân viên đã ở cùng với công ty
32	YearsInCurrentRole	Số năm mà nhân viên đã ở trong vai trò hiện tại của họ
33	YearsSinceLastPromotion	Số năm kể từ khi thăng chức cuối cùng của nhân viên
34	YearsWithCurrManager	Số năm mà nhân viên đã làm với người quản lý hiện tại của họ
35	Attrition	Nhân viên có rời khỏi tổ chức hay không

[⬆️ Back to Table of Contents ⬆️](#)

## B. Khám phá dữ liệu

### I. Mục tiêu

🗨️ Ở phần này chúng ta sẽ thực hiện khám phá dữ liệu đã thu thập bằng cách sử dụng thống kê mô tả để hiểu dữ liệu tốt hơn, tức là để xác định các vấn đề về dữ liệu (dữ liệu bị thiếu giá trị, giá trị không hợp lệ, cột có kiểu dữ liệu không phù hợp để xử lý thêm,...). Thông qua việc khám phá dữ liệu, có thể ta sẽ phát hiện ra những điểm bất thường, không hợp lý của dữ liệu, từ đó thực hiện tiền xử lý để dữ liệu trở nên rõ ràng và dễ hiểu hơn, phục vụ tốt cho các mục đích khác.

[⬆️ Back to Table of Contents ⬆️](#)

### II. Nội dung

## 1. Đọc dữ liệu và tính số dòng và cột

➡ Tiếp đến đọc file "HR\_Analytics.csv" vào dataframe df và in ra 5 dòng đầu tiên của dataframe.

```
df = pd.read_csv("./datasets/HR_Analytics.csv")
df.head()
```

	Age	Attrition	BusinessTravel	DailyRate	Department
0	41	Yes	Travel_Rarely	1102	Sales
1	49	No	Travel_Frequently	279	Research & Development
2	37	Yes	Travel_Rarely	1373	Research & Development
3	33	No	Travel_Frequently	1392	Research & Development
4	27	No	Travel_Rarely	591	Research & Development

	DistanceFromHome	Education	EducationField	EmployeeCount
0	1	2	Life Sciences	1
1	8	1	Life Sciences	1
2	2	2	Other	1
4	3	4	Life Sciences	1
5	2	1	Medical	1
7				

	RelationshipSatisfaction	StandardHours	StockOptionLevel
0	1	80	0
1	4	80	1
2	2	80	0
3	3	80	0
4	4	80	1

	TotalWorkingYears	TrainingTimesLastYear	WorkLifeBalance
0	8	0	1
6			
1	10	3	3
10			
2	7	3	3
0			
3	8	3	3

8			
4	6	3	3
2			

	YearsInCurrentRole	YearsSinceLastPromotion	YearsWithCurrManager
0	4	0	5
1	7	1	7
2	0	0	0
3	7	3	0
4	2	2	2

[5 rows x 35 columns]

☞ Tính số dòng và số cột và lưu vào 2 biến `num_rows` và `num_cols`.

```
num_rows, num_cols = df.shape
print(f'Number of rows: {num_rows}\nNumber of columns: {num_cols}')
```

Number of rows: 1470  
Number of columns: 35

2. Mỗi dòng có ý nghĩa gì? Có vấn đề các dòng có ý nghĩa khác nhau không?

💬 **Nhận xét:**

- Tập dữ liệu cung cấp một phân tích toàn diện và đa dạng về nhân viên ở một tổ chức.
- Mỗi dòng là thông tin của một nhân viên tập trung vào các khía cạnh như chi tiêu, các yếu tố cá nhân, công việc và tài chính.

=> **Vì thế sẽ không xuất hiện dòng nào có ý nghĩa khác.**

3. Dữ liệu có các dòng bị lặp không?

☞ Ta kiểm tra xem có dòng nào bị lặp không bằng cách sử dụng các phương thức `uplicated()` và `any()` trên dataframe `df` và lưu kết quả vào biến `have_duplicated_rows`. Biến này sẽ có giá trị `True` nếu dữ liệu có các dòng bị lặp và có giá trị `False` nếu ngược.

```
have_duplicate_rows = df.duplicated().any()
have_duplicate_rows
```

False

💬 **Nhận xét:** Ta thấy rằng dữ liệu không có dòng nào bị trùng lặp.

4. Tỷ lệ giá trị thiếu và thống kê mô tả của từng cột

☞ Để tính tỷ lệ giá trị thiếu, sử dụng phương thức `isna()` và `mean()` và lưu vào biến `missing_rate`

☞ Để thống kê mô tả mỗi cột, phương thức `describe()`.



Trong đó:

- count: số lượng giá trị không bị thiếu trong cột.
- mean : giá trị trung bình của các giá trị trong cột.
- std : độ lệch chuẩn của các giá trị trong cột.
- min : giá trị nhỏ nhất trong cột.
- 25%, 50%, 75% : các **phân vị** tương ứng với các mức phân chia dữ liệu là 25%, 50% và 75%.
- max : giá trị lớn nhất trong cột.

🔗 Thống kê các cột numerical

```
describe = df.describe()
missing_rates = df[describe.columns].isna().mean()
missing_rates.name = 'missing_rate'
describe = pd.concat([describe, missing_rates.to_frame().T])
describe
```

	Age	DailyRate	DistanceFromHome	Education
\count	1470.000000	1470.000000	1470.000000	1470.000000
mean	36.923810	802.485714	9.192517	2.912925
std	9.135373	403.509100	8.106864	1.024165
min	18.000000	102.000000	1.000000	1.000000
25%	30.000000	465.000000	2.000000	2.000000
50%	36.000000	802.000000	7.000000	3.000000
75%	43.000000	1157.000000	14.000000	4.000000
max	60.000000	1499.000000	29.000000	5.000000
missing_rate	0.000000	0.000000	0.000000	0.000000

	EmployeeCount	EmployeeNumber	
EnvironmentSatisfaction \count	1470.0	1470.000000	1470.000000
mean	1.0	1024.865306	2.721769
std	0.0	602.024335	1.093082
min	1.0	1.000000	1.000000

25%	1.0	491.250000	2.000000
50%	1.0	1020.500000	3.000000
75%	1.0	1555.750000	4.000000
max	1.0	2068.000000	4.000000
missing_rate	0.0	0.000000	0.000000

	HourlyRate	JobInvolvement	JobLevel	...	\
count	1470.000000	1470.000000	1470.000000	...	
mean	65.891156	2.729932	2.063946	...	
std	20.329428	0.711561	1.106940	...	
min	30.000000	1.000000	1.000000	...	
25%	48.000000	2.000000	1.000000	...	
50%	66.000000	3.000000	2.000000	...	
75%	83.750000	3.000000	3.000000	...	
max	100.000000	4.000000	5.000000	...	
missing_rate	0.000000	0.000000	0.000000	...	

	RelationshipSatisfaction	StandardHours
StockOptionLevel \		
count	1470.000000	1470.0
1470.000000		
mean	2.712245	80.0
0.793878		
std	1.081209	0.0
0.852077		
min	1.000000	80.0
0.000000		
25%	2.000000	80.0
0.000000		
50%	3.000000	80.0
1.000000		
75%	4.000000	80.0
1.000000		
max	4.000000	80.0
3.000000		
missing_rate	0.000000	0.0
0.000000		

	TotalWorkingYears	TrainingTimesLastYear
WorkLifeBalance \		
count	1470.000000	1470.000000
1470.000000		
mean	11.279592	2.799320

2.761224		
std	7.780782	1.289271
0.706476		
min	0.000000	0.000000
1.000000		
25%	6.000000	2.000000
2.000000		
50%	10.000000	3.000000
3.000000		
75%	15.000000	3.000000
3.000000		
max	40.000000	6.000000
4.000000		
missing_rate	0.000000	0.000000
0.000000		

	YearsAtCompany	YearsInCurrentRole
YearsSinceLastPromotion \		
count	1470.000000	1470.000000
1470.000000		
mean	7.008163	4.229252
2.187755		
std	6.126525	3.623137
3.222430		
min	0.000000	0.000000
0.000000		
25%	3.000000	2.000000
0.000000		
50%	5.000000	3.000000
1.000000		
75%	9.000000	7.000000
3.000000		
max	40.000000	18.000000
15.000000		
missing_rate	0.000000	0.000000
0.000000		

	YearsWithCurrManager
count	1470.000000
mean	4.123129
std	3.568136
min	0.000000
25%	2.000000
50%	3.000000
75%	7.000000
max	17.000000
missing_rate	0.000000

[9 rows x 26 columns]

👉 Thống kê các cột category

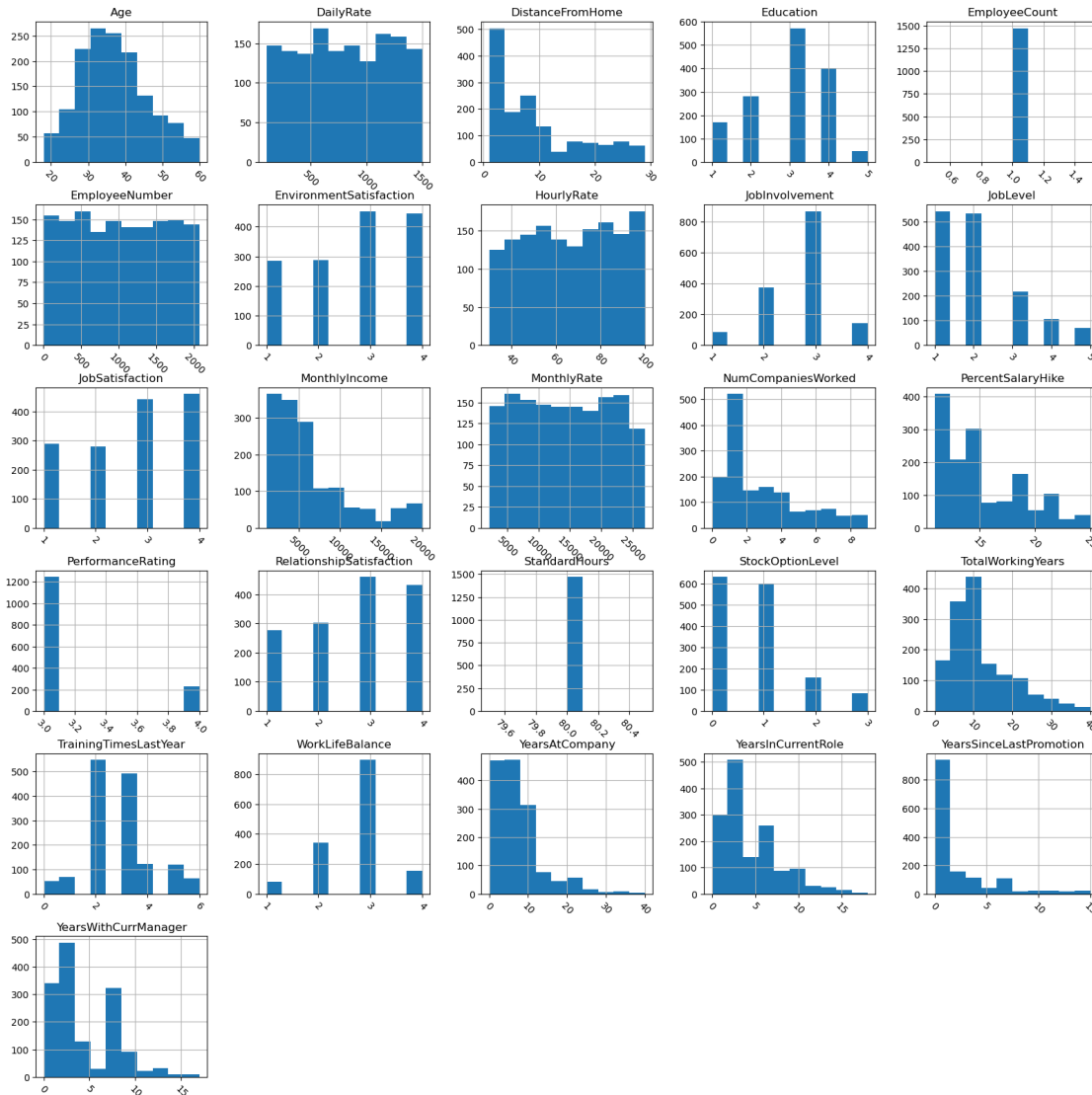
```
describe = df.describe(include=['O'])
missing_rates = df[describe.columns].isna().mean()
missing_rates.name = 'missing_rate'
describe = pd.concat([describe, missing_rates.to_frame().T])
describe
```

	Attrition	BusinessTravel	Department
EducationField \			
count	1470	1470	1470
unique	2	3	3
top	No	Travel_Rarely	Research & Development
Sciences			Life
freq	1233	1043	961
missing_rate	0.0	0.0	0.0

	Gender	JobRole	MaritalStatus	Over18	OverTime
count	1470	1470	1470	1470	1470
unique	2	9	3	1	2
top	Male	Sales Executive	Married	Y	No
freq	882	326	673	1470	1054
missing_rate	0.0	0.0	0.0	0.0	0.0

👉 Biểu đồ histogram cho các cột numerical

```
df.hist(figsize=(20,20), xrot=-45)
plt.show()
```



**Nhận xét:** Các cột sau đây có thể lược bỏ bởi vì giá trị của chúng không ảnh hưởng đến kết quả phân tích

1. Over18: Các giá trị đều là Y
2. EmployeeCount: các giá trị đều là 1.0
3. StandardHours: các giá trị đều là 80.0
4. EmployeeNumber: là id của nhân viên có ý nghĩa tương tự như index của mỗi dòng

## 5. Kiểu dữ liệu của mỗi cột

Ta sử dụng phương thức `dtypes` trên dataframe `df` để xem kiểu dữ liệu của mỗi cột. Kết quả được lưu vào series `col_dtypes`; series này có index là tên các cột và giá trị là kiểu dữ liệu của các cột tương ứng.

```
col_dtype = df.dtypes
col_dtype
```

Age	int64
Attrition	object
BusinessTravel	object
DailyRate	int64
Department	object
DistanceFromHome	int64
Education	int64
EducationField	object
EmployeeCount	int64
EmployeeNumber	int64
EnvironmentSatisfaction	int64
Gender	object
HourlyRate	int64
JobInvolvement	int64
JobLevel	int64
JobRole	object
JobSatisfaction	int64
MaritalStatus	object
MonthlyIncome	int64
MonthlyRate	int64
NumCompaniesWorked	int64
Over18	object
OverTime	object
PercentSalaryHike	int64
PerformanceRating	int64
RelationshipSatisfaction	int64
StandardHours	int64
StockOptionLevel	int64
TotalWorkingYears	int64
TrainingTimesLastYear	int64
WorkLifeBalance	int64
YearsAtCompany	int64
YearsInCurrentRole	int64
YearsSinceLastPromotion	int64
YearsWithCurrManager	int64
dtype:	object

☞ Ta sử dụng phương thức `select_dtypes` để liệt kê các cột kiểu numerical và category.

```
cat_coulmns = df.select_dtypes(['object']).columns
num_coulmns = df.select_dtypes(['number']).columns
print(cat_coulmns)
print(num_coulmns)
```

```
Index(['Attrition', 'BusinessTravel', 'Department', 'EducationField',
      'Gender',
      'JobRole', 'MaritalStatus', 'Over18', 'OverTime'],
      dtype='object')
Index(['Age', 'DailyRate', 'DistanceFromHome', 'Education',
      'EmployeeCount',
```

```

    'EmployeeNumber', 'EnvironmentSatisfaction', 'HourlyRate',
    'JobInvolvement', 'JobLevel', 'JobSatisfaction',
'MonthlyIncome',
    'MonthlyRate', 'NumCompaniesWorked', 'PercentSalaryHike',
    'PerformanceRating', 'RelationshipSatisfaction',
'StandardHours',
    'StockOptionLevel', 'TotalWorkingYears',
'TrainingTimesLastYear',
    'WorkLifeBalance', 'YearsAtCompany', 'YearsInCurrentRole',
    'YearsSinceLastPromotion', 'YearsWithCurrManager'],
dtype='object')

```

## 6. Xem xét tập giá trị của các thuộc tính phân loại

☞ Tên các thuộc tính dạng số và không phải dạng số lần lượt được lưu vào `num_cols` và `cat_cols`.

```

num_cols = list(set(df._get_numeric_data()))
cat_cols = list(set(df.columns) - set(df._get_numeric_data()))

```

☞ Xem xét mỗi thuộc tính phân loại có bao nhiêu giá trị phân biệt bằng phương thức `set()`.

```

for col in cat_cols:
    print('Unique values of ', col, set(df[col]))

```

```

Unique values of EducationField {'Medical', 'Life Sciences', 'Other',
'Human Resources', 'Marketing', 'Technical Degree'}
Unique values of Gender {'Female', 'Male'}
Unique values of Department {'Sales', 'Human Resources', 'Research &
Development'}
Unique values of JobRole {'Sales Representative', 'Manufacturing
Director', 'Sales Executive', 'Laboratory Technician', 'Healthcare
Representative', 'Research Director', 'Human Resources', 'Research
Scientist', 'Manager'}
Unique values of Over18 {'Y'}
Unique values of BusinessTravel {'Travel_Rarely',
'Travel_Frequently', 'Non-Travel'}
Unique values of MaritalStatus {'Divorced', 'Single', 'Married'}
Unique values of Attrition {'Yes', 'No'}
Unique values of OverTime {'Yes', 'No'}

```

💬 **Nhận xét:**

Tập giá trị của các thuộc tính phân loại đầy đủ, dễ hiểu nên ta không cần thực hiện các bước tiền xử lý đối với các thuộc tính này.

## 7. Xem xét sự phân bố giá trị của các cột dữ liệu dạng số

☞ Thực hiện thống kê trên các cột dữ liệu dạng số và xem xét các giá trị cụ thể như sau:

- Tỷ lệ % (từ 0 đến 100) các giá trị thiếu (`missing_ratio`).

- Giá trị min (min).
- Giá trị lower quartile (phần vị 25) (lower\_quartile).
- Giá trị median (phần vị 50) (median).
- Giá trị upper quartile (phần vị 75) (upper\_quartile).
- Giá trị max (max).

👉 Tính tổng giá trị thiếu của từng cột bằng phương thức `isnull()` và `sum()`, rồi chia cho số dòng để được tỉ lệ giá trị thiếu `missing_ratio`.

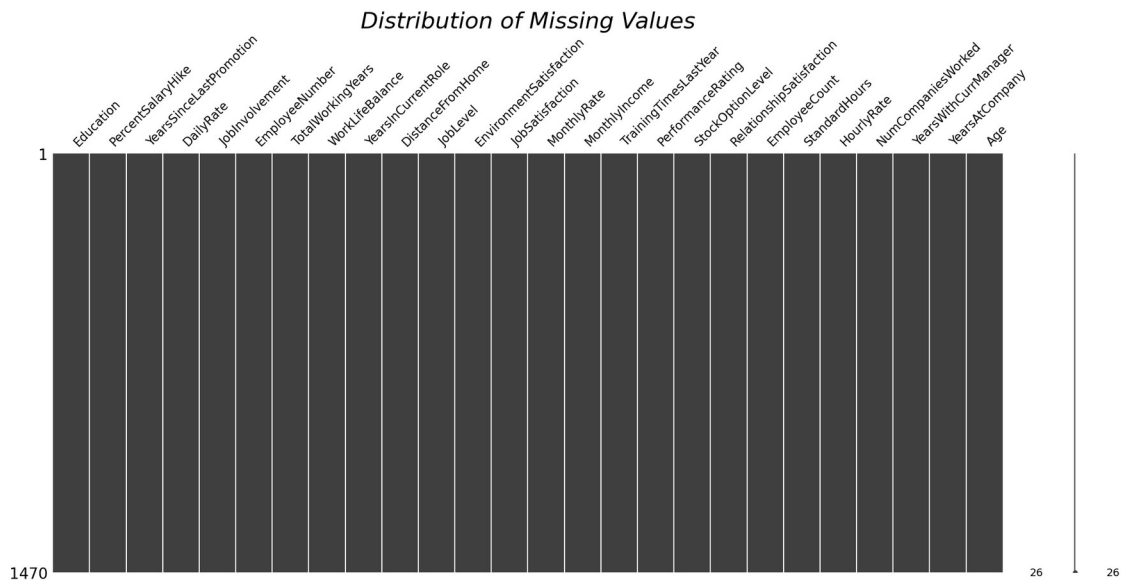
```
missing_ratio = df[num_cols].isnull().sum()
missing_ratio = missing_ratio / num_rows
missing_ratio_df = pd.DataFrame({'missing_ratio': missing_ratio})
missing_ratio_df
```

	missing_ratio
Education	0.0
PercentSalaryHike	0.0
YearsSinceLastPromotion	0.0
DailyRate	0.0
JobInvolvement	0.0
EmployeeNumber	0.0
TotalWorkingYears	0.0
WorkLifeBalance	0.0
YearsInCurrentRole	0.0
DistanceFromHome	0.0
JobLevel	0.0
EnvironmentSatisfaction	0.0
JobSatisfaction	0.0
MonthlyRate	0.0
MonthlyIncome	0.0
TrainingTimesLastYear	0.0
PerformanceRating	0.0
StockOptionLevel	0.0
RelationshipSatisfaction	0.0
EmployeeCount	0.0
StandardHours	0.0
HourlyRate	0.0
NumCompaniesWorked	0.0
YearsWithCurrManager	0.0
YearsAtCompany	0.0
Age	0.0

Trực quan sự phân bố các giá trị thiếu bằng thư viện `missingno`

```
msno.matrix(df[num_cols])
plt.title('Distribution of Missing Values', fontsize=30, fontstyle=
'oblique');
```





👉 Tính các giá trị thống kê mô tả bằng phương thức `describe()`.

```
num_cols_info_df = df[num_cols].describe()
num_cols_info_df
```

	Education	PercentSalaryHike	YearsSinceLastPromotion
DailyRate \			
count	1470.000000	1470.000000	1470.000000
mean	2.912925	15.209524	2.187755
std	1.024165	3.659938	3.222430
min	1.000000	11.000000	0.000000
25%	2.000000	12.000000	0.000000
50%	3.000000	14.000000	1.000000
75%	4.000000	18.000000	3.000000
max	5.000000	25.000000	15.000000

	JobInvolvement	EmployeeNumber	TotalWorkingYears
WorkLifeBalance \			
count	1470.000000	1470.000000	1470.000000
mean	2.729932	1024.865306	11.279592
std	0.711561	602.024335	7.780782
min	1.000000	1.000000	0.000000

1.000000			
25%	2.000000	491.250000	6.000000
2.000000			
50%	3.000000	1020.500000	10.000000
3.000000			
75%	3.000000	1555.750000	15.000000
3.000000			
max	4.000000	2068.000000	40.000000
4.000000			

	YearsInCurrentRole	DistanceFromHome	...	PerformanceRating	\
count	1470.000000	1470.000000	...	1470.000000	
mean	4.229252	9.192517	...	3.153741	
std	3.623137	8.106864	...	0.360824	
min	0.000000	1.000000	...	3.000000	
25%	2.000000	2.000000	...	3.000000	
50%	3.000000	7.000000	...	3.000000	
75%	7.000000	14.000000	...	3.000000	
max	18.000000	29.000000	...	4.000000	

	StockOptionLevel	RelationshipSatisfaction	EmployeeCount	\
count	1470.000000	1470.000000	1470.0	
mean	0.793878	2.712245	1.0	
std	0.852077	1.081209	0.0	
min	0.000000	1.000000	1.0	
25%	0.000000	2.000000	1.0	
50%	1.000000	3.000000	1.0	
75%	1.000000	4.000000	1.0	
max	3.000000	4.000000	1.0	

	StandardHours	HourlyRate	NumCompaniesWorked
YearsWithCurrManager	\		
count	1470.0	1470.000000	1470.000000
1470.000000			
mean	80.0	65.891156	2.693197
4.123129			
std	0.0	20.329428	2.498009
3.568136			
min	80.0	30.000000	0.000000
0.000000			
25%	80.0	48.000000	1.000000
2.000000			
50%	80.0	66.000000	2.000000
3.000000			
75%	80.0	83.750000	4.000000
7.000000			
max	80.0	100.000000	9.000000
17.000000			

YearsAtCompany

Age

count	1470.000000	1470.000000
mean	7.008163	36.923810
std	6.126525	9.135373
min	0.000000	18.000000
25%	3.000000	30.000000
50%	5.000000	36.000000
75%	9.000000	43.000000
max	40.000000	60.000000

[8 rows x 26 columns]

Gộp missing\_ratio\_df và num\_cols\_info\_df để quan sát đầy đủ các giá trị thống kê cần thiết.

```
num_cols_info_df = pd.concat([missing_ratio_df.transpose(),
num_cols_info_df], axis=0)
pd.set_option("display.max_columns", None)
display(num_cols_info_df)
pd.reset_option('display.max_columns')
```

	Education	PercentSalaryHike	YearsSinceLastPromotion
\			
missing_ratio	0.000000	0.000000	0.000000
count	1470.000000	1470.000000	1470.000000
mean	2.912925	15.209524	2.187755
std	1.024165	3.659938	3.222430
min	1.000000	11.000000	0.000000
25%	2.000000	12.000000	0.000000
50%	3.000000	14.000000	1.000000
75%	4.000000	18.000000	3.000000
max	5.000000	25.000000	15.000000

	DailyRate	JobInvolvement	EmployeeNumber
TotalWorkingYears			
\			
missing_ratio	0.000000	0.000000	0.000000
0.000000			
count	1470.000000	1470.000000	1470.000000
1470.000000			
mean	802.485714	2.729932	1024.865306
11.279592			
std	403.509100	0.711561	602.024335

7.780782			
min	102.000000	1.000000	1.000000
0.000000			
25%	465.000000	2.000000	491.250000
6.000000			
50%	802.000000	3.000000	1020.500000
10.000000			
75%	1157.000000	3.000000	1555.750000
15.000000			
max	1499.000000	4.000000	2068.000000
40.000000			

	WorkLifeBalance	YearsInCurrentRole	
DistanceFromHome \			
missing_ratio	0.000000	0.000000	0.000000
count	1470.000000	1470.000000	1470.000000
mean	2.761224	4.229252	9.192517
std	0.706476	3.623137	8.106864
min	1.000000	0.000000	1.000000
25%	2.000000	2.000000	2.000000
50%	3.000000	3.000000	7.000000
75%	3.000000	7.000000	14.000000
max	4.000000	18.000000	29.000000

	JobLevel	EnvironmentSatisfaction	
JobSatisfaction \			
missing_ratio	0.000000	0.000000	0.000000
count	1470.000000	1470.000000	1470.000000
mean	2.063946	2.721769	2.728571
std	1.106940	1.093082	1.102846
min	1.000000	1.000000	1.000000
25%	1.000000	2.000000	2.000000
50%	2.000000	3.000000	3.000000

75%	3.000000	4.000000	4.000000
max	5.000000	4.000000	4.000000

	MonthlyRate	MonthlyIncome	TrainingTimesLastYear \
missing_ratio	0.000000	0.000000	0.000000
count	1470.000000	1470.000000	1470.000000
mean	14313.103401	6502.931293	2.799320
std	7117.786044	4707.956783	1.289271
min	2094.000000	1009.000000	0.000000
25%	8047.000000	2911.000000	2.000000
50%	14235.500000	4919.000000	3.000000
75%	20461.500000	8379.000000	3.000000
max	26999.000000	19999.000000	6.000000

	PerformanceRating	StockOptionLevel
RelationshipSatisfaction \		
missing_ratio	0.000000	0.000000
0.000000		
count	1470.000000	1470.000000
1470.000000		
mean	3.153741	0.793878
2.712245		
std	0.360824	0.852077
1.081209		
min	3.000000	0.000000
1.000000		
25%	3.000000	0.000000
2.000000		
50%	3.000000	1.000000
3.000000		
75%	3.000000	1.000000
4.000000		
max	4.000000	3.000000
4.000000		

	EmployeeCount	StandardHours	HourlyRate
NumCompaniesWorked \			
missing_ratio	0.0	0.0	0.000000
0.000000			
count	1470.0	1470.0	1470.000000
1470.000000			
mean	1.0	80.0	65.891156
2.693197			
std	0.0	0.0	20.329428
2.498009			
min	1.0	80.0	30.000000
0.000000			

25%	1.0	80.0	48.000000
1.000000			
50%	1.0	80.0	66.000000
2.000000			
75%	1.0	80.0	83.750000
4.000000			
max	1.0	80.0	100.000000
9.000000			

	YearsWithCurrManager	YearsAtCompany	Age
missing_ratio	0.000000	0.000000	0.000000
count	1470.000000	1470.000000	1470.000000
mean	4.123129	7.008163	36.923810
std	3.568136	6.126525	9.135373
min	0.000000	0.000000	18.000000
25%	2.000000	3.000000	30.000000
50%	3.000000	5.000000	36.000000
75%	7.000000	9.000000	43.000000
max	17.000000	40.000000	60.000000

#### Nhận xét:

- Các cột dữ liệu dạng số hoàn toàn không có giá trị thiếu.
- Các giá trị min, lower quartile, median, upper quartile và max không cho thấy điều bất thường nên ta không cần thực hiện các bước tiền xử lý đối với các thuộc tính dạng số này.

### 8. Xem xét sự phân bố giá trị của các cột dữ liệu không phải dạng số

☞ Thực hiện thống kê trên các cột dữ liệu dạng số và xem xét các giá trị cụ thể như sau:

- Tỷ lệ % (từ 0 đến 100) các giá trị thiếu (missing\_ratio).
- Số lượng các giá trị khác nhau (không xét giá trị thiếu) (n\_values).
- Tỷ lệ % (từ 0 đến 100) của mỗi giá trị được sắp xếp giảm dần (không xét giá trị thiếu, tỷ lệ là tỷ lệ so với số lượng các giá trị không thiếu). Các tỷ lệ này được lưu vào một dictionary cho mỗi thuộc tính, key là giá trị, value là tỷ lệ % (value\_ratios).

☞ Tính tổng giá trị thiếu của từng cột bằng phương thức `isnull()` và `sum()`, rồi chia cho số dòng để được tỷ lệ giá trị thiếu `missing_ratio`.

```
missing_ratio = df[cat_cols].isnull().sum()
missing_ratio = missing_ratio / num_rows
missing_ratio_df = pd.DataFrame({'missing_ratio': missing_ratio})
missing_ratio_df
```

	missing_ratio
EducationField	0.0
Gender	0.0
Department	0.0
JobRole	0.0
Over18	0.0

```

BusinessTravel      0.0
MaritalStatus       0.0
Attrition           0.0
OverTime            0.0

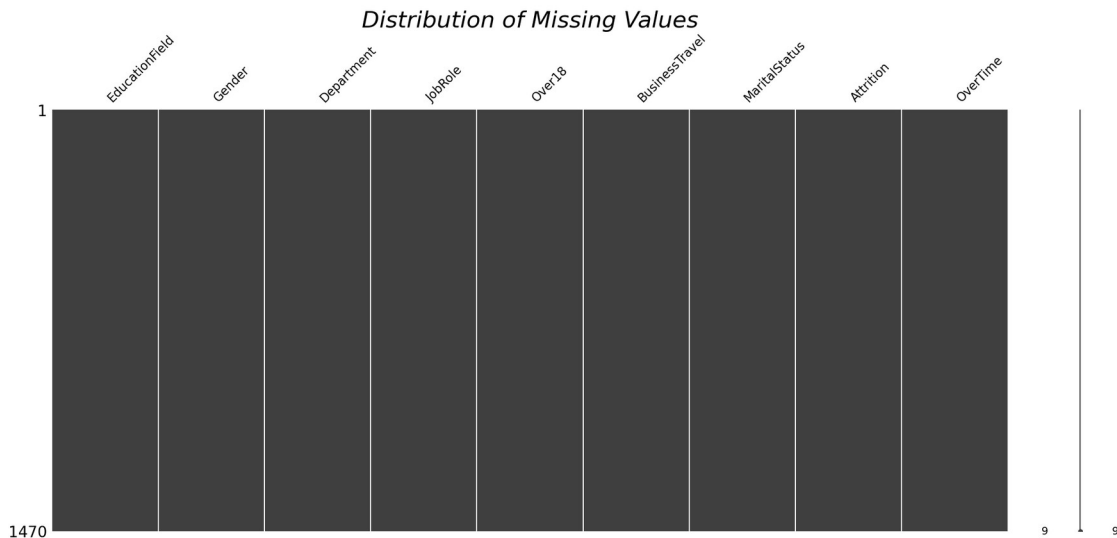
```

Trực quan sự phân bố các giá trị thiếu bằng thư viện missingno

```

msno.matrix(df[cat_cols])
plt.title('Distribution of Missing Values', fontsize=30, fontstyle=
'oblique');

```



👉 Tính số lượng các giá trị khác nhau bằng phương thức nunique().

```

n_values_df = pd.DataFrame({'n_values': df[cat_cols].nunique()})
n_values_df

```

```

          n_values
EducationField      6
Gender              2
Department          3
JobRole             9
Over18              1
BusinessTravel      3
MaritalStatus       3
Attrition           2
OverTime            2

```

👉 Tính tỉ lệ của mỗi giá trị bằng phương thức value\_counts().

```

value_ratios_dict = {}
for col in cat_cols:
    value_ratios_dict[col] = dict(df[col].value_counts(normalize=True)
* 100)
value_ratios_df = pd.DataFrame({'value_ratios': value_ratios_dict})

```

```
value_ratios_df = value_ratios_df.transpose()[cat_cols]
value_ratios_df
```

```
EducationField \
value_ratios    {'Life Sciences': 41.224489795918366, 'Medical...
```

```
Gender \
value_ratios    {'Male': 60.0, 'Female': 40.0}
```

```
Department \
value_ratios    {'Research & Development': 65.37414965986395, ...
```

```
JobRole
Over18 \
value_ratios    {'Sales Executive': 22.176870748299322, 'Resea... {'Y':
100.0}
```

```
BusinessTravel \
value_ratios    {'Travel_Rarely': 70.95238095238095, 'Travel_F...
```

```
MaritalStatus \
value_ratios    {'Married': 45.78231292517007, 'Single': 31.97...
```

```
Attrition \
value_ratios    {'No': 83.87755102040816, 'Yes': 16.1224489795...
```

```
OverTime
value_ratios    {'No': 71.70068027210884, 'Yes': 28.2993197278...
```

🔗 Gộp missing\_ratio\_df, n\_values\_df và value\_ratios\_df để quan sát đầy đủ các giá trị thống kê cần thiết.

```
cat_cols_info_df = pd.concat([missing_ratio_df.transpose(),
n_values_df.transpose(), value_ratios_df], axis=0)
pd.set_option('display.max_colwidth', None)
display(cat_cols_info_df)
pd.reset_option('display.max_colwidth')
```

```
EducationField \
missing_ratio
0.0
n_values
6
value_ratios    {'Life Sciences': 41.224489795918366, 'Medical':
31.564625850340132, 'Marketing': 10.816326530612246, 'Technical
Degree': 8.979591836734693, 'Other': 5.578231292517007, 'Human
Resources': 1.8367346938775513}
```

```
Gender \
```



```
missing_ratio      0.0
n_values           2
value_ratios      {'Male': 60.0, 'Female': 40.0}
```

```
Department \
missing_ratio
0.0
n_values
3
value_ratios  {'Research & Development': 65.37414965986395, 'Sales':
30.34013605442177, 'Human Resources': 4.285714285714286}
```

```
JobRole \
missing_ratio
0.0
n_values
9
value_ratios  {'Sales Executive': 22.176870748299322, 'Research
Scientist': 19.86394557823129, 'Laboratory Technician':
17.61904761904762, 'Manufacturing Director': 9.863945578231291,
'Healthcare Representative': 8.91156462585034, 'Manager':
6.938775510204081, 'Sales Representative': 5.646258503401361,
'Research Director': 5.442176870748299, 'Human Resources':
3.537414965986395}
```

```
Over18 \
missing_ratio      0.0
n_values           1
value_ratios      {'Y': 100.0}
```

```
BusinessTravel \
missing_ratio
0.0
n_values
3
value_ratios  {'Travel_Rarely': 70.95238095238095,
'Travel_Frequently': 18.843537414965986, 'Non-Travel':
10.204081632653061}
```

```
MaritalStatus \
missing_ratio
0.0
n_values
3
value_ratios  {'Married': 45.78231292517007, 'Single':
31.97278911564626, 'Divorced': 22.244897959183675}
```

```

missing_ratio      Attrition \
n_values           0.0
value_ratios      {'No': 83.87755102040816, 'Yes': 16.122448979591837}

missing_ratio      OverTime
n_values           0.0
value_ratios      {'No': 71.70068027210884, 'Yes': 28.29931972789116}


```

#### Nhận xét:

- Các cột dữ liệu không phải dạng số hoàn toàn không có giá trị thiếu.
- Thuộc tính 'Over18' chỉ có 1 giá trị duy nhất là 'Y', cho thấy toàn bộ nhân viên trong bộ dữ liệu đều trên 18 tuổi.

[!\[\]\(99f58673407353e96a019fbca558fd72\_img.jpg\)](#) Back to Table of Contents [!\[\]\(2113e5cba4d11862fa536c379e9b61cd\_img.jpg\)](#)

## C. Khám phá mối quan hệ trong dữ liệu

| Biểu đồ 1 

**Tiêu đề:** Xem xét tỉ lệ nhân viên rời khỏi và không rời khỏi tổ chức.

**Loại biểu đồ:** Doughnut chart (biểu đồ bánh rán).

**Lý do lựa chọn:** Doughnut chart là lựa chọn tốt để trình bày dữ liệu phân phối theo tỷ lệ phần trăm. Do đó dùng Doughnut chart sẽ giúp so sánh nhanh giữa các nhóm có trong thuộc tính 'Attrition' (rời khỏi tổ chức/không rời khỏi tổ chức), để xem chúng chiếm bao nhiêu phần trăm trong tổng số nhân viên của tổ chức.

Điều quan trọng cần lưu ý khi sử dụng biểu đồ bánh là nó chỉ hiển thị được một chiều (phần trăm) của tập dữ liệu, không so sánh được giữa các giá trị cụ thể. Nếu muốn so sánh giá trị giữa các nhóm, thì biểu đồ cột hoặc biểu đồ đường có thể là sự lựa chọn tốt hơn.

**Thực quan hóa:**

*#Tạo dữ liệu*

```

plot_df = df.copy()
plot_df= plot_df['Attrition'].value_counts()

```

*#Thực quan hóa*

```

fig = make_subplots(rows=1, cols=2, specs=[[{"type": "pie"}, {"type":
"pie"}]], subplot_titles=(' ', ' '))
fig.add_trace(go.Pie(values=plot_df.values, labels=plot_df.index, hole=0
.3), row=1, col=1)

```

```

fig.update_traces(hoverinfo='label',
                  textfont_size = 18,
                  textposition = 'auto',
                  marker=dict(colors = ["#335C42", "#7E7F92"],
                              line = dict(color = 'white',
                                          width = 2)))

fig.add_layout_image(
    dict(

source="https://raw.githubusercontent.com/ntclai/PictureForMyProject/
main/Voluntary-Resignation--1-.png",
        xref="paper",
        yref="paper",
        x=1.2, y=0.25,
        sizex=0.7, sizey=1,
        xanchor="right", yanchor="bottom", sizing= "contain",
    )
)

fig.update_layout(title ={'text' : "<b>Attrition rate of the
organization's forces</b>",
                    'x' : 0.21},
                  template = 'xgridoff',
                  width = 900, height = 600,
                  legend=dict(title_font_family="Times New Roman",

font=dict(family="Courier",size=25,color="black" ),
                  bgcolor="#E4F5CA",
                  bordercolor="Black",
                  borderwidth=2.5)
)

iplot(fig)

```

Attrition rate of the organization's forces



#### Nhận xét

- Tỉ lệ nhân viên thôi việc của tổ chức này là 16.1%. Và theo các chuyên gia về lĩnh vực nhân sự cho rằng, tỷ lệ tiêu hao nhân lực của mỗi doanh nghiệp từ 4% đến 6% là mức ổn định.

=> Tỉ lệ thôi việc của tổ chức này đang ở mức nguy hiểm. Do đó tổ chức nên có các biện pháp để làm giảm tỉ lệ này.

| Biểu đồ 2

**Tiêu đề:** Tỉ lệ nam nữ trong tổ chức.

**Loại biểu đồ:** Pie chart (biểu đồ tròn).

**Lý do lựa chọn:**

- Pie chart là lựa chọn hợp lý vì nó cấp cho chúng ta cái nhìn tổng thể về tỷ lệ của các phần khác nhau tạo thành một tập dữ liệu.
- Tuy nhiên, khi có quá nhiều phần tử cần hiển thị, trực quan hóa với Pie Chart có thể trở nên khó hiểu và làm giảm tính rõ ràng của dữ liệu. Ở đây thuộc tính Gender chỉ có 2 giá trị là Male và Female nên pie chart sẽ thể hiện được hết khả năng trực quan hóa của nó.

**Trực quan hóa:**

*#Tạo dữ liệu*

```
plot_df = df.copy()  
plot_df= plot_df['Gender'].value_counts()
```

*#Trực quan hóa*

```

fig = make_subplots(rows=1, cols=2, specs=[[{"type": "pie"}, {"type":
"pie"}]], subplot_titles=('', ''))
fig.add_trace(go.Pie(values=plot_df.values, labels=plot_df.index), row=1
,col=2)

fig.update_traces(hoverinfo='label',
                    textfont_size = 18,
                    textposition = 'auto',
                    marker=dict(colors = ["#AC1F29", "#7E7F92"],
                                line = dict(color = 'white',
                                            width = 2)))

fig.add_layout_image(
    dict(

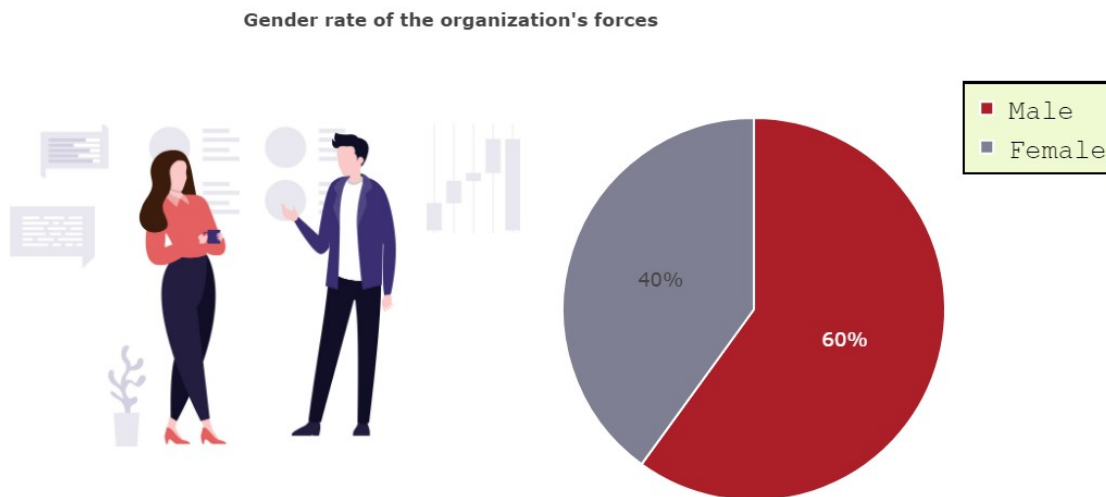
source="https://raw.githubusercontent.com/ntclai/PictureForMyProject/
main/viz.jpg",
        xref="paper",
        yref="paper",
        x=0.5, y=0.20,
        sizex=0.6, sizey=1,
        xanchor="right", yanchor="bottom", sizing= "contain",
    )
)

fig.update_layout(title ={'text' : "<b>Gender rate of the
organization's forces</b>",
                        'x' : 0.21},
                  template = 'xgridoff',
                  width = 900, height = 600,
                  legend=dict(title_font_family="Times New Roman",

font=dict(family="Courier", size=25, color="black" ),
                  bgcolor="#EFFAD3",
                  bordercolor="Black",
                  borderwidth=2.5)
)


iplot(fig)

```



#### Nhận xét

Số lượng nhân viên nam trong công ty chiếm tỉ lệ nhiều hơn nhân viên nữ (hơn 20%).

| Biểu đồ 3 

**Tiêu đề:** Tỉ lệ các cấp bậc trong công việc

**Loại biểu đồ:** TreeMap

**Lý do lựa chọn:** Để so sánh tỉ lệ sự phân bố của các cấp độ trong công việc có trong thuộc tính 'Job Level', thì biểu đồ TreeMap là một lựa chọn phù hợp. Vì TreeMap là một loại biểu đồ thống kê hiển thị dữ liệu dưới dạng các hình chữ nhật có diện tích khác nhau, trong đó độ lớn của hình chữ nhật sẽ đại diện cho giá trị lượng của một biến. Ta sẽ biểu diễn số lượng các giá trị có trong thuộc tính 'JobLevel' bằng diện tích các hình chữ nhật.

**Thực quan hóa:**

```
plot_df = df.copy()
plot_df['JobLevel'] = pd.Categorical(
    plot_df['JobLevel']).rename_categories(
    ['Entry level', 'Mid level', 'Senior', 'Lead', 'Executive'])

plot_df = plot_df['JobLevel'].value_counts()

fig = px.treemap(plot_df,
    path=[plot_df.index],
    values=plot_df.values,
    title = 'Distribution of Job Level',
    color=plot_df.index,
    color_discrete_sequence=px.colors.sequential.PuBuGn,
```

```

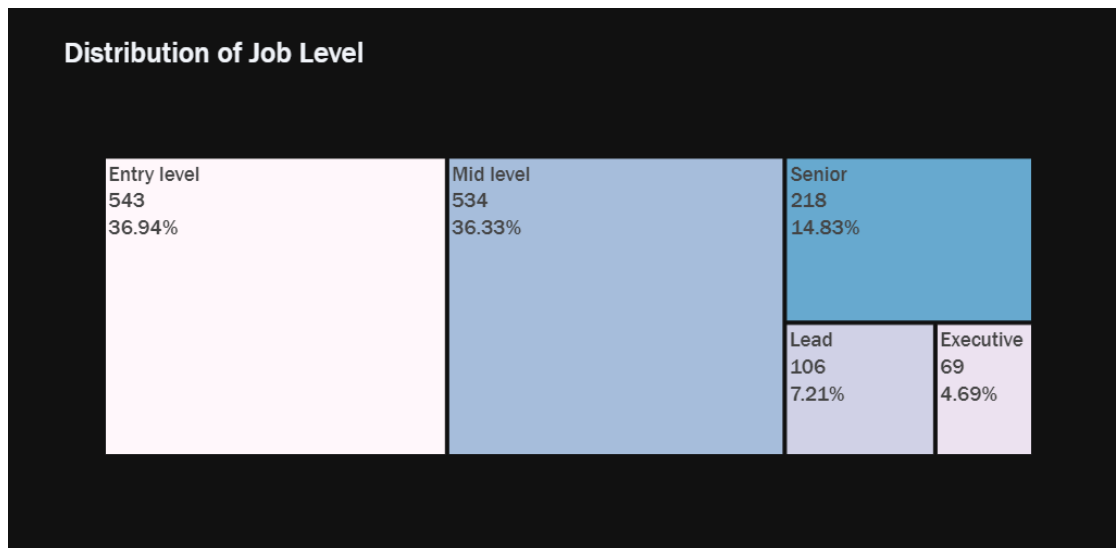
        template='plotly_dark',
        width=1000, height=500)

percents = np.round((100*plot_df.values /
sum(plot_df.values)).tolist(),2)

fig.data[0].customdata = [36.94, 4.69, 7.21, 36.33, 14.83]
fig.data[0].texttemplate = '%{label}<br>{%value}<br>{%customdata}%'

fig.update_layout(
    font=dict(size=19, family="Franklin Gothic"))
fig.show()

```




### Nhận xét

Từ biểu đồ TreeMap, ta thấy:

- 'Entry level' (vị trí công việc của những người có ít kinh nghiệm làm việc) chiếm số lượng lớn nhất với 543 người (chiếm 36.94%), tương ứng với hình chữ nhật có diện tích lớn nhất. Xếp ngay sau đó là 'Mid level' với 534 người (chiếm 36.33%).
- Chiếm tỉ lệ ít nhất là 'Executive' (vị trí cấp cao, có vai trò rất quan trọng trong mỗi phòng ban) với chỉ 69 người (chiếm 4.69%).

=> Cấp bậc trong công việc càng cao thì số lượng nhân sự càng ít.

| Biểu đồ 4 

**Tiêu đề:** Tỉ lệ các cấp độ trong công việc

**Loại biểu đồ:** BarPlot

**Lý do lựa chọn:** Để so sánh tỉ lệ bỏ việc của những người ở các cấp độ công việc được lưu vào hai thuộc tính là 'Attrition' và 'JobLevel', thì biểu đồ BarPlot là một lựa chọn phù

hợp. Vì BarPlot là một loại biểu đồ thống kê hiển thị dữ liệu dưới dạng các cột có chiều cao khác nhau, trong đó chiều cao của cột sẽ đại diện cho giá trị lượng của một biến.

### Thực quan hóa:

*#Tạo dữ liệu*

```
plot_df = df.copy()
plot_df['JobLevel'] = pd.Categorical(
    plot_df['JobLevel']).rename_categories(
    ['Entry level', 'Mid level', 'Senior', 'Lead', 'Executive'])
```

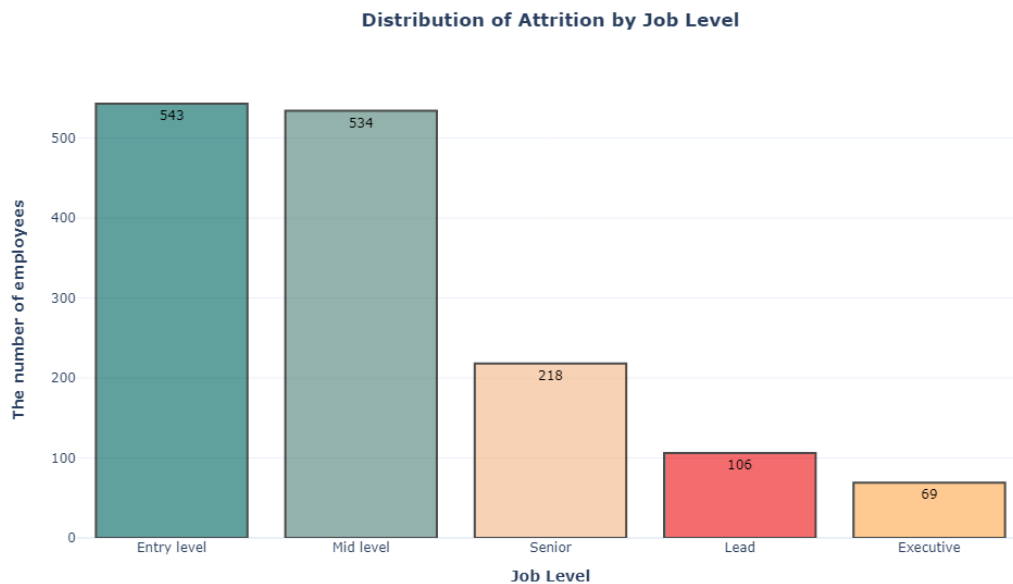
```
plot_df= plot_df['JobLevel'].value_counts()
```

*#Thực quan hóa*

```
colors = ["#1d7874", "#679289", "#f4c095", "#ee2e31", "#ffb563", "#918450"]
data = go.Bar(x=plot_df.index, y = plot_df.values, text =
    plot_df.values , textposition = 'inside',
    textfont = dict(size = 12,
        color = 'black'),
    marker = dict(color = colors,
        opacity = 0.7,
        line_color = 'black',
        line_width = 2))
layout = go.Layout(title = {'text': "<b>Distribution of Attrition by
Job Level</b>",
    'x':0.5,
    'xanchor': 'center'},
    xaxis = dict(title='<b>Job Level</b>'),
    yaxis =dict(title='<b>The number of
employees</b>'),
    width = 900,
    height = 600,
    template = 'plotly_white')
fig=go.Figure(data = data, layout = layout)

iplot(fig)
```






### Nhận xét

Dựa vào biểu đồ cột ở trên ta có thể thấy:

- Nhân viên ở Level 1 (Entry level) có tỷ lệ rời khỏi công ty rất cao (60%). Họ thường là những người rất trẻ, mới vào nghề.
- Nhân viên ở Level 2 (Middle Level) có tỷ lệ rời khỏi tương đối cao (21%)
- Những nhân viên đạt được Level 4 (Lead) và 5 (Executive) có tỷ lệ rời khỏi rất ít.

=> Những nhân viên *trẻ* mới vào công ty có khả năng "*nhảy việc*" rất cao.

| Biểu đồ 5 

**Tiêu đề:** Phân bố mức độ làm việc tăng ca trong công việc ở các vai trò công việc.

**Loại biểu đồ:** Stacked bar chart.

**Lý do lựa chọn:** So sánh số lượng nhân viên làm việc tăng ca ở các vai trò công việc khác nhau, biểu đồ cột chồng là phù hợp vì thể hiện được tổng số lượng và cả số lượng của mỗi thành phần trong nó.

**Trực quan hóa:**

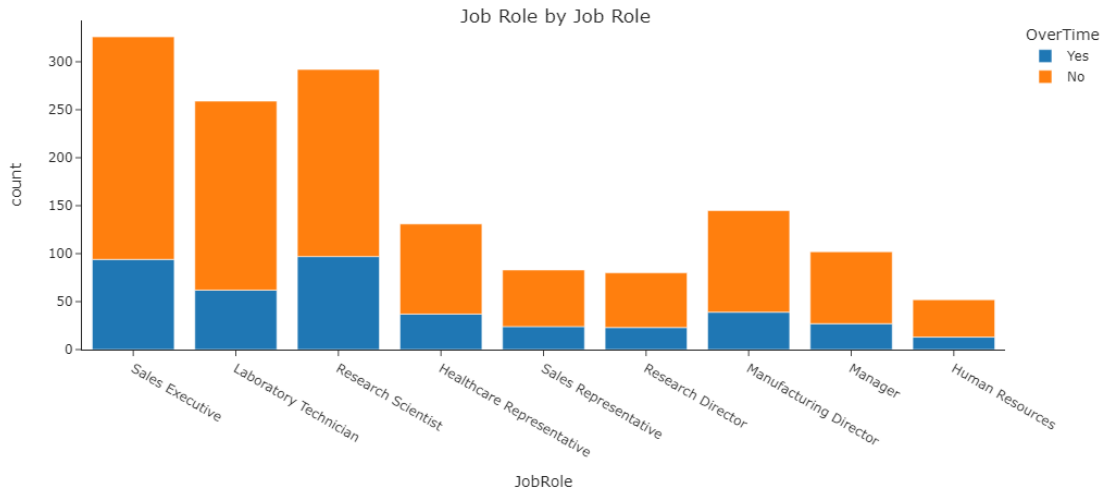
```
fig = px.histogram(data_frame = df, x = 'JobRole', color = 'OverTime',
                   width=1000, height=500, template="simple_white")

fig.update_layout(
    title={
        'text': "Job Role by Job Role",
        'y':0.9,
        'x':0.5,
```

```


        'xanchor': 'center',
        'yanchor': 'top'})
fig.show()

```



#### Nhận xét

- 3 vị trí có nhiều nhân sự nhất là: Sale Executive, Research Scientist và Laboratory Technician.
- Tỷ lệ làm việc OverTime gấp gần 2 lần nhóm không làm việc quá giờ. Xu hướng làm việc OT đang ngày càng phổ biến.

| Biểu đồ 6 

**Tiêu đề:** Mối quan hệ giữa tỉ lệ rời đi và sự hài lòng về môi trường làm việc.

**Loại biểu đồ:** Bar chart + line plot.

**Lý do lựa chọn:** Dùng line plot đếm số lượng nhân viên theo từng nhóm đánh giá hài lòng về môi trường làm việc ở 4 mức 1, 2, 3, 4. Sau đó tính tỉ lệ rời đi trong nhóm đó và vẽ các tỉ lệ bằng bar chart.

**Thực quan hóa:**

*#Tạo dữ liệu*

```

new_df=df.groupby('EnvironmentSatisfaction').count()
idx=list(new_df.index) #get groups as group 1, group 2, group 3, group 4
count=list(new_df['Age'].values)

```

```

percent_dropout=[] #list contains employees dropout percent
for i in range(len(idx)):

```

```

    percent_dropout.append(round((len(df[df['EnvironmentSatisfaction']==idx[i]][df['Attrition']=='Yes'])/count[i])*100,3))

```

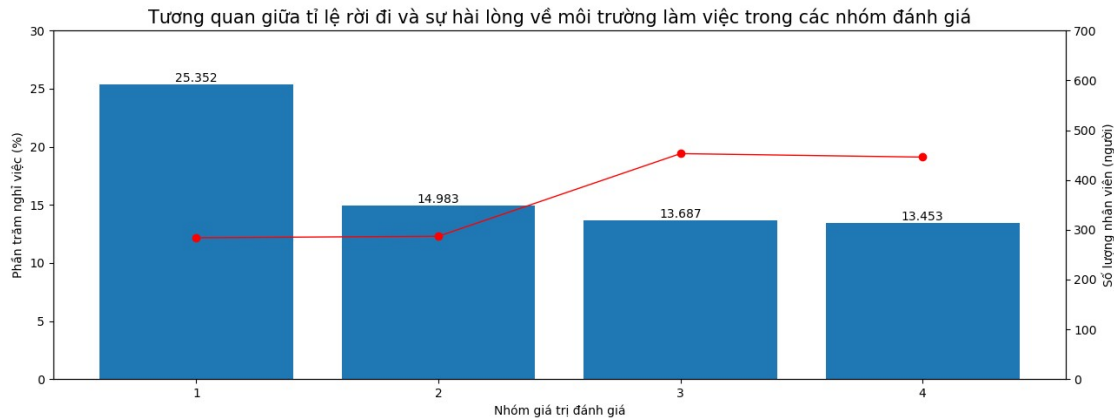
```
C:\Users\Administrator\AppData\Local\Temp\
ipykernel_13416\4025534587.py:8: UserWarning:
```

Boolean Series key will be reindexed to match DataFrame index.

```
envSat_attrition_df={idx[i]:[count[i],percent_dropout[i]] for i in
range(len(idx))}
envSat_attrition_df['Thông kê']=['Số lượng nhân viên','Phần trăm nghỉ
việc']
envSat_attrition_df=pd.DataFrame(envSat_attrition_df).set_index('Thông
kê')
envSat_attrition_df.head()
```

	1	2	3	4
Thông kê				
Số lượng nhân viên	284.000	287.000	453.000	446.000
Phần trăm nghỉ việc	25.352	14.983	13.687	13.453

```
groups=['1','2','3','4']
fig,ax1=plt.subplots(figsize=(13,5))# bar chart
bars=ax1.bar(groups,envSat_attrition_df.loc[['Phần trăm nghỉ
việc']].values[0])
for rect in bars:
    height = rect.get_height()
    plt.text(rect.get_x() + rect.get_width() / 2.0, height,
f'{height:.3f}', ha='center', va='bottom')
ax2=ax1.twinx()
ax2.plot(groups,envSat_attrition_df.loc[['Số lượng nhân
viên']].values[0],lw=1,marker='o',color='r')
ax1.set_xlabel='Nhóm giá trị đánh giá', ylabel='Phần trăm nghỉ việc
(%)'
ax2.set_ylabel='Số lượng nhân viên (người)'
ax1.set_ylim(0,30)
ax2.set_ylim(0,700)
ax1.set_title(f'Tương quan giữa tỉ lệ rời đi và sự hài lòng về môi
trường làm việc trong các nhóm đánh giá', size=15)
fig.tight_layout()
plt.show();
```



#### Nhận xét

- Số lượng nhân viên hài lòng về môi trường làm việc cũng rất cao tương tự hài lòng về công việc.
- Tỉ lệ rời đi trung bình ở các nhóm đánh giá thấp về môi trường làm việc cao hơn so với các nhóm đánh giá cao. Điều này cũng rất hợp lý với thực tế.
- Từ đây cho thấy bên cạnh tiền lương thì yếu tố môi trường làm việc cũng là yếu tố quan trọng quyết định lớn đến việc nhân viên có thật sự gắn bó lâu dài hay không.

| Biểu đồ 7

**Tiêu đề:** Mức lương trung bình thay đổi như thế nào theo số năm làm việc tại công ty?

**Loại biểu đồ:** Line chart

**Lý do lựa chọn:** Biểu đồ line chart là một cách hiệu quả để thể hiện các xu hướng tăng giảm hoặc sự biến động của dữ liệu qua thời gian. Do đó ta dùng line chart để xem xét sự thay đổi mức lương trung bình của nhân viên trong công ty theo số năm làm việc.

**Thực quan hóa:**

*#Tạo dữ liệu cần thiết*

```
temp=df.copy()
conditions=[(temp['YearsAtCompany']>=0) & (temp['YearsAtCompany']<=5),
             (temp['YearsAtCompany']>5) & (temp['YearsAtCompany']<=10),
             (temp['YearsAtCompany']>10) &
             (temp['YearsAtCompany']<=15),
             (temp['YearsAtCompany']>15) &
             (temp['YearsAtCompany']<=20),
             (temp['YearsAtCompany']>20) &
             (temp['YearsAtCompany']<=25),
             (temp['YearsAtCompany']>25) &
             (temp['YearsAtCompany']<=30),
             (temp['YearsAtCompany']>30) &
             (temp['YearsAtCompany']<=35),
             (temp['YearsAtCompany']>35) &
             (temp['YearsAtCompany']<=40)]
values=['0-5', '05-10', '10-15', '15-20', '20-25', '25-30', '30-35', '35-40']
```

```
temp['YearsRange']=np.select(conditions, values)
```

```
plot_df=temp.groupby(['YearsRange'])  
['MonthlyIncome'].mean().reset_index()
```

*#Trực quan hóa*

```
sns.set(rc={"axes.facecolor": "#ECF4F5", "figure.facecolor": "#ECF4F5"})  
sns.set_context("poster", font_scale = .7)  
plt.subplots(figsize=(20,8))  
p=sns.lineplot(x=plot_df["YearsRange"], y=plot_df["MonthlyIncome"], data=plot_df, color="#0E2269", marker="o", linewidth=6, markersize=18, markerfacecolor="orange", markeredgewidth=3)  
p.axes.set_title("\nRelationship between monthly salary and number of years at the company\n", fontsize=25, color="#307149", family='Verdana')  
p.axes.set_xlabel("\nYears at company", fontsize=20, color="#307149", family='Verdana')  
p.axes.set_ylabel("Monthly income", fontsize=20, color="#307149", family='Verdana')  
sns.despine(left=True, bottom=True)  
plt.show()
```



#### Nhận xét

- Trong khoảng thời gian gắn bó với công ty từ 0-35 năm, mức lương trung bình của nhân viên tăng dần theo thời gian gắn bó. Điều này chứng tỏ khi thâm niên làm việc lâu thì nhân viên đó sẽ có kinh nghiệm làm việc càng nhiều và là một nhân viên trung thành với công ty. Nên có được mức lương hậu hĩnh là điều xứng đáng.
- Tuy nhiên, mức lương cho khoảng thời gian làm việc từ 35-40 năm lại sụt giảm nhưng không quá lớn. Nguyên nhân sự sụt giảm này có thể là khi đã gắn bó với công ty từ 35-40 năm, thì các nhân viên này ở độ tuổi sắp nghỉ hưu nên năng suất làm việc giảm hoặc họ đã nhường lại các vị trí cấp cao cho lớp trẻ lãnh đạo,...

**Tiêu đề:** Mức độ đi công tác ở mỗi phòng ban

**Loại biểu đồ:** Radar chart

**Lý do lựa chọn:** Radar Chart là một biểu đồ đa diện được sử dụng để biểu thị dữ liệu đa chiều với các đặc trưng cho trước được đặt ở các trục khác nhau. Khi áp dụng cho thuộc tính 'Department' và 'BusinessTravel' ta sẽ có được tỉ lệ mức độ đi công tác của từng phòng ban, trong đó:

- Mức độ đi công tác (Non-Travel, Travel\_Frequently, Travel\_Rarely): sẽ nằm ở các trục có chia tỷ lệ.
- Màu sắc hình đa diện đại diện cho tên phòng ban (Human\_Resource, Research\_Development, Sales).

**Thực quan hóa:**

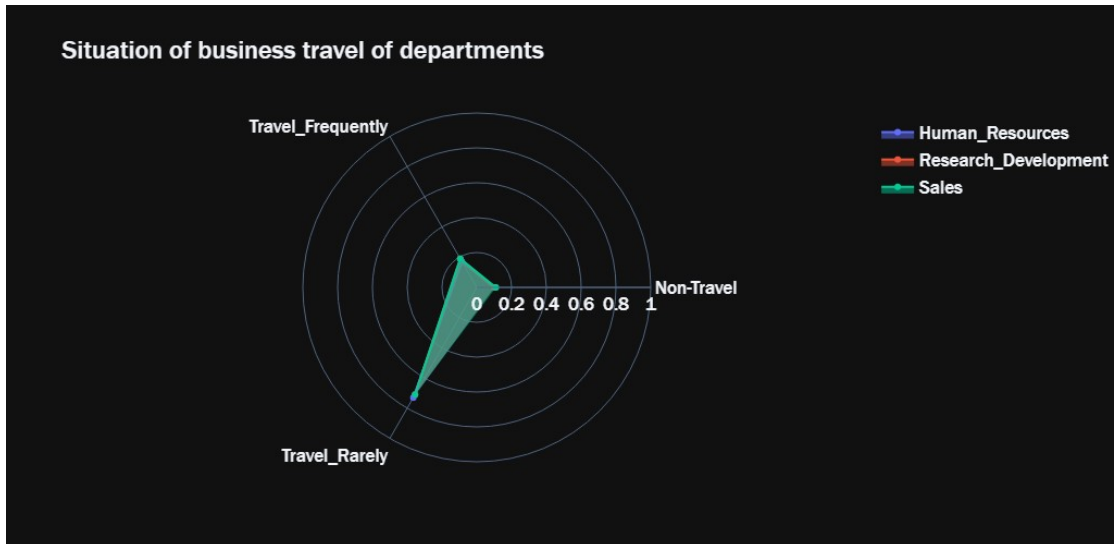
```
plot_df = df.groupby(['Department', 'BusinessTravel']).size()
Human_Resources=np.round(plot_df['Human
Resources'].values/(plot_df['Human Resources'].values.sum()),2)
Research_Development=np.round(plot_df['Research &
Development'].values/(plot_df['Research &
Development'].values.sum()),2)
Sales=np.round(plot_df['Sales'].values/(plot_df['Sales'].values.sum()),2)

categories = ['Non-Travel', 'Travel_Frequently', 'Travel_Rarely']
fig = go.Figure()
fig.add_trace(go.Scatterpolar(
    r = Human_Resources,
    theta = categories,
    fill = 'toself',
    name = 'Human_Resources'
))
fig.add_trace(go.Scatterpolar(
    r = Research_Development,
    theta = categories,
    fill = 'toself',
    name = 'Research_Development'
))
fig.add_trace(go.Scatterpolar(
    r = Sales,
    theta = categories,
    fill = 'toself',
    name = 'Sales'
))
fig.update_layout(
    polar=dict(
        radialaxis=dict(
```

```

        range=[0, 1]
    )),
    font = dict(family="Franklin Gothic", size=17),
    showlegend=True,
    title = 'Situation of business travel of departments',
    height = 500,
    width = 1000
)
fig.layout.template = 'plotly_dark'
fig.show()

```



#### 💬 Nhận xét

Dựa vào biểu đồ Radar chart bên trên, ta có thể thấy:

- Các hình tam giác gần như xếp chồng lên nhau, cho thấy mức độ đi công tác ở các phòng ban khá tương đồng.
- Tỷ lệ 'Travel\_Rarely' (hiếm khi đi công tác) là lớn nhất và 'Non-Travel' (không bao giờ đi công tác) là thấp nhất ở tất cả các phòng ban. Tỷ lệ này khá phổ biến ở các công ty, do thông thường việc đi công tác hay thuộc về các vị trí có cấp bậc cao trong công việc, và cấp bậc càng cao thì số lượng nhân sự càng ít (được minh chứng ở biểu đồ 1).

| Biểu đồ 9 📊

**Tiêu đề:** Tỷ lệ nhân viên theo độ tuổi trong công ty.

**Loại biểu đồ:** TreeMap

**Lý do lựa chọn:** Để so sánh tỷ lệ sự phân bố của độ tuổi có trong thuộc tính 'Age\_Range' được biến đổi từ thuộc tính 'Age', thì biểu đồ TreeMap là một lựa chọn phù hợp. Vì TreeMap là một loại biểu đồ thống kê hiển thị dữ liệu dưới dạng các hình chữ nhật có diện tích khác nhau, trong đó độ lớn của hình chữ nhật sẽ đại diện cho giá trị lượng của

một biến. Ta sẽ biểu diễn số lượng các giá trị có trong thuộc tính 'Age\_Range' bằng diện tích các hình chữ nhật.

### Xử lý dữ liệu:

```
new_df = df.loc[:, ['Age',  
'Attrition', 'BusinessTravel', 'Department', 'EducationField', 'EnvironmentSatisfaction',
```

```
'Gender', 'JobInvolvement', 'JobLevel', 'JobRole', 'JobSatisfaction', 'MaritalStatus',
```

```
'MonthlyIncome', 'Over18', 'OverTime', 'StockOptionLevel', 'TotalWorkingYears', 'YearsAtCompany',  
        'YearsInCurrentRole', 'YearsWithCurrManager']]
```

```
new_df['Attrition'] = new_df['Attrition'].replace('Yes',  
1).replace('No', 0)
```

```
dodi = new_df.copy()
```

```
conditions = [(dodi['Age'] >= 18) & (dodi['Age'] <= 20),  
              (dodi['Age'] > 20) & (dodi['Age'] <= 25),  
              (dodi['Age'] > 25) & (dodi['Age'] <= 30),  
              (dodi['Age'] > 30) & (dodi['Age'] <= 35),  
              (dodi['Age'] > 35) & (dodi['Age'] <= 40),  
              (dodi['Age'] > 40) & (dodi['Age'] <= 45),  
              (dodi['Age'] > 45) & (dodi['Age'] <= 50),  
              (dodi['Age'] > 50) & (dodi['Age'] <= 55),  
              (dodi['Age'] > 55) & (dodi['Age'] <= 60)  
            ]
```

```
values = ['18-20', '21-25', '26-30', '31-35', '36-40', '41-45', '46-50', '51-55', '56-60']
```

```
new_df['Age_Range'] = np.select(conditions, values)
```

```
percent_age_range = new_df['Age_Range'].value_counts().values /  
new_df.shape[0] * 100
```

```
for i in range(len(percent_age_range)):  
    percent_age_range[i] = '{:.2f}'.format(percent_age_range[i])
```

```
plot_df = new_df['Age_Range'].value_counts()  
plot_df = pd.DataFrame({'Age_Range': plot_df.index, 'Count':  
plot_df.values})  
plot_df['Percent'] = percent_age_range
```

### Trực quan hóa:

```
fig = px.treemap(plot_df,  
                 path=['Age_Range'],  
                 values='Count',  
                 title = 'Distribution of Job Level',  
                 color=plot_df.index,
```



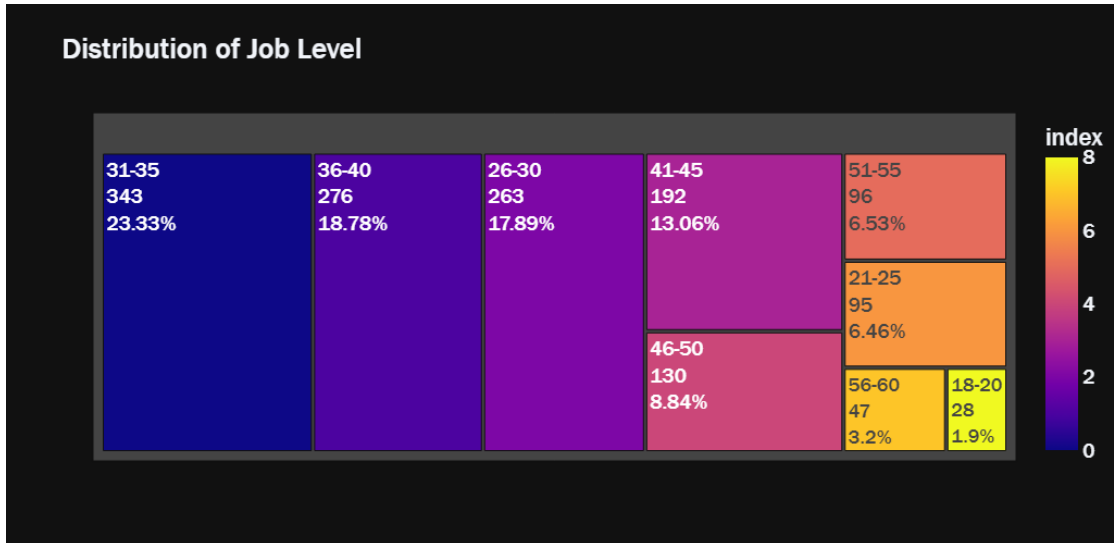
```

color_discrete_sequence=px.colors.sequential.PuBuGn,
template='plotly_dark', hover_data=['Percent'],
width=1000, height=500)

fig.data[0].texttemplate = '%{label}<br>{%value}<br>{%customdata[0]}%'

fig.update_layout(
    font=dict(size=19, family="Franklin Gothic"))
fig.show()

```



#### 💬 Nhận xét

- Số nhân viên có độ tuổi trong khoảng 31-35 chiếm tỷ lệ cao nhất (23%)
- Nhân viên trong khoảng độ tuổi 26-30 và 36-40 chiếm tỷ lệ cũng khá cao (18%)
- Số nhân viên trong độ tuổi 18-20 chiếm tỷ lệ thấp nhất (2%).
- Nhân viên trong khoảng độ tuổi 56-60 cũng khá thấp (3%)

=> Đây là phân bố hợp lý vì các độ tuổi chiếm tỷ lệ cao là những độ tuổi có năng suất làm việc hiệu quả nhất.

| Biểu đồ 10 📊

**Tiêu đề:** Sự khác biệt về mức lương hằng tháng giữa các cấp bậc trong công việc

**Loại biểu đồ:** Violin plot

**Lý do lựa chọn:** Biểu đồ violin plot thường được sử dụng với dữ liệu liên tục hoặc dữ liệu số. Nó rất hữu ích để hiển thị phân phối của một biến dữ liệu và so sánh phân phối đó giữa nhiều nhóm. Do đó để xem xét sự phân bố và so sánh phân phối dữ liệu theo mức lương giữa các cấp bậc có trong thuộc tính "JobLevel" thì Violin plot là lựa chọn lý tưởng.

**Trực quan hóa:**

```

palette = ["#11264e", "#6faea4", "#FEE08B", "#D4A1E7", "#E7A1A1"]
sns.set(rc={"axes.facecolor": "#F4F2F0", "figure.facecolor": "#F4F2F0"})
sns.set_context("poster", font_scale = .7)

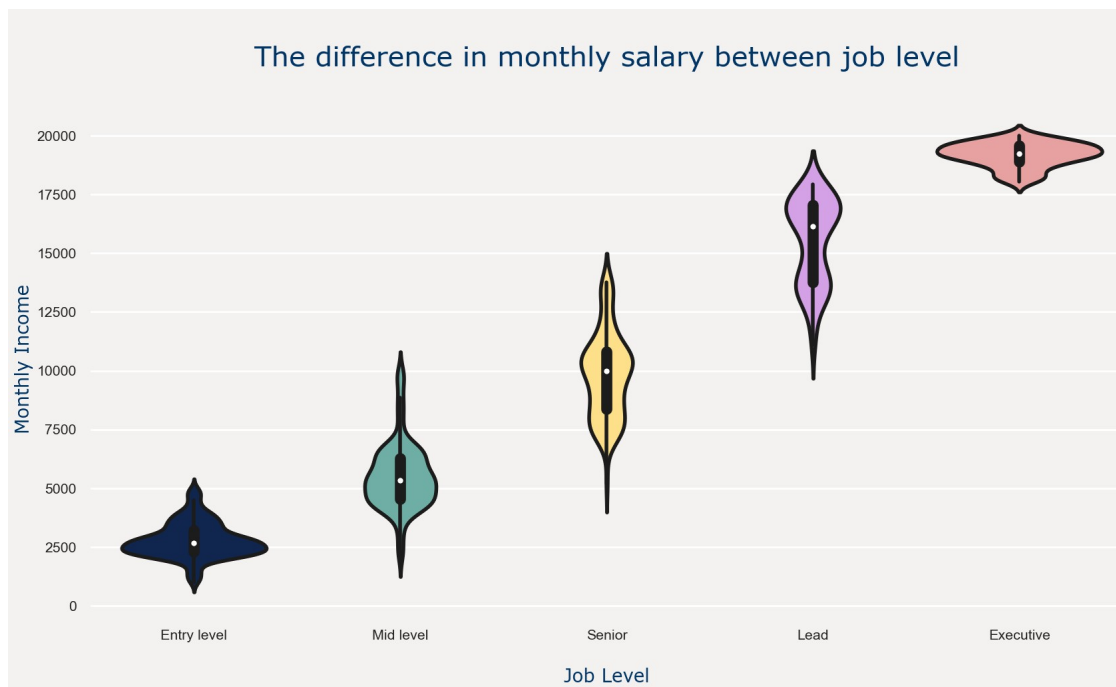
plot_df = df.copy()
plot_df['JobLevel'] = pd.Categorical(
    plot_df['JobLevel']).rename_categories(
    ['Entry level', 'Mid level', 'Senior', 'Lead', 'Executive'])

plt.subplots(figsize=(20, 10))

p=sns.violinplot(x=plot_df["JobLevel"],y=plot_df["MonthlyIncome"],order=
r=plot_df["JobLevel"].value_counts().index,palette=palette,saturation=
1,linewidth=4,edgecolor="black")
p.axes.set_title("\nThe difference in monthly salary between job
level\n", fontsize=30, family='Verdana', color='#003566')
p.axes.set_xlabel("\nJob Level", fontsize=20, family='Verdana', color=
'#003566')
p.axes.set_ylabel("Monthly Income", fontsize=20, family='Verdana',
color='#003566')

sns.despine(left=True, bottom=True)

```




### Nhận xét

Dựa vào biểu đồ Violin plot bên trên, ta có nhận xét:

- Một điều hiển nhiên dễ thấy là nhân viên có cấp bậc trong công việc càng cao thì thu nhập hàng tháng càng nhiều.

- Cấp bậc Mid level, Senior, Lead có phạm vi phân bố mức lương rộng, có thể xem là có nhiều giá trị ngoại lai (outlier).

| Biểu đồ 11 

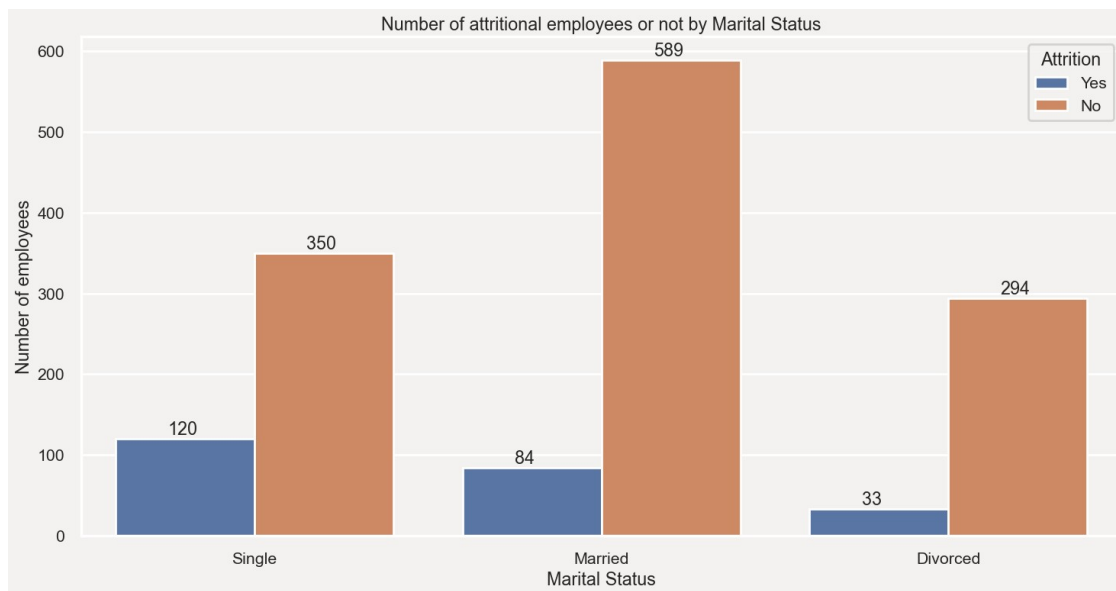
**Tiêu đề:** Tình trạng hôn nhân có liên quan gì với quyết định thôi việc hay không?

**Loại biểu đồ:** Countplot

**Lý do lựa chọn:**

**Trực quan hóa:**

```
plt.rcParams["figure.figsize"] = [15, 8]
plt.rcParams["figure.autolayout"] = True
g = sns.countplot(x='MaritalStatus', hue='Attrition', data=df)
g.set(title='Number of attritional employees or not by Marital Status')
g.set_xlabel('Marital Status')
g.set_ylabel('Number of employees')
g.set_yticks(range(0, 700, 100))
for p in g.patches:
    g.annotate('{}' .format(p.get_height()), (p.get_x()+0.15,
p.get_height()+5))
plt.show()
```




 **Nhận xét**

Dựa vào biểu đồ cột ghép ta có nhận xét:

- Tỷ lệ nhân viên đã kết hôn chiếm số lượng lớn nhất, điều này hoàn toàn phù hợp khi độ tuổi nhân viên công ty này từ 26-40 chiếm khoảng 60% (biểu đồ 9), đây là độ tuổi phù hợp để lập gia đình.

- Tỷ lệ nhân viên độc thân tuy đứng thứ 2 nhưng lại có tỷ lệ nghỉ việc cao nhất. Điều này có thể thấy những người trẻ tuổi và tự do (không vướng bận gia đình) có xu hướng "nhảy việc" để trải nghiệm nhiều môi trường làm việc khác nhau và tìm được công việc phù hợp với sở thích.

| Biểu đồ 12 

**Tiêu đề:** Thu nhập hàng tháng theo tổng số năm làm việc và cấp bậc trong công việc

**Loại biểu đồ:** Scatter plot

**Lý do lựa chọn:**

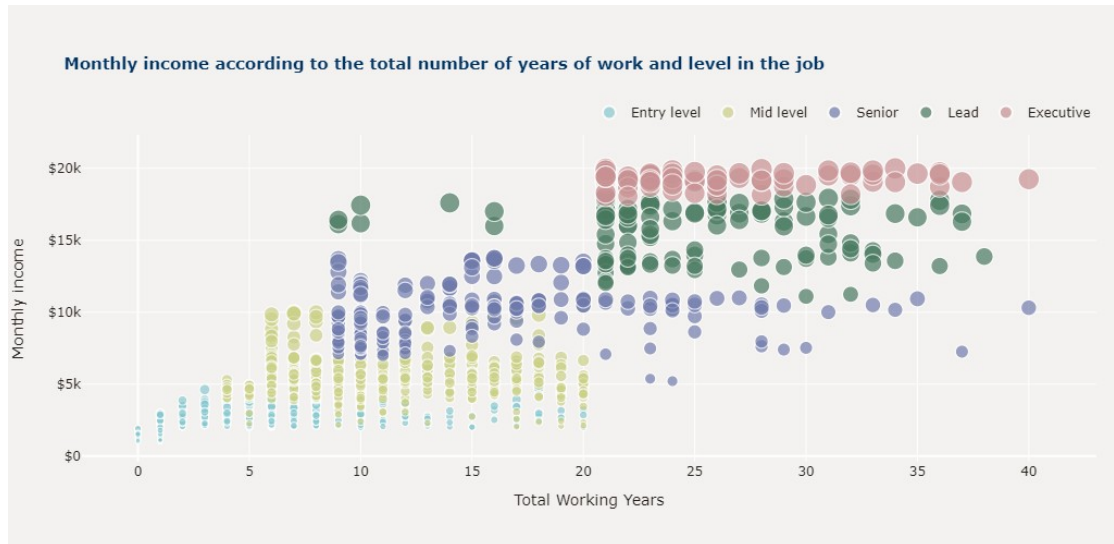
- Scatter plot là một công cụ mạnh mẽ để thể hiện quan hệ giữa hai biến số liên tục là 'MonthlyIncome' và 'TotalWorkingYears'. Ngoài ra để phân loại điểm dữ liệu thuộc 'JobLevel' nào ta có thể dùng tham số 'color', hoặc để thể hiện độ lớn của 'MonthlyIncome' ta có thể sử dụng tham số 'size' trong syntax của Scatter plot để biểu thị.

**Thực quan hóa:**

```
plot_df = df.copy()
plot_df['JobLevel'] = pd.Categorical(
    plot_df['JobLevel']).rename_categories(
    ['Entry level', 'Mid level', 'Senior', 'Lead', 'Executive'])
col=['#88C9D1', '#CCD188', '#707BAD', '#48795E', '#C99193']
fig = px.scatter(plot_df, x='TotalWorkingYears', y='MonthlyIncome',
    color='JobLevel', size='MonthlyIncome',
    color_discrete_sequence=col,
    category_orders={'JobLevel': ['Entry level', 'Mid
level', 'Senior', 'Lead', 'Executive']})

fig.update_layout(legend=dict(orientation="h", yanchor="bottom",
y=1.02, xanchor="right", x=1),
    title='<b>Monthly income according to the total
number of years of work and level in the job </b>',
    title_font = dict(size = 15, family = 'Verdana',
color = '#003566'),
    xaxis_title='Total Working Years',
yaxis=dict(title='Monthly income', tickprefix='$'),
    legend_title='', font_color='#28221D',
margin=dict(l=40, r=30, b=80,
t=120), paper_bgcolor='#F4F2F0', plot_bgcolor='#F4F2F0',
height = 500,
width = 1000)


fig.show()
```



### Nhận xét

Dựa vào biểu đồ phân tán (scatter plot) bên trên, ta có thể thấy:

- Cấp bậc trong công việc càng cao thì thu nhập hằng tháng có xu hướng càng nhiều.
- Các cấp bậc như : Entry level, Mid level có số năm làm việc trong khoảng từ 0 - 20 năm. Trong khi để có thể ở vị trí cao nhất là Executive thì cần số năm làm việc trên 20 năm, thậm chí trên 35 năm.

| Biểu đồ 13 

**Tiêu đề:** Mức độ hài lòng và cân bằng trong công việc theo độ tuổi.

**Loại biểu đồ:** Line plot

**Lý do lựa chọn:** Để trực quan sự thay đổi mức độ đánh giá theo độ tuổi, line plot là một loại biểu đồ thích hợp. Vì các điểm dữ liệu có giá trị gần nhau và nằm khoảng giá trị khá nhỏ, nên thang đo đánh giá không bắt đầu từ 1. Điều này không ảnh hưởng đến độ chính xác trong đánh giá vì biểu đồ chỉ so sánh sự cao thấp giữa các giá trị, không đánh giá gấp bao nhiêu lần,...

**Trực quan hóa:**

```
def year_stats(year):
    return year//5;

df_ = df.copy()
df_['year_stats'] = df_['YearsAtCompany'].map(year_stats)

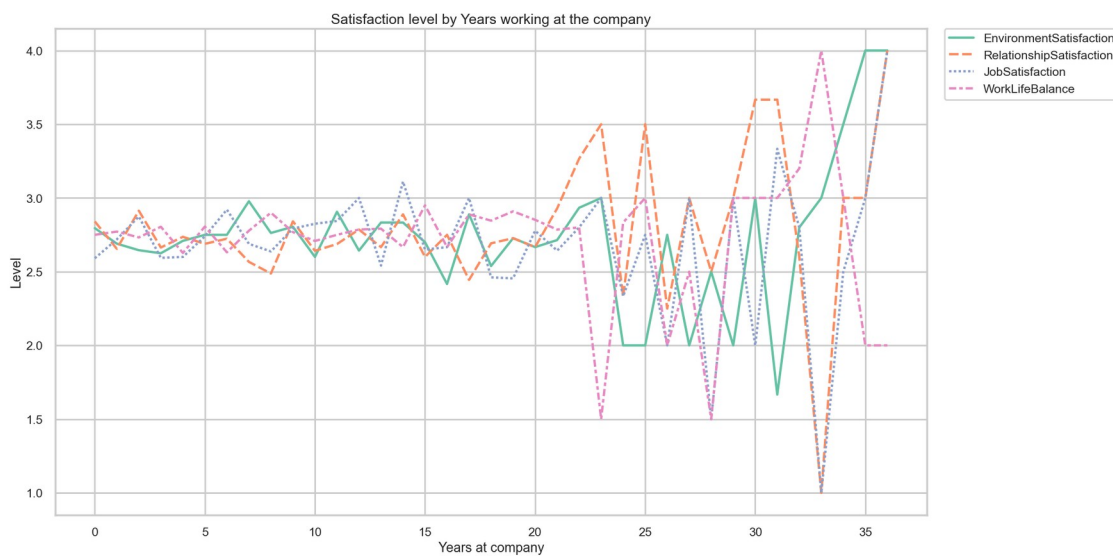
df_satis = df_[['EnvironmentSatisfaction', 'RelationshipSatisfaction',
'JobSatisfaction', 'WorkLifeBalance', 'YearsAtCompany', 'year_stats']]
df_satis_year = df_satis.groupby('YearsAtCompany')
[['EnvironmentSatisfaction', 'RelationshipSatisfaction', 'WorkLifeBalance',
'JobSatisfaction']].mean().reset_index()
```

```

sns.set_palette("Set2")
sns.set_style('whitegrid')
plt.subplots(figsize=(20, 10))
sns.lineplot(data =
df_satis_year[['EnvironmentSatisfaction', 'RelationshipSatisfaction',
'JobSatisfaction', 'WorkLifeBalance']])
plt.legend(bbox_to_anchor=(1.02, 1), loc='upper left',
borderaxespad=0)
plt.title('Satisfaction level by Years working at the company',
fontsize = 18)

plt.xlabel('Years at company')
plt.ylabel('Level')
plt.show()

```



### 💬 Nhận xét

- Có sự khác biệt rõ rệt giữa sự hài lòng của nhân viên theo độ tuổi, được phân cách rõ rệt ở năm làm việc thứ 20.
- Nhóm dưới 20 tuổi: ít có biến động hơn, dao động ở mức 2.5-3.
- Nhóm trên 20 tuổi:
  - Biến động nhiều hơn. Càng gắn bó với công ty, càng tiếp cận được nhiều khía cạnh khác nhau của công việc, nhu cầu mà mọi người đòi hỏi trong công việc cũng đa dạng hơn.
  - Work-life balance thường xu hướng ngược lại so với các đặc điểm còn lại. Điều này thể hiện sự hài lòng trong công việc ở 3 khía cạnh (môi trường, mối quan hệ, công việc) thường song hành và có liên quan với nhau. Ngoài ra sự hài lòng còn là động lực lớn với employees, họ có xu hướng dành nhiều thời gian cho công việc hơn. Từ đó thời gian cho các hoạt động khác của cuộc sống cũng giảm, ảnh hưởng work-life balance.
  - Càng gắn bó với công ty, sự ảnh hưởng ngày càng rõ rệt. Dù khi đó employees đã có kinh nghiệm làm việc ở công ty đó.

**Tiêu đề:** Phân bố mức lương theo vị trí công việc và mức độ hài lòng của mọi người.

**Loại biểu đồ:** Dot and jitter plot.

**Lý do lựa chọn:** Để trực quan sự phân bố giữa các categorical variables khác nhau, ta chọn jitter plots. Sau đó ta thêm việc trực quan Attrition vote của nhân viên qua việc thể hiện các chấm dữ liệu bằng các màu sắc khác nhau. Từ đó có thể nhìn thấy sự liên quan về mức lương với attrition vote giữa nhân viên ở các vai trò công việc khác nhau. **Trực quan hóa:**

```
df_['satisfaction'] = (df_['JobSatisfaction'] +
df_['EnvironmentSatisfaction'] + df_['RelationshipSatisfaction'])/3

stripplot = alt.Chart(df_, width=120, title="Monthly income by Job
Roles and Attrition rate").mark_circle(size=11).encode(
    #color = "Attrition",
    color=alt.Color('Attrition', scale=alt.Scale(scheme='dark2')),
    x=alt.X(
        'jitter:Q',
        title=None,
        axis=alt.Axis(values=[0], ticks=True, grid=False,
labels=False),
        scale=alt.Scale()
    ),
    y=alt.Y('MonthlyIncome:Q'),
    #color=alt.Color('JobRole:N', legend=None),
    column=alt.Column(
        'JobRole:N',
        header=alt.Header(
            labelAngle=0,
            titleOrient='top',
            labelOrient='bottom',
            labelAlign='center',
            labelPadding=10,
        ),
    ),

    tooltip = ["JobRole", "MonthlyIncome"]
).transform_calculate(
    # Generate Gaussian jitter with a Box-Muller transform
    jitter='sqrt(-2*log(random()))*cos(2*PI*random())'
).configure_facet(
    spacing=0
).properties(
    width=125,
    height=500
).configure_title(
    fontSize=20,
    anchor='middle', # <---- does not take effect, why?
```


```

        color='gray'
    )
    stripplot
alt.Chart(...)

```

#### Nhận xét

- Phân bố mức lương của các vị trí công việc có sự khác biệt. Trong đó cao nhất là Manager và Research Director, đây là các vị trí quản lý. Đây cũng là 2 nhóm có số lượng nhân viên "hối hận" thấp nhất.
- Vị trí Human Resources có số lượng ít nhưng range lương cũng trải ra một khoảng khá rộng, thể hiện dù nghề này ít nhân sự nhưng mức lương cũng có sự đa dạng giữa các cá nhân.
- Khi phân chia về Attrition rating, ta thấy có sự phân nhóm khá rõ nét với 2 vị trí là Human Resources và Sales Representative. Những người vote Yes thì nằm ở nhóm có mức lương thấp hơn, do đó có thể thấy mức lương ảnh hưởng nhiều đến đánh giá của nhân viên về công việc ở hai ngành này. Các ngành còn lại, Attrition vote không có sự khác biệt nhiều ở các rank lương (các dot nằm khá trộn lẫn vào nhau).

| Biểu đồ 15 

**Tiêu đề:** Tên các lĩnh vực học tập thể hiện qua đám mây từ

**Loại biểu đồ:** WordCloud

**Lý do lựa chọn:** Một đám mây từ trong Python sẽ biểu thị trực quan cho dữ liệu văn bản. Còn được gọi là đám mây thẻ, nó sử dụng các cỡ chữ và màu sắc khác nhau để làm nổi bật tầm quan trọng của mỗi từ. Bằng cách này sẽ làm nổi trội tên các lĩnh vực học tập của nhân viên (Education Field) để lại ấn tượng cho người xem.

**Trực quan hóa:**

```

text = " ".join(EducationField for EducationField in
df["EducationField"])
wordcloud = WordCloud(width = 800, height = 250,
                        background_color
                        ="black", colormap="RdYlGn", max_font_size=100, stopwords =None, repeat=
True).generate(text)
plt.figure(figsize = (20, 8), facecolor= "black")
plt.imshow(wordcloud)
plt.axis("off")
plt.margins(x=0, y=0)
plt.tight_layout(pad = 0)
plt.show()

```





[Nhận xét](#)

Có rất nhiều tên ngành học được thể hiện một cách đặc sắc trên WordCloud như: human resources, sciences medical, life sciences, marketing, technical degree, ...

[⬆️ Back to Table of Contents ⬆️](#)

## D. Mô hình học máy

### I. Bài toán đặt ra

#### Bài toán đặt ra:

Dự đoán liệu rằng nhân viên có nguy cơ nghỉ việc ở công ty đang làm việc hay không?

#### Giới thiệu chung:

- Trong học máy, học có giám sát là một nhóm các thuật toán phổ biến trong lĩnh vực này và một trong những vấn đề quan trọng của học có giám sát là phân loại/phân lớp(classification problem).
- Có 2 dạng classification thường gặp là: binary classification và multiclassification, và trong bài toán mà nhóm đặt ra thì đây là một vấn đề binary classification: từ những thuộc tính đầu vào của một nhân viên như tổng số năm làm việc, mức lương, sự hài lòng về môi trường làm việc, ... dự đoán rằng nhân viên đó nguy cơ nghỉ làm ở công ty hiện tại hay không (Công ty có bị mất mát nhân viên hay không) (0: Không/ 1: Có).
- Nhóm sẽ tạo một mô hình logistic regression cho bài toán phân loại nhị phân.

[⬆️ Back to Table of Contents ⬆️](#)

### II. Tiền xử lý dữ liệu

#### 1. Mã hóa các thuộc tính dạng danh mục về dạng số

*#lọc ra các thuộc tính dạng danh mục*

```
cat_cols=df.select_dtypes(exclude=['int32','int64','float32','float64'])
```

```

])
cat_cols.head()

```

	Attrition	BusinessTravel	Department	EducationField
0	Yes	Travel_Rarely	Sales	Life Sciences
1	No	Travel_Frequently	Research & Development	Life Sciences
2	Yes	Travel_Rarely	Research & Development	Other
3	No	Travel_Frequently	Research & Development	Life Sciences
4	No	Travel_Rarely	Research & Development	Medical

	JobRole	MaritalStatus	Over18	OverTime
0	Sales Executive	Single	Y	Yes
1	Research Scientist	Married	Y	No
2	Laboratory Technician	Single	Y	Yes
3	Research Scientist	Married	Y	Yes
4	Laboratory Technician	Married	Y	No

*#thông kê số lượng các giá trị riêng biệt trong môi thuộc tính danh mục ở trên*

```

count_uvalue=[cat_cols[c].nunique() for c in list(cat_cols.columns)]
count_uvalue

```

```

[2, 3, 3, 6, 2, 9, 3, 1, 2]

```

Một vài nhận xét như sau:

- Cột Over18 chỉ chứa 1 loại giá trị duy nhất là 'Y' chứng tỏ cột này sẽ không ảnh hưởng đến kết quả của việc huấn luyện mô hình.
- Có 3 thuộc tính là Attrition, Gender, và OverTime là có 2 loại giá trị khác nhau, do đó với các thuộc tính này chúng ta có thể mã hóa bằng cách gán nhãn 0/1 cho chúng.
- Đối với các thuộc tính có nhiều hơn 2 loại giá trị, chúng ta sẽ mã hóa bằng one-hot vector, lý do sử dụng one-hot mà không dùng ordinal hay label để tránh xảy ra hiện tượng bias do mã hóa thành các giá trị lớn nếu số lượng giá trị lớn.

*#xóa thuộc tính 'Over18'*

```

df.drop(['Over18'],axis=1, inplace=True)

```

*#chuyển các thuộc tính danh mục chỉ có 2 giá trị riêng biệt về dạng số bằng cách gán nhãn*

```

from sklearn.preprocessing import LabelEncoder
label_encoder=LabelEncoder()
df['Attrition']=label_encoder.fit_transform(df['Attrition'])
df['OverTime']=label_encoder.fit_transform(df['OverTime'])
df['Gender']=label_encoder.fit_transform(df['Gender'])

```

*#chuyên các thuộc tính danh mục có hơn 2 giá trị riêng biệt về dạng số bằng one-hot vector*

```
df=pd.get_dummies(df, columns=['BusinessTravel', 'Department',
                                'EducationField',
                                'JobRole', 'MaritalStatus'])
```

```
df.head()
```

	Age	Attrition	DailyRate	DistanceFromHome	Education
EmployeeCount \					
0	41	1	1102	1	2
1					
1	49	0	279	8	1
1					
2	37	1	1373	2	2
1					
3	33	0	1392	3	4
1					
4	27	0	591	2	1
1					

	EmployeeNumber	EnvironmentSatisfaction	Gender	HourlyRate	...	\
0		1	2	0	94	...
1		2	3	1	61	...
2		4	4	1	92	...
3		5	4	0	56	...
4		7	1	1	40	...

	JobRole_Laboratory Technician	JobRole_Manager	\
0	0	0	
1	0	0	
2	1	0	
3	0	0	
4	1	0	

	JobRole_Manufacturing Director	JobRole_Research Director	\
0	0	0	
1	0	0	
2	0	0	
3	0	0	
4	0	0	

	JobRole_Research Scientist	JobRole_Sales Executive	\
0	0	1	
1	1	0	
2	0	0	
3	1	0	
4	0	0	

	JobRole_Sales Representative	MaritalStatus_Divorced	\
--	------------------------------	------------------------	---

0	0	0
1	0	0
2	0	0
3	0	0
4	0	0

	MaritalStatus_Married	MaritalStatus_Single
0	0	1
1	1	0
2	0	1
3	1	0
4	1	0

[5 rows x 53 columns]

2. Loại những thuộc tính không có ý nghĩa cho bài toán

**Cách 1:**

- **Cách thô sơ nhất có thể nghĩ đến là cách loại các thuộc tính có thể thấy ngay về mặt ý nghĩa là không cần thiết cho bài toán phân loại:** Ví dụ thuộc tính 'Over18' được loại bỏ ở trên do chúng chỉ mang một giá trị 'yes', để hiểu khi bộ dữ liệu được cung cấp bởi IBM, có trụ sở ở Mỹ và nước Mỹ chỉ cho phép đi làm khi đủ 18 tuổi trở lên. Ngoài ra khi nhìn vào bảng ý nghĩa các thuộc tính mà nhóm đã cung cấp ở phần thu thập dữ liệu thì cột 'StandardHours' cũng sẽ không có ý nghĩa cho bài toán vì cột này mang cùng một giá trị cho tất cả nhân viên là giờ làm chuẩn mà họ phải đi làm, có thể xóa thuộc tính này đi.

*#xóa 'StandardHours'*

```
df.drop(['StandardHours'],axis=1,inplace=True)
```

**Cách 2:**

- **Cách tiếp theo là sử dụng giá trị correlations giữa từng biến biến độc lập với biến phụ thuộc:**
  - Correlation là một thuật ngữ thống kê được sử dụng phổ biến để cập đến mức độ liên quan của hai biến để có mối quan hệ tuyến tính với nhau hay không.
  - Correlation cao nhất có giá trị là 1 (hai biến hoàn toàn có quan hệ tuyến tính) và thấp nhất dần nếu hai biến càng không có quan hệ tuyến tính.
  - Nhóm sẽ tạo một dataframe tên là 'correlations' chứa các correlations của từng cột trong bộ dữ liệu để dễ dàng nhận xét mức độ tương quan giữa các biến.

```
correlations=df.corr()
correlations
```

	Age	Attrition	DailyRate	\
Age	1.000000	-0.159205	0.010661	
Attrition	-0.159205	1.000000	-0.056652	
DailyRate	0.010661	-0.056652	1.000000	

DistanceFromHome	-0.001686	0.077924	-0.004985
Education	0.208034	-0.031373	-0.016806
EmployeeCount	NaN	NaN	NaN
EmployeeNumber	-0.010145	-0.010577	-0.050990
EnvironmentSatisfaction	0.010146	-0.103369	0.018355
Gender	-0.036311	0.029453	-0.011716
HourlyRate	0.024287	-0.006846	0.023381
JobInvolvement	0.029820	-0.130016	0.046135
JobLevel	0.509604	-0.169105	0.002966
JobSatisfaction	-0.004892	-0.103481	0.030571
MonthlyIncome	0.497855	-0.159840	0.007707
MonthlyRate	0.028051	0.015170	-0.032182
NumCompaniesWorked	0.299635	0.043494	0.038153
OverTime	0.028062	0.246118	0.009135
PercentSalaryHike	0.003634	-0.013478	0.022704
PerformanceRating	0.001904	0.002889	0.000473
RelationshipSatisfaction	0.053535	-0.045872	0.007846
StockOptionLevel	0.037510	-0.137145	0.042143
TotalWorkingYears	0.680381	-0.171063	0.014515
TrainingTimesLastYear	-0.019621	-0.059478	0.002453
WorkLifeBalance	-0.021490	-0.063939	-0.037848
YearsAtCompany	0.311309	-0.134392	-0.034055
YearsInCurrentRole	0.212901	-0.160545	0.009932
YearsSinceLastPromotion	0.216513	-0.033019	-0.033229
YearsWithCurrManager	0.202089	-0.156199	-0.026363
BusinessTravel_Non-Travel	-0.011215	-0.074457	0.012096
BusinessTravel_Travel_Frequently	-0.024743	0.115143	-0.011776
BusinessTravel_Travel_Rarely	0.028791	-0.049538	0.002078
Department_Human Resources	0.020523	0.016832	-0.026726
Department_Research & Development	0.017883	-0.085293	0.014871
Department_Sales	-0.027549	0.080855	-0.003616
EducationField_Human Resources	0.001696	0.036466	-0.043144
EducationField_Life Sciences	0.016824	-0.032703	0.004028
EducationField_Marketing	0.038162	0.055781	-0.064449
EducationField_Medical	-0.006354	-0.046999	0.034202
EducationField_Other	-0.041466	-0.017898	-0.003893
EducationField_Technical Degree	-0.027604	0.069355	0.030869
JobRole_Healthcare Representative	0.098825	-0.078696	0.040141
JobRole_Human Resources	-0.029856	0.036215	-0.021156
JobRole_Laboratory Technician	-0.143176	0.098290	-0.006728
JobRole_Manager	0.294248	-0.083316	-0.013224
JobRole_Manufacturing Director	0.049726	-0.082994	-0.005302
JobRole_Research Director	0.185891	-0.088870	-0.000021
JobRole_Research Scientist	-0.146518	-0.000360	-0.002624
JobRole_Sales Executive	-0.002001	0.019774	-0.000513
JobRole_Sales Representative	-0.175785	0.157234	0.005375
MaritalStatus_Divorced	0.033120	-0.087716	0.037080
MaritalStatus_Married	0.083919	-0.090984	0.040035
MaritalStatus_Single	-0.119185	0.175419	-0.075835

	DistanceFromHome	Education
EmployeeCount \		
Age	-0.001686	0.208034
NaN		
Attrition	0.077924	-0.031373
NaN		
DailyRate	-0.004985	-0.016806
NaN		
DistanceFromHome	1.000000	0.021042
NaN		
Education	0.021042	1.000000
NaN		
EmployeeCount	NaN	NaN
NaN		
EmployeeNumber	0.032916	0.042070
NaN		
EnvironmentSatisfaction	-0.016075	-0.027128
NaN		
Gender	-0.001851	-0.016547
NaN		
HourlyRate	0.031131	0.016775
NaN		
JobInvolvement	0.008783	0.042438
NaN		
JobLevel	0.005303	0.101589
NaN		
JobSatisfaction	-0.003669	-0.011296
NaN		
MonthlyIncome	-0.017014	0.094961
NaN		
MonthlyRate	0.027473	-0.026084
NaN		
NumCompaniesWorked	-0.029251	0.126317
NaN		
OverTime	0.025514	-0.020322
NaN		
PercentSalaryHike	0.040235	-0.011111
NaN		
PerformanceRating	0.027110	-0.024539
NaN		
RelationshipSatisfaction	0.006557	-0.009118
NaN		
StockOptionLevel	0.044872	0.018422
NaN		
TotalWorkingYears	0.004628	0.148280
NaN		
TrainingTimesLastYear	-0.036942	-0.025100
NaN		
WorkLifeBalance	-0.026556	0.009819
NaN		

YearsAtCompany	0.009508	0.069114
NaN		
YearsInCurrentRole	0.018845	0.060236
NaN		
YearsSinceLastPromotion	0.010029	0.054254
NaN		
YearsWithCurrManager	0.014406	0.069065
NaN		
BusinessTravel_Non-Travel	0.023605	0.004524
NaN		
BusinessTravel_Travel_Frequently	0.005081	-0.008292
NaN		
BusinessTravel_Travel_Rarely	-0.020116	0.004126
NaN		
Department_Human Resources	-0.012901	0.011435
NaN		
Department_Research & Development	-0.008117	-0.018604
NaN		
Department_Sales	0.014085	0.014215
NaN		
EducationField_Human Resources	-0.002624	0.026479
NaN		
EducationField_Life Sciences	-0.024499	0.013184
NaN		
EducationField_Marketing	0.039294	0.072405
NaN		
EducationField_Medical	0.013486	-0.072335
NaN		
EducationField_Other	-0.007969	0.038043
NaN		
EducationField_Technical Degree	-0.014802	-0.026742
NaN		
JobRole_Healthcare Representative	0.022916	0.024270
NaN		
JobRole_Human Resources	-0.024089	-0.005295
NaN		
JobRole_Laboratory Technician	0.012369	-0.063566
NaN		
JobRole_Manager	-0.039190	0.028453
NaN		
JobRole_Manufacturing Director	0.011848	-0.005290
NaN		
JobRole_Research Director	-0.022351	0.049694
NaN		
JobRole_Research Scientist	-0.010986	0.000709
NaN		
JobRole_Sales Executive	0.030761	0.053398
NaN		
JobRole_Sales Representative	-0.015994	-0.091465
NaN		

MaritalStatus_Divorced	-0.005440	-0.002439
NaN		
MaritalStatus_Married	0.030232	-0.001865
NaN		
MaritalStatus_Single	-0.027445	0.004168
NaN		

	EmployeeNumber	
EnvironmentSatisfaction \		
Age	-0.010145	
0.010146		
Attrition	-0.010577	-
0.103369		
DailyRate	-0.050990	
0.018355		
DistanceFromHome	0.032916	-
0.016075		
Education	0.042070	-
0.027128		
EmployeeCount	NaN	
NaN		
EmployeeNumber	1.000000	
0.017621		
EnvironmentSatisfaction	0.017621	
1.000000		
Gender	0.022556	
0.000508		
HourlyRate	0.035179	-
0.049857		
JobInvolvement	-0.006888	-
0.008278		
JobLevel	-0.018519	
0.001212		
JobSatisfaction	-0.046247	-
0.006784		
MonthlyIncome	-0.014829	-
0.006259		
MonthlyRate	0.012648	
0.037600		
NumCompaniesWorked	-0.001251	
0.012594		
OverTime	-0.024037	
0.070132		
PercentSalaryHike	-0.012944	-
0.031701		
PerformanceRating	-0.020359	-
0.029548		
RelationshipSatisfaction	-0.069861	
0.007665		
StockOptionLevel	0.062227	



0.003432		
TotalWorkingYears	-0.014365	-
0.002693		
TrainingTimesLastYear	0.023603	-
0.019359		
WorkLifeBalance	0.010309	
0.027627		
YearsAtCompany	-0.011240	
0.001458		
YearsInCurrentRole	-0.008416	
0.018007		
YearsSinceLastPromotion	-0.009019	
0.016194		
YearsWithCurrManager	-0.009197	-
0.004999		
BusinessTravel_Non-Travel	0.022272	
0.003568		
BusinessTravel_Travel_Frequently	-0.007980	-
0.012624		
BusinessTravel_Travel_Rarely	-0.007976	
0.008496		
Department_Human Resources	0.063431	-
0.007597		
Department_Research & Development	-0.041923	
0.027976		
Department_Sales	0.015441	-
0.025606		
EducationField_Human Resources	0.035345	-
0.006898		
EducationField_Life Sciences	-0.000609	-
0.024526		
EducationField_Marketing	-0.014487	
0.000479		
EducationField_Medical	-0.008689	-
0.021299		
EducationField_Other	0.010432	
0.064602		
EducationField_Technical Degree	0.005938	
0.027713		
JobRole_Healthcare Representative	0.025945	
0.014090		
JobRole_Human Resources	0.067287	-
0.022014		
JobRole_Laboratory Technician	-0.019722	-
0.001533		
JobRole_Manager	-0.035058	
0.010730		
JobRole_Manufacturing Director	-0.014350	
0.059178		
JobRole_Research Director	-0.013983	-

0.048689		
JobRole_Research Scientist	-0.017686	
0.001940		
JobRole_Sales Executive	0.023263	-
0.024421		
JobRole_Sales Representative	0.006255	
0.002949		
MaritalStatus_Divorced	-0.025149	
0.016439		
MaritalStatus_Married	0.053933	-
0.022180		
MaritalStatus_Single	-0.035189	
0.009035		

	Gender	HourlyRate	...	\
Age	-0.036311	0.024287	...	
Attrition	0.029453	-0.006846	...	
DailyRate	-0.011716	0.023381	...	
DistanceFromHome	-0.001851	0.031131	...	
Education	-0.016547	0.016775	...	
EmployeeCount	NaN	NaN	...	
EmployeeNumber	0.022556	0.035179	...	
EnvironmentSatisfaction	0.000508	-0.049857	...	
Gender	1.000000	-0.000478	...	
HourlyRate	-0.000478	1.000000	...	
JobInvolvement	0.017960	0.042861	...	
JobLevel	-0.039403	-0.027853	...	
JobSatisfaction	0.033252	-0.071335	...	
MonthlyIncome	-0.031858	-0.015794	...	
MonthlyRate	-0.041482	-0.015297	...	
NumCompaniesWorked	-0.039147	0.022157	...	
OverTime	-0.041924	-0.007782	...	
PercentSalaryHike	0.002733	-0.009062	...	
PerformanceRating	-0.013859	-0.002172	...	
RelationshipSatisfaction	0.022868	0.001330	...	
StockOptionLevel	0.012716	0.050263	...	
TotalWorkingYears	-0.046881	-0.002334	...	
TrainingTimesLastYear	-0.038787	-0.008548	...	
WorkLifeBalance	-0.002753	-0.004607	...	
YearsAtCompany	-0.029747	-0.019582	...	
YearsInCurrentRole	-0.041483	-0.024106	...	
YearsSinceLastPromotion	-0.026985	-0.026716	...	
YearsWithCurrManager	-0.030599	-0.020123	...	
BusinessTravel_Non-Travel	0.050461	-0.016994	...	
BusinessTravel_Travel_Frequently	-0.022015	-0.018819	...	
BusinessTravel_Travel_Rarely	-0.014682	0.027541	...	
Department_Human Resources	0.035652	-0.016551	...	
Department_Research & Development	0.015760	0.018686	...	
Department_Sales	-0.032017	-0.012047	...	
EducationField_Human Resources	0.028956	-0.033670	...	

EducationField_Life Sciences	0.006770	0.038759	...
EducationField_Marketing	-0.024143	0.004452	...
EducationField_Medical	-0.013146	-0.020418	...
EducationField_Other	0.022992	-0.042163	...
EducationField_Technical Degree	0.003886	0.011283	...
JobRole_Healthcare Representative	0.006823	0.014599	...
JobRole_Human Resources	0.036082	-0.016189	...
JobRole_Laboratory Technician	0.067793	0.018028	...
JobRole_Manager	-0.033880	0.012659	...
JobRole_Manufacturing Director	-0.065197	-0.014394	...
JobRole_Research Director	-0.006121	-0.025128	...
JobRole_Research Scientist	0.009745	0.020034	...
JobRole_Sales Executive	-0.005348	-0.011886	...
JobRole_Sales Representative	-0.028877	-0.018703	...
MaritalStatus_Divorced	0.046076	-0.006150	...
MaritalStatus_Married	-0.007804	0.036432	...
MaritalStatus_Single	-0.032752	-0.033436	...

	JobRole_Laboratory Technician \
Age	-0.143176
Attrition	0.098290
DailyRate	-0.006728
DistanceFromHome	0.012369
Education	-0.063566
EmployeeCount	NaN
EmployeeNumber	-0.019722
EnvironmentSatisfaction	-0.001533
Gender	0.067793
HourlyRate	0.018028
JobInvolvement	-0.022724
JobLevel	-0.344608
JobSatisfaction	-0.015710
MonthlyIncome	-0.320906
MonthlyRate	-0.016056
NumCompaniesWorked	-0.021121
Overtime	-0.044774
PercentSalaryHike	-0.020628
PerformanceRating	0.010796
RelationshipSatisfaction	-0.010691
StockOptionLevel	0.013386
TotalWorkingYears	-0.215426
TrainingTimesLastYear	0.053998
WorkLifeBalance	-0.028209
YearsAtCompany	-0.150181
YearsInCurrentRole	-0.131322
YearsSinceLastPromotion	-0.110099
YearsWithCurrManager	-0.107072
BusinessTravel_Non-Travel	0.009270
BusinessTravel_Travel_Frequently	0.010023
BusinessTravel_Travel_Rarely	-0.014815

Department_Human Resources	-0.097859
Department_Research & Development	0.336570
Department_Sales	-0.305208
EducationField_Human Resources	-0.063260
EducationField_Life Sciences	0.044359
EducationField_Marketing	-0.161055
EducationField_Medical	0.066262
EducationField_Other	0.058759
EducationField_Technical Degree	-0.026589
JobRole_Healthcare Representative	-0.144652
JobRole_Human Resources	-0.088561
JobRole_Laboratory Technician	1.000000
JobRole_Manager	-0.126280
JobRole_Manufacturing Director	-0.152987
JobRole_Research Director	-0.110947
JobRole_Research Scientist	-0.230248
JobRole_Sales Executive	-0.246873
JobRole_Sales Representative	-0.113130
MaritalStatus_Divorced	-0.011224
MaritalStatus_Married	-0.009233
MaritalStatus_Single	0.019873

	JobRole_Manager \
Age	0.294248
Attrition	-0.083316
DailyRate	-0.013224
DistanceFromHome	-0.039190
Education	0.028453
EmployeeCount	NaN
EmployeeNumber	-0.035058
EnvironmentSatisfaction	0.010730
Gender	-0.033880
HourlyRate	0.012659
JobInvolvement	0.017112
JobLevel	0.552744
JobSatisfaction	-0.005620
MonthlyIncome	0.619573
MonthlyRate	0.031717
NumCompaniesWorked	0.042125
OverTime	-0.011086
PercentSalaryHike	-0.005394
PerformanceRating	0.032050
RelationshipSatisfaction	0.025638
StockOptionLevel	-0.015637
TotalWorkingYears	0.465837
TrainingTimesLastYear	0.003052
WorkLifeBalance	0.005137
YearsAtCompany	0.330965
YearsInCurrentRole	0.167499
YearsSinceLastPromotion	0.224255

YearsWithCurrManager	0.164695
BusinessTravel_Non-Travel	0.014078
BusinessTravel_Travel_Frequently	-0.042583
BusinessTravel_Travel_Rarely	0.027294
Department_Human Resources	0.087615
Department_Research & Development	-0.071356
Department_Sales	0.035248
EducationField_Human Resources	0.082271
EducationField_Life Sciences	-0.011143
EducationField_Marketing	0.025577
EducationField_Medical	-0.001128
EducationField_Other	-0.008046
EducationField_Technical Degree	-0.038946
JobRole_Healthcare Representative	-0.085409
JobRole_Human Resources	-0.052290
JobRole_Laboratory Technician	-0.126280
JobRole_Manager	1.000000
JobRole_Manufacturing Director	-0.090330
JobRole_Research Director	-0.065508
JobRole_Research Scientist	-0.135949
JobRole_Sales Executive	-0.145765
JobRole_Sales Representative	-0.066797
MaritalStatus_Divorced	0.001997
MaritalStatus_Married	0.049982
MaritalStatus_Single	-0.055176

	JobRole_Manufacturing Director \
Age	0.049726
Attrition	-0.082994
DailyRate	-0.005302
DistanceFromHome	0.011848
Education	-0.005290
EmployeeCount	NaN
EmployeeNumber	-0.014350
EnvironmentSatisfaction	0.059178
Gender	-0.065197
HourlyRate	-0.014394
JobInvolvement	-0.021939
JobLevel	0.114896
JobSatisfaction	-0.013747
MonthlyIncome	0.055684
MonthlyRate	0.007711
NumCompaniesWorked	0.009580
OverTime	-0.010302
PercentSalaryHike	0.034682
PerformanceRating	0.029775
RelationshipSatisfaction	0.003640
StockOptionLevel	0.007735
TotalWorkingYears	0.064077
TrainingTimesLastYear	-0.013987

WorkLifeBalance	0.002011
YearsAtCompany	0.031968
YearsInCurrentRole	0.067877
YearsSinceLastPromotion	-0.007241
YearsWithCurrManager	0.076207
BusinessTravel_Non-Travel	-0.013536
BusinessTravel_Travel_Frequently	0.009783
BusinessTravel_Travel_Rarely	0.000598
Department_Human Resources	-0.070000
Department_Research & Development	0.240754
Department_Sales	-0.218320
EducationField_Human Resources	-0.045251
EducationField_Life Sciences	0.052023
EducationField_Marketing	-0.115206
EducationField_Medical	0.035496
EducationField_Other	-0.010820
EducationField_Technical Degree	0.007817
JobRole_Healthcare Representative	-0.103472
JobRole_Human Resources	-0.063349
JobRole_Laboratory Technician	-0.152987
JobRole_Manager	-0.090330
JobRole_Manufacturing Director	1.000000
JobRole_Research Director	-0.079362
JobRole_Research Scientist	-0.164700
JobRole_Sales Executive	-0.176592
JobRole_Sales Representative	-0.080924
MaritalStatus_Divorced	0.020543
MaritalStatus_Married	0.002819
MaritalStatus_Single	-0.021331

	JobRole_Research Director \
Age	0.185891
Attrition	-0.088870
DailyRate	-0.000021
DistanceFromHome	-0.022351
Education	0.049694
EmployeeCount	NaN
EmployeeNumber	-0.013983
EnvironmentSatisfaction	-0.048689
Gender	-0.006121
HourlyRate	-0.025128
JobInvolvement	0.015200
JobLevel	0.414319
JobSatisfaction	-0.006217
MonthlyIncome	0.485818
MonthlyRate	0.025875
NumCompaniesWorked	0.097925
OverTime	0.002400
PercentSalaryHike	-0.017017
PerformanceRating	-0.035744

RelationshipSatisfaction	-0.005492
StockOptionLevel	0.015807
TotalWorkingYears	0.312148
TrainingTimesLastYear	-0.004527
WorkLifeBalance	0.034403
YearsAtCompany	0.153918
YearsInCurrentRole	0.136332
YearsSinceLastPromotion	0.074455
YearsWithCurrManager	0.131279
BusinessTravel_Non-Travel	-0.021431
BusinessTravel_Travel_Frequently	-0.023579
BusinessTravel_Travel_Rarely	0.034600
Department_Human Resources	-0.050765
Department_Research & Development	0.174596
Department_Sales	-0.158327
EducationField_Human Resources	-0.032816
EducationField_Life Sciences	0.018401
EducationField_Marketing	-0.083548
EducationField_Medical	0.062898
EducationField_Other	-0.006044
EducationField_Technical Degree	-0.022905
JobRole_Healthcare Representative	-0.075038
JobRole_Human Resources	-0.045941
JobRole_Laboratory Technician	-0.110947
JobRole_Manager	-0.065508
JobRole_Manufacturing Director	-0.079362
JobRole_Research Director	1.000000
JobRole_Research Scientist	-0.119442
JobRole_Sales Executive	-0.128066
JobRole_Sales Representative	-0.058687
MaritalStatus_Divorced	0.037524
MaritalStatus_Married	0.008271
MaritalStatus_Single	-0.042299

	JobRole_Research Scientist \
Age	-0.146518
Attrition	-0.000360
DailyRate	-0.002624
DistanceFromHome	-0.010986
Education	0.000709
EmployeeCount	NaN
EmployeeNumber	-0.017686
EnvironmentSatisfaction	0.001940
Gender	0.009745
HourlyRate	0.020034
JobInvolvement	0.047604
JobLevel	-0.387788
JobSatisfaction	0.020503
MonthlyIncome	-0.345180
MonthlyRate	-0.027008

NumCompaniesWorked	-0.043981
OverTime	0.054378
PercentSalaryHike	0.032537
PerformanceRating	0.019416
RelationshipSatisfaction	-0.003116
StockOptionLevel	-0.011635
TotalWorkingYears	-0.228119
TrainingTimesLastYear	-0.052126
WorkLifeBalance	-0.058613
YearsAtCompany	-0.154062
YearsInCurrentRole	-0.131314
YearsSinceLastPromotion	-0.105237
YearsWithCurrManager	-0.127608
BusinessTravel_Non-Travel	-0.010116
BusinessTravel_Travel_Frequently	-0.004461
BusinessTravel_Travel_Rarely	0.010588
Department_Human Resources	-0.105352
Department_Research & Development	0.362340
Department_Sales	-0.328576
EducationField_Human Resources	-0.068103
EducationField_Life Sciences	0.043729
EducationField_Marketing	-0.173387
EducationField_Medical	0.039735
EducationField_Other	0.005286
EducationField_Technical Degree	0.076218
JobRole_Healthcare Representative	-0.155727
JobRole_Human Resources	-0.095342
JobRole_Laboratory Technician	-0.230248
JobRole_Manager	-0.135949
JobRole_Manufacturing Director	-0.164700
JobRole_Research Director	-0.119442
JobRole_Research Scientist	1.000000
JobRole_Sales Executive	-0.265775
JobRole_Sales Representative	-0.121792
MaritalStatus_Divorced	-0.012115
MaritalStatus_Married	-0.039987
MaritalStatus_Single	0.053522

	JobRole_Sales Executive \
Age	-0.002001
Attrition	0.019774
DailyRate	-0.000513
DistanceFromHome	0.030761
Education	0.053398
EmployeeCount	NaN
EmployeeNumber	0.023263
EnvironmentSatisfaction	-0.024421
Gender	-0.005348
HourlyRate	-0.011886
JobInvolvement	-0.011413



JobLevel	0.127490
JobSatisfaction	0.012604
MonthlyIncome	0.047792
MonthlyRate	0.011854
NumCompaniesWorked	0.005913
OverTime	0.006341
PercentSalaryHike	-0.046683
PerformanceRating	-0.041401
RelationshipSatisfaction	-0.004836
StockOptionLevel	0.015756
TotalWorkingYears	-0.012241
TrainingTimesLastYear	0.013241
WorkLifeBalance	0.032092
YearsAtCompany	0.042602
YearsInCurrentRole	0.092349
YearsSinceLastPromotion	0.049202
YearsWithCurrManager	0.083028
BusinessTravel_Non-Travel	0.031022
BusinessTravel_Travel_Frequently	-0.010175
BusinessTravel_Travel_Rarely	-0.011920
Department_Human Resources	-0.112959
Department_Research & Development	-0.733497
Department_Sales	0.808869
EducationField_Human Resources	-0.073020
EducationField_Life Sciences	-0.091122
EducationField_Marketing	0.457308
EducationField_Medical	-0.133532
EducationField_Other	-0.036995
EducationField_Technical Degree	-0.058843
JobRole_Healthcare Representative	-0.166971
JobRole_Human Resources	-0.102226
JobRole_Laboratory Technician	-0.246873
JobRole_Manager	-0.145765
JobRole_Manufacturing Director	-0.176592
JobRole_Research Director	-0.128066
JobRole_Research Scientist	-0.265775
JobRole_Sales Executive	1.000000
JobRole_Sales Representative	-0.130586
MaritalStatus_Divorced	-0.013853
MaritalStatus_Married	0.005751
MaritalStatus_Single	0.006210

	JobRole_Sales Representative \
Age	-0.175785
Attrition	0.157234
DailyRate	0.005375
DistanceFromHome	-0.015994
Education	-0.091465
EmployeeCount	NaN
EmployeeNumber	0.006255

EnvironmentSatisfaction	0.002949
Gender	-0.028877
HourlyRate	-0.018703
JobInvolvement	-0.027282
JobLevel	-0.216559
JobSatisfaction	0.001413
MonthlyIncome	-0.201514
MonthlyRate	-0.001200
NumCompaniesWorked	-0.104494
OverTime	0.003347
PercentSalaryHike	0.031102
PerformanceRating	-0.006214
RelationshipSatisfaction	-0.024859
StockOptionLevel	-0.048067
TotalWorkingYears	-0.207726
TrainingTimesLastYear	0.040377
WorkLifeBalance	0.045148
YearsAtCompany	-0.163464
YearsInCurrentRole	-0.149751
YearsSinceLastPromotion	-0.085622
YearsWithCurrManager	-0.168743
BusinessTravel_Non-Travel	-0.033780
BusinessTravel_Travel_Frequently	0.055469
BusinessTravel_Travel_Rarely	-0.025257
Department_Human Resources	-0.051764
Department_Research & Development	-0.336127
Department_Sales	0.370667
EducationField_Human Resources	-0.033462
EducationField_Life Sciences	-0.043208
EducationField_Marketing	0.133065
EducationField_Medical	-0.051990
EducationField_Other	-0.033774
EducationField_Technical Degree	0.057185
JobRole_Healthcare Representative	-0.076515
JobRole_Human Resources	-0.046845
JobRole_Laboratory Technician	-0.113130
JobRole_Manager	-0.066797
JobRole_Manufacturing Director	-0.080924
JobRole_Research Director	-0.058687
JobRole_Research Scientist	-0.121792
JobRole_Sales Executive	-0.130586
JobRole_Sales Representative	1.000000
MaritalStatus_Divorced	-0.052890
MaritalStatus_Married	-0.023659
MaritalStatus_Single	0.072439

	MaritalStatus_Divorced \
Age	0.033120
Attrition	-0.087716
DailyRate	0.037080

DistanceFromHome	-0.005440
Education	-0.002439
EmployeeCount	NaN
EmployeeNumber	-0.025149
EnvironmentSatisfaction	0.016439
Gender	0.046076
HourlyRate	-0.006150
JobInvolvement	0.016815
JobLevel	0.037087
JobSatisfaction	-0.015197
MonthlyIncome	0.032203
MonthlyRate	-0.000227
NumCompaniesWorked	0.040824
OverTime	0.023462
PercentSalaryHike	-0.023478
PerformanceRating	-0.010310
RelationshipSatisfaction	0.006199
StockOptionLevel	0.446285
TotalWorkingYears	0.036291
TrainingTimesLastYear	0.008405
WorkLifeBalance	-0.009080
YearsAtCompany	0.025728
YearsInCurrentRole	0.018532
YearsSinceLastPromotion	-0.005279
YearsWithCurrManager	0.014095
BusinessTravel_Non-Travel	0.057455
BusinessTravel_Travel_Frequently	0.005779
BusinessTravel_Travel_Rarely	-0.043287
Department_Human Resources	0.016037
Department_Research & Development	0.035158
Department_Sales	-0.043451
EducationField_Human Resources	0.012107
EducationField_Life Sciences	-0.002672
EducationField_Marketing	-0.007212
EducationField_Medical	0.013316
EducationField_Other	0.005411
EducationField_Technical Degree	-0.019243
JobRole_Healthcare Representative	0.027897
JobRole_Human Resources	0.021541
JobRole_Laboratory Technician	-0.011224
JobRole_Manager	0.001997
JobRole_Manufacturing Director	0.020543
JobRole_Research Director	0.037524
JobRole_Research Scientist	-0.012115
JobRole_Sales Executive	-0.013853
JobRole_Sales Representative	-0.052890
MaritalStatus_Divorced	1.000000
MaritalStatus_Married	-0.491506
MaritalStatus_Single	-0.366691

MaritalStatus_Single	MaritalStatus_Married	
Age	0.083919	-
0.119185		
Attrition	-0.090984	
0.175419		
DailyRate	0.040035	-
0.075835		
DistanceFromHome	0.030232	-
0.027445		
Education	-0.001865	
0.004168		
EmployeeCount	NaN	
NaN		
EmployeeNumber	0.053933	-
0.035189		
EnvironmentSatisfaction	-0.022180	
0.009035		
Gender	-0.007804	-
0.032752		
HourlyRate	0.036432	-
0.033436		
JobInvolvement	0.028324	-
0.045253		
JobLevel	0.050547	-
0.087072		
JobSatisfaction	-0.010315	
0.024571		
MonthlyIncome	0.056767	-
0.089361		
MonthlyRate	-0.034689	
0.037260		
NumCompaniesWorked	-0.016142	-
0.019161		
Overtime	-0.013502	-
0.006498		
PercentSalaryHike	0.020895	-
0.001386		
PerformanceRating	0.009585	-
0.001045		
RelationshipSatisfaction	-0.043382	
0.040817		
StockOptionLevel	0.225574	-
0.638957		
TotalWorkingYears	0.053512	-
0.089529		
TrainingTimesLastYear	-0.029602	
0.024129		
WorkLifeBalance	-0.006388	
0.014921		

YearsAtCompany 0.070935	0.044925	-
YearsInCurrentRole 0.086486	0.065488	-
YearsSinceLastPromotion 0.053090	0.054102	-
YearsWithCurrManager 0.047793	0.032972	-
BusinessTravel_Non-Travel 0.004622	-0.043635	-
BusinessTravel_Travel_Frequently 0.027734	-0.030785	
BusinessTravel_Travel_Rarely 0.020808	0.055613	-
Department_Human Resources 0.051443	0.034767	-
Department_Research & Development 0.009990	-0.019997	-
Department_Sales 0.033002	0.005378	
EducationField_Human Resources 0.072051	0.057339	-
EducationField_Life Sciences 0.021469	-0.017866	
EducationField_Marketing 0.013323	0.018491	-
EducationField_Medical 0.004249	-0.007139	-
EducationField_Other 0.004972	-0.009171	
EducationField_Technical Degree 0.014265	0.002710	
JobRole_Healthcare Representative 0.030126	0.004913	-
JobRole_Human Resources 0.052320	0.030995	-
JobRole_Laboratory Technician 0.019873	-0.009233	
JobRole_Manager 0.055176	0.049982	-
JobRole_Manufacturing Director 0.021331	0.002819	-
JobRole_Research Director 0.042299	0.008271	-
JobRole_Research Scientist 0.053522	-0.039987	
JobRole_Sales Executive 0.006210	0.005751	
JobRole_Sales Representative 0.072439	-0.023659	

```

MaritalStatus_Divorced          -0.491506      -
0.366691
MaritalStatus_Married            1.000000      -
0.629981
MaritalStatus_Single            -0.629981
1.000000

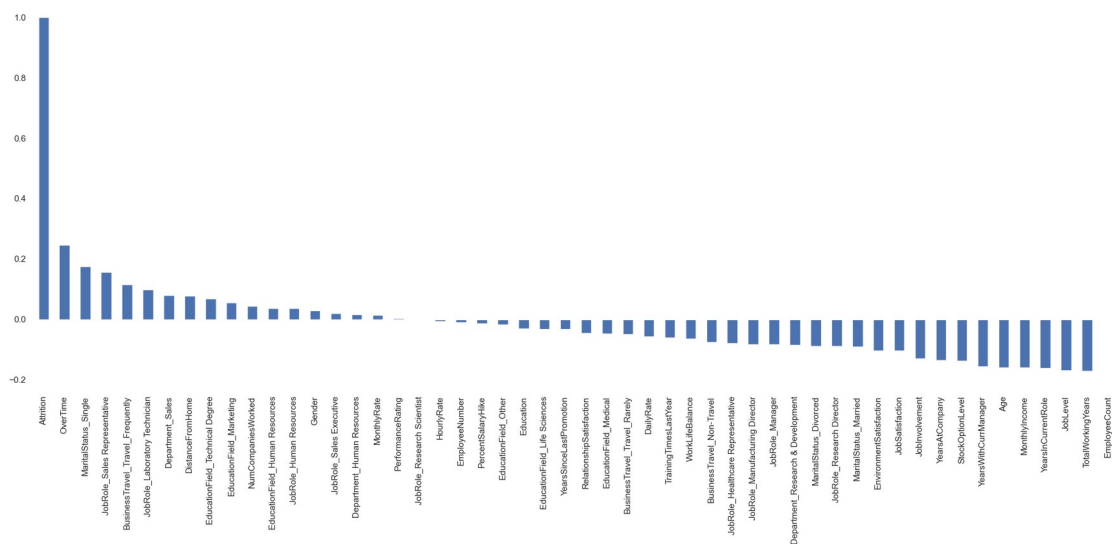
```

[52 rows x 52 columns]

```

sns.set(rc={"axes.facecolor":"white","figure.facecolor":"white"})
plt.figure(figsize=(20,10))
correlations['Attrition'].sort_values(ascending =
False).plot(kind='bar');

```



Dễ nhận thấy ngay là 'EmployeeCount' hoàn toàn không có quan hệ với 'Attrition'.

```
df.drop(['EmployeeCount'],axis=1,inplace=True)
```

```
correlations.drop(['EmployeeCount'],axis=1,inplace=True)
```

```
correlations.drop(['EmployeeCount'],axis=0,inplace=True)
```

### Cách 3:

- Sử dụng correlations giữa các biến độc lập với nhau:
  - Hai biến độc lập khi có giá trị correlation càng cao thì chứng tỏ chúng càng mang thông tin giống nhau cho ngữ cảnh của bài toán.
  - Do đó khi hai biến độc lập có correlation cao thì có thể chọn một trong hai để áp dụng vào việc huấn luyện mô hình.

```

threshold=0.7
cols=list(correlations.columns)
cols_at_index=list(correlations.index)
feature1, feature2, correlation=[], [], []
for i in range(len(cols)):

```

```

    for j in range(len(correlations)):
        if correlations[cols[i]][j]>=threshold and
correlations[cols[i]][j]<1 and cols_at_index[j] not in feature1:
            feature1.append(cols[i])
            feature2.append(cols_at_index[j])
            correlation.append(correlations[cols[i]][j])
new_df=pd.DataFrame({'feature1':[feature1[i] for i in
range(len(feature1))],
                    'feature2':[feature2[i] for i in
range(len(feature2))],
                    'correlation':[correlation[i] for i in
range(len(correlation))]},)
new_df

```

	feature1	feature2	correlation
0	JobLevel	MonthlyIncome	0.950300
1	JobLevel	TotalWorkingYears	0.782208
2	MonthlyIncome	TotalWorkingYears	0.772893
3	PercentSalaryHike	PerformanceRating	0.773550
4	YearsAtCompany	YearsInCurrentRole	0.758754
5	YearsAtCompany	YearsWithCurrManager	0.769212
6	YearsInCurrentRole	YearsWithCurrManager	0.714365
7	Department_Human Resources	JobRole_Human Resources	0.904983
8	Department_Sales	JobRole_Sales Executive	0.808869

- **TotalWorkingYears, JobLevel và MonthlyIncome:** Có giá trị correlation rất cao. Chọn giữ lại MonthlyIncome.
- **PercentSalaryHike và PerformanceRating:** Có giá trị correlation là 0.77. Chọn giữ lại PerformanceRating.
- **YearsAtCompany, YearsInCurrentRole, và YearsWithCurrManager:** Có giá trị correlation cao. Chọn giữ lại YearsAtCompany.
- **Department\_Human Resources và JobRole\_Human Resources:** Có giá trị correlation là 0.9. Chọn giữ lại JobRole\_Human Resources.
- **Department\_Sales và JobRole\_Sales Executive:** Có giá trị correlation là 0.8. Chọn giữ lại JobRole\_Sales Executive.

```

df.drop(['TotalWorkingYears', 'JobLevel', 'PercentSalaryHike',
'YearsInCurrentRole',
        'YearsWithCurrManager', 'Department_Human Resources',
'Department_Sales'],axis=1,inplace=True)

```

```
df.head()
```

	Age	Attrition	DailyRate	DistanceFromHome	Education
EmployeeNumber \					
0	41	1	1102	1	2
1					
1	49	0	279	8	1
2					
2	37	1	1373	2	2

4							
3	33	0	1392	3	4		
5							
4	27	0	591	2	1		
7							

	EnvironmentSatisfaction	Gender	HourlyRate	JobInvolvement	...	\
0		2	0	94	3	...
1		3	1	61	2	...
2		4	1	92	2	...
3		4	0	56	3	...
4		1	1	40	3	...

	JobRole_Laboratory Technician	JobRole_Manager	\
0		0	0
1		0	0
2		1	0
3		0	0
4		1	0

	JobRole_Manufacturing Director	JobRole_Research Director	\
0		0	0
1		0	0
2		0	0
3		0	0
4		0	0

	JobRole_Research Scientist	JobRole_Sales Executive	\
0		0	1
1		1	0
2		0	0
3		1	0
4		0	0

	JobRole_Sales Representative	MaritalStatus_Divorced	\
0		0	0
1		0	0
2		0	0
3		0	0
4		0	0

	MaritalStatus_Married	MaritalStatus_Single	
0		0	1
1		1	0
2		0	1
3		1	0
4		1	0

[5 rows x 44 columns]



### 3. Xử lý các giá trị thiếu

Thống kê số lượng giá trị thiếu của mỗi cột

```
df.isnull().sum()
```

Age	0
Attrition	0
DailyRate	0
DistanceFromHome	0
Education	0
EmployeeNumber	0
EnvironmentSatisfaction	0
Gender	0
HourlyRate	0
JobInvolvement	0
JobSatisfaction	0
MonthlyIncome	0
MonthlyRate	0
NumCompaniesWorked	0
OverTime	0
PerformanceRating	0
RelationshipSatisfaction	0
StockOptionLevel	0
TrainingTimesLastYear	0
WorkLifeBalance	0
YearsAtCompany	0
YearsSinceLastPromotion	0
BusinessTravel_Non-Travel	0
BusinessTravel_Travel_Frequently	0
BusinessTravel_Travel_Rarely	0
Department_Research & Development	0
EducationField_Human Resources	0
EducationField_Life Sciences	0
EducationField_Marketing	0
EducationField_Medical	0
EducationField_Other	0
EducationField_Technical Degree	0
JobRole_Healthcare Representative	0
JobRole_Human Resources	0
JobRole_Laboratory Technician	0
JobRole_Manager	0
JobRole_Manufacturing Director	0
JobRole_Research Director	0
JobRole_Research Scientist	0
JobRole_Sales Executive	0
JobRole_Sales Representative	0
MaritalStatus_Divorced	0
MaritalStatus_Married	0
MaritalStatus_Single	0
dtype: int64	

Bộ dữ liệu không tồn tại bất kì giá trị thiếu nào trong tất cả các cột thuộc tính. Vậy có thể bỏ qua bước này.

#### 4. Feature Scaling

- Khi khoảng giá trị giữa 2 thuộc tính quá cách xa nhau thì việc mô hình hóa cũng như trực quan mối quan hệ có thể gặp khó khăn, do đó phải thực hiện kĩ thuật 'Feature Scaling' hay viết hóa là 'Co giãn thuộc tính'.
- Có 3 phương pháp feature scaling chính là:
  - Standardisation (Chính quy hóa): Làm cho tập dữ liệu có trung bình là 0 và độ lệch chuẩn là 1 và được áp dụng cho hầu hết các trường hợp cần feature scaling.
  - Normalisation (Tiêu chuẩn hóa): Làm cho các giá trị trong tập dữ liệu thuộc đoạn  $[0, 1]$  và được áp dụng nếu tập dữ liệu tuân theo phân phối chuẩn.
  - MinMax Scaler: Đưa các giá trị về khoảng giữa 2 giá trị min và max trong miền giá trị của thuộc tính, có thể là đoạn  $[-1, 0]$ ,  $[0, 1]$ ,  $[-1, 1]$ ,...
- Trong bài này nhóm chọn phương pháp Standardisation để scaling khoảng giá trị của thuộc tính về khoảng gần hơn với giá trị của tập y.
- **Nhóm sẽ không cài đặt ở bước này mà sẽ tích hợp việc feature scaling vào Pipeline của thư viện sklearn trong bước xây dựng mô hình về sau.**

[⬆️ Back to Table of Contents ⬆️](#)

### III. Xây dựng mô hình học máy

#### 1. Logistic Regression cho Phân loại nhị phân

##### Mô tả bài toán:

- Mục tiêu của bài toán phân loại nhị phân là dự đoán xác suất thuộc về một trong hai lớp cần phân lớp của một biến phụ thuộc dựa vào các biến độc lập (hay còn gọi là thuộc tính).
- Trong bộ dữ liệu này chúng ta sẽ thử dự đoán **xác suất thuộc về lớp 0 hoặc 1 của biến Attrition dựa vào các thuộc tính độc lập khác của một người nhân viên.**

##### Cost Function:

- Bắt đầu với việc xây dựng một hypothesis tương tự như bài toán linear regression:  $h_{\theta}(x) = \theta_0 + \theta_1 x$  với  $\theta_i$  ( $i=0,1$ ) là các tham số (parameters) của công thức hồi quy và  $\theta_0$  còn được gọi là hệ số tự do.
- Tuy nhiên sử dụng đường thẳng tuyến tính là không phù hợp cho mục tiêu của bài toán khi giá trị dự đoán cần thuộc vào một trong hai lớp là 0 hoặc 1, nhưng với một đường thẳng tuyến tính thì  $h_{\theta}(x)$  có thể lớn hơn 1 và nhỏ hơn 0.
- Lúc này cần 1 giải pháp để giá trị dự đoán có thể nằm trong khoảng 0 đến 1 => truyền hypothesis qua một hàm sigmoid mà giá trị trả về của một hàm sigmoid nằm trong khoảng 0 đến 1. Đặt  $z = h_{\theta}(x)$

$$\text{sigmoid}(z) = \frac{1}{1 + e^{-z}}$$

(Hình ảnh minh họa cho hàm sigmoid sẽ được trình bày kĩ trong file báo cáo của nhóm).

- Lúc này, ta có giá trị dự đoán  $\hat{y} = \text{sigmoid}(z)$  nằm trong khoảng từ 0 đến 1 hay nói cách khác đây chính là giá trị xác suất mà  $\hat{y}$  thuộc về một trong hai lớp với điều kiện cho trước là các thuộc tính đầu vào khác.
- Thông thường, quy ước giá trị trả về của hàm sigmoid biểu thị cho xác suất thuộc lớp 1 của  $\hat{y}$ :  $\hat{y} = \text{sigmoid}(z) = P(y=1|x) \Rightarrow P(y=0|x) = 1 - \hat{y} = 1 - P(y=1|x)$ .
- Vậy làm thế nào có thể quy định lớp mà  $\hat{y}$  thuộc về?
  - Chúng ta sẽ sử dụng một giá trị ngưỡng mà nếu  $\hat{y}$  lớn hơn hoặc bằng ngưỡng này sẽ thuộc về lớp 1, ngược lại thuộc về lớp 0.
  - Ngưỡng giá trị thông thường sẽ là 0.5, điều này tương ứng với việc nếu  $z \geq 0$  thì  $\hat{y}$  thuộc về lớp 1 (do  $\text{sigmoid}(z \geq 0) \geq 0.5$ ).
- Sau khi có được tập các giá trị dự đoán của các mẫu đầu vào, để đánh giá xem hypothesis đã tốt hay chưa chúng ta xây dựng một hàm chi phí để tính toán độ sai lệch giữa giá trị dự đoán và giá trị thực tế.

$$J(\theta) = -\frac{1}{N} \sum_{i=1}^N (y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i))$$

với  $\hat{y}_i = y_{\hat{h}} a t_i$

- Do càng nhiều giá trị dự đoán giống với giá trị thực tế càng tốt nên  $J(\theta)$  có giá trị càng bé càng tốt.

### Gradient Descent:

- Với mục tiêu là cực tiểu hóa hàm chi phí

$$J(\theta) = -\frac{1}{N} \sum_{i=1}^N (y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i))$$

- Thì các tham số  $\theta$  sẽ là những giá trị mà chúng ta cần phải thay đổi để tối ưu hóa Cost Function, và một trong những cách để thực hiện việc này là thuật toán Gradient Descent.
- Thuật toán được thực hiện như sau: **Trong mỗi lần lặp cập nhật một cách đồng thời các tham số  $\theta_j$  theo công thức như sau:**

$$\theta_j = \theta_j - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

Trong đó: alpha là 'learning rate' giúp việc học được tối ưu hơn.

## 2. Sử dụng Pipeline và Các độ đo được dùng để đánh giá mô hình

### Pipeline:

- Pipeline là một công cụ giúp kết hợp nhiều bước xử lý dữ liệu và huấn luyện mô hình thành một quy trình hoàn chỉnh.

- Các bước để thực hiện xây dựng một mô hình học máy sẽ được xếp tuần tự trong một đối tượng Pipeline (có thể xem như một đường ống để dẫn lần lượt đi qua các bước).
- Pipeline giúp tiết kiệm thời gian và tối ưu hóa quá trình huấn luyện mô hình.

### Các độ đo:

- Các độ đo được nhóm sử dụng trong bài này sẽ là:
  - Accuracy score.
  - Precision.
  - Recall.
  - F1-score.

**(Hình ảnh về pipeline và định nghĩa các độ đo được nhóm trình bày trong file báo cáo đính kèm).**

### 3. Cài đặt

***Lý do cụ thể cho các bước thực hiện dưới đây sẽ được nhóm giải thích một cách chi tiết trong file báo cáo đi kèm.***

**Tạo tập các thuộc tính đầu vào và tập các biến mục tiêu từ bộ dữ liệu ban đầu.**

```
X=np.array(df.drop(['Attrition'],axis=1))
y=np.array(df['Attrition'])
```

X

```
array([[ 41, 1102,    1, ...,    0,    0,    1],
       [ 49,  279,    8, ...,    0,    1,    0],
       [ 37, 1373,    2, ...,    0,    0,    1],
       ...,
       [ 27,  155,    4, ...,    0,    1,    0],
       [ 49, 1023,    2, ...,    0,    1,    0],
       [ 34,  628,    8, ...,    0,    1,    0]], dtype=int64)
```

y

```
array([1, 0, 1, ..., 0, 0, 0])
```

**Chia tập dữ liệu thành tập huấn luyện và tập kiểm tra:**

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    random_state=0, test_size=0.2)
```

```
print(X_train.shape)
print(X_test.shape)
```

```
(1176, 43)
(294, 43)
```

**Cài đặt mô hình:**

```
pipe.fit(X_train, y_train)
pipe.named_steps['classifier'].get_params()
```

```
{'C': 1.0,  
 'class_weight': {0: 1, 1: 1},  
 'dual': False,  
 'fit_intercept': True,  
 'intercept_scaling': 1,  
 'l1_ratio': None,  
 'max_iter': 10000,  
 'multi_class': 'auto',  
 'n_jobs': None,  
 'penalty': 'l2',  
 'random_state': None,  
 'solver': 'liblinear',  
 'tol': 0.0001,  
 'verbose': 0,  
 'warm_start': False}
```

- Đầu tiên tạo một thành phần 'scaler' thực hiện việc chuẩn hóa dữ liệu như đã đề cập ở mục cuối của phần tiền xử lý dữ liệu. Thành phần này sử dụng lớp StandardScaler() được cung cấp bởi sklearn để thực hiện chuẩn hóa giá trị của từng cột trong X theo phương pháp Standardisation (Chính quy hóa).
- Tiếp theo là tạo một bộ phân lớp 'classifier' để thực hiện việc phân loại các mẫu đầu vào về một trong hai lớp của bài toán phân loại nhị phân. Bộ phân lớp này áp dụng thuật toán Logistic Regression mà nhóm đã giới thiệu trước đó (Các tham số trong lớp LogisticRegression sẽ được giải thích trong file báo cáo đi kèm).

```
predictions=pipe.predict(X_test)
predictions
```

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

```

0,      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0,      1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0,      0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,      0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,      0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,      0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,      0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0,
0,      0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0,      0, 0, 0, 0, 0, 1, 0, 0])

```

**Đánh giá mô hình:** Trước tiên xem xét giá trị của accuracy score.

```

acc_score=accuracy_score(predictions, y_test)
print(acc_score)

```

0.8775510204081632

Accuracy Score đạt khá cao nhưng liệu đã đủ tốt cho việc đánh giá mức độ hiệu quả của mô hình?

Hãy cùng xem xét thêm về các độ đo khác.

```

report=classification_report(predictions, y_test)
print(report)

```

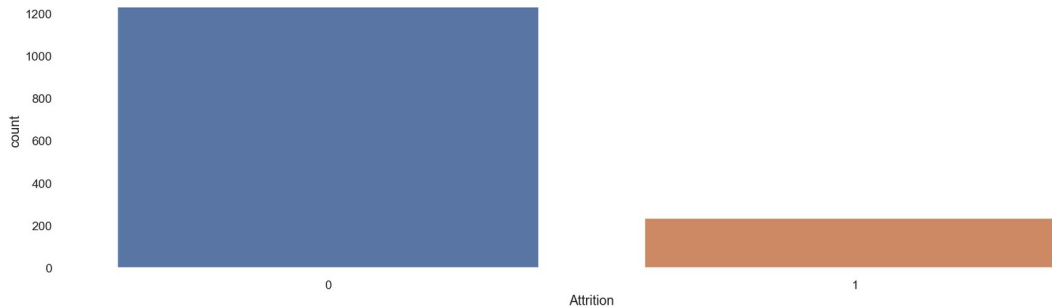
	precision	recall	f1-score	support
0	0.97	0.89	0.93	267
1	0.41	0.74	0.53	27
accuracy			0.88	294
macro avg	0.69	0.82	0.73	294
weighted avg	0.92	0.88	0.89	294

Đúng là đừng nên vội tin những điều màu hồng trước mắt 😊

- Các độ đo khác như Precision, Recall và F1-score đều rất thấp ở lớp 1 hơn so với ở lớp 0, chứng tỏ mô hình dự đoán chưa tốt cho các trường dự đoán mẫu thuộc thực tế thuộc về lớp 1 (hay nói trong ngữ cảnh này là các mẫu thuộc về lớp tiêu cực).
- Và điều này làm chúng ta có thể nghĩ đến trường hợp mất cân bằng dữ liệu (imbalanced data).

## Kiểm tra tính cân bằng của dữ liệu:

```
sns.set(rc={"axes.facecolor":"white","figure.facecolor":"white"})
sns.set_context("poster",font_scale = .7)
plt.subplots(figsize=(20,6))
sns.countplot(data=df,x=df['Attrition']);
```



- Rõ ràng xảy ra hiện tượng mất cân bằng giữa hai lớp 0 và 1 khi số lượng các phần tử thuộc lớp 0 trong bộ dữ liệu này nhiều gấp 6 lần số lượng các phần tử thuộc lớp 1.
- Sự mất cân bằng nghiêm trọng này ảnh hưởng rất lớn đến khả năng dự đoán chính xác của mô hình.

## Mất cân bằng dữ liệu và cách giải quyết:

- Dữ liệu bị mất cân bằng hiểu sự phân bố các mẫu trên các lớp chênh lệch nhau quá lớn dẫn tới việc mô hình chỉ tập trung học những đặc trưng của lớp có số lượng mẫu là chiếm đa số. Làm cho việc dự đoán của mô hình xảy ra tình trạng thiên vị, mất đi tính tổng quát cho dữ liệu thực tế sau này.
- Có nhiều cách xử lý việc mất cân bằng dữ liệu như: Thu thập thêm dữ liệu, các thuật toán tăng mẫu dữ liệu (Oversampling), các thuật toán giảm mẫu dữ liệu (Undersampling), sử dụng trọng số để phạt mô hình cho các giá trị dự đoán của các lớp,....
- Trong phần này nhóm sẽ thực hiện hai phương pháp:
  - Sử dụng **trọng số** để phạt mô hình.
  - Ứng dụng **Thuật toán SMOTE** thuộc nhóm Oversampling.

*(Chi tiết về mặt lý thuyết của mất cân bằng dữ liệu, tác hại của mất cân bằng dữ liệu, phương pháp sử dụng trọng số, phương pháp sử dụng thuật toán SMOTE sẽ được nhóm trình bày trong file báo cáo đính kèm).*

## Sử dụng trọng số để tăng việc phạt mô hình nếu dự đoán sai lớp:

```
pipe2 = Pipeline([
    ('scaler', StandardScaler()),
    ('classifier', LogisticRegression(solver="liblinear",penalty="l2",
class_weight={0:1,1:2},max_iter=10000))
])
```

```

pipe2.fit(X_train, y_train)

Pipeline(steps=[('scaler', StandardScaler()),
                 ('classifier',
                  LogisticRegression(class_weight={0: 1, 1: 2},
                                     max_iter=10000,
                                     solver='liblinear'))])

predictions2=pipe2.predict(X_test)

acc_score2=accuracy_score(predictions2, y_test)
print(acc_score2)

0.8741496598639455

report2=classification_report(predictions2, y_test)
print(report2)

```

	precision	recall	f1-score	support
0	0.93	0.92	0.92	246
1	0.61	0.62	0.62	48
accuracy			0.87	294
macro avg	0.77	0.77	0.77	294
weighted avg	0.88	0.87	0.87	294

- Với việc sử dụng một trọng số lớn hơn cho lớp 1m đã giúp cải thiện precision và f1-score của lớp 1 một cách đáng kể, cho thấy tính tổng quát của mô hình đã được tăng lên.
- Tuy nhiên các độ đo này vẫn chưa đạt được đến giá trị thật sự tốt lắm cho một bài toán phân loại.
- Và nhóm cũng đã thực nghiệm với nhiều cặp trọng số khác nhau nhưng kết quả thu được tốt nhất và là {0:1, 1:2}.

### Áp dụng thuật toán SMOTE:

```

from imblearn.over_sampling import SMOTE
X,y=SMOTE(sampling_strategy=1, random_state=0).fit_resample(X, y)

sns.set(rc={"axes.facecolor":"white", "figure.facecolor":"white"})
sns.set_context("poster", font_scale = .7)
plt.subplots(figsize=(20,6))
sns.countplot(x=y);

```





### Chia tập dữ liệu thành tập huấn luyện và tập kiểm tra:

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
random_state=0, test_size=0.2)
```

```
print(X_train.shape)
print(X_test.shape)
```

```
(1972, 43)
```

```
(494, 43)
```

### Cài đặt mô hình:

```
pipe3 = Pipeline([
    ('scaler', StandardScaler()),
    ('classifier', LogisticRegression(solver="liblinear",penalty="l2",
class_weight={0:1,1:1},max_iter=10000))
])
```

```
pipe3.fit(X_train, y_train)
pipe3.named_steps['classifier'].get_params()
```

```
{'C': 1.0,
 'class_weight': {0: 1, 1: 1},
 'dual': False,
 'fit_intercept': True,
 'intercept_scaling': 1,
 'l1_ratio': None,
 'max_iter': 10000,
 'multi_class': 'auto',
 'n_jobs': None,
 'penalty': 'l2',
 'random_state': None,
 'solver': 'liblinear',
 'tol': 0.0001,
 'verbose': 0,
 'warm_start': False}
```

*#dự đoán*

```
predictions3=pipe3.predict(X_test)
```

```
predictions3
```

```
array([[0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0,
0,
      0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0,
1,
      0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0,
1,
      0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0,
0,
      1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1,
1,
      0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0,
0,
      1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0,
1,
      0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0,
1,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0,
1,
      0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0,
1,
      0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0,
1,
      0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1,
0,
      1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0,
0,
      0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0,
1,
      0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1,
1,
      0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0,
0,
      0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0,
1,
      0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1,
0,
      0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0,
0,
      0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0,
      1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1,
1,
      1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1,
0,
      0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1]])
```

## Đánh giá mô hình:

```
#Trước tiên xem xét accuracy score cu'a mô hình
acc_score3=accuracy_score(predictions3, y_test)
print(acc_score3)
```

0.9048582995951417

```
#Xem xét chi tiết hơn đến các giá trị metric khác như F1-score hay
Recall score
report3=classification_report(predictions3, y_test)
print(report3)
```

	precision	recall	f1-score	support
0	0.98	0.84	0.91	268
1	0.84	0.98	0.90	226
accuracy			0.90	494
macro avg	0.91	0.91	0.90	494
weighted avg	0.92	0.90	0.90	494

- Tất cả các độ đo đều đạt ở mức 90% hoặc hơn, cho thấy sự hiệu quả của thuật toán SMOTE trên bộ dữ liệu này.

[⬆️ Back to Table of Contents ⬆️](#)

## V. Tổng kết

- Nhóm sẽ không đi sâu vào việc rút ra ý nghĩa nào từ bộ dữ liệu thông qua việc xây dựng mô hình học máy. Điều này được thực hiện ở một giai đoạn khác chính là EDA bộ dữ liệu.
- Thông qua mô hình học máy nhóm chú trọng vào các khía cạnh:
  - Tiền xử lý dữ liệu.
  - Rút trích đặc trưng.
  - Xây dựng mô hình học máy hiệu quả.
- Nhóm đã phát hiện vấn đề về mất cân bằng dữ liệu xảy ra trên bộ dữ liệu này, một điều gây nhiều khó khăn cho quá trình xây dựng các bộ phân lớp.
- Thực hiện các phương pháp để giải quyết vấn đề mất cân bằng dữ liệu và xây dựng mô hình học máy để sử dụng như bộ phân lớp với các độ đo đạt kết quả rất khả quan:
  - Accuracy score: 90%.
  - Precision macro average: 91%.
  - Recall macro average: 91%.
  - F1-Score macro average: 90%.



[⬆️ Back to Table of Contents ⬆️](#)