**TRƯỜNG ĐẠI HỌC KINH TẾ ĐÀ NẴNG - ĐẠI HỌC ĐÀ NẴNG**

**------\*\*\*------**

**A blue and white logo

Description automatically generated**

**BÁO CÁO GIỮA KÌ**

**PHÂN TÍCH DỮ LIỆU BẰNG PYTHON**

**ĐỀ TÀI: PHÂN TÍCH DỰ BÁO NGHỈ VIỆC CỦA CÔNG TY IBM**

**BẰNG PHƯƠNG PHÁP HỒI QUY LOGISTICS**

|  |  |  |
| --- | --- | --- |
| GVHD | : | Lê Diên Tuấn |
| Nhóm | : | 2 |
| Lớp | : | 47K21.2 |
| Thành viên | : | Trần Văn Lợi (100%)  0357985692 |
|  | Trần Diệu Thùy (100%) |
|  | Trịnh Thị Kim Oanh (100%) |
|  | Phan Thanh Tiến (100%)  Nguyễn Anh Tuấn (100%) |

*Đà Nẵng, ngày 6 tháng 4 năm 2**024*

MỤC LỤC

[**I.** **Giới thiệu tổng quan** 2](#_Toc163317179)

[**II.** **Cơ sở lý thuyết** 2](#_Toc163317180)

[1. Khái niệm về hồi quy (Regression) 2](#_Toc163317181)

[a. Khái niệm 2](#_Toc163317182)

[b. Các bước cơ bản trong phân tích hồi quy 2](#_Toc163317183)

[c. Các thành phần trong hồi quy 3](#_Toc163317184)

[2. Hồi quy Logistic 3](#_Toc163317185)

[a. Khái niệm 3](#_Toc163317186)

[b. Cách thức hoạt động 4](#_Toc163317187)

[**III.** **Triển khai dự án Python** 7](#_Toc163317188)

[1. Framework dự án 7](#_Toc163317189)

[2. Mô tả dữ liệu 7](#_Toc163317190)

[a. Các thuộc tính tập dữ liệu 7](#_Toc163317191)

[b. Build model 10](#_Toc163317192)

[**IV. Kết luận và giải pháp** 32](#_Toc163317193)

# **Giới thiệu tổng quan**

* Trong xã hội ngày càng phát triển về mọi lĩnh vực, công nghệ và trí tuệ nhân tạo được áp dụng mạnh mẽ trong quản lý một doanh nghiệp, đặc biệt các tập đoàn lớn trên thế giới. Việc dự báo tỷ lệ nghỉ việc của nhân viên là một nhiệm vụ vô cùng quan trọng nhưng còn là một thách thức đối với các doanh nghiệp, nhất là trong thời điểm hiện tại kinh tế khủng hoảng và môi trường đầy biến động.
* Việc dự báo xu hướng nhân viên có nghỉ việc hay không đều phụ thuộc vào những thuật toán, công cụ phân tích với độ chính xác cao. Trong số đó, thuật toán phân tích hồi quy logistic là một công cụ rất đáng sử dụng trong việc phân tích và dự đoán sự biến động của các biến độc lập trong nhiều lĩnh vực khác nhau. Với sự hỗ trợ về mặt thuật toán nhanh và tiện lợi, những lợi ích vô vàn của mô hình này đem lại xu hướng, dự báo được những nhân viên có khả năng nghỉ việc trong tương lai. Từ đó, các nhà quản lý có một cái nhìn tổng quát nhất về tình trạng doanh nghiệp và đưa ra những quyết định sáng suốt về mặt nhân sự.
* Mục tiêu của nghiên cứu đề tài hồi quy logistic để phân tích các yếu tố ảnh hưởng đến tỷ lệ nghỉ việc của nhân viên công ty IBM. Bằng phương pháp này, chúng em sẽ tiến hành phân tích dữ liệu và cung cấp thông tin chính xác và đáng tin cậy về xu hướng tỷ lệ nghỉ việc để hỗ trợ đưa ra quyết định cho doanh nghiệp. Để thực hiện dự án hiệu quả tối đa, chúng em cùng nhau xây dựng hệ thống phân tích như sau:
* Tiền xử lý dữ liệu
* Xây dựng mô hình hồi quy logistic
* Dự báo và đánh giá mô hình có thật sự hiệu quả hay là không

# **Cơ sở lý thuyết**

## Khái niệm về hồi quy (Regression)

### Khái niệm

* Phân tích hồi quy là một phương pháp thống kê được sử dụng để nghiên cứu mối quan hệ giữa một biến phụ thuộc (biến mục tiêu) và một hoặc nhiều biến độc lập (biến dự đoán). Mục tiêu chính của phân tích hồi quy là dự đoán hoặc mô tả biến phụ thuộc dựa trên các biến độc lập.
* Trong phân tích hồi quy, biến phụ thuộc là biến chúng ta muốn dự đoán hoặc hiểu rõ hơn. Các biến độc lập (còn gọi là biến dự đoán) là những yếu tố mà chúng ta cho là có ảnh hưởng đến biến phụ thuộc.
* Phân tích hồi quy thường được thực hiện bằng cách xác định một mô hình hồi quy để mô tả mối quan hệ giữa các biến.

### Các bước cơ bản trong phân tích hồi quy

* Đầu tiên cần xác định loại hồi quy: Xác định loại hồi quy phù hợp cho bài toán của bạn, chẳng hạn như hồi quy tuyến tính đa biến, hồi quy logistic, và nhiều loại hồi quy khác.
* Tiếp theo là tiến hành thu thập dữ liệu: Thu thập dữ liệu về biến phụ thuộc và biến độc lập. Bạn cần thu thập dữ liệu về các biến độc lập và biến phụ thuộc. Đảm bảo rằng dữ liệu được thu thập một cách chính xác và đầy đủ.
* Xử lý làm sạch dữ liệu : Chuẩn bị dữ liệu bằng cách xử lý các giá trị thiếu, loại bỏ nhiễu, chuẩn hóa dữ liệu nếu cần thiết.
* Tiếp theo là xây dựng mô hình: Xác định một mô hình hồi quy phù hợp để dự đoán hoặc mô tả biến phụ thuộc dựa trên các biến độc lập.
* Sau đó lựa chọn biến: Xác định các biến độc lập mà bạn muốn sử dụng trong mô hình hồi quy. Cân nhắc cả các biến định tính (categorical) và biến liên tục (continuous).
* Sau đó Phân tích dữ liệu: Thực hiện phân tích thống kê cơ bản để hiểu rõ dữ liệu của bạn, bao gồm việc tính mean, median, độ lệch chuẩn và xem xét sự phân phối của các biến.
* Ước tính tham số: Tìm các tham số của mô hình hồi quy bằng cách sử dụng các phương pháp thống kê như phương pháp bình phương tối thiểu.
* Kiểm tra mô hình: Đánh giá độ phù hợp của mô hình bằng cách sử dụng các chỉ số như R-squared và kiểm tra giả thuyết thống kê xem các biến độc lập có tác động đến biến phụ thuộc không.
* Bước cuối cùng là Dự đoán và diễn giải: Sử dụng mô hình để dự đoán giá trị biến phụ thuộc dựa trên các giá trị của biến độc lập. Đồng thời, diễn giải ý nghĩa thống kê và thực tiễn của các tham số mô hình.

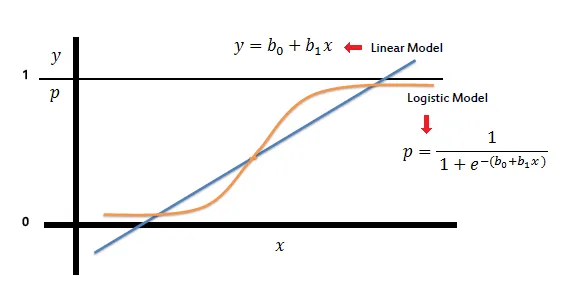
### Các thành phần trong hồi quy

* Biến độc lập (Independent variable): là một biến mà bạn nghiên cứu hoặc quan sát để xem liệu nó có ảnh hưởng đến biến phụ thuộc hay không. Ví dụ, nếu bạn đang nghiên cứu tác động của giá cả (biến độc lập) đối với dự định mua iPhone (biến phụ thuộc), thì giá cả là biến độc lập. Bạn quan sát các giá trị khác nhau của giá cả để xem liệu chúng có tác động đến dự định mua iPhone hay không.
* Biến độc lập có thể là các yếu tố, điều kiện, thuộc tính, hoặc thông số mà bạn tùy ý thay đổi để xem cách chúng ảnh hưởng đến kết quả hoặc biến phụ thuộc mà bạn quan tâm. Trong phân tích hồi quy, biến độc lập thường được sử dụng để dự đoán hoặc - giải thích biến phụ thuộc.
* Biến phụ thuộc (Dependent variable): là biến mà bạn quan tâm đến và muốn nghiên cứu hoặc dự đoán tùy theo sự thay đổi của biến độc lập. Trong phân tích thống kê, biến phụ thuộc là biến mà bạn cố gắng hiểu rõ hơn thông qua việc tìm hiểu mối quan hệ với biến độc lập.
* Ví dụ, nếu bạn đang nghiên cứu tác động của giá cả (biến độc lập) đối với dự định mua Iphone (biến phụ thuộc), thì dự định mua Iphone là biến phụ thuộc. Bạn quan tâm đến cách mà giá cả thay đổi có thể ảnh hưởng đến dự định mua Iphone.
* Biến phụ thuộc thường là kết quả, hiện tượng, hoặc sự biến đổi mà bạn mong muốn nghiên cứu hoặc dự đoán. Trong phân tích hồi quy, biến phụ thuộc thường được sử dụng để đo lường và phân tích để hiểu rõ cách biến độc lập ảnh hưởng đến nó.

## Hồi quy Logistic

### Khái niệm

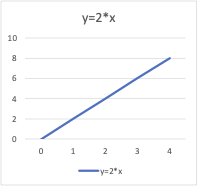
Hồi quy Logistic là một mô hình thống kê được sử dụng để phân loại nhị phân, tức dự đoán một đối tượng thuộc vào một trong hai nhóm. Hồi quy Logistic làm việc dựa trên nguyên tắc của hàm sigmoid – một hàm phi tuyến tự chuyển đầu vào của nó thành xác suất thuộc về một trong hai lớp nhị phân.



### Cách thức hoạt động

#### Phương trình

* Trong toán học, phương trình cho ta mối quan hệ giữa hai biến: x và y. Bạn có thể sử dụng các phương trình hoặc hàm này để vẽ đồ thị theo trục x và trục y bằng cách nhập các giá trị khác nhau của x và y. Ví dụ: nếu bạn vẽ đồ thị cho hàm y = 2\*x, bạn sẽ có một đường thẳng như hình dưới đây. Do đó hàm này còn được gọi là hàm tuyến tính.

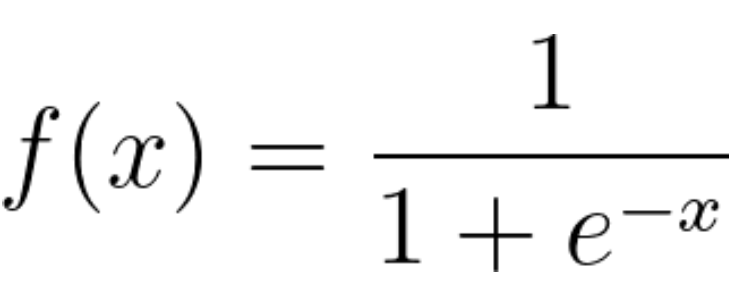


#### Biến

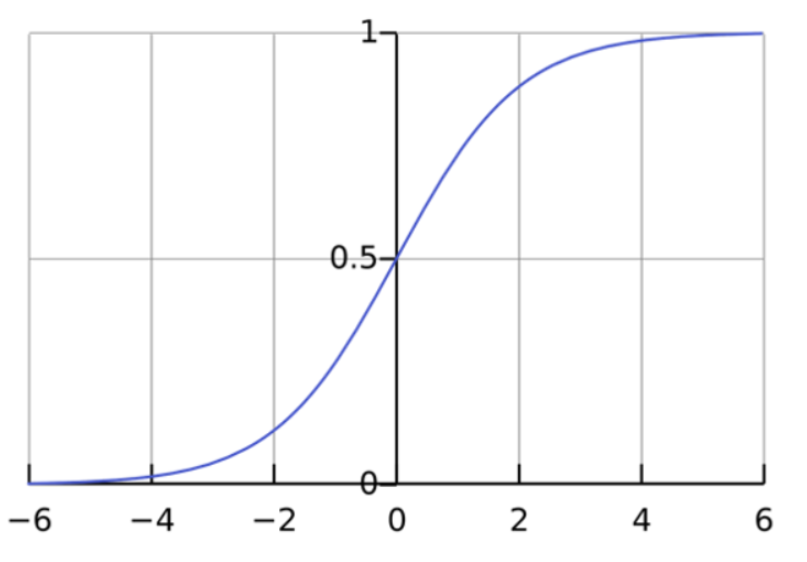
* Trong thống kê, biến là các yếu tố dữ liệu hoặc thuộc tính có giá trị khác nhau. Bất kỳ phân tích nào cũng có một số biến nhất định là biến độc lập hoặc biến giải thích. Những thuộc tính này là nguyên nhân của một kết quả. Các biến khác là biến phụ thuộc hoặc biến đáp ứng; giá trị của chúng phụ thuộc vào các biến độc lập. Nhìn chung, hồi quy logistic khám phá cách các biến độc lập ảnh hưởng đến một biến phụ thuộc bằng cách xem xét các giá trị dữ liệu lịch sử của cả hai biến.
* Trong ví dụ ở trên của chúng tôi, x được gọi là biến độc lập, biến dự đoán hoặc biến giải thích vì nó có một giá trị đã xác định. Y được gọi là biến phụ thuộc, biến kết quả hoặc biến đáp ứng vì giá trị của nó không xác định.

#### Hàm hồi quy logistic

* Hồi quy logistic là một mô hình thống kê sử dụng hàm logistic, hay hàm logit trong toán học làm phương trình giữa x và y. Hàm logit ánh xạ y làm hàm sigmoid của x.



* Nếu vẽ phương trình hồi quy logistic này, bạn sẽ có một đường cong hình chữ S như hình dưới đây.



* Như bạn có thể thấy, hàm logit chỉ trả về các giá trị giữa 0 và 1 cho biến phụ thuộc, dù giá trị của biến độc lập là gì. Đây là cách hồi quy logistic ước tính giá trị của biến phụ thuộc. Phương pháp hồi quy logistic cũng lập mô hình phương trình giữa nhiều biến độc lập và một biến phụ thuộc.

#### Phân tích hồi quy logistic với nhiều biến độc lập

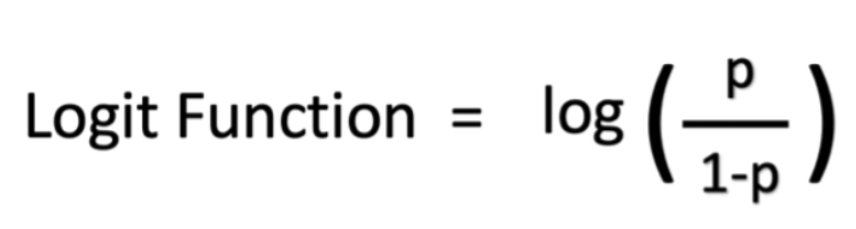
* Trong nhiều trường hợp, nhiều biến giải thích ảnh hưởng đến giá trị của biến phụ thuộc. Để lập mô hình các tập dữ liệu đầu vào như vậy, công thức hồi quy logistic phải giả định mối quan hệ tuyến tính giữa các biến độc lập khác nhau. Bạn có thể sửa đổi hàm sigmoid và tính toán biến đầu ra cuối cùng như sau

**y = f(β0 + β1x1 + β2x2+… βnxn)**

* Ký hiệu β đại diện cho hệ số hồi quy. Mô hình logit có thể đảo ngược tính toán các giá trị hệ số này khi bạn cho nó một tập dữ liệu thực nghiệm đủ lớn có các giá trị đã xác định của cả hai biến phụ thuộc và biến độc lập.

#### Log của tỷ số odds

* Mô hình logistic cũng có thể xác định tỷ số thành công trên thất bại hay log của tỷ số odds.
* Ví dụ: nếu bạn đang chơi poker với bạn bè và thắng bốn ván trên mười ván, tỷ số chiến thắng của bạn là 4/6 và đó là tỷ số thành công trên thất bại của bạn. Mặt khác, xác suất thắng là 4/10.
* Về mặt toán học, tỉ số odds về mặt xác suất của bạn là p/(1 - p) và log của tỷ số odds là log(p/(1 - p)). Bạn có thể biểu diễn hàm logistic bằng log của tỷ số odds như hình dưới đây:



#### Phân tích hồi quy logistic có những loại nào?

* Có ba cách tiếp cận phân tích hồi quy logistic dựa trên kết quả của biến phụ thuộc.

##### \*\*Hồi quy logistic nhị phân

* Hồi quy logistic nhị phân phù hợp với các vấn đề phân lớp nhị phân chỉ có hai kết quả có thể xảy ra. Biến phụ thuộc chỉ có thể có hai giá trị, chẳng hạn như có và không hoặc 0 và 1.
  + Dù hàm logistic tính toán một phạm vi giá trị giữa 0 và 1, mô hình hồi quy nhị phân vẫn sẽ làm tròn kết quả đến các giá trị gần nhất. Nói chung, kết quả dưới 0,5 sẽ được làm tròn thành 0 và kết quả trên 0,5 sẽ được làm tròn thành 1, do đó hàm logistic trả về một kết quả nhị phân.

##### \*\*Hồi quy logistic đa thức

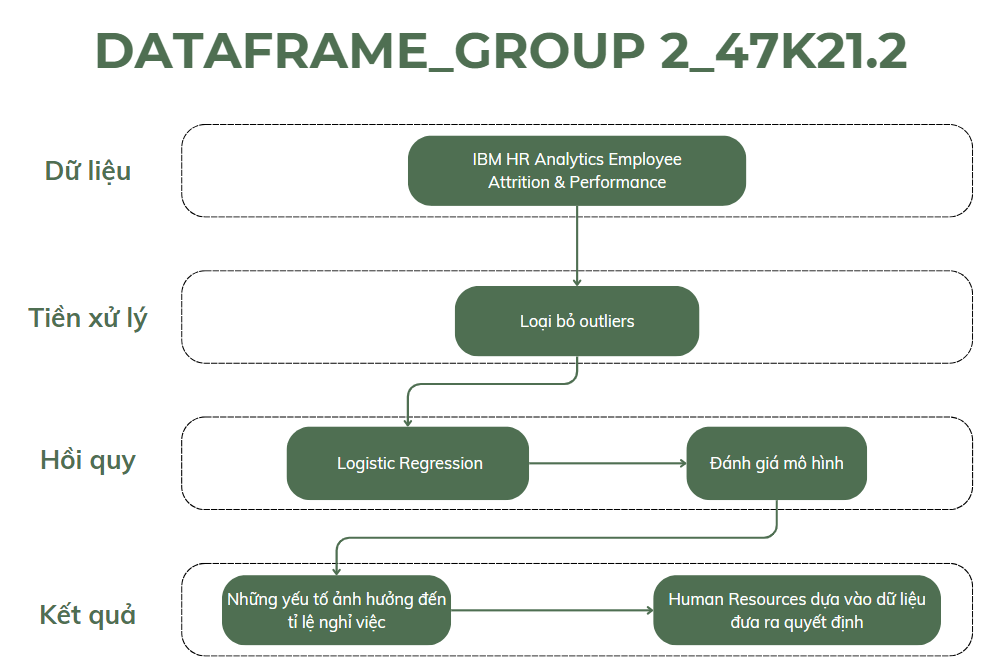
* Hồi quy đa thức có thể phân tích các vấn đề có một số kết quả có thể xảy ra, miễn là số kết quả hữu hạn. Ví dụ: kỹ thuật này có thể dự đoán xem giá nhà sẽ tăng 25%, 50%, 75% hay 100% dựa trên dữ liệu dân số, nhưng sẽ không thể dự đoán được giá trị chính xác của một ngôi nhà.
* Hồi quy logistic đa thức hoạt động bằng cách ánh xạ các giá trị kết quả cho các giá trị khác nhau giữa 0 và 1. Hàm logistic có thể trả về một khoảng dữ liệu liên tục như 0,1, 0,11, 0,12, v.v., do đó hồi quy đa thức cùng nhóm đầu ra đến các giá trị gần nhất có thể có.

##### \*\*Hồi quy logistic thứ tự

* Hồi quy logistic thứ tự, hay mô hình logit có thứ tự, là một loại hồi quy đa thức đặc biệt cho các vấn đề trong đó các số đại diện cho các bậc chứ không phải là giá trị thực tế. Ví dụ: bạn sẽ sử dụng hồi quy thứ tự để dự đoán đáp án cho câu hỏi khảo sát yêu cầu khách hàng đánh giá dịch vụ của bạn ở mức kém, ổn, tốt hoặc xuất sắc dựa trên một giá trị số, chẳng hạn như số lượng mặt hàng họ mua từ bạn trong năm.

# **Triển khai dự án Python**

## Framework dự án

****

## Mô tả dữ liệu

### Các thuộc tính tập dữ liệu

* Đây là bộ dữ liệu được lấy từ [tại đây](https://www.kaggle.com/datasets/pavansubhasht/ibm-hr-analytics-attrition-dataset?fbclid=IwAR0tXC4K565k0s5kf35qWTOfHhOAw_Bc3hUkq_8Vt0qQWeGp7cOlpVPPGj8) trên Kaggle. Bộ dữ liệu bao gồm:

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Tên cột | Ý nghĩa | Kiểu dữ liệu |
| 1 | Age | Tuổi của nhân viên | Integer |
| 2 | Attrition | Sự tiêu hao nhân viên | Boolean |
| 3 | BusinessTravel | Tần suất một nhân viên đi công tác vì mục đích kinh doanh | String |
| 4 | DailyRate | Tiền lương hàng ngày của một nhân viên | Integer |
| 5 | Department | Phòng nhân viên | String |
| 6 | Distance from home | Khoảng cách từ nhà đến cơ quan ở KM's | Integer |
| 7 | Education | Trình độ chuyên môn của nhân viên | Integer |
| 8 | EducationField | Stream of Education | String |
| 9 | EmployeeCount | Số nhân viên | Integer |
| 10 | EmployeeNumber | Mã nhân viên | Integer |
| 11 | Environmentsatisfaction | Mức độ thỏa mãn môi trường làm việc | Integer |
| 12 | Gender | Giới tính | String |
| 13 | HourlyRate | Mức lương theo giờ của nhân viên | Integer |
| 14 | JobInvolvement | Sự tích cực trong công việc | Integer |
| 15 | JobLevel | Cấp độ công việc | Integer |
| 16 | JobRole | Tên công việc của nhân viên | String |
| 17 | JobSatisfaction | Mức độ hài lòng với công việc | Integer |
| 18 | MaritalStatus | Tình trạng hôn nhân | String |
| 19 | MonthlyIncome | Thu nhập hàng tháng | Integer |
| 20 | MonthlyRate | Lương hàng tháng của nhân viên | Integer |
| 21 | NumberCompaniesWorked | Số công ty đã làm việc | Integer |
| 22 | Over18 | Trên 18 tuổi | Boolean |
| 23 | OverTime | Nhân viên làm việc tăng ca | String |
| 24 | PercentSalaryHike | Tăng lương | Integer |
| 25 | PerformanceRating | Tỷ lệ hiệu suất | Integer |
| 26 | Relationship satisfaction | Sự hài lòng về mối quan hệ | Integer |
| 27 | StandardHours | Giờ làm việc tiêu chuẩn mỗi tuần | Integer |
| 28 | StockOptionLevel | Cấp độ tùy chọn chứng khoán | Integer |
| 29 | TotalWorkingYears | Tổng số năm làm việc | Integer |
| 30 | TrainingTimeLastYear | Thời gian đào tạo | Integer |
| 31 | Work life balance | Cân bằng cuộc sống công việc | Integer |
| 32 | YearsAtCompany | Tổng số năm làm việc tại công ty hiện tại | Integer |
| 33 | YearsInCurrentRole | Tổng số năm ở vai trò hiện tại | Integer |
| 34 | YearsSinceLastPromotion | Năm kể từ lần thăng chức cuối cùng | Integer |
| 35 | YearsWorkedUnderCurrentManager | Số năm làm việc dưới sự quản lý hiện tại | Integer |

s

* Kích thước dữ liệu gồm:
* 1470 hàng
* 35 cột

### Build model

#### Mô tả dữ liệu

# Khai báo các thư viện sẽ được sử dụng

import pandas as pd

import matplotlib.pyplot as plt

from google.colab import drive

import seaborn as sns

import numpy as np

%matplotlib inline

# Lưu file vào trong thư mục Visualization ở google drive sau đó tiến hành mount drive để đọc dữ liệu

folder\_ds='/content/drive/MyDrive/Visualization/'

drive.mount('/content/drive', force\_remount=True)

# đọc dữ liệu và lưu vào biến df

df=pd.read\_csv(folder\_ds+'WA\_Fn-UseC\_-HR-Employee-Attrition.csv')

#cho biết kiểu dữ liệu của df

print(df.dtypes)

#cho biết kích thước của df

print(df.shape)

#Hiển thị các tên cột của df

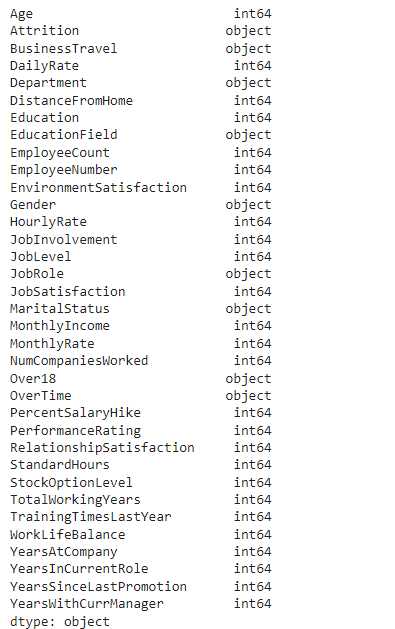
print(df.columns)

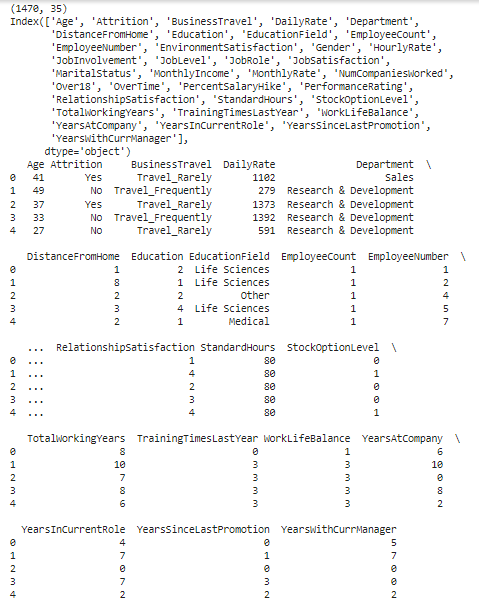
#Hiển thị thông tin 5 dòng đầu của df

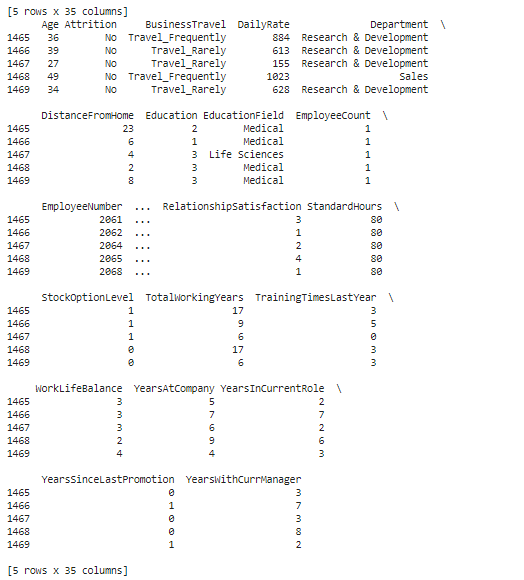
print(df.head(5))

#Hiển thị thông tin 5 dòng cuối của df

print(df.tail(5))

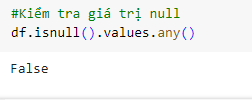






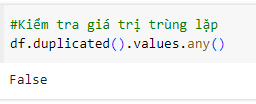
#Kiểm tra giá trị null

df.isnull().values.any()



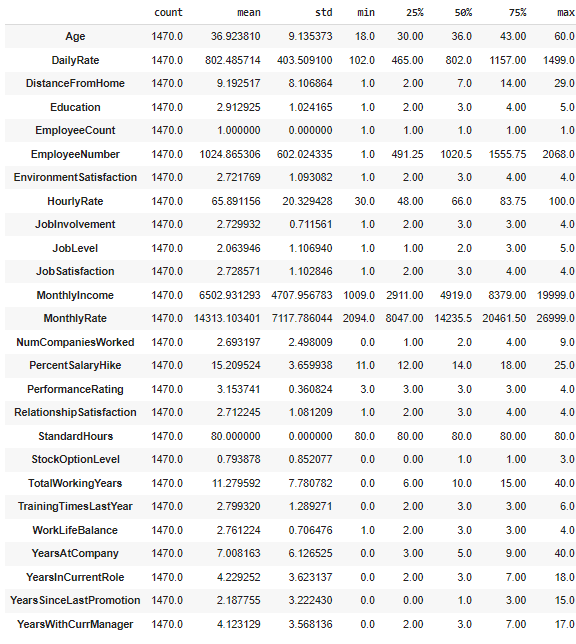
#Kiểm tra giá trị trùng lặp

df.duplicated().values.any()



#Hiển thị thông tin mô tả của df

df.describe().T



#### Tiền xử lý dữ liệu

##### Data Cleansing

#loại bỏ các thuộc tính thừa và đồng nhất

for key in df.keys():

if len(df[key].unique())==1:

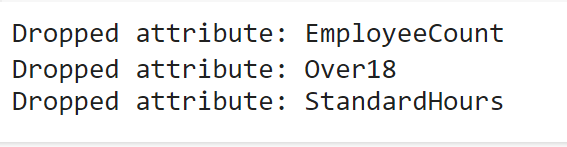
df.drop(key,axis=1,inplace=True)

print('Dropped attribute: %s'%key)

# Ở vòng lặp trên chúng ta sẽ kiểm tra xem số lượng giá trị duy nhất của cột đó có bằng 1 không. Nếu bằng 1 thì đồng nghĩa với việc cột đó là duy nhất ngược lại thì chúng ta sẽ loại bỏ các thuộc tính không duy nhất

# Bỏ đi một số thuộc tính dư thừa không được sử dụng

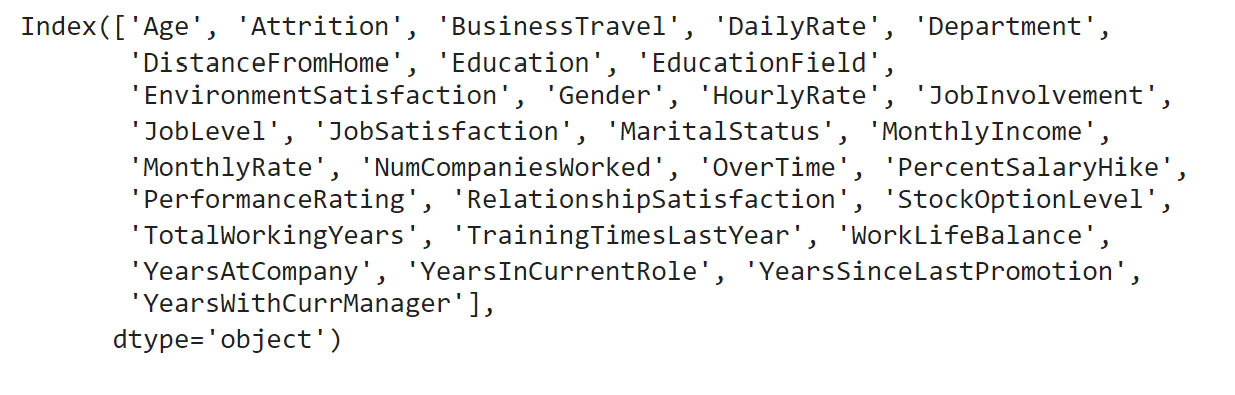
df.drop({'EmployeeNumber','JobRole'}, axis=1, inplace=True)



# Xác định ordinal variables

ordinal\_vars=['Education','EnvironmentSatisfaction','JobInvolvement','JobSatisfaction','PerformanceRating','RelationshipSatisfaction','WorkLifeBalance']

df.keys()



# Phân biệt thuộc tính để kiểm tra tương quan gồm các biến về thời gian và các biến về lương

time\_params=['Age','TotalWorkingYears','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager']

salary\_params=['DailyRate','HourlyRate','MonthlyIncome','MonthlyRate','PercentSalaryHike','StockOptionLevel']

# Tìm mối tương quan giữa các tham số thời gian

corr\_df=df[time\_params].corr().round(2)

# Xác định giá trị tối đa cho phép của tương quan là 0.4 (biến max\_corr).

max\_corr = 0.4

plt.figure(figsize=(8,5))

#Tạo một mask để che đi nửa phía trên của ma trận tương quan (vì ma trận tương quan là ma trận đối xứng).

mask = np.zeros\_like(corr\_df)

mask[np.triu\_indices\_from(mask)] = True

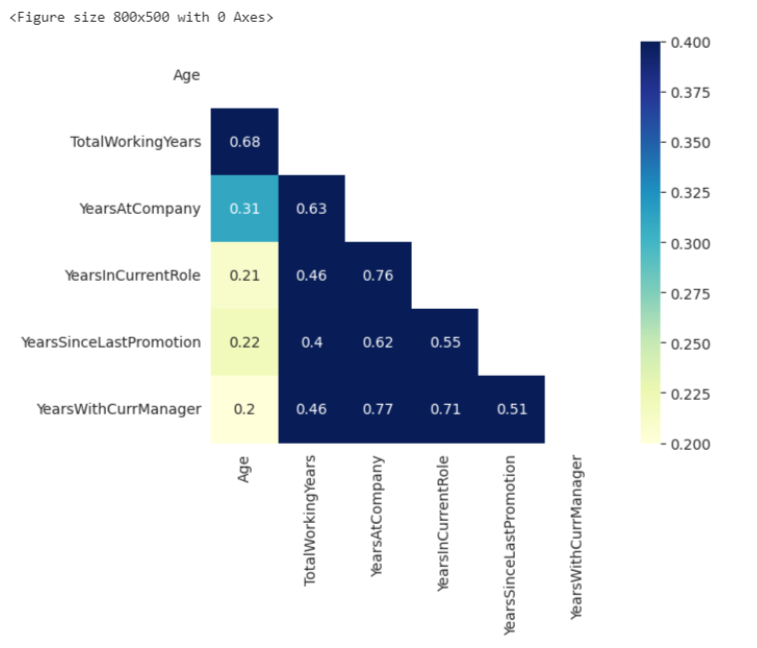
mask[np.triu\_indices\_from(mask)] = True

#Sử dụng sns.heatmap để vẽ heatmap với các thiết lập như sau: Mask được áp dụng để che đi nửa phía trên của heatmap.

with sns.axes\_style("white"):

f, ax = plt.subplots(figsize=(7, 5))

ax = sns.heatmap(corr\_df, mask=mask,vmax=max\_corr, square=True, annot=True, cmap="YlGnBu")



Ở kết quả đầu ra ở trên, chúng ta có thể thấy rằng tất cả các tham số đều có mối tương quan cao với nhau ngoại trừ 'Age'. Điều này là hoàn toàn hợp lý với bối cảnh. Vì vậy, chúng ta phải loại bỏ tất cả các tham số tương quan ngoại trừ một tham số (YearsAtCompany) để nó đại diện cho phần còn lại của các tham số tương quan cùng với 'Age'.

# Loại bỏ các tính năng có mối tương quan cao  
df.drop({'TotalWorkingYears','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager'}, axis=1, inplace=True)

Bây giờ chúng ta sẽ thực hiện phân tích tương tự cho các thông số liên quan đến Mức lương.

# Tìm mối tương quan giữa các thông số Lương

corr\_df=df[salary\_params].corr().round(2)

# Tương quan tối đa cho phép là 0,4

max\_corr = 0.4

plt.figure(figsize=(8,5))

#Tạo một mask để che đi nửa phía trên của ma trận tương quan (vì ma trận tương quan là ma trận đối xứng).

mask = np.zeros\_like(corr\_df)

mask[np.triu\_indices\_from(mask)] = True

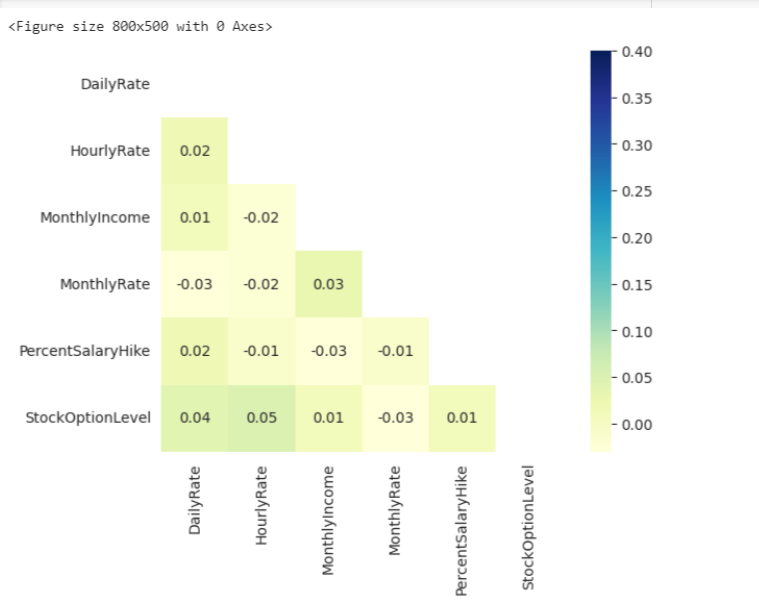
mask[np.triu\_indices\_from(mask)] = True

#Sử dụng sns.heatmap để vẽ heatmap với các thiết lập như sau: Mask được áp dụng để che đi nửa phía trên của heatmap.

with sns.axes\_style('white'):

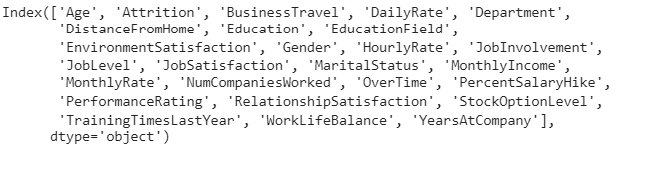
f, ax = plt.subplots(figsize=(7, 5))

ax = sns.heatmap(corr\_df, mask=mask,vmax=max\_corr, square=True, annot=True, cmap='YlGnBu')



#Hàm dùng để trả về danh sách tên cột trong df sau khi đã xóa cột

df.keys()



Ở đây chúng ta có thể thấy rằng chúng ta đã loại bỏ thành công nhiều biến độc lập.

##### Creating Dummies

# Phân tích các biến định tính

#Sử dụng phương thức select\_dtypes('object') để chọn các cột chứa dữ liệu kiểu đối tượng (object) từ DataFrame

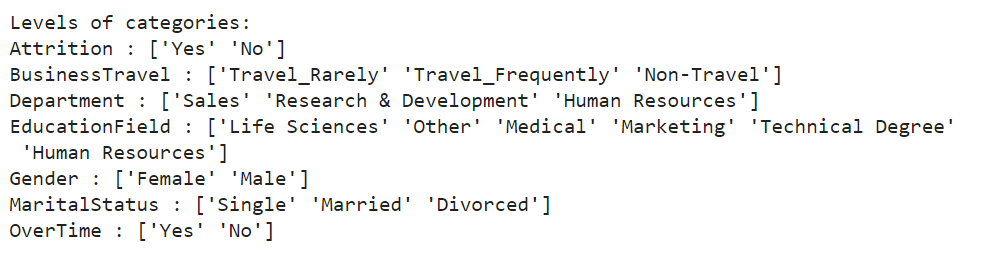
df.select\_dtypes('object').head()

print('Levels of categories: ')

#Dùng vòng lặp for để duyệt qua từng cột dữ liệu định tính được chọn. Trong mỗi vòng lặp, sử dụng phương thức unique() để hiển thị các giá trị duy nhất của cột đó.

for key in df.select\_dtypes('object').keys():

print(key ,':' ,df[key].unique())



Đoạn mã trên là một phần của quy trình tiền xử lý dữ liệu trước khi xây dựng mô hình hồi quy logistic. Khi chúng ta có các biến phân loại, chúng ta cần biến đổi chúng thành các biến giả (dummy variables) để sử dụng trong mô hình hồi quy logistic. Trong trường hợp này, mỗi biến phân loại có nhiều hơn một cấp độ (hoặc loại), và chúng ta muốn tránh rơi vào "bẫy biến giả" bằng cách chỉ tạo ra (n-1) biến giả.

Ví dụ, nếu chúng ta có một biến phân loại là 'Gender' với hai cấp độ là 'Male' và 'Female', chúng ta chỉ cần tạo một biến giả để đại diện cho 'Gender'. Trong trường hợp này, nếu ‘Male’ là 1, thì giới tính của cá nhân là nam giới; nếu là 0, thì giới tính của cá nhân là nữ giới. Tương tự, trong mô hình hồi quy logistic, chúng ta muốn biết tác động của mỗi cấp độ của biến phân loại đối với biến phụ thuộc.

Do đó, chúng ta chỉ cần tạo (n-1) biến giả để tránh việc bị mắc kẹt trong "bẫy biến giả", với một cấp độ được chọn làm cơ sở. Điều này giúp chúng ta hiểu rõ hơn về tác động của từng cấp độ đối với mô hình.

# Lấy biến giả cho các biến định tính

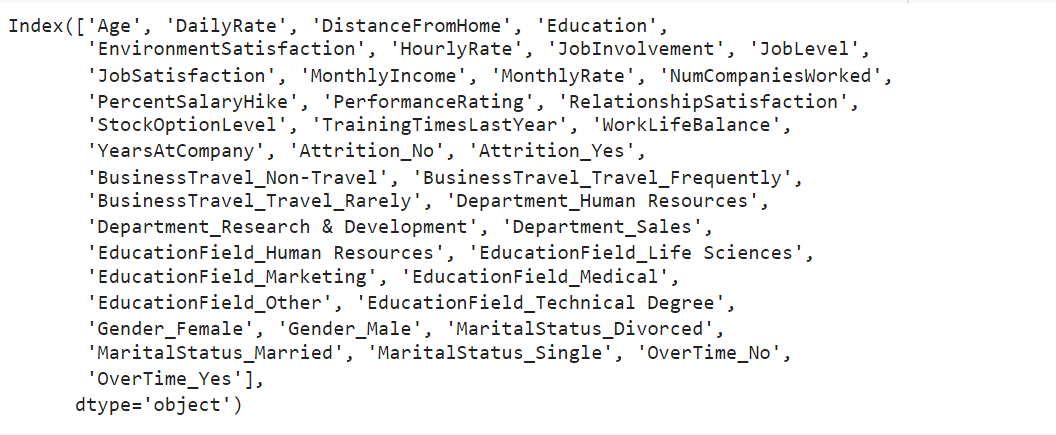
#Sử dụng phương thức get\_dummies() của pandas để tạo các biến giả cho các biến định tính đã chọn Trong đó mỗi giá trị duy nhất của biến định tính được biến đổi thành một biến giả, và các giá trị trong mỗi biến giả là 0 hoặc 1.

df=pd.get\_dummies(df,columns=df.select\_dtypes('object').keys(),

prefix=df.select\_dtypes('object').keys())

#Truyền danh sách các cột cần tạo biến giả vào tham số columns của phương thức get\_dummies(). Sử dụng tham số prefix để thêm tiền tố cho tên của các biến giả tạo ra, nhằm phân biệt chúng với các cột gốc

df.keys()



Mã trên là một bước tiếp theo trong việc xử lý dữ liệu trước khi xây dựng mô hình hồi quy logistic. Chúng ta sử dụng phương pháp get\_dummies từ thư viện pandas để tạo biến giả cho các biến phân loại trong dữ liệu. Mỗi biến phân loại sẽ được chia thành nhiều biến giả tương ứng với từng cấp độ của biến phân loại đó.

Ví dụ, nếu chúng ta có một biến phân loại 'Gender' với hai cấp độ là 'Male' và 'Female', sau khi áp dụng get\_dummies, chúng ta sẽ có hai biến giả: 'Gender\_Male' và 'Gender\_Female'. Nếu cá nhân có giới tính là 'Male', giá trị của 'Gender\_Male' sẽ là 1 và giá trị của 'Gender\_Female' sẽ là 0.

Trong mã trên, chúng ta sử dụng select\_dtypes('object') để chọn các biến có kiểu dữ liệu là 'object', tức là các biến phân loại. Sau đó, chúng ta sử dụng keys() để lấy danh sách các biến phân loại. Cuối cùng, chúng ta sử dụng get\_dummies để tạo các biến giả cho các biến phân loại và đặt tiền tố là tên của từng biến phân loại.

Điều này giúp chúng ta tạo ra một bộ dữ liệu mới với các biến giả đã được tạo, và sau đó chúng ta có thể sử dụng bộ dữ liệu này để xây dựng mô hình hồi quy logistic.

# Xoá base dummy attributes để tránh dummy variable trap

#Dùng phương thức drop() để loại bỏ các cột được chỉ định từ DataFrame df.

df.drop({'Attrition\_No','BusinessTravel\_Non-Travel','Department\_Human Resources','EducationField\_HumanResources','Gender\_Female', 'MaritalStatus\_Single','OverTime\_No'}, axis=1,inplace=True)

# Vi phân cho các biến giả

#cont\_vars dùng để chưa các biến liến tục

cont\_vars=df.select\_dtypes('int').keys()

dummies= df.select\_dtypes('uint8').keys().drop('Attrition\_Yes')

dummies

Loại bỏ biến giả cơ sở: Để tránh bẫy biến giả, chúng ta cần loại bỏ một biến giả cơ sở từ mỗi tập hợp các biến giả tương ứng với mỗi biến phân loại. Trong ví dụ này, chúng ta đã loại bỏ các biến giả cơ sở như 'Attrition\_No', 'BusinessTravel\_Non-Travel', 'Department\_HumanResources', 'EducationField\_HumanResources', 'Gender\_Female', 'MaritalStatus\_Single', và 'OverTime\_No'.

Phân biệt giữa biến liên tục và biến giả: Sau khi đã loại bỏ các biến giả cơ sở, chúng ta cần phân biệt giữa các biến liên tục và biến giả để sử dụng chúng trong việc xây dựng mô hình. Trong đoạn mã này, chúng ta đã tạo hai tập hợp các biến: cont\_vars chứa các biến liên tục và dummies chứa các biến giả sau khi đã loại bỏ biến giả cơ sở 'Attrition\_Yes'.

##### Eliminate Outliers

#Vẽ biểu đồ distplot cho các cột nằm trong Salary\_Params

#Bỏ qua các cảnh báo

import warnings

warnings.filterwarnings("ignore")

def distplot(dataset, columns\_list, rows, cols, suptitle):

fig, axs = plt.subplots(rows, cols, figsize=(16, 8))

# Tiêu đề chung cho toàn bộ biểu đồ được đặt bằng cách sử dụng suptitle().

fig.suptitle, size=16)

# Tiếp theo, một mảng 1D axs được tạo ra từ các trục con được trả về bởi subplots(), và được phẳng hóa bằng cách sử dụng flatten().

axs = axs.flatten()

#Vẽ biểu đồ displot cho các cột trong salary\_params

for i, data in enumerate(salary\_params):

sns.distplot(dataset[data], ax=axs[i])

axs[i].set\_title(data + ', Độ lệch: ' + str(round(dataset[data].skew(axis=0, skipna=True), 2)))

plt.tight\_layout() # Đảm bảo không có biểu đồ nào bị đè lên nhau

#Gọi lại hàm và truyền các tham số đầu vào

distplot(dataset=df, columns\_list=salary\_params, rows=2, cols=3, suptitle='Đồ thị phân phối các cột numeric\_columns')

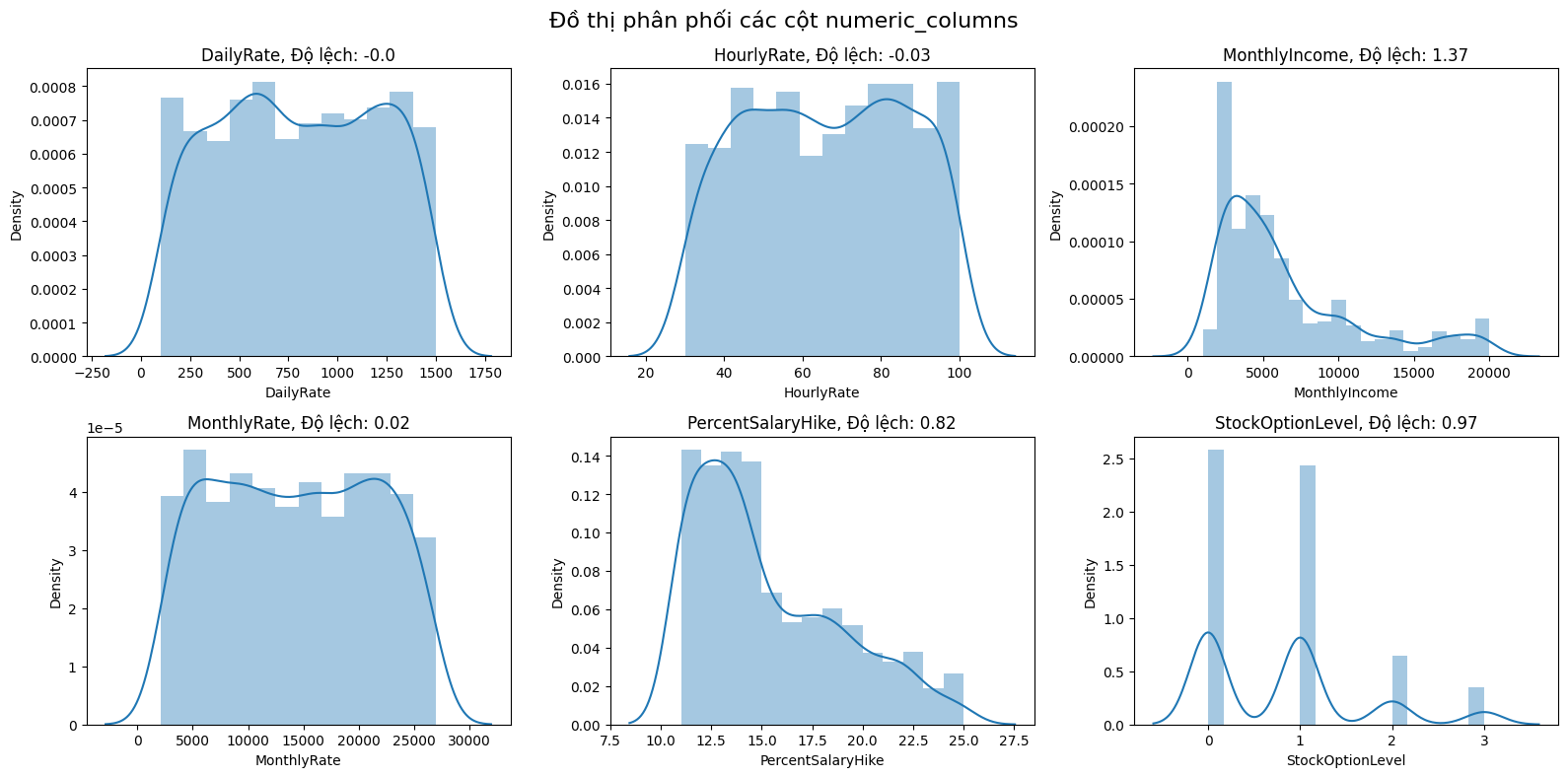
#Hàm dùng để show kết quả

plt.show()

=> Nhận xét đoạn code phía trên:

Với đoạn code trên, chúng ta sử dụng biểu đồ Displot được khai báo trong thư viện Matplotlib dựa trên những yếu tố như dataset, danh sách các thuộc tính, dòng, cột và tiêu đề. Đoạn code xây dựng hàm, cho chạy hết các phần tử trong dataset salary\_params gồm 6 thuộc tính (DailyRate, HourlyRate, MonthlyIncome, MonthlyRate, PercentSalaryHike và StockOptionLevel). Trong vòng lặp, hàm displot đã tạo ra các biểu đồ phân phối bằng cách sử dụng hàm distplot của Seaborn cho cột data trong dataset. Ngoài ra nó cũng đặt tiêu đề của subplot để bao gồm tên cột và độ lệch của dữ liệu.

Sau khi xây dựng hàm thì áp dụng hàm cho dataset trong đoạn code, xuất ra 6 biểu đồ 2 dòng 3 cột. Cuối cùng, kết quả đưa ra 6 biểu đồ cho 6 thuộc tính về đồ thị phân phối các cột trên.



# Vẽ biểu đồ boxplot theo các cột nằm trong Salary\_Params

plt.figure(figsize = (15,25)) # Tạo một hình với kích thước là 15x25 inch.

for idx, i in enumerate(salary\_params):

# Tạo một subplot trong lưới có kích thước 8 dòng và 3 cột, với chỉ mục của subplot là idx + 1

plt.subplot(8, 3, idx + 1)

#Vẽ biểu đồ boxplot bằng thư viện seaborn

sns.boxplot(x = i, data = df)

plt.title(i,backgroundcolor='black',color='white',fontsize=15)

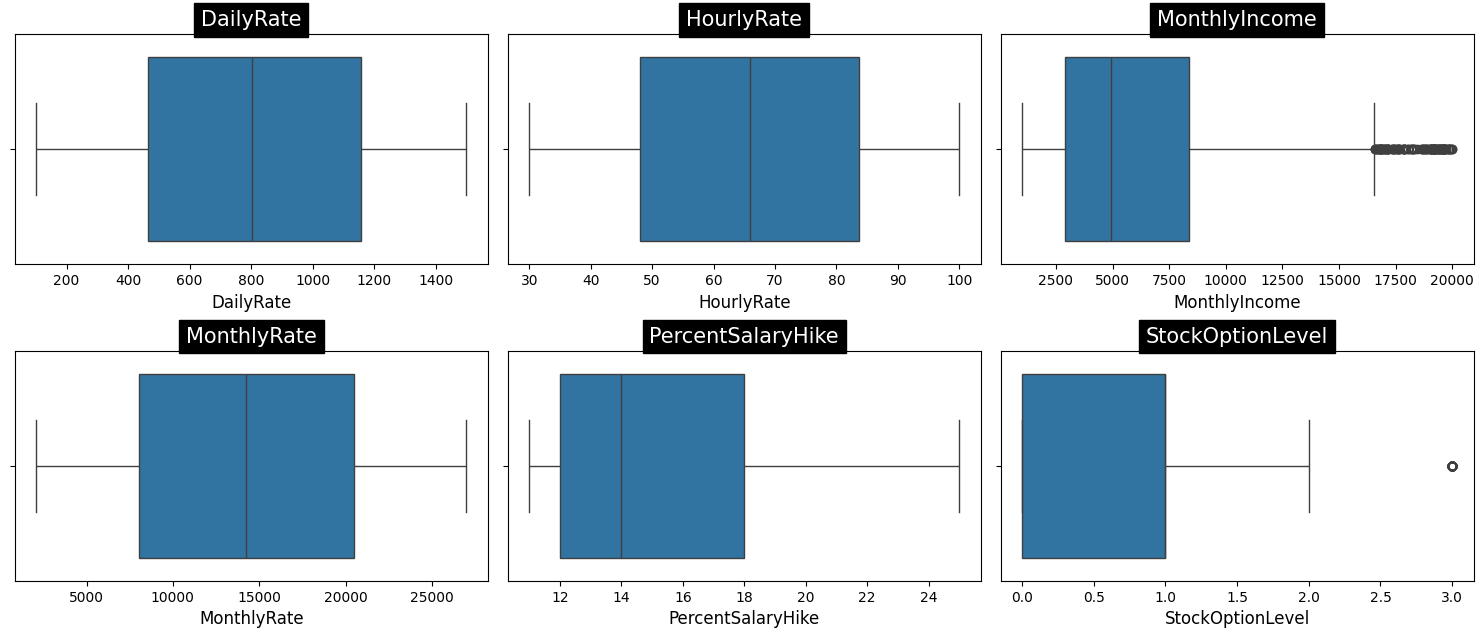
plt.xlabel(i, size = 12)

plt.tight\_layout() #Đảm bảo các biểu đồ không chồng lên nhau

plt.show()

=> Nhận xét đoạn code phía trên:

Sau khi vẽ bảng phân phối của 6 cột thuộc tính của dataset salary\_params, chúng ta có thể thấy được độ lệch của một số thuộc tính từ dataset. Nhận thấy rằng thuộc tính MonthlyIncome có độ lệch giá trị cao nhất giữa trong 6 thuộc tính trên. Với bài toán này sẽ giải quyết bằng cách sử dụng đồ thị Boxplot để có thể phát hiện ra những điểm ngoại lai bất thường có thể ảnh hưởng đến quá trình phân tích. Tại đoạn code trên, chúng ta sử dụng hàm Boxplot của thư viện Seaborn dùng vòng lặp for chạy tất cả các thuộc tính của dataset để in ra 6 biểu đồ thể hiện những điểm outlier của mỗi thuộc tính như thế nào.



#Tính IQR cho các cột trong Salary\_Params

plt.figure(figsize=(15, 25))

drop\_outlier\_col = salary\_params

for idx, i in enumerate(drop\_outlier\_col):

plt.subplot(8, 3, idx + 1)

Q1 = df[i].quantile(0.25)

Q3 = df[i].quantile(0.75)

IQR = Q3 - Q1

lower\_bound = Q1 - 1.5 \* IQR

upper\_bound = Q3 + 1.5 \* IQR

df = df[(df[i] >= lower\_bound) & (df[i] <= upper\_bound)]

#Vẽ biểu đồ boxplot cho Salary\_Params sau khi đã loại bỏ outliers

sns.boxplot(x=i, data=df)

plt.title(i, backgroundcolor='black', color='white', fontsize=15)

plt.xlabel(i, size=12)

plt.tight\_layout()

plt.show()

=> Nhận xét đoạn code:

Sau khi nhận ra những điểm bất thường của một số thuộc tính trong dataset, chúng ta phải xử lý những điểm ngoại lai bằng phương pháp tính độ trải giữa IQR. Tại phương pháp tiền xử lý này, chúng ta sử dụng phân vị thứ 25 (Q1) và phân vị thứ 75 (Q3) và áp dụng công thức như sau:

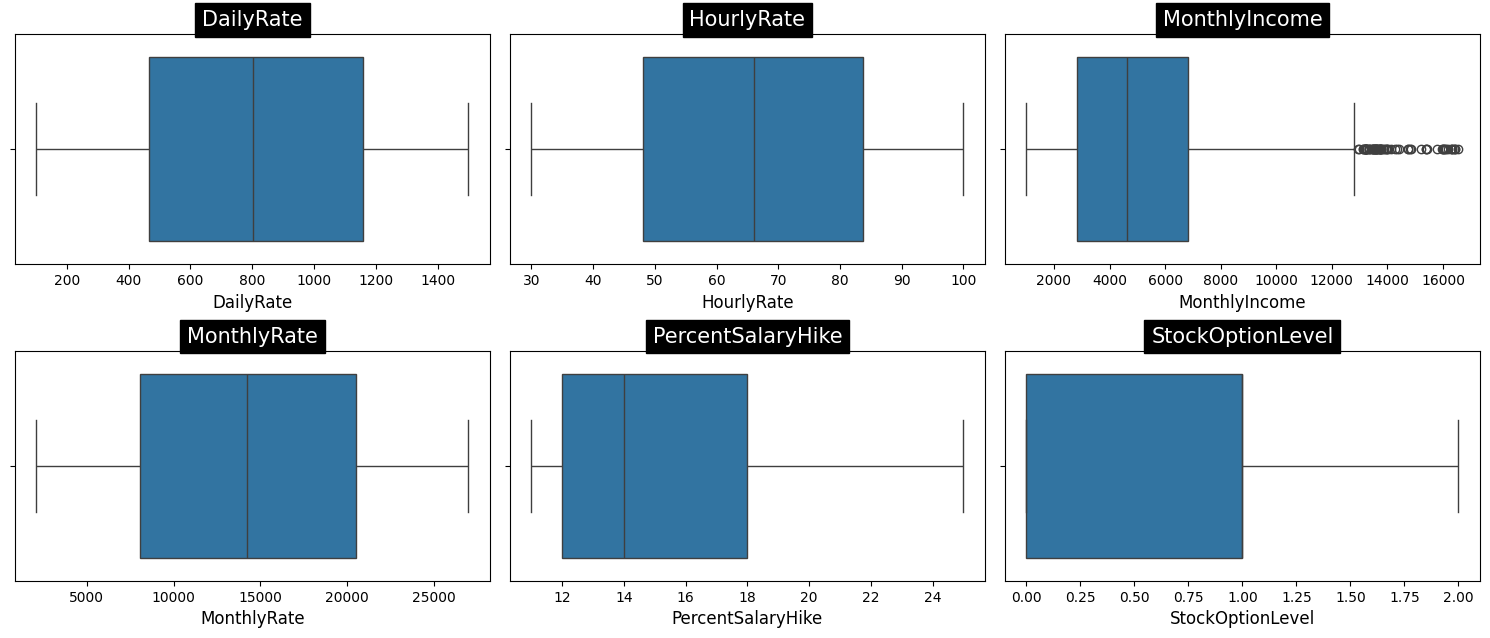
IQR = Q3 - Q1

Giới hạn dưới = Q1 - 1.5\*IQR

Giới hạn trên = Q3 + 1.5\*IQR

Chúng ta lấy những điểm nào vượt qua giới hạn trên và giới hạn dưới thì sẽ loại bỏ ra khỏi dataset để file dữ liệu khi phân tích mang lại tính chính xác, hiệu quả tốt nhất. Với những hướng đi tiền xử lý như vậy thì chúng ta dữ liệu còn 1274 dòng và 26 cột có khả năng ảnh hưởng đến yếu tố tỷ lệ nghỉ việc của nhân viên công ty IBM.





#### Machine Learning Models

import statsmodels.api as sm

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

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

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LogisticRegression

# Xác định X và y

y=df['Attrition\_Yes']

X=df.drop('Attrition\_Yes',axis=1)

# Thêm hằng số hồi quy cho biến độc lập.

X=sm.add\_constant(X)

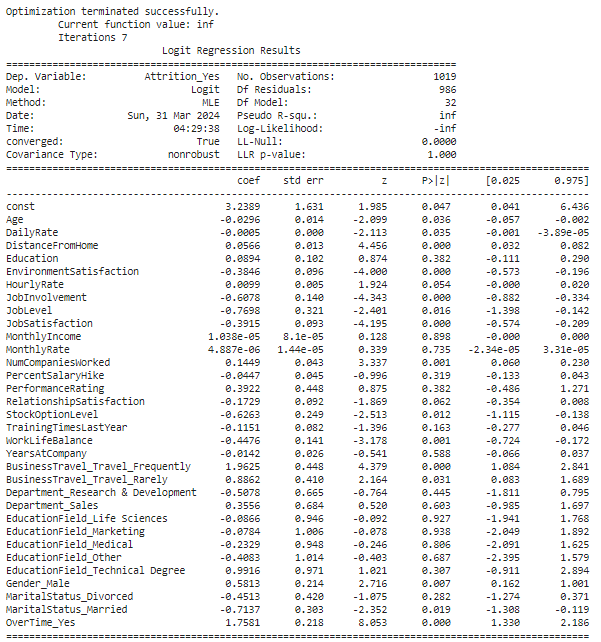
#Tách tập dữ liệu

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)

model=sm.Logit(y\_train,X\_train,random\_state=0)

result=model.fit()

print(result.summary())



Ở trên, nhóm đã tạo ra 2 nhóm X và y, với y là biến phụ thuộc trong bài toán này là cột Attrition và X là các biến độc lập bao gồm tất cả các cột khác ngoài cột Attrition. Sau đó chúng ta phải thêm vào hằng số hồi quy constant cho các biến độc lập.

Sau khi tạo ra X và y, tiếp theo ta cần phải tách bộ dữ liệu ra thành 2 phần là train và test, với train có số dòng bằng 80% số dòng của bộ dữ liệu sau tiền xử lý nhằm mục đích huấn luyện mô hình, và test dùng để thử nghiệm, kiểm tra tính chính xác của mô hình chiếm 20% số dòng còn lại. Random\_state=0 để đảm bảo dữ liệu được ngẫu nhiên là giống nhau sau từng lần chạy.

Sau đó ta tạo ra một mô hình hồi quy logistic (Logit) bằng cách sử dụng statsmodels. Tiêp theo là bắt đầu huấn luyện mô hình với y\_train là biến mục tiêu và X\_train là ma trận các biến độc lập cho huấn luyện.

**Bảng logistic regression model fit:** Bảng thể hiện kết quả cho các biến và mối quan hệ của chúng với biến phụ thuộc attrition

* Hệ số tương quan (Coefficient): Những giá trị này thể hiện thay đổi của tỷ lệ nghỉ việc với mỗi gia tăng đơn vị của từng biến độc lập mà giữ nguyên giá trị của các biến khác. Ví dụ như với tuổi, hệ số tương quan là -0.0296. Điều này có nghĩa là với mỗi năm tuổi được gia tăng thì log odds của attrition giảm 0.0296.
* p-value: Những giá trị này thể hiện ý nghĩa thống kê của từng hệ số tương quan. Nếu p-value thấp (Thông thường là nhỏ hơn 0.05) cho biết rằng mối quan hệ giữa biến phụ thuộc và biến độc lập không phải là do ngẫu nhiên. Trong bảng này, hầu hết p-value là rất thấp, cho biết rằng hầu hết mối quan hệ các biến độc lập và biến có ý nghĩa thống kê cao.
* Những hệ số khác: Bảng này cũng bao gồm standard errors, z-statistics, và confidence intervals, cung cấp nhiều thông tin hơn về độ phù hợp mô hình.

# Lấy số liệu thống kê

stat\_df=pd.DataFrame({'coefficients':result.params,'p-value':result.pvalues,'odds\_ratio': np.exp(result.params)})

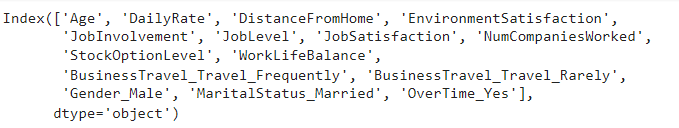
stat\_df

# Điều kiện cho tham số có ý nghĩa

significant\_params=stat\_df[stat\_df['p-value']<=0.05].index

significant\_params= significant\_params.drop('const')

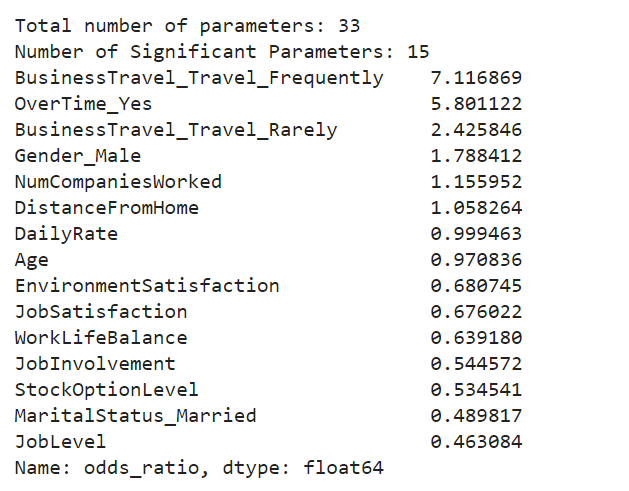
significant\_params



print('Total number of parameters: %s '%len(X.keys()) )

print('Number of Significant Parameters: %s'%(len(significant\_params)))

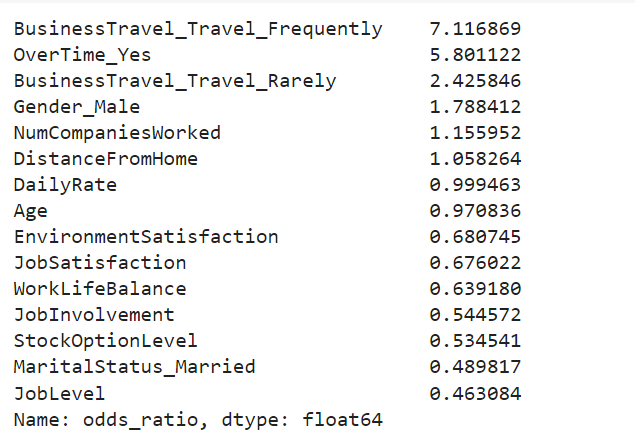
stat\_df.loc[significant\_params].sort\_values('odds\_ratio', ascending=False)['odds\_ratio']



Chúng ta lọc ra những biến giả có ý nghĩa thống kê, tức là những biến có p-value thấp hơn 0.05 ngoại trừ biến const. Sau khi kiểm tra thì từ 33 biến thì còn lại 15 biến p-value<0.05 và có ý nghĩa thống kê.

##### Odds Ratio

stat\_df.loc[significant\_params].sort\_values('odds\_ratio', ascending=False)['odds\_ratio']



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

Ta có thể thấy cột “BusinessTravel\_Travel\_Frequently” có tỷ lệ odds cao nhất là 7.11 cùng với “BusinessTravel\_Travel\_Rarely” tỷ lệ odds là 2.42. Có nghĩa là trong những người ở cột ‘BusinessTravel’, những người đi công tác thường xuyên và hiếm khi có tỷ lệ nghỉ việc cao hơn lần lượt là 7.11 và 2.42 lần so với những người không đi công tác. Tương tự như vậy, những người làm việc thêm giờ có tỷ lệ nghỉ việc cao hơn 580%.

Tương tự như vậy, những nhân viên đã kết hôn có tỷ lệ nghỉ việc 49% thấp hơn so với những nhân viên chưa kết hôn.

Với những biến liên tục như tuổi, mỗi lần tăng một đơn vị sẽ làm giảm tỷ lệ nghỉ việc đi 3%. Cùng với đó, việc tăng job involvement thì tỷ lệ nghỉ việc sẽ gấp 0.54 lần.

##### ROC curve

from sklearn.metrics import roc\_auc\_score, roc\_curve

# Ở đây bạn cần có y\_pred từ mô hình logistic regression của bạn

fpr, tpr, thresholds = roc\_curve(y\_test, y\_pred)

logit\_roc\_auc = roc\_auc\_score(y\_test, y\_pred)

ROC là một đường cong biểu diễn xác suất được sử dụng để trực quan hóa hiệu suất của bộ phân loại nhị phân. Đường cong ROC dùng để đánh giá các kết quả của một [dự đoán](http://vi.wikipedia.org/w/index.php?title=D%E1%BB%B1_%C4%91o%C3%A1n&action=edit&redlink=1). Đường cong ROC biểu diễn các cặp chỉ số (TPR, FPR) tại mỗi ngưỡng với TPR là trục tung và FPR là trục hoành.

# Sau đó, bạn có thể vẽ ROC curve:

plt.figure(figsize=(12, 8))

plt.plot(fpr,tpr,label='Logistic Regression (AUC = %0.3f)' % logit\_roc\_auc, marker='o')

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

plt.xlim(0, 1)

plt.ylim(0, 1.05)

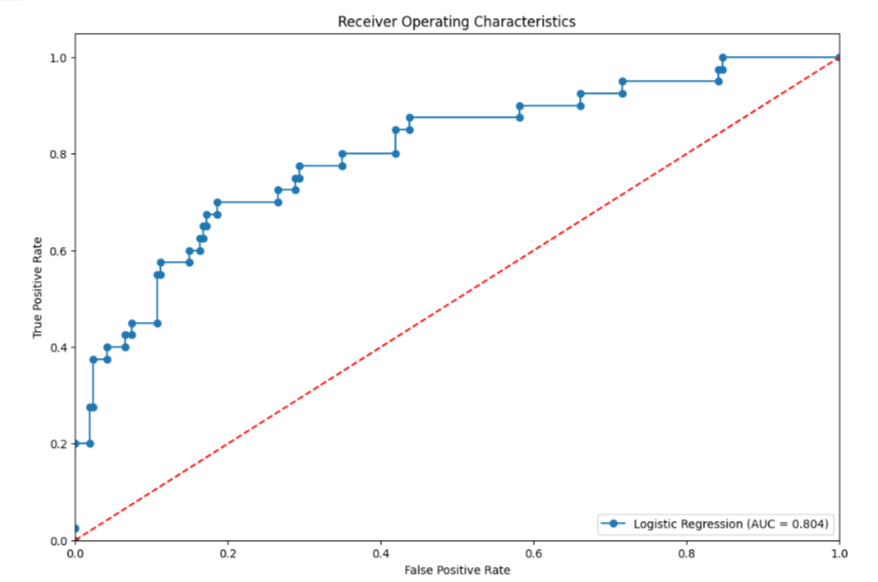
plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristics')

plt.legend(loc='lower right')

plt.show()



Đường cong càng đi dọc theo biên trái và rồi đi dọc theo biên phía trên của không gian ROC, thì chứng tỏ kết quả kiểm tra càng chính xác. Diện tích phía dưới đường cong, giới hạn trong không gian ROC, là thước đo cho độ chính xác hay còn gọi là AUC- đánh giá khả năng phân loại của mô hình tốt như thế nào. Theo kết quả AUC=0.804 chứng tỏ mô hình hoạt động tốt vì AUC càng gần 1 thì mô hình càng phân loại chính xác

# Chuyển đổi y thành dạng nhị phân. Ở đây 0.5 chính là ngưỡng

opt\_thr=0.5

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

y\_pred= np.where(y\_pred>opt\_thr,1,0)

Tiếp đến là chọn mức ngưỡng cho bài toán, mức ngưỡng được chọn ở bài toán này là 0.5. Tức là nếu kết quả dự đoán của y <0.5 thì sẽ xem là 0 tức là nhân viên không nghỉ việc và ngược lại nếu y dự đoán >=0.5 thì sẽ được dự đoán là nhân viên nghỉ việc.

##### Accuracy Measurement

Bắt đầu với confusion matrix:

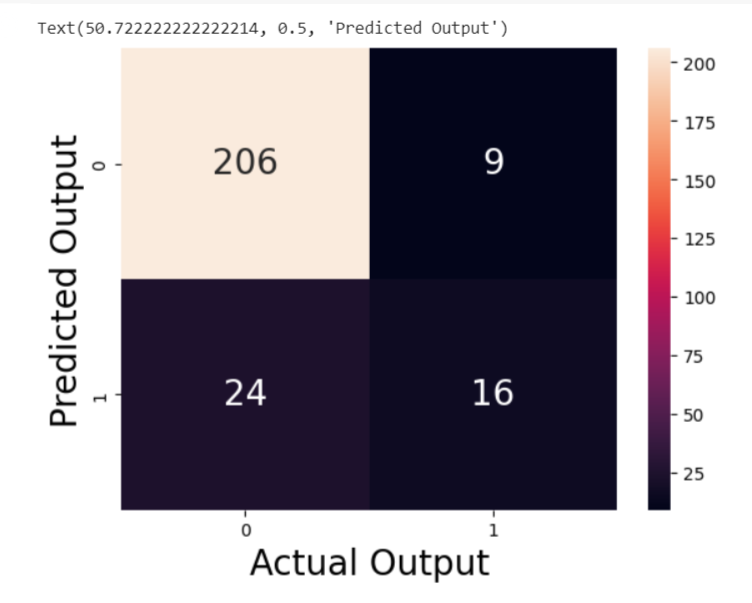
# Xây dựng confusion Matrix

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

ax=sns.heatmap(cm, annot=True, fmt="0", annot\_kws={"size":20})

plt.xlabel('Actual Output', fontsize=20)

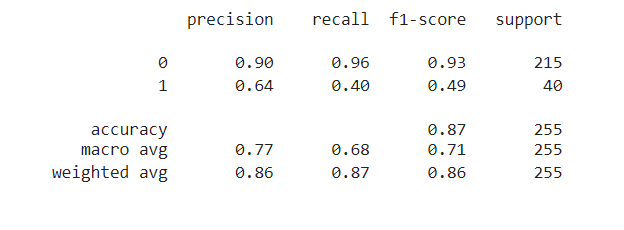
plt.ylabel('Predicted Output', fontsize=20)



Trong hình thì ta có thể nhận thấy mô hình đã dự đoán đúng 222 nhân viên trên tổng số 255 nhân viên của dữ liệu thử nghiệm (test), trong đó có 206 dự đoán đúng nhân viên không nghỉ việc (True Negative TN) và 16 dự đoán đúng nhân viên nghỉ việc (True Positive TP). Cùng lúc đó, mô hình dự đoán sai 33 trường hợp trong đó có 9 trường hợp nhân viên nghỉ việc mà mô hình dự đoán không nghỉ việc (False Negative FN) và 24 trường hợp nhân viên không nghỉ việc mà mô hình lại dự đoán là nghỉ việc (False Positive FP).

# Báo cáo phân lớp

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



**Accuracy:** Độ chính xác là thước đo hiệu suất của mô hình trực quan nhất. Accuracy là phần trăm của những quan sát được dự đoán đúng so với tổng số quan sát.  
Accuracy = (TP+TN) /(TP+FP+FN+TN)  
Chỉ số Accuracy = 87% có nghĩa mô hình dự đoán chính xác 87% những nhân viên nghỉ việc. Trong 255 kết quả trả về thì có 222 kết quả chính xác bao gồm 206 người không thực sự nghỉ việc và 16 người thực sự nghỉ việc.  
**Precision:** Precision là tỉ số của những quan sát nhân viên nghỉ việc được dự đoán đúng trên tổng những quan sát được dự đoán là nghỉ việc.  
Precision = TP/(TP+FP)  
Precision (Class 0): chỉ số này cho biết rằng với class 0, 90% dự báo của mô hình là thật sự đúng.  
**Recall (Sensitivity):** Recall là tỷ lệ của những quan sát nhân viên nghỉ việc được dự đoán đúng trên cho tất cả quan sát nhân viên thực sự nghỉ việc. Chỉ số Recall nên cao hơn 0.5  
Recall = TP/(TP+FN)  
Recall (class 0): 0.96 - chỉ số này cho biết rằng với class 0, mô hình đã xác định chính xác 96% trường hợp true positive trong bộ dữ liệu.  
**F1 score:** F1 score là trung bình của Precision và Recall. Do đó, chỉ số này tính cả false positives và false negatives. Nó được sử dụng khi chi phí của false positive và false negative có sự khác biệt lớn.  
F1 Score = 2\*(Recall \* Precision) / (Recall + Precision)  
⇒Trong kết quả dự đoán, chỉ số Accuracy khá cao là 87% nhưng Recall thì rất thấp khi kết quả là 40%. Điều đó có nghĩa là chúng ta chỉ đang dự đoán chính xác 40% số nhân viên thực sự nghỉ việc.

##### Refine lại mô hình:

# Xác định x và y

y=df['Attrition\_Yes']

X=df[significant\_params]

X=sm.add\_constant(X)

#Chia bộ dữ liệu thành train và test

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)

model=sm.Logit(y\_train,X\_train)

result=model.fit()

print(result.summary())

# Lấy số liệu thống kê

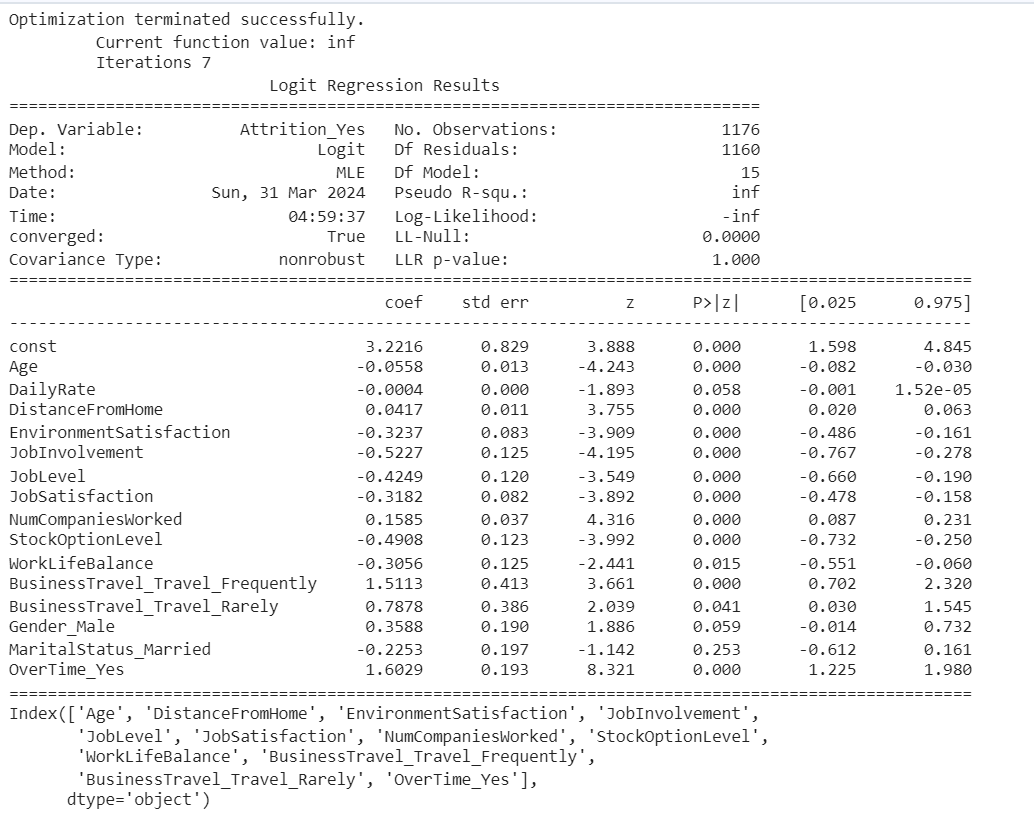
stat\_df=pd.DataFrame({'coefficients':result.params,'p-value':result.pvalues,'odds\_ratio': np.exp(result.params)})

# Điều kiện cho các tham số có ý nghĩa

significant\_params=stat\_df[stat\_df['p-value']<=0.05].index

significant\_params= significant\_params.drop('const')

significant\_params

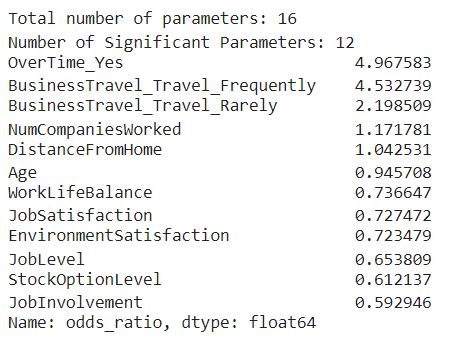


In các

print('Total number of parameters: %s '%len(X.keys()) )

print('Number of Significant Parameters: %s'%(len(significant\_params)))

stat\_df.loc[significant\_params].sort\_values('odds\_ratio', ascending=False)['odds\_ratio']



Sau đó ta tiếp tục hoàn thiện mô hình bằng cách xác định những biến ảnh hưởng đáng kể đến biến phụ thuộc lần hai, sau khi thực hiện lần 2 thì từ 16 biến có ảnh hưởng, nay chỉ còn 12 biến thực sự ảnh hưởng đến biến phụ thuộc.

# **IV. Kết luận và giải pháp**

Sau một khoảng thời gian tìm tòi, nghiên cứu và phân tích chủ đề ***“Phân tích các yếu tố ảnh hưởng đến tỷ lệ nghỉ việc của nhân viên tại công ty IBM”*** dựa trên mô hình thuật toán ***hồi quy logistic***, đánh giá mô hình dựa trên kết quả thực nghiệm cho thấy rằng những yếu tố ảnh hưởng mạnh mẽ đến tỷ lệ nghỉ việc bao gồm:

* Yếu tố về làm việc quá thời gian
* Yếu tố về mức độ đi công tác
* Yếu tố về số lượng công ty đã từng làm việc (mức độ nhảy việc)
* Yếu tố khoảng cách từ nhà đến công ty

Ngoài ra, với những yếu tố khác sẽ ảnh hưởng mạnh mẽ nếu chúng ta làm tốt, cải thiện nhiều thì sẽ giảm tỷ lệ nghỉ việc, từ đó thúc đẩy sự phát triển của doanh nghiệp tốt hơn với chất lượng nhân viên xuất sắc bao gồm:

* Yếu tố mức độ tham gia nhiệt tình của nhân viên
* Yếu tố về vị trí chức vụ trong công ty
* Yếu tố về mức độ hài lòng môi trường doanh nghiệp
* Yếu tố về mức độ hài lòng trong công việc của mỗi cá nhân

Dựa vào những thuộc tính đã được phân tích dựa theo mô hình thuật toán hồi quy logistic, chúng ta có thể thấy rằng những yếu tố trên nếu theo mức độ quá tiêu cực rất nhiều đến tỷ lệ nghỉ việc của nhân viên IBM, ngoài ra nếu chúng ra có những biện pháp khắc phục những yếu tố này sẽ góp phần nâng cao chất lượng hiệu quả làm việc của nhân viên và nâng cao văn hoá, môi trường doanh nghiệp. Từ những con số được biểu thị thông qua các biểu đồ, chúng ta có những biện pháp để giảm tỷ lệ nghỉ việc và đưa ra những quyết định đúng đắn với việc quản lý nhân sự tại công ty IBM nói riêng và tại mỗi doanh nghiệp nói chung như:

* Cải thiện môi trường làm việc dựa trên các cuộc khảo sát để hiểu rõ các nguyên nhân và các nhà quản lý tìm cách khắc phục.
* Tổ chức các chương trình đào tạo và phát triển năng lực cá nhân, đồng đội để giúp nhân viên hay tổ chức nâng cao kỹ năng chuyên môn và tiến xa hơn trong sự nghiệp.
* Tăng cường giao tiếp, phản hồi để xây dựng một quy trình giao tiếp thân thiện giữa nhân viên và quản lý. Khuyến khích nhân viên có thể đề xuất ý kiến xây dựng văn hoá, môi trường lành mạnh trong doanh nghiệp.
* Đưa ra những chính sách thưởng phạt cho mỗi cá nhân trong công ty để có thể nâng cao chất lượng làm việc và đào thải những nhân viên có hiệu quả làm việc thấp.

Mô hình thuật toán hồi quy logistics được áp dụng nhiều trong việc phân tích các yếu tố khách quan ảnh hưởng đến tỷ lệ của một yếu tố nào đó. Với chủ đề nghiên cứu về hồi quy, nhóm chúng em xây dựng mô hình trên để đánh giá các yếu tố khách quan tác động đến yếu tố nghỉ việc của nhân viên. Với dự án này, nhóm đã có trải nghiệm xây dựng thuật toán và nghiên cứu vận hành của hàm hồi quy logistic như thế nào. Hy vọng trong tương lai, mỗi thành viên trong nhóm có thể vận dụng được mô hình thuật toán để đánh giá một môi trường tổ chức dựa vào dữ liệu nội bộ doanh nghiệp.

**LỜI CẢM ƠN**

Phân tích dữ liệu bằng Python là một phần không thể không thiếu trong các doanh nghiệp để phân tích, đưa ra đề xuất cho dữ liệu và hỗ trợ ra quyết định của doanh nghiệp. Mục đích chính của phân tích dữ liệu bằng Python là khám phá, hiểu biết và rút ra thông tin hữu ích từ các bộ dữ liệu. Với những chức năng vô cùng nổi trội và phù hợp thúc đẩy phát triển doanh nghiệp cùng với việc sử dụng các kỹ thuật và công cụ trong ngôn ngữ lập trình Python sẽ giúp doanh nghiệp có thể đạt được những thành tựu đáng kể. Bằng cách sử dụng Python, doanh nghiệp có thể tạo ra những giải pháp thông minh và hiệu quả hơn, giúp họ: tối ưu hóa quy trình kinh doanh, hiểu sâu hơn về khách hàng, dự đoán và đánh giá rủi ro, tăng cường hiệu suất và lợi nhuận,..

Môn học Phân tích dữ liệu bằng Python là môn học hay và thiết thực cho các sinh viên kinh tế, đặc biệt sinh viên ngành Hệ thống thông tin quản lý. Trong khoảng thời gian học tập, tìm tòi và nghiên cứu, mỗi thành viên trong nhóm đều nhận thấy được tầm quan trọng của phân tích dữ liệu bằng Python nói chung và chức năng vượt trội của Regression Logistic nói riêng trong thời đại dữ liệu lớn như hiện nay. Những nghiên cứu này của mỗi thành viên đều bổ sung thêm nhiều kiến thức nền tảng và nâng cao để phục vụ cho bản thân trong tương lai.

Lời cuối cùng, chúng em trân trọng khoảng thời gian cùng nhau làm việc, nghiên cứu, xây dựng đề tài với sự hướng dẫn nhiệt tình của thầy **Lê Diên Tuấn – Giảng viên Khoa Thương mại điện tử - Trường Đại học Kinh tế - Đại học Đà Nẵng**. Cuối cùng, chúng em xin cảm ơn thầy đã tận tình chỉ dạy trong suốt khoảng thời gian qua. Chúng em xin chúc thày luôn nhiều sức khoẻ, hạnh phúc và đạt nhiều thành công trong cuộc sống.

Trân trọng,

Nhóm 2, Lớp 47K21.2.

**TÀI LIỆU THAM KHẢO**

<https://aws.amazon.com/vi/what-is/logistic-regression/>

<https://trituenhantao.io/machine-learning-co-ban/bai-6-logistic-regression-hoi-quy-logistic/>

**LINK VIDEO BÁO CÁO**

Drive:

<https://drive.google.com/file/d/1yV6PJgk5ekgyl5X2hGqulXsPonLoMs-S/view?usp=sharing>