Human Resources Analytics

July 16, 2024

Name: Nguyễn Thanh Hòa

_Project Title: Enhancing Human Resources Insights through Advanced Analytics (Nâng cao hiểu biết sâu sắc về nguồn nhân lực thông qua phân tích nâng cao)

Link Report: Enhancing Human Resources Insights through Advanced Analytics

The aim of this project: + Sử dụng phân tích dữ liệu và học máy để hiểu sâu hơn về hoạt động nhân sự (HR) của chúng tôi

=> Có thể giúp tối ưu hóa việc quản lý lực lượng lao động, cải thiện sự hài lòng của nhân viên và nâng cao hiệu quả tổng thể của tổ chức. (tập trung vào các số liệu nhân sự khác nhau như nhân khẩu học, điểm hiệu suất, và lương thưởng)

Goals: + Cải thiện quản lý lực lượng lao động + Tăng sự hài lòng của nhân viên + Tối ưu hóa chiến lược lương thưởng + Thông tin chi tiết có thể dự đoán

0.1 Các bước sẽ thực hiện:

Chuẩn bị dữ liệu và làm sạch data: + Thực hiện xử lý giá trị thiếu(Thay thế giá trị thiếu cho các thuộc tính số bằng giá trị trung bình) + Khai thác đặc trưng(Thay thế giá trị thiếu cho các thuộc tính hạng mục bằng giá trị xuất hiện nhiều nhất) + Mã hóa, chuẩn hóa dữ liệu,Xử lý ngoại lệ bằng cách sử dụng IQR.

Thực hiện Data Analysis (EDA): + Tính toán thống kê mô tả, trực quan hóa phân bố dữ liệu (tình trạng nghỉ việc, tỷ lệ nghỉ việc theo phòng ban…) + Biểu đồ nhiệt tương quan.Giúp phát hiện và hỗ trợ đưa ra quyết định.

Trực quan hóa dữ liệu: + Phân bố tình trạng nghỉ việc . + Phân bố nhân viên theo chức vụ . + Tỷ lệ nghỉ việc theo phòng ban . + Phân bố thu nhập hàng tháng theo vai trò công việc . + Biểu đồ nhiệt tương quan .

Xây dựng và đánh giá mô hình dự đoán:để dự đoán mức độ tiêu hao và hiệu suất của nhân viên. + Logistic Regression (Hồi quy Logistic):Dùng để phân loại tình trạng nghỉ việc (Attrition) của nhân viên. + Random Forest (Rừng ngẫu nhiên):phân loại tình trạng nghỉ việc.tốt hơn Logistic Regression + Hồi quy (Regression):Dự đoán PerformanceRating (Đánh giá hiệu suất). + XGBoost:Phân loại tình trạng nghỉ việc. + LightGBM:Phân loại tình trạng nghỉ việc. + Gradient Boosting (Tăng cường độ dốc):Dự đoán PerformanceRating

Đánh giá kết luận và Đưa ra Đề xuất: : Kết luận đánh giá kết quả của mô hình và đưa ra kết luận.

1 0.Imporrt thư viện

[27]:
```
import pandas as pd
import matplotlib.pyplot as plt import
seaborn as sns
import numpy as np
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler, OneHotEncoder, LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, RandomForestRegressor,␣

  ↪GradientBoostingRegressor
from xgboost import XGBClassifier
from lightgbm import LGBMClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score,␣

  ↪f1_score, roc_auc_score, mean_squared_error, mean_absolute_error, r2_score,␣

  ↪confusion_matrix, classification_report, roc_curve, auc
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
import shap
```

2.0.1 1. Nạp và tiền xử lý dữ liệu

[28]:
```
data = pd.read_csv("50000 HRA Records.csv")
```

3 Thông tin data:

[29]:
```
print("Mô tả  về  data")
print(data.describe())
```

```
Mô tả  về  data
                Age      DailyRate  DistanceFromHome     Education  ¥
count   50000.000000   50000.000000      50000.000000  50000.000000
mean       38.971480     798.677560         25.539780      3.004600
std        12.420834     405.080217         14.339956      1.414249
min        18.000000     100.000000          1.000000      1.000000
25%        28.000000     445.000000         13.000000      2.000000
50%        39.000000     798.000000         25.000000      3.000000
75%        50.000000    1151.000000         38.000000      4.000000
max        60.000000    1500.000000         50.000000      5.000000
```

	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	¥
count	50000.0	50000.000000	50000.000000	50000.000000	
mean	1.0	25000.500000	2.498360	115.432940	
std	0.0	14433.901067	1.119671	49.424867	
min	1.0	1.000000	1.000000	30.000000	
25%	1.0	12500.750000	1.000000	73.000000	
50%	1.0	25000.500000	2.000000	116.000000	
75%	1.0	37500.250000	4.000000	158.000000	
max	1.0	50000.000000	4.000000	200.000000	

	JobInvolvement	JobLevel	..	RelationshipSatisfaction	¥
count	50000.000000	50000.000000	..	50000.000000	
mean	2.502620	2.994640	..	2.502220	
std	1.120544	1.415998	..	1.117918	
min	1.000000	1.000000	..	1.000000	
25%	1.000000	2.000000	..	2.000000	
50%	3.000000	3.000000	..	2.000000	
75%	4.000000	4.000000	..	4.000000	
max	4.000000	5.000000	..	4.000000	

	StandardHours	StockOptionLevel	TotalWorkingYears	¥
count	50000.0	50000.000000	50000.000000	
mean	80.0	2.503780	20.496860	
std	0.0	1.118933	11.575819	
min	80.0	1.000000	1.000000	
25%	80.0	2.000000	11.000000	
50%	80.0	3.000000	20.000000	
75%	80.0	4.000000	31.000000	
max	80.0	4.000000	40.000000	

	TrainingTimesLastYear	WorkLifeBalance	YearsAtCompany	¥
count	50000.000000	50000.00000	50000.00000	
mean	3.493980	2.49872	10.77110	
std	1.708152	1.11412	8.93423	
min	1.000000	1.00000	1.00000	
25%	2.000000	2.00000	3.00000	
50%	3.000000	2.00000	8.00000	
75%	5.000000	3.00000	16.00000	
max	6.000000	4.00000	40.00000	

	YearsInCurrentRole	YearsSinceLastPromotion	YearsWithCurrManager
count	50000.000000	50000.000000	50000.000000
mean	5.907300	5.871820	5.889040
std	6.034378	5.999056	6.009613
min	1.000000	1.000000	1.000000
25%	1.000000	2.000000	1.000000
50%	4.000000	4.000000	4.000000

75%	8.000000	8.000000	8.000000
max	40.000000	40.000000	40.000000

[8 rows x 26 columns]

[30]:
```
print("Thông tin về  data")
print(data.info())
```

```
Thông tin về  data
<class   'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 35 columns):
 #   Column                     Non-Null Count  Dtype
---  -------                    --------------  -----
 0   Age                        50000  non-null  int64
 1   Attrition                  50000  non-null  object
 2   BusinessTravel             50000  non-null  object
 3   DailyRate                  50000  non-null  int64
 4   Department                 50000  non-null  object
 5   DistanceFromHome           50000  non-null  int64
 6   Education                  50000  non-null  int64
 7   EducationField             50000  non-null  object
 8   EmployeeCount              50000  non-null  int64
 9   EmployeeNumber             50000  non-null  int64
 10  EnvironmentSatisfaction    50000  non-null  int64
 11  Gender                     50000  non-null  object
 12  HourlyRate                 50000  non-null  int64
 13  JobInvolvement             50000  non-null  int64
 14  JobLevel                   50000  non-null  int64
 15  JobRole                    50000  non-null  object
 16  JobSatisfaction            50000  non-null  int64
 17  MaritalStatus              50000  non-null  object
 18  MonthlyIncome              50000  non-null  int64
 19  MonthlyRate                50000  non-null  int64
 20  NumCompaniesWorked         50000  non-null  int64
 21  Over18                     50000  non-null  object
 22  OverTime                   50000  non-null  object
 23  PercentSalaryHike          50000  non-null  int64
 24  PerformanceRating          50000  non-null  int64
 25  RelationshipSatisfaction   50000  non-null  int64
 26  StandardHours              50000  non-null  int64
 27  StockOptionLevel           50000  non-null  int64
 28  TotalWorkingYears          50000  non-null  int64
 29  TrainingTimesLastYear      50000  non-null  int64
 30  WorkLifeBalance            50000  non-null  int64
 31  YearsAtCompany             50000  non-null  int64
 32  YearsInCurrentRole         50000  non-null  int64
 33  YearsSinceLastPromotion    50000  non-null  int64
```

```
 34  YearsWithCurrManager        50000 non-null  int64
dtypes: int64(26), object(9)
memory usage: 13.4+ MB None
```

[31]:
```
print("Data") print(data.head(10))
```

```
Data
   Age Attrition      BusinessTravel  DailyRate        Department  ¥
0   31        No          Non-Travel        158          Software
1   38        No       Travel_Rarely        985  Human Resources
2   59       Yes          Non-Travel       1273             Sales
3   52       Yes       Travel_Rarely        480           Support
4   32        No          Non-Travel        543  Human Resources
5   19       Yes          Non-Travel        779          Hardware
6   42       Yes          Non-Travel        934           Support
7   30        No       Travel_Rarely        380           Support
8   41        No  Travel_Frequently       1464          Software
9   45        No  Travel_Frequently       1020  Human Resources

    DistanceFromHome  Education     EducationField   EmployeeCount   ¥
0                  7          3            Medical               1
1                 33          5      Life Sciences               1
2                  5          2  Technical Degree               1
3                  2          5          Marketing               1
4                  7          5    Human Resources               1
5                 43          1            Medical               1
6                 26          4    Human Resources               1
7                 19          3          Marketing               1
8                 16          1      Life Sciences               1
9                 17          5      Life Sciences               1

   EmployeeNumber   ..  RelationshipSatisfaction   StandardHours   ¥
0               1   ..                         1              80
1               2   ..                         3              80
2               3   ..                         2              80
3               4   ..                         2              80
4               5   ..                         4              80
5               6   ..                         3              80
6               7   ..                         1              80
7               8   ..                         2              80
8               9   ..                         3              80
9              10   ..                         3              80

   StockOptionLevel  TotalWorkingYears  TrainingTimesLastYear WorkLifeBalance  ¥
0                 2                 15                      1                2
1                 4                  5                      4                3
```

```
2                   2                    9                    5                    1
3                   2                   22                    4                    4
4                   2                   30                    3                    4
5                   1                   33                    4                    2
6                   1                    4                    3                    4
7                   1                    2                    2                    2
8                   2                    8                    1                    2
9                   4                    6                    4                    4

   YearsAtCompany  YearsInCurrentRole  YearsSinceLastPromotion   ¥
0              12                   4                       10
1               1                   1                        1
2               6                   6                        4
3              10                   9                        5
4              29                  27                        9
5              16                   4                       14
6               2                   1                        1
7               2                   2                        2
8               2                   1                        2
9               5                   3                        4

   YearsWithCurrManager
0                    11
1                     1
2                     3
3                     6
4                     7
5                     3
6                     2
7                     2
8                     2
9                     1

[10 rows x 35 columns]
```

3.0.1 1.1 Xử lý giá trị thiếu

Thay thế giá trị thiếu cho các thuộc tính số bằng giá trị trung bình

[32]:
```
numerical_features = ['Age', 'DailyRate', 'DistanceFromHome', 'MonthlyIncome',
                      'NumCompaniesWorked', 'PercentSalaryHike',␣
  ↪'TotalWorkingYears',TrainingTimesLastYear', 'YearsAtCompany',␣
  ↪'YearsInCurrentRole',
                      'YearsSinceLastPromotion', 'YearsWithCurrManager']
data[numerical_features] = data[numerical_features].
  ↪fillna(data[numerical_features].mean())
```

Thay thế giá trị thiếu cho các thuộc tính hạng mục bằng giá trị xuất hiện nhiều nhất

[33]:
```
categorical_features = ['Gender', 'Department', 'JobRole', 'MaritalStatus',
    'OverTime', 'BusinessTravel', 'EducationField']
data[categorical_features] = data[categorical_features].
    fillna(data[categorical_features].mode().iloc[0])
```

Xử lý ngoại lệ (ví dụ sử dụng IQR)

[34]:
```
numerical_features = ['Age', 'DailyRate', 'DistanceFromHome', 'MonthlyIncome',
    'NumCompaniesWorked', 'PercentSalaryHike',
                      'TotalWorkingYears', 'TrainingTimesLastYear',
    'YearsAtCompany','YearsInCurrentRole', 'YearsSinceLastPromotion',
    'YearsWithCurrManager']
for feature in numerical_features:
    Q1 = data[feature].quantile(0.25) Q3
    = data[feature].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    data[feature] = np.where(data[feature] < lower_bound, lower_bound,
    data[feature])
    data[feature] = np.where(data[feature] > upper_bound, upper_bound,
    data[feature])
# Kiểm tra dữ liệu trùng lặp
print("Số lượng bản ghi trùng lặp:", data.duplicated().sum())
data.drop_duplicates(inplace=True)
```

```
Số lượng bản ghi trùng lặp: 0
```

4 1.2 Khai thác đặc trưng

4.0.1 Tạo nhóm tuổi

[35]:
```
bins = [18, 25, 35, 45, 55, 60]
labels = ['18-24', '25-34', '35-44', '45-54', '55-60']
data['AgeGroup'] = pd.cut(data['Age'], bins=bins, labels=labels, right=False)
```

4.0.2 1.3 Mã hóa các thuộc tính hạng mục

[36]:
```
categorical_transformer = Pipeline(steps=[
    ('onehot', OneHotEncoder(handle_unknown='ignore'))
])
```

4.0.3 1.4 Chuẩn hóa các thuộc tính số

[37]:
```
numerical_transformer = Pipeline(steps=[
    ('scaler', StandardScaler())
])
```

4.0.4 1.5 Kết hợp các bộ chuyển đổi

[38]:
```
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numerical_transformer, numerical_features),
        ('cat', categorical_transformer, categorical_features + ['AgeGroup'])
    ])
```

5 2.Thực hiện Data Analysis (EDA):

5.0.1 2.1 Thống kê mô tả

[39]:
```
print("Thố ng kê mô tả :")
print(data.describe(include='all'))
```

```
Thố ng kê mô tả :
                Age Attrition BusinessTravel      DailyRate Department  ¥
count   50000.000000     50000          50000   50000.000000      50000
unique           NaN         2              3            NaN          6
top              NaN       Yes     Non-Travel            NaN      Sales
freq             NaN     25105          16919            NaN       8453
mean       38.971480       NaN            NaN     798.677560        NaN
std        12.420834       NaN            NaN     405.080217        NaN
min        18.000000       NaN            NaN     100.000000        NaN
25%        28.000000       NaN            NaN     445.000000        NaN
50%        39.000000       NaN            NaN     798.000000        NaN
75%        50.000000       NaN            NaN    1151.000000        NaN
max        60.000000       NaN            NaN    1500.000000        NaN

        DistanceFromHome     Education EducationField  EmployeeCount  ¥
count       50000.000000  50000.000000          50000        50000.0
unique               NaN           NaN              6            NaN
top                  NaN           NaN        Medical            NaN
freq                 NaN           NaN           8607            NaN
mean           25.539780      3.004600            NaN            1.0
std            14.339956      1.414249            NaN            0.0
min             1.000000      1.000000            NaN            1.0
25%            13.000000      2.000000            NaN            1.0
50%            25.000000      3.000000            NaN            1.0
75%            38.000000      4.000000            NaN            1.0
max            50.000000      5.000000            NaN            1.0
```

	EmployeeNumber	…	StandardHours	StockOptionLevel	¥
count	50000.000000	…	50000.0	50000.000000	
unique	NaN	…	NaN	NaN	
top	NaN	…	NaN	NaN	
freq	NaN	…	NaN	NaN	
mean	25000.500000	…	80.0	2.503780	
std	14433.901067	…	0.0	1.118933	
min	1.000000	…	80.0	1.000000	
25%	12500.750000	…	80.0	2.000000	
50%	25000.500000	…	80.0	3.000000	
75%	37500.250000	…	80.0	4.000000	
max	50000.000000	…	80.0	4.000000	

	TotalWorkingYears	TrainingTimesLastYear	WorkLifeBalance	¥
count	50000.000000	50000.000000	50000.00000	
unique	NaN	NaN	NaN	
top	NaN	NaN	NaN	
freq	NaN	NaN	NaN	
mean	20.496860	3.493980	2.49872	
std	11.575819	1.708152	1.11412	
min	1.000000	1.000000	1.00000	
25%	11.000000	2.000000	2.00000	
50%	20.000000	3.000000	2.00000	
75%	31.000000	5.000000	3.00000	
max	40.000000	6.000000	4.00000	

	YearsAtCompany	YearsInCurrentRole	YearsSinceLastPromotion	¥
count	50000.000000	50000.000000	50000.000000	
unique	NaN	NaN	NaN	
top	NaN	NaN	NaN	
freq	NaN	NaN	NaN	
mean	10.754170	5.636480	5.514880	
std	8.884751	5.222247	4.968657	
min	1.000000	1.000000	1.000000	
25%	3.000000	1.000000	2.000000	
50%	8.000000	4.000000	4.000000	
75%	16.000000	8.000000	8.000000	
max	35.500000	18.500000	17.000000	

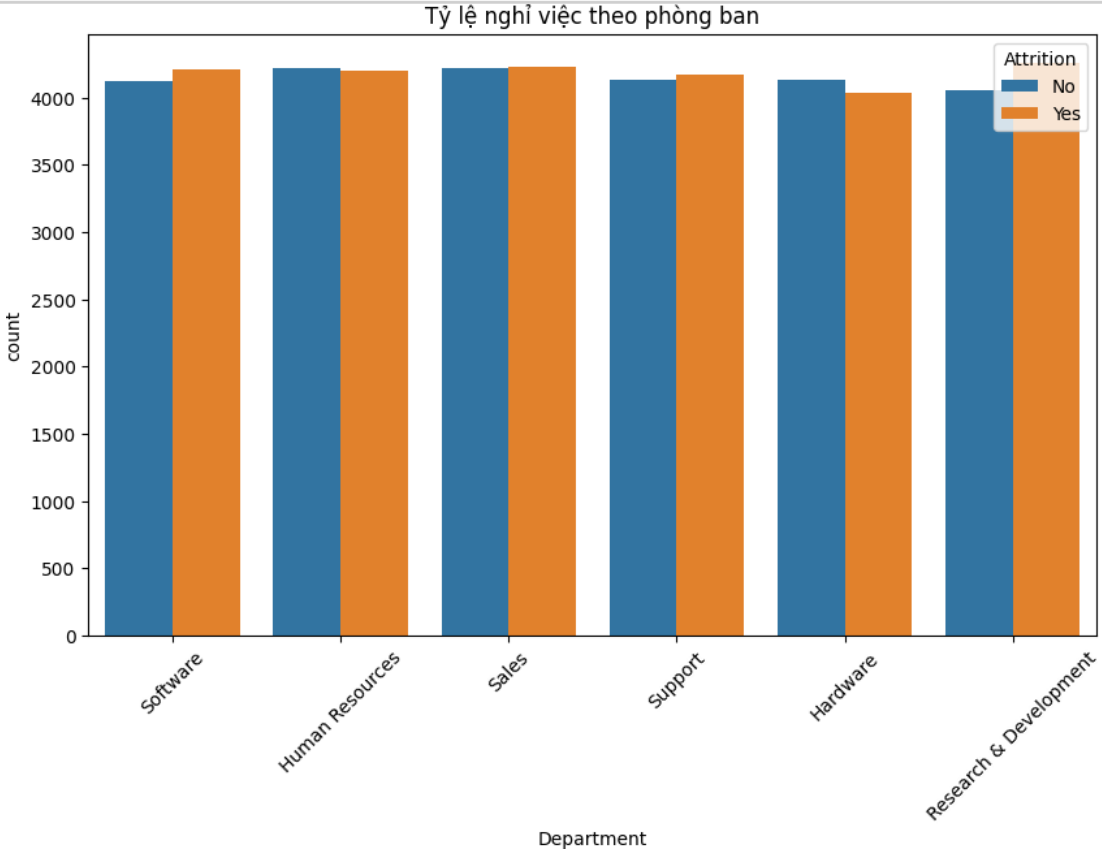
	YearsWithCurrManager	AgeGroup
count	50000.000000	48842
unique	NaN	5
top	NaN	45-54
freq	NaN	11685
mean	5.618110	NaN
std	5.185144	NaN
min	1.000000	NaN

25%	1.000000	NaN
50%	4.000000	NaN
75%	8.000000	NaN
max	18.500000	NaN

[11 rows x 36 columns]

5.0.2 2.2.2 Tỷ lệ nghỉ việc theo phòng ban

[40]:
```
plt.figure(figsize=(10, 6))
sns.countplot(x='Department', hue='Attrition', data=data)
plt.title('Tỷ lệ nghỉ việc theo phòng ban')
plt.xticks(rotation=45)
plt.show()
```

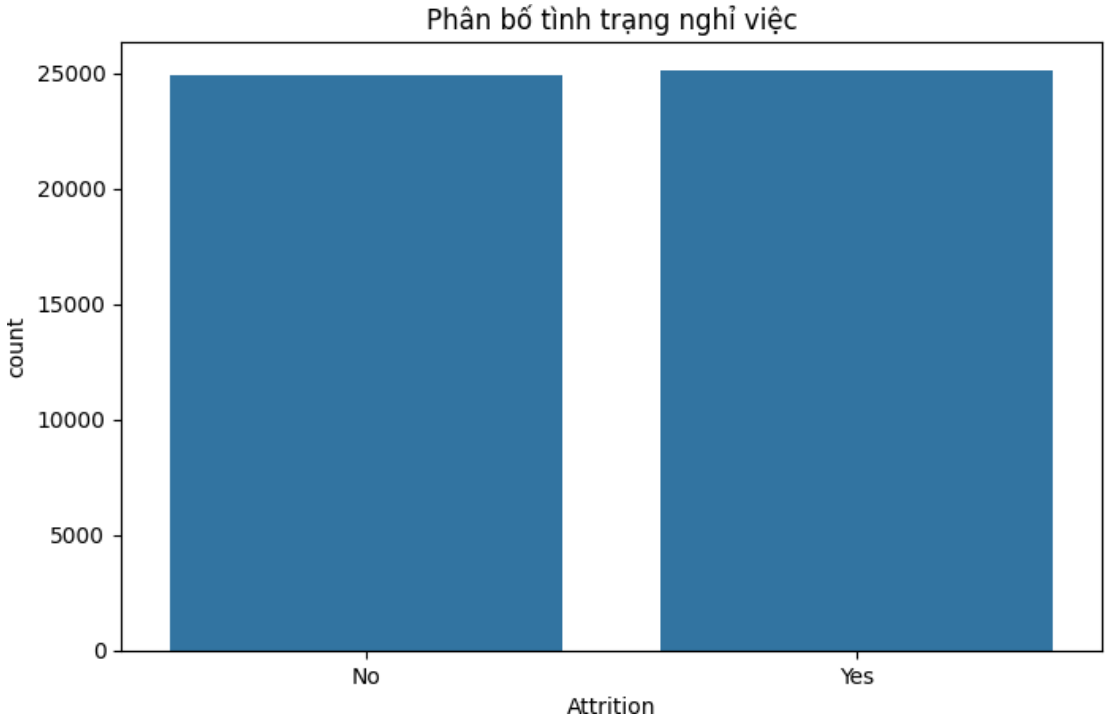


Nhận xét: Cho thấy sự khác biệt về tỷ lệ nghỉ việc giữa các phòng ban. Ví dụ: Software và Research & Development có tỷ lệ nghỉ việc cao hơn so với Human Resources và Hardware

5.0.3 3.Trực quan hóa dữ liệu

3.3.1_a Phân bố của tình trạng nghỉ việc

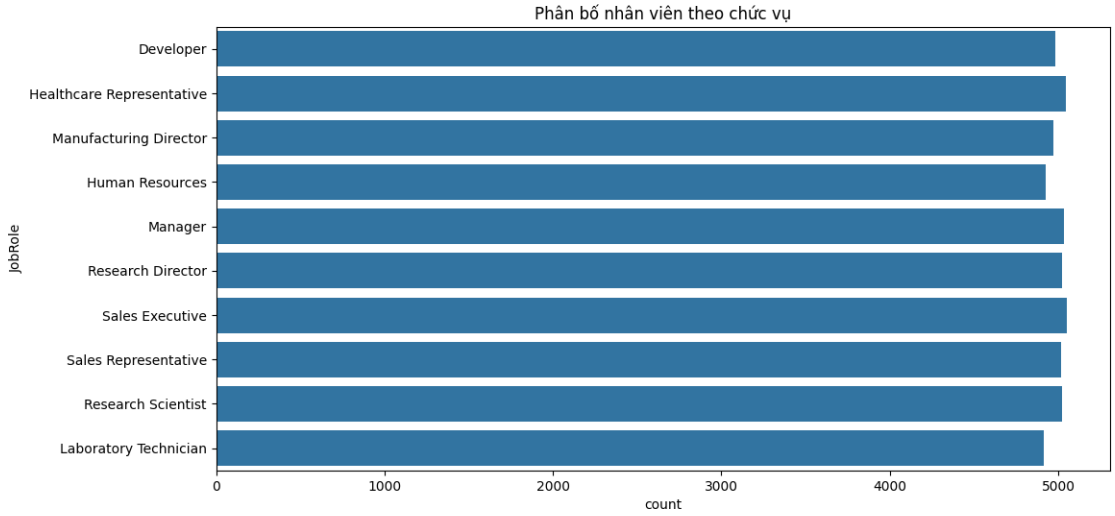
[41]:
```
plt.figure(figsize=(8, 5))
sns.countplot(x='Attrition', data=data)
plt.title('Phân bố  tình trạ ng nghỉ  việ c')
plt.show()
```



Nhận xét:Cho thấy tỷ lệ nghỉ việc (“Yes”) chiếm một phần đáng kể, có thể là dấu hiệu cần quan tâm để tìm hiểu nguyên nhân và có giải pháp giữ chân nhân viên.

3.3.1_b Phân bố nhân viên theo chức vụ

[42]:
```
# Phân bố  nhân viên theo chứ c vụ
plt.figure(figsize=(12, 6))
sns.countplot(y='JobRole', data=data)
plt.title('Phân bố  nhân viên theo chứ c vụ ')
plt.show()
```

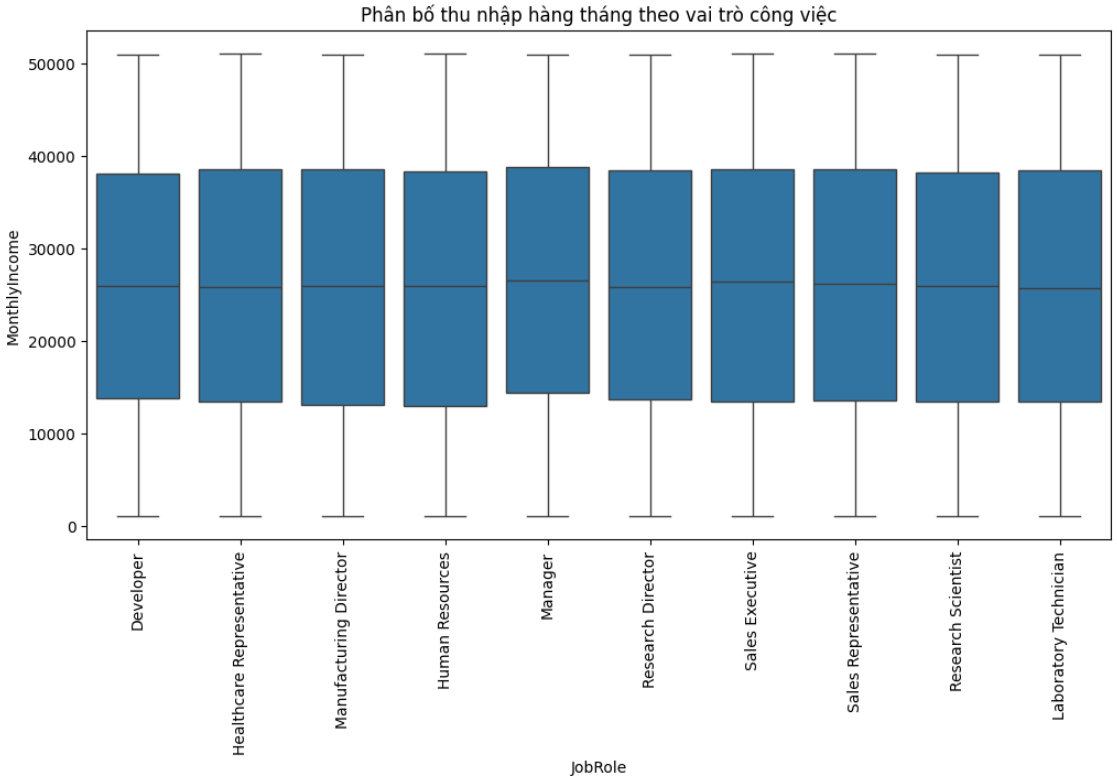


Nhận Xét:Cho thấy số lượng nhân viên phân bố tương đối đồng đều ở hầu hết các vị trí trong công ty

5.0.4 Mô tả: Biểu đồ cột hiển thị số lượng nhân viên cho mỗi chức vụ (JobRole).

5.0.5 3.3.1_c Phân bố thu nhập hàng tháng theo vai trò công việc (Diễn tả khác)

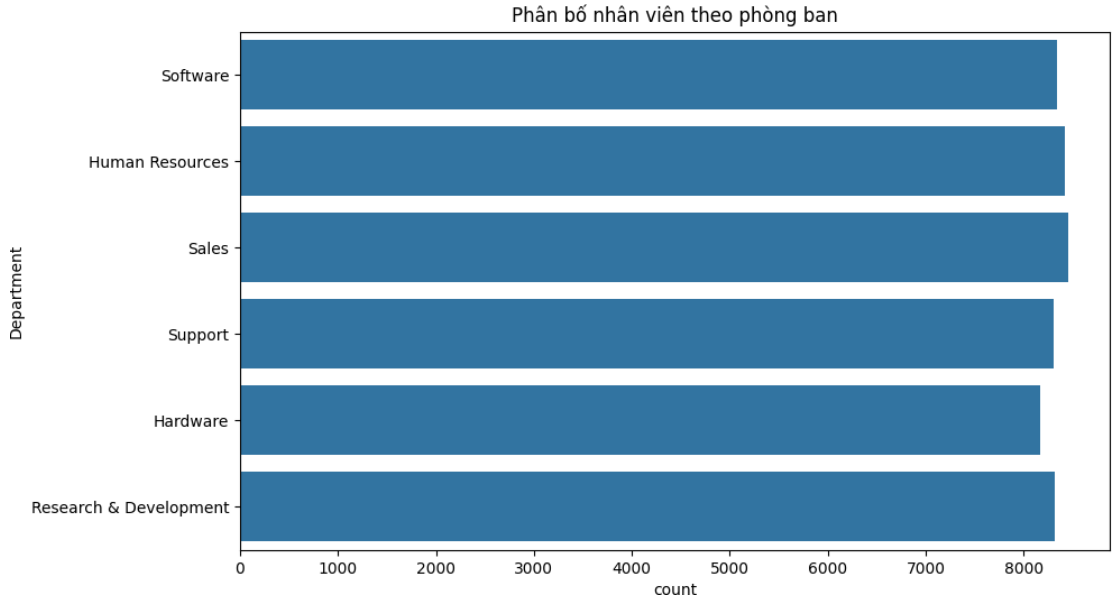
[43]:
```
plt.figure(figsize=(12, 6))
sns.boxplot(x='JobRole', y='MonthlyIncome', data=data)
plt.title('Phân bố  thu nhậ p hàng tháng theo vai trò công việ c')
plt.xticks(rotation=90)
plt.show()
```



Nhận xét:Chính sách lương của công ty tương đối công bằng , nhưng vẫn có sự phân hóa thu nhập giữa các cá nhân cùng vị trí công việc. Biên độ lương rộng cho thấy công ty có thể đang áp dụng chính sách lương thưởng dựa trên năng lực và kinh nghiệm cá nhân.

5.0.6 3.3.1.d Phân bố nhân viên theo phòng ban

[44]:
```
plt.figure(figsize=(10, 6))
sns.countplot(y='Department', data=data) plt.title('Phân
bố  nhân viên theo phòng ban') plt.show()
```

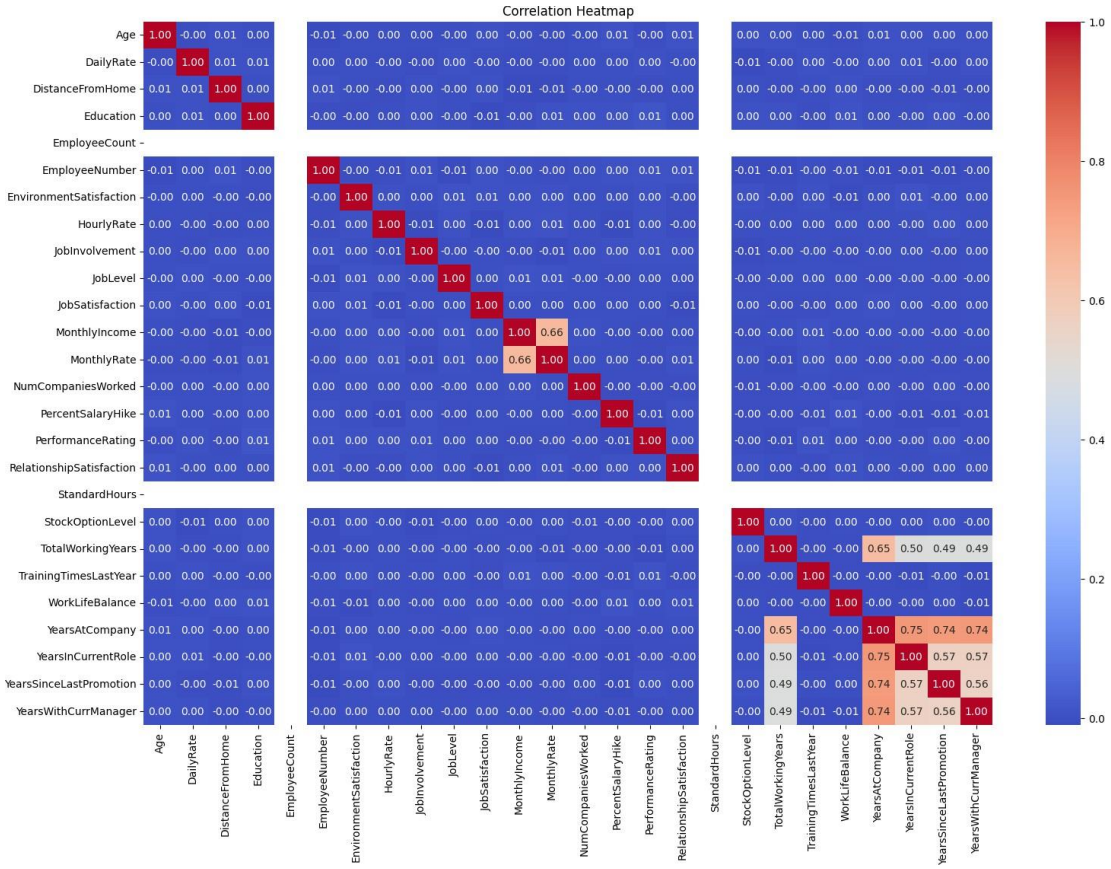


Nhận xét:Cho thấy sự phân bổ nhân sự tương đối đồng đều giữa các phòng ban trong công ty.

5.0.7 3.3.2 Biểu đồ nhiệt tương quan

[45]:
```
data_numeric = data.select_dtypes(include=[float, int])

plt.figure(figsize=(18, 12)) correlation_matrix
= data_numeric.corr()
sns.heatmap(correlation_matrix, annot=True, fmt=".2f", cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```



Nhận xét: + JobLevel,MonthlyIncome,Age có số liệu > 0.5 cho thấy tuong quan mạnh ,nhân viên làm việc càng lâu năm thì thường có cấp bậc,thu nhập cao hơn tuổi đời càng cao (điều này khá hiển nhiên) trong công việc cao hơn. + YearsInCurrent-Role,YearsWithCurrManager,YearsSinceLastPromotion,PercentSalaryHike với PerformanceRating tương quan mạnh từ 0.7 trở lên cho thấy nhân viên gắn bó lâu với công ty thường có thời gian làm việc và liên kết với công ty hiện tại lâu hơn.

Đánh Giá Chung: + **Công ty có cơ cấu tổ chức**: khá cân bằng, các phòng ban có số lượng nhân viên tương đối đồng đều. + **Chính sách lương thưởng**: tương đối công bằng, tuy nhiên vẫn có sự phân hóa thu nhập giữa các cá nhân cùng vị trí công việc,dựa trên năng lực và kinh nghiệm, tuy nhiên cần theo dõi sát sao để đảm bảo tính công bằng và tránh bất mãn trong nội bộ.

+ **Nhân viên làm việc lâu năm**: Thường có cấp bậc, thu nhập và tuổi đời cao hơn.

5.0.8 3.3.3 Phân bố lương

[46]:
```
print("Column names in the dataset:")
print(data.columns)
```

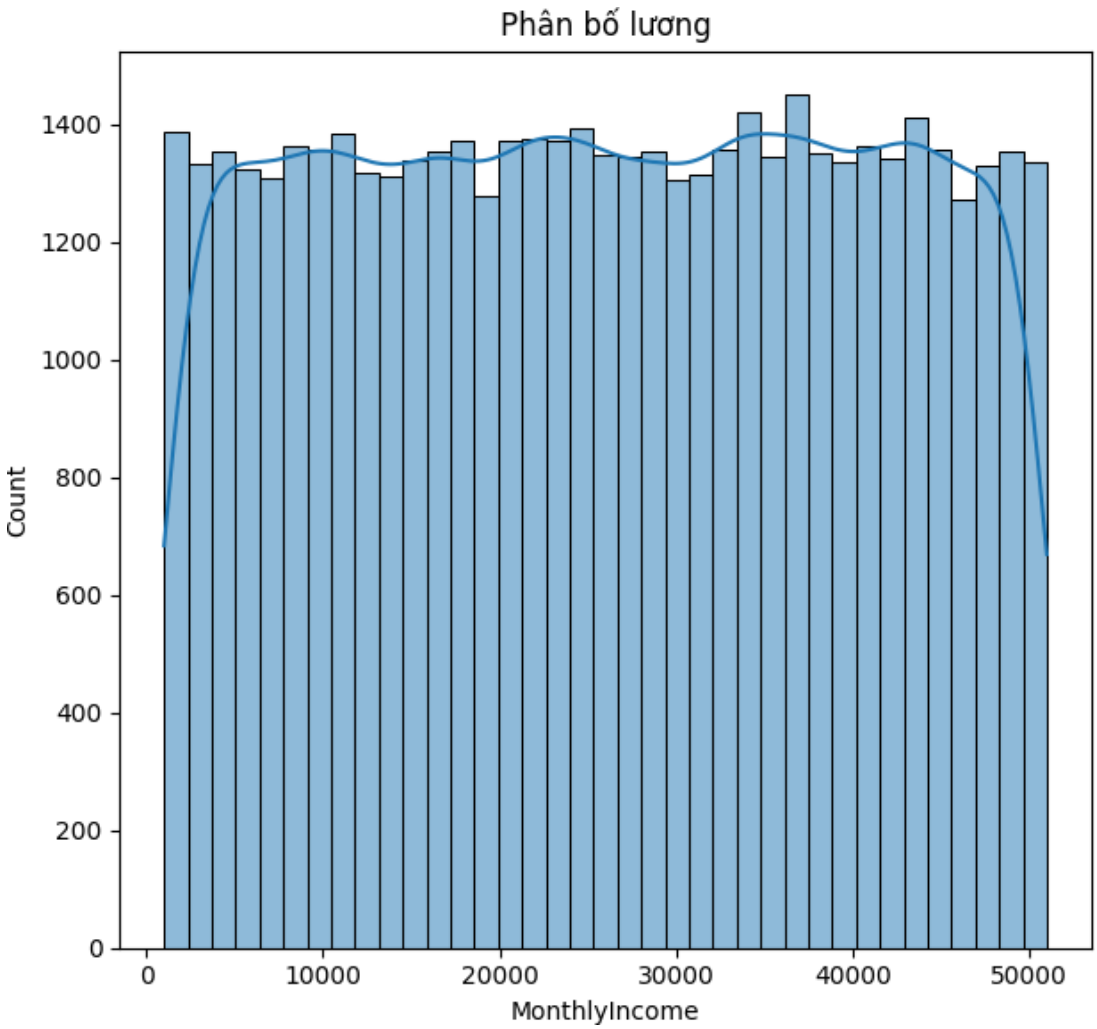
```
Column names in the dataset:
Index(['Age', 'Attrition', 'BusinessTravel', 'DailyRate', 'Department',
       'DistanceFromHome', 'Education', 'EducationField', 'EmployeeCount',
```

```
       'EmployeeNumber', 'EnvironmentSatisfaction', 'Gender', 'HourlyRate',
       'JobInvolvement', 'JobLevel', 'JobRole', 'JobSatisfaction',
       'MaritalStatus', 'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked',
       'Over18', 'OverTime', 'PercentSalaryHike', 'PerformanceRating',
       'RelationshipSatisfaction', 'StandardHours', 'StockOptionLevel',
       'TotalWorkingYears', 'TrainingTimesLastYear', 'WorkLifeBalance',
       'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion',
       'YearsWithCurrManager', 'AgeGroup'],
      dtype='object')
```

[47]:
```
#Vẽ biểu đồ histogram thể hiện phân bố của lương tháng
plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1) sns.histplot(data['MonthlyIncome'],
kde=True) plt.title('Phân bố lương')

plt.tight_layout() plt.show()
```



Nhận xét: + Chính sách lương của công ty có vẻ khá công bằng khi mức lương phân bố tương đối đồng đều trong khoảng từ 0 đến 50,000. Không có sự chênh lệch quá lớn giữa các nhóm thu nhập. + Mức lương phổ biến: Phần lớn nhân viên tập trung ở nhóm thu nhập từ 5,000 đến 15,000

6 4. Xây dựng mô hình dự đoán

4.1 Chuẩn bị dữ liệu cho mô hình

[48]:
```
print("Column names in the dataset:")
print(data.columns)
X = data.rename(columns={"Tên_cộ t_cũ_1": "ActualColumn1", "Tên_cộ t_cũ_2":␣
 ↪"ActualColumn2"})
y_classification = data['Attrition']
y_regression = data['PerformanceRating']
```

```
Column names in the dataset:
```

```
Index(['Age', 'Attrition', 'BusinessTravel', 'DailyRate', 'Department',
       'DistanceFromHome', 'Education', 'EducationField', 'EmployeeCount',
       'EmployeeNumber', 'EnvironmentSatisfaction', 'Gender', 'HourlyRate',
       'JobInvolvement', 'JobLevel', 'JobRole', 'JobSatisfaction', 'MaritalStatus',
       'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked', 'Over18', 'OverTime',
       'PercentSalaryHike', 'PerformanceRating', 'RelationshipSatisfaction',
       'StandardHours', 'StockOptionLevel', 'TotalWorkingYears',
       'TrainingTimesLastYear', 'WorkLifeBalance', 'YearsAtCompany',
       'YearsInCurrentRole', 'YearsSinceLastPromotion', 'YearsWithCurrManager',
       'AgeGroup'],
      dtype='object')
```

[49]:
```
X_train_clf, X_test_clf, y_train_clf, y_test_clf = train_test_split(X,
 y_classification, test_size=0.2, random_state=42)
X_train_reg, X_test_reg, y_train_reg, y_test_reg = train_test_split(X,
 y_regression, test_size=0.2, random_state=42)
```

7 4.2 Huấn luyện và đánh giá mô hình phân loại

7.0.1 4.2.1 Hồi quy Logistic

[50]:
```
log_reg = LogisticRegression(max_iter=1000, random_state=42)
log_reg.fit(preprocessor.fit_transform(X_train_clf), y_train_clf) y_pred_clf_log_reg
= log_reg.predict(preprocessor.transform(X_test_clf)) y_pred_proba_log_reg =
log_reg.predict_proba(preprocessor.
 transform(X_test_clf))[:, 1]
```

7.0.2 Hiển thị kết quả của mô hình Hồi quy Logistic

[51]:
```
print("¥nHồ i quy Logistic:")
print(classification_report(y_test_clf, y_pred_clf_log_reg))
print("Ma trậ n nhầ m lẫ n:")
print(confusion_matrix(y_test_clf, y_pred_clf_log_reg))
print("ROC-AUC:", roc_auc_score(y_test_clf, y_pred_proba_log_reg))
```

```
Hồ i quy Logistic:
              precision    recall  f1-score   support

          No       0.49      0.47      0.48      4931
         Yes       0.50      0.53      0.52      5069

    accuracy                           0.50     10000
   macro avg       0.50      0.50      0.50     10000
weighted avg       0.50      0.50      0.50     10000

Ma trậ n lẫ n:
   nhầ m
```

```
[[2301 2630]
 [2398 2671]]
ROC-AUC: 0.4985242989674954
```

7.0.3 4.2.2 Mô hình rfing ngẫu nhiên (Phân loại)

[52]:
```
rf_clf = RandomForestClassifier(random_state=42)
param_grid_rf_clf = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 5, 10]
}
grid_search_rf_clf = GridSearchCV(estimator=rf_clf, ␣
  ↪param_grid=param_grid_rf_clf, cv=5, n_jobs=-1, verbose=2, scoring='accuracy')
grid_search_rf_clf.fit(preprocessor.fit_transform(X_train_clf), y_train_clf)
y_pred_clf_rf = grid_search_rf_clf.predict(preprocessor.transform(X_test_clf))
y_pred_proba_rf_clf = grid_search_rf_clf.predict_proba(preprocessor.
  ↪transform(X_test_clf))[:, 1]
```

7.0.4 Hiển thị kết quả mô hình RandomForestClassifier

[53]:
```
print("¥nMô hình rừ ng ngẫ u nhiên (Phân loạ i):")
print(classification_report(y_test_clf, y_pred_clf_rf))
print("Ma trậ n nhầ m lẫ n:")
print(confusion_matrix(y_test_clf, y_pred_clf_rf))
print("ROC-AUC:", roc_auc_score(y_test_clf, y_pred_proba_rf_clf)) print("Tham
số  tố t nhấ t:", grid_search_rf_clf.best_params_)
```

```
Mô hình rừ ng ngẫ u       (Phân loạ i):
              nhiên         recall  f1-score     support
              precision

          No       0.50      0.49      0.49        4931
         Yes       0.51      0.51      0.51        5069

    accuracy                           0.50       10000
   macro avg       0.50      0.50      0.50       10000
weighted avg       0.50      0.50      0.50       10000

Ma trậ n      lẫ n:
nhầ m [[2428
2503]
 [2473 2596]]
ROC-AUC: 0.49875282248751446
Tham số  tố t nhấ t: {'max_depth': 10, 'min_samples_split': 10, 'n_estimators': 50}
```

8 4.2.3 Mô hình XGBoost (Phân loại)

[54]:
```
from sklearn.preprocessing import LabelEncoder
from xgboost import XGBClassifier
from sklearn.model_selection import GridSearchCV

# Khở i tạ o LabelEncoder
label_encoder = LabelEncoder()

# Huấ n luyệ n LabelEncoder vớ i dữ liệ u huấ n luyệ n và chuyể n đổ i nhã n y
y_train_encoded = label_encoder.fit_transform(y_train_clf)

# Khở i tạ o mô hình XGBClassifier
xgb_clf = XGBClassifier(random_state=42)

# Thiế t lậ p tham số cho Grid Search
param_grid_xgb_clf = { 'n_estimators':
    [50, 100, 200],
    'learning_rate': [0.01, 0.1, 0.3],
    'max_depth': [3, 5, 7]
}

# Huấ n luyệ n mô hình vớ i nhã n đã đượ c chuyể n đổ i
grid_search_xgb_clf = GridSearchCV(estimator=xgb_clf,␣
 ↪param_grid=param_grid_xgb_clf, cv=5, n_jobs=-1, verbose=2,␣
 ↪scoring='accuracy')
grid_search_xgb_clf.fit(preprocessor.fit_transform(X_train_clf),␣
 ↪y_train_encoded)
# Dự đoá n nhã n
y_pred_clf_xgb = grid_search_xgb_clf.predict(preprocessor.transform(X_test_clf))
y_pred_clf_xgb = label_encoder.inverse_transform(y_pred_clf_xgb)

# Dự đoá n xá c suấ t
y_pred_proba_xgb_clf = grid_search_xgb_clf.predict_proba(preprocessor.
 ↪transform(X_test_clf))[:, 1]
```

```
Fitting 5 folds for each of 27 candidates, totalling 135 fits
```

8.0.1 Hiển thị kết quả Mô hình XGBoost

[55]:
```
print("¥nMô hình XGBoost (Phân loạ i):")
print(classification_report(y_test_clf, y_pred_clf_xgb))
print("Ma trậ n nhầ m lẫ n:")
print(confusion_matrix(y_test_clf, y_pred_clf_xgb))
print("ROC-AUC:", roc_auc_score(y_test_clf, y_pred_proba_xgb_clf)) print("Tham
số tố t nhấ t:", grid_search_xgb_clf.best_params_)
```

```
Mô hình XGBoost (Phân loạ i):
              precision    recall  f1-score   support

          No       0.49      0.46      0.47      4931
         Yes       0.51      0.54      0.52      5069

    accuracy                           0.50     10000
   macro avg       0.50      0.50      0.50     10000
weighted avg       0.50      0.50      0.50     10000

Ma trậ n      lẫ n:
nhầ m [[2244
2687]
 [2322 2747]]
ROC-AUC: 0.5021823156001829
Tham số  tố t nhấ t: {'learning_rate': 0.01, 'max_depth': 5, 'n_estimators': 50}
```

9 4.2.4 Mô hình LightGBM (Phân loại) :

[ ]:
```
lgb_clf = LGBMClassifier(random_state=42)
param_grid_lgb_clf = {
    'n_estimators': [50, 100, 200],
    'learning_rate': [0.01, 0.1, 0.3],
    'max_depth': [3, 5, 7]
}
grid_search_lgb_clf = GridSearchCV(estimator=lgb_clf, ␣
  ↪param_grid=param_grid_lgb_clf, cv=5, n_jobs=-1, verbose=2, ␣
  ↪scoring='accuracy')
grid_search_lgb_clf.fit(preprocessor.fit_transform(X_train_clf), y_train_clf)
y_pred_clf_lgb = grid_search_lgb_clf.predict(preprocessor.transform(X_test_clf))
y_pred_proba_lgb_clf = grid_search_lgb_clf.predict_proba(preprocessor.
  ↪transform(X_test_clf))[:, 1]
```

```
[LightGBM] [Info] Auto-choosing col-wise multi-threading, the overhead of testing
was 0.004368 seconds.
You can set `force_col_wise=true` to remove the overhead. [LightGBM]
[Info] Total Bins 952
[LightGBM] [Info] Number of data points in the train set: 40000, number of used
features: 50
[LightGBM] [Info] [binary:BoostFromScore]: pavg=0.500900 -> initscore=0.003600
[LightGBM] [Info] Start training from score 0.003600
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
```

```
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
```

9.0.1 Hiển thị kết quả Mô hình LightGBM

[ ]:
```
print("¥nMô hình LightGBM (Phân loạ i):")
print(classification_report(y_test_clf, y_pred_clf_lgb))
print("Ma trậ n nhầ m lẫ n:")
print(confusion_matrix(y_test_clf, y_pred_clf_lgb))
print("ROC-AUC:", roc_auc_score(y_test_clf, y_pred_proba_lgb_clf)) print("Tham
số  tố t nhấ t:", grid_search_lgb_clf.best_params_)
```

```
Mô hình LightGBM (Phân loạ i):
              precision    recall  f1-score   support

          No       0.49      0.50      0.49      4931
         Yes       0.50      0.50      0.50      5069

    accuracy                           0.50     10000
   macro avg       0.50      0.50      0.50     10000
weighted avg       0.50      0.50      0.50     10000

Ma trậ n     lẫ n:
nhầ m [[2449
2482]
 [2558 2511]]
ROC-AUC: 0.4991459773599284
Tham số  tố t nhấ t: {'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 100}
```

10 4.2.5 Huấn luyện và đánh giá mô hình hồi quy

10.0.1 4.2.5_a RandomForestRegressor:

[57]:
```
rf_reg = RandomForestRegressor(random_state=42)
param_grid_rf_reg = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20],
```

```
    'min_samples_split': [2, 5, 10]
}
grid_search_rf_reg = GridSearchCV(estimator=rf_reg,␣
  ▫param_grid=param_grid_rf_reg, cv=5, n_jobs=-1, verbose=2,␣
  ▫scoring='neg_mean_squared_error')
grid_search_rf_reg.fit(preprocessor.fit_transform(X_train_reg), y_train_reg)
y_pred_reg_rf = grid_search_rf_reg.predict(preprocessor.transform(X_test_reg))
```

```
Fitting 5 folds for each of 27 candidates, totalling 135 fits
```

11 Hiển thị kết quả RandomForestRegressor

[60]:
```
print("¥nMô hình Random Forest Regressor:")
print("MSE:", mean_squared_error(y_test_reg, y_pred_reg_rf)) print("RMSE:",
np.sqrt(mean_squared_error(y_test_reg, y_pred_reg_rf))) print("MAE:",
mean_absolute_error(y_test_reg, y_pred_reg_rf)) print("R2:",
r2_score(y_test_reg, y_pred_reg_rf))
print("Tham số  tố t nhấ t:", grid_search_rf_reg.best_params_)
```

```
Mô hình Random Forest Regressor:
MSE: 1.2460844533233608
RMSE: 1.116281529598766
MAE: 0.9967712956859366
R2: -0.0019254828813703995
Tham số  tố t nhấ t: {'max_depth': 10, 'min_samples_split': 10, 'n_estimators':
200}
```

12 4.2.5 b Mô hình Gradient Boosting (Hồi quy)

[ ]:
```
gb_reg = GradientBoostingRegressor(random_state=42)
param_grid_gb_reg = {
    'n_estimators': [50, 100, 200],
    'learning_rate': [0.01, 0.1, 0.3],
    'max_depth': [3, 5, 7]
}
grid_search_gb_reg = GridSearchCV(estimator=gb_reg,␣
  ▫param_grid=param_grid_gb_reg, cv=5, n_jobs=-1, verbose=2,␣
  ▫scoring='neg_mean_squared_error')
grid_search_gb_reg.fit(preprocessor.fit_transform(X_train_reg), y_train_reg)
y_pred_reg_gb = grid_search_gb_reg.predict(preprocessor.transform(X_test_reg))
```

13 Hiện kết quả Mô hình Gradient Boosting

[ ]:
```
print("¥nMô hình Gradient Boosting (Hồ i quy):") print("MSE:",
mean_squared_error(y_test_reg, y_pred_reg_gb))
print("RMSE:", np.sqrt(mean_squared_error(y_test_reg, y_pred_reg_gb)))
print("MAE:", mean_absolute_error(y_test_reg, y_pred_reg_gb)) print("R2:",
r2_score(y_test_reg, y_pred_reg_gb))
print("Tham số  tố t nhấ t:", grid_search_gb_reg.best_params_)
```

```
Mô hình Gradient Boosting (Hồ i quy):
MSE: 1.2441419266228786
RMSE: 1.1154111020708368
MAE: 0.9970273038475392
R2: -0.00036357670623132776
Tham số  tố t nhấ t: {'learning_rate': 0.01, 'max_depth': 3, 'n_estimators': 50}
```

14 5 Đánh giá kết luận và Đưa ra Đề xuất

[59]:

14.0.1 5.1 Tầm quan trọng của đặc trưng

```
importances = grid_search_rf_clf.best_estimator_.feature_importances_ feature_names =
preprocessor.get_feature_names_out()
feature_importance_df = pd.DataFrame({'Đặ c trư ng': feature_names, 'Tầ m quan␣
↪trọ ng': importances})
feature_importance_df = feature_importance_df.sort_values(by='Tầ m quan trọ ng', ␣
↪ascending=False)
print("¥nTầ m quan trọ ng củ a đặ c trư ng:")
print(feature_importance_df.head(10))
```

	Đặ c trư ng	Tầ m quan trọ ng
3	num__MonthlyIncome	0.101511
1	num__DailyRate	0.097276
2	num__DistanceFromHome	0.069968
5	num__PercentSalaryHike	0.069916
6	num__TotalWorkingYears	0.064528
0	num__Age	0.057827
8	num__YearsAtCompany	0.056697
11	num__YearsWithCurrManager	0.042766
9	num__YearsInCurrentRole	0.041448
4	num__NumCompaniesWorked	0.041207

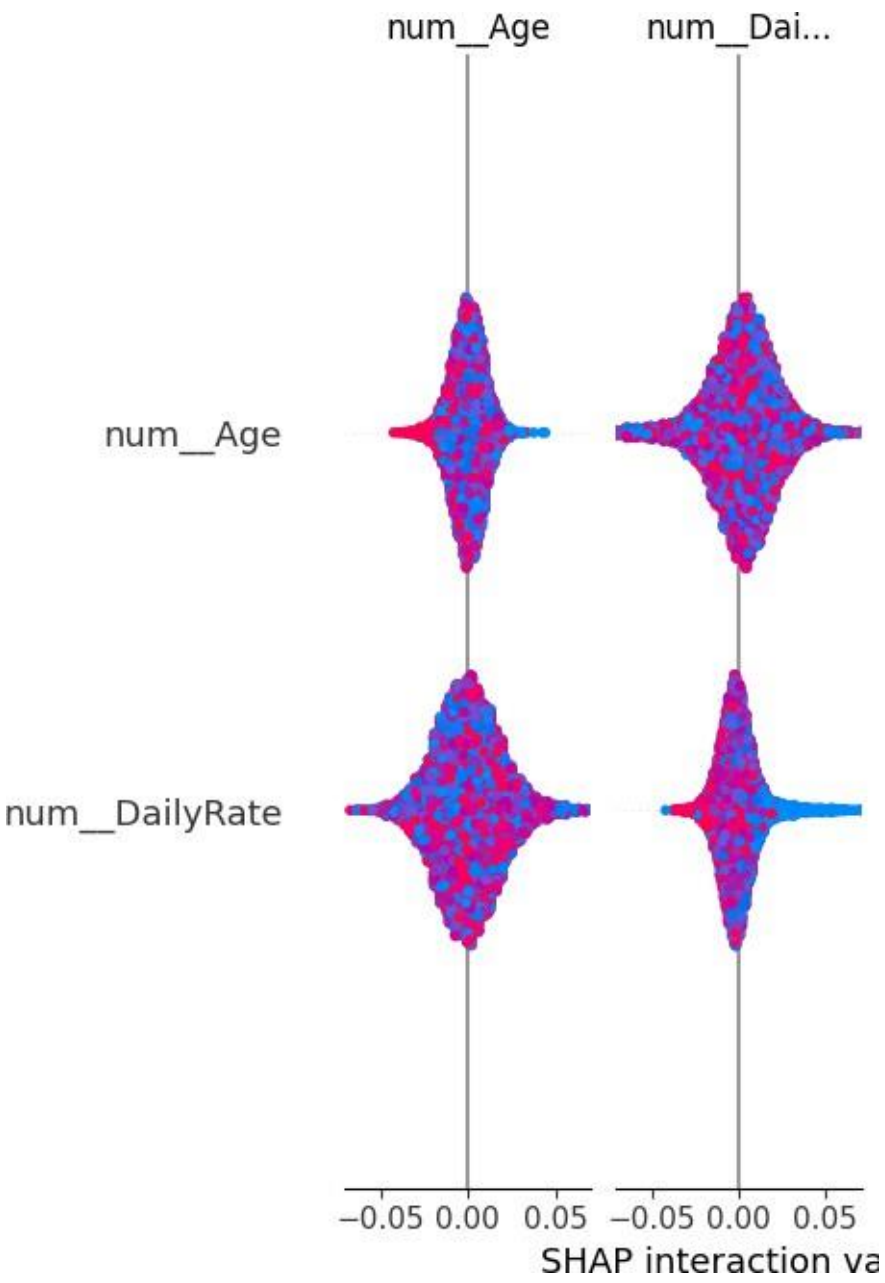
MonthlyIncome (thu nhập hàng tháng) là yếu tố quan trọng nhất trong việc dự đoán, chiếm **10.15%** tầm quan trọng. Điều này cho thấy **mức lương** có thể là một trong những yếu tố **ảnh hưởng lớn** đến quyết định nghỉ việc của nhân viên.

Các yếu tố tiếp theo có tầm quan trọng **tương đối đồng đều**, dao động từ **4% đến 7%**, bao gồm:

- **DailyRate** (lương ngày)
- **DistanceFromHome** (khoảng cách từ nhà đến công ty)
- **PercentSalaryHike** (phần trăm tăng lương)
- **TotalWorkingYears** (tổng số năm kinh nghiệm)
- **Age** (tuổi)
- **YearsAtCompany** (số năm làm việc tại công ty)
- **YearsWithCurrManager** (số năm làm việc với quản lý hiện tại)
- **YearsInCurrentRole** (số năm làm việc ở vị trí hiện tại)
- **NumCompaniesWorked** (số công ty đã làm việc)

15. 5.2 Giá trị SHAP

[ ]:
```
explainer = shap.TreeExplainer(grid_search_rf_clf.best_estimator_)
shap_values = explainer.shap_values(preprocessor.transform(X_test_clf))
shap.summary_plot(shap_values, preprocessor.transform(X_test_clf),␣
 feature_names=feature_names)
```



Nhận xét giá trị Shap

Tương tác giữa ”num__Age” và ”num__DailyRate”: Hai biểu đồ ở phía trên bên phải và phía dưới bên trái cho thấy **sfi tương tác yếu** giữa tuổi và lương ngày. Các điểm dữ liệu phân bố tương đối đồng đều, không có xu hướng rõ ràng cho thấy **tác động kết hợp của hai biến này không đáng kể**.

Tác động chính của ”num__Age” và ”num__DailyRate”: Hai biểu đồ ở phía trên bên trái và phía dưới bên phải cho thấy tác động chính của từng biến đến kết quả dự đoán.

- **num__Age:** Các điểm dữ liệu màu xanh (giá trị ”num__Age” thấp) tập trung ở phía âm, cho thấy **tuổi thấp** có xu hướng **làm giảm kết quả dfi đoán**. Ngược lại, các điểm dữ liệu màu đỏ (giá trị ”num__Age” cao) tập trung nhiều hơn ở phía dương, cho thấy **tuổi cao** có xu hướng **làm tăng kết quả dfi đoán**.
- **num__DailyRate:** Các điểm dữ liệu phân bố tương đối đồng đều ở cả hai phía âm và dương, cho thấy **lương ngày** có **tác động không rõ ràng** đến kết quả dự đoán.

Kết luận:

- **Tương tác gifia tuổi và lương ngày** có **ảnh hưởng yếu** đến kết quả dự đoán.
- **Tuổi** có **ảnh hưởng đáng kể** đến kết quả dự đoán, trong khi **lương ngày** có **ảnh hưởng không rõ ràng**.

Nhận xét chung: + Các mô hình dự đoán :Hồi quy Logistic,RandomForestClassifier,Mô hình XGBoost ,Mô hình LightGBM có tỉ lệ dự đoán tương đối chưa hiệu quả với tỉ lệ Accuracy khoảng dao động ở mức 50% + Hai mô hình Gradient Boosting và Random Forest Regressor có MSE,RMSE,MAE xấp xỉ từ 0.99 tơi 1.24 cho thấy sai số dự đoán lớn,Đặc biệt đáng chú ý là R2 gần bằng 0 và âm cho thấy mô hình dự đoán kém hiệu quả hơn so với việc chỉ sử dụng giá trị trung bình ,Gradient Boostin dự đoán ổn hơn Random Forest Regressor