Machine Learning II Project

on

Employee Attrition Prediction with ML Algorithms in Python

Submitted by

Ranjana Chakraverty

Registration No.: 12014011

Program Name: B.Tech. (CSE - Data Science (ML and AI))

Under the guidance of

Aishwary Shukla

School of Computer Science and Engineering

Lovely Professional University, Phagwara

(August-October, 2023)

DECLARATION

I hereby declare that I have completed my machine learning project from 23rd August 2023 to 25th October 2023 under the guidance of Aishwary Shukla. I have worked with full dedication during these eight weeks of training and my learning outcomes fulfil the requirements of training for the award of degree of B.Tech. (CSE - Data Science (ML and AI)), Lovely Professional University, Phagwara.

Ranjana Chakraverty

Registration Number: 12014011

Date: 24th October 2023

ACKNOWLEDGEMENT

I would like to thank my professors at LPU of the School of Computer Science and Engineering for their support. I would like to thank the instructors at Upgrad, specially Aishwary Shukla and Ejaz Ahmed Qazi, who were very helpful and always available to give clear my doubts.

TABLE OF CONTENTS

1. Cover Page 01
2. Declaration 02
3. Acknowledgement 03
4. Table of Contents 04
5. Objective and Scope 05
6. Introduction 06
7. Profile & Analysis of the Problem 08
8. Method 10
9. Result 11
10. Bibliography 12
11. Python Notebook 13

OBJECTIVE AND SCOPE

Predicting whether an employee is going to leave the job. In ML jargon, this is known as attrition. Attrition refers to departures that are voluntary and/or "natural"—such as retirement or a position being discontinued.

One of the biggest concerns employers have while hiring employees is the possibility of an early attrition. A lot of capital is spent on hiring by companies. Very often, if they hire unskilled graduates, fresh out of a Bachelor's degree, they also must spend money to train the theoretically strong but practically weak beginners. Acceptable salaries are also negotiated upon. After spending so much on taking in new employees, companies are quite rudely shocked when their employees start resigning, sometimes within a year of working.

It is an endeavour of companies to spot any such parts of an employee's profile which make them more likely to leave jobs early without contributing proportionately. With this knowledge, companies can make sure to tailor their hiring process to only take in those candidates who are committed enough to stick around till the long haul.

This project is concerned with building a machine learning model for binary classification on whether an employee will attrition or not.

An unusually high rate of employee attrition is considered indicative of problems within the organization. Uncompetitive pay scales, micromanagement, ineffective human resource management (HRM) practices and unreasonable expectations can all lead to unacceptable levels of attrition.

Employees who stay at a company for longer tend to be less stressed, more committed to the long-term success of the company, more productive and more likely to recommend the company as a place to work.

The model I will build in this project is probably already in use in many companies to predict employee attrition. Companies can use information obtained from common trends in employees who attrition and use it to improve their working conditions or incentives. It has high applicability in real life and scope for use in many companies with similar employee profiles.

INTRODUCTION

An employee's departure is considered attrition if it meets the following criteria:

- The departure is voluntary.
- The company is not rehiring or re-filling the position.

An example of employee attrition may be an employee who retired or quit because they are moving to another location. Attrition is not always a sign that something is wrong in an organization and may be a part of large enterprise strategic decision.

Attrition, as applied to an organization's workforce, is a measurement of the reduction of staff during a set period of time. Most companies measure it annually. It encompasses all reasons for separation including resignation, termination, or retirement. If an employee is replaced, the separation is not included in the rate of attrition.

Employers need to monitor the employee attrition rate because it impacts productivity, business performance and growth.

Causes

While many factors add to employee attrition, some easily stand out and need management attention.

- Lack of growth plan and opportunities: Every employee needs to have a defined growth opportunities path that must be conveyed clearly. Lack of this can lead to dissent and doubts that their efforts are unrecognized. This could lead to resentment and employees leaving the organisation.
- Skewed work-life balance: Striking the right work-life balance is crucial for employees, be it for family, hobbies, or pursuing higher studies. Not having the right balance or organizations not helping find it can cause employees to exit the organization.
- Bad workplace culture: Having a positive culture is essential for organizations. A culture where employees are appreciated, valued, and rewarded is highly appreciated. If employees feel the work culture is toxic and non-conducive to their growth, they will leave the organization.
- Dissatisfactory appraisals: Many organizations follow the yearly appraisal process. Employees eagerly wait for that for the entire year. They may get disheartened if they feel the assessment is not in line with their efforts and contributions. Some employees

may stick it out for another cycle, but most will leave as early as possible. Because attrition occurs voluntarily rather than as a result of a lack of job satisfaction, it usually isn't seen as something bad but rather as a normal part of the employee life cycle

- Low staff morale: The above factors can affect employee morale and bring it down significantly. It could either be one of the elements or a combination, but employees with low morale are a significant employee

Adverse Effects

A high attrition rate impacts an organization in several ways. First, it can cripple a company financially because costs for recruiting, hiring, and training are significant. Employers who reduce their attrition rate can save money on hiring and related costs which can, in turn, increase their profit margin.

Secondly, a high attrition rate limits a company because it results in a staff that is perennially composed of less-experienced employees. This puts a burden on the employees with more skills and experience. They may grow to resent shouldering the bulk of the workload. If it leads to experienced employees quitting, it can trigger a downward spiral that is hard to reverse.

Thirdly, a high attrition rate damages the company's reputation which, in turn, can dissuade customers, investors and make it harder to attract new employees.

Reduction Measures

It is critical for employers to compare their attrition rate to other employers in their industry and employment market. Employers who identify an increase in their attrition rate can take measures to address it. These include using a structured onboarding process, making sure the company's benefits package is competitive, improving management practices, providing flexible schedules and supporting employee work/life balance in other ways, conducting exit interviews to determine why employees are quitting, and providing professional development programs so employees can progress along a career path in the organization.

Managing for employee retention involves strategic actions to keep employees motivated and focused so they elect to remain employed and fully productive for the benefit of the organization. A comprehensive employee retention program can play a vital role in both attracting and retaining key employees, as well as in reducing turnover and its related costs. All of these contribute to an organization's productivity and overall business performance.

PROFILE & ANALYSIS OF THE PROBLEM

The dataset used for this project has been sourced from Kaggle. It is human resourced (HR) data from IBM. It contains employee data that are thought to contribute to attrition. This is a fictional data set that has been created by data scientists at IBM.

The columns in the dataset are:

- Age (numerical)
- Attrition (categorical)
- BusinessTravel (categorical)
- DailyRate (numerical)
- Department (categorical)
- DistanceFromHome (numerical)
- Education (categorical)
- EducationField (categorical)
- EmployeeCount (numerical)
- EmployeeNumber (numerical)
- EnvironmentSatisfaction (numerical)
- Gender (categorical)
- HourlyRate (numerical)
- JobInvolvement (numerical)
- JobLevel (numerical)
- JobRole (categorical)
- JobSatisfaction (numerical)
- MaritalStatus (categorical)
- MonthlyIncome (numerical)
- MonthlyRate (numerical)
- NumCompaniesWorked (numerical)
- Over18 (categorical)
- OverTime (categorical)

- PercentSalaryHike (numerical)
- PerformanceRating (numerical)
- RelationshipSatisfaction (numerical)
- StandardHours (numerical)
- StockOptionLevel (numerical)
- TotalWorkingYears (numerical)
- TrainingTimesLastYear (numerical)
- WorkLifeBalance (numerical)
- YearsAtCompany (numerical)
- YearsInCurrentRole (numerical)
- YearsSinceLastPromotion (numerical)
- YearsWithCurrManager (numerical)

Based on the above columns, the task was to build a classification-based machine learning model that can correctly predict employee attrition with high performance and accuracy. This will output values in the form of either 0 or 1 where 0 means no attrition and 1 means attrition.

METHOD

Out of the many possible machine learning algorithms available for binary classification, I chose the random forest classifier. This is an algorithm that works excellently for classification problems. It is an advanced ML algorithm which is very apt for this problem.

As with any machine learning project, I followed a set of well-defined steps before proceeding with the model building. These were the steps I took.

1. Importing relevant libraries and data
2. Exploratory Data Analysis (EDA)
    a. Data exploration
    b. Univariate analysis
    c. Bivariate Analysis
    d. Multivariate Analysis
    e. Visualisations
    f. Outlier analysis
    g. Checking for null values
    h. Checking for missing values
    i. Removing irrelevant columns
    j. Statistical Analysis
    k. Checking for multicollinearity
3. Data preprocessing
4. Building the random forest classifier
5. Evaluating the model with evaluation metrics

The python notebook can be found at the end of this document.

RESULT

The model has an accuracy of 97.9 %.

	precision	recall
0.0	0.98	1.00
1.0	1.00	0.87

Where 0 means no attrition and 1 means attrition

This is an excellent value for accuracy and goes to show how powerful ensemble models like random forest classifiers are. They incentivise choosing them over models like logistic regression which pale in comparison in terms of most classification evaluation metrics.

BIBLIOGRAPHY

nilohealth.com

workforcehub.com

questionpro.com

whatfix.com

This project aims to accurately predict whether or not an employee will leave his/her job in the future.

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

In [2]:
```
df = pd.read_csv("/content/WA_Fn-UseC_-HR-Employee-Attrition.csv")
```

Exploratory Data Analysis

In [3]:
```
df.shape
```

Out[3]: (1470, 35)

In [4]:
```
df.describe()
```

Out[4]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	1470.000000
mean	36.923810	802.485714	9.192517	2.912925	1.0	1024.865306
std	9.135373	403.509100	8.106864	1.024165	0.0	602.024335
min	18.000000	102.000000	1.000000	1.000000	1.0	1.000000
25%	30.000000	465.000000	2.000000	2.000000	1.0	491.250000
50%	36.000000	802.000000	7.000000	3.000000	1.0	1020.500000
75%	43.000000	1157.000000	14.000000	4.000000	1.0	1555.750000
max	60.000000	1499.000000	29.000000	5.000000	1.0	2068.000000

8 rows × 26 columns

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 35 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   Age                       1470 non-null   int64 
 1   Attrition                 1470 non-null   object
 2   BusinessTravel            1470 non-null   object
 3   DailyRate                 1470 non-null   int64 
 4   Department                1470 non-null   object
 5   DistanceFromHome          1470 non-null   int64 
 6   Education                 1470 non-null   int64 
```

```
 7   EducationField            1470 non-null   object
 8   EmployeeCount             1470 non-null   int64
 9   EmployeeNumber            1470 non-null   int64
 10  EnvironmentSatisfaction   1470 non-null   int64
 11  Gender                    1470 non-null   object
 12  HourlyRate                1470 non-null   int64
 13  JobInvolvement            1470 non-null   int64
 14  JobLevel                  1470 non-null   int64
 15  JobRole                   1470 non-null   object
 16  JobSatisfaction           1470 non-null   int64
 17  MaritalStatus             1470 non-null   object
 18  MonthlyIncome             1470 non-null   int64
 19  MonthlyRate               1470 non-null   int64
 20  NumCompaniesWorked        1470 non-null   int64
 21  Over18                    1470 non-null   object
 22  OverTime                  1470 non-null   object
 23  PercentSalaryHike         1470 non-null   int64
 24  PerformanceRating         1470 non-null   int64
 25  RelationshipSatisfaction  1470 non-null   int64
 26  StandardHours             1470 non-null   int64
 27  StockOptionLevel          1470 non-null   int64
 28  TotalWorkingYears         1470 non-null   int64
 29  TrainingTimesLastYear     1470 non-null   int64
 30  WorkLifeBalance           1470 non-null   int64
 31  YearsAtCompany            1470 non-null   int64
 32  YearsInCurrentRole        1470 non-null   int64
 33  YearsSinceLastPromotion   1470 non-null   int64
 34  YearsWithCurrManager      1470 non-null   int64
dtypes: int64(26), object(9)
memory usage: 402.1+ KB
```

In [6]:

```
df.head()
```

Out[6]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	Educatio
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sc
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sc
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sc
4	27	No	Travel_Rarely	591	Research & Development	2	1	M

5 rows × 35 columns

In [7]:

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

Out[7]:

```
Age                         0
Attrition                   0
BusinessTravel              0
DailyRate                   0
Department                  0
DistanceFromHome            0
Education                   0
EducationField              0
EmployeeCount               0
EmployeeNumber              0
EnvironmentSatisfaction     0
```

```
Gender                      0
HourlyRate                  0
JobInvolvement              0
JobLevel                    0
JobRole                     0
JobSatisfaction             0
MaritalStatus               0
MonthlyIncome               0
MonthlyRate                 0
NumCompaniesWorked          0
Over18                      0
OverTime                    0
PercentSalaryHike           0
PerformanceRating           0
RelationshipSatisfaction    0
StandardHours               0
StockOptionLevel            0
TotalWorkingYears           0
TrainingTimesLastYear       0
WorkLifeBalance             0
YearsAtCompany              0
YearsInCurrentRole          0
YearsSinceLastPromotion     0
YearsWithCurrManager        0
dtype: int64
```

In [8]:

```
print(df.EmployeeCount.unique())
print(df.Over18.unique())
print(df.StandardHours.unique())
```

```
[1]
['Y']
[80]
```

Since, these columns have only one value in the whole dataset, they have absolutely no use in this classification problem. I will remove these columns.

In [9]:

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

The EmployeeNumber is basically the unique ID of each employee. It has no relevance in the analysis and shall be removed.

In [10]:

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

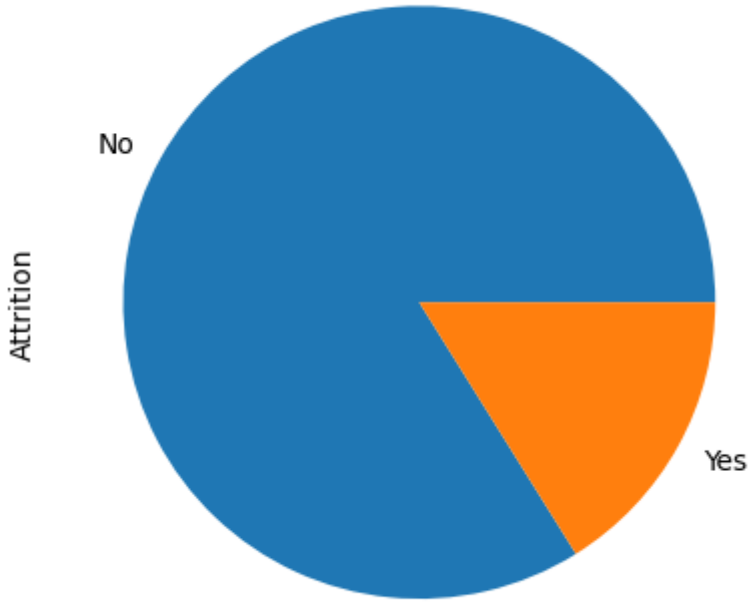
Univariate Analysis

In [11]:

```
df.Attrition.value_counts().plot.pie()
```

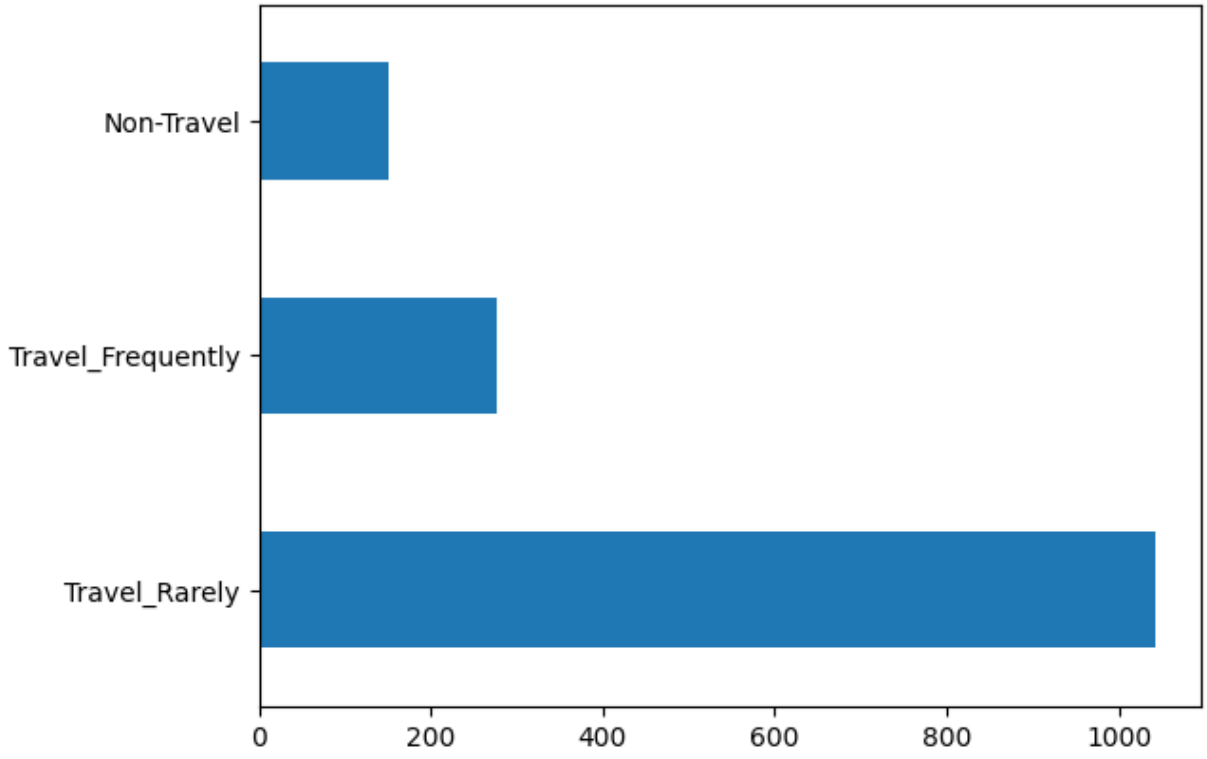
Out[11]:

```
<Axes: ylabel='Attrition'>
```



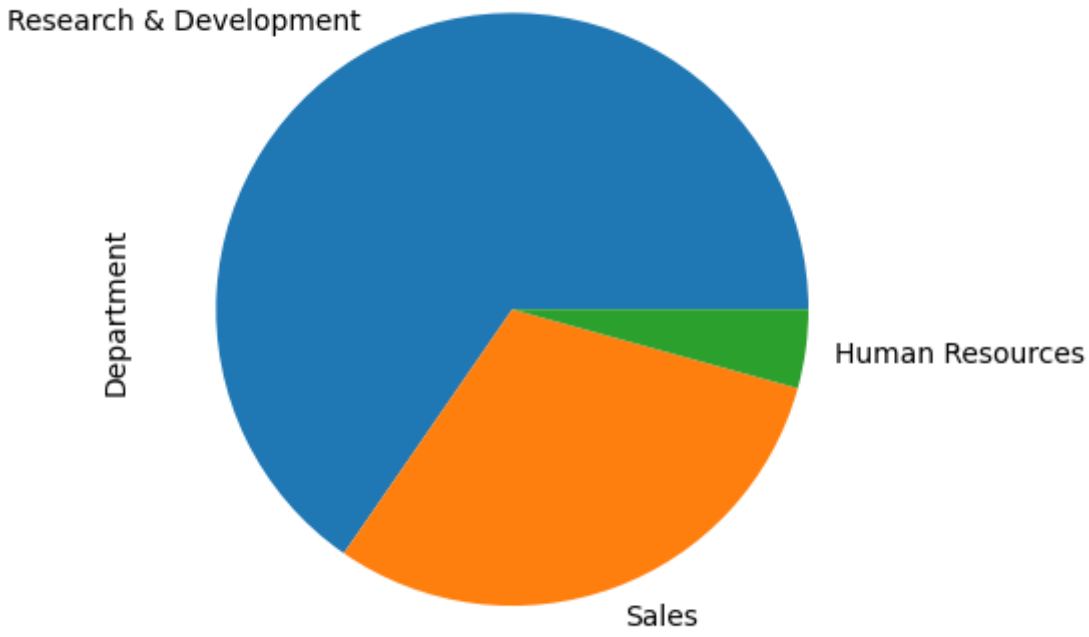
In [12]:
```
df.BusinessTravel.value_counts().plot.barh()
```

Out[12]: <Axes: >



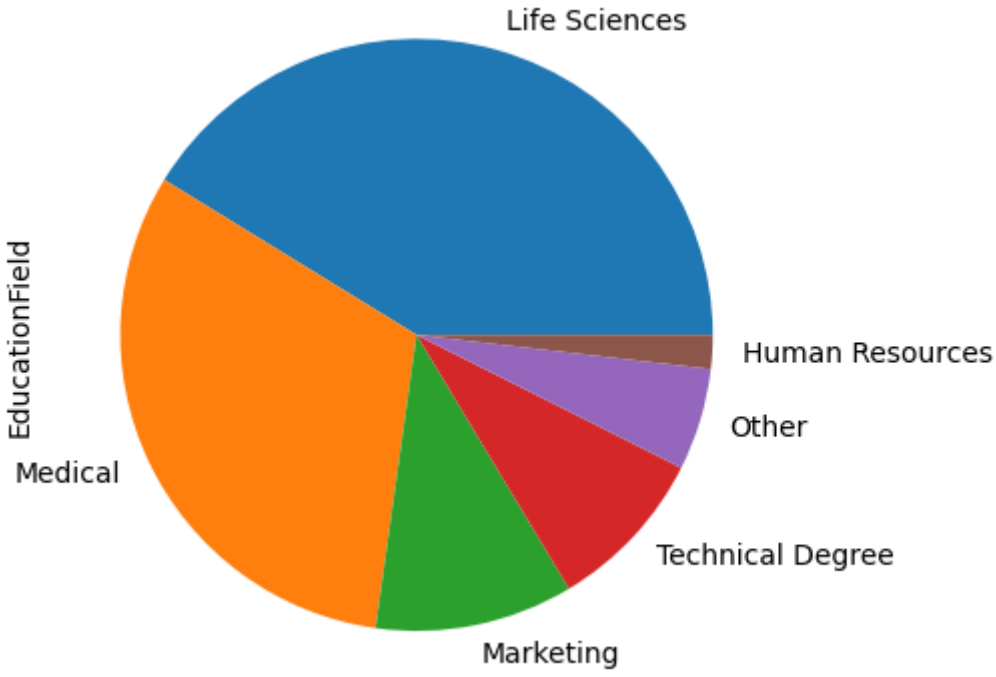
In [13]:
```
df.Department.value_counts().plot.pie()
```

Out[13]: <Axes: ylabel='Department'>



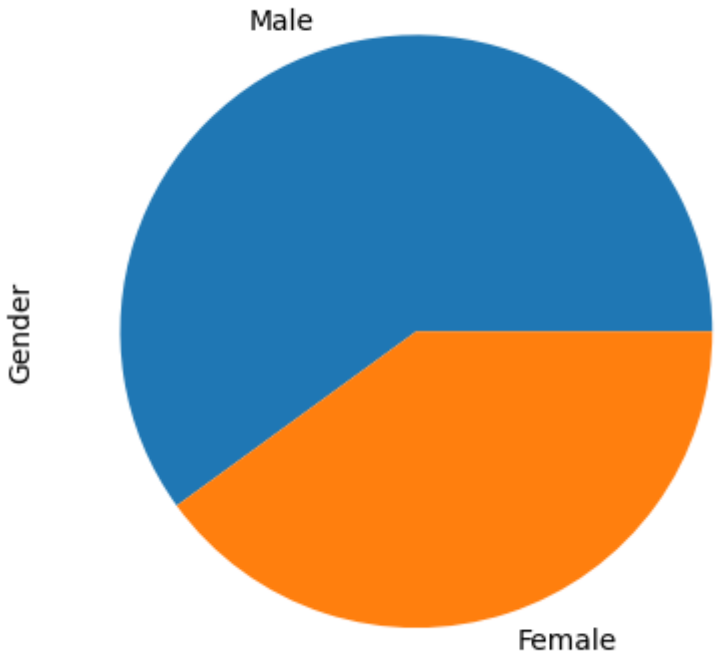
In [14]:
```
df.EducationField.value_counts().plot.pie()
```

Out[14]: <Axes: ylabel='EducationField'>



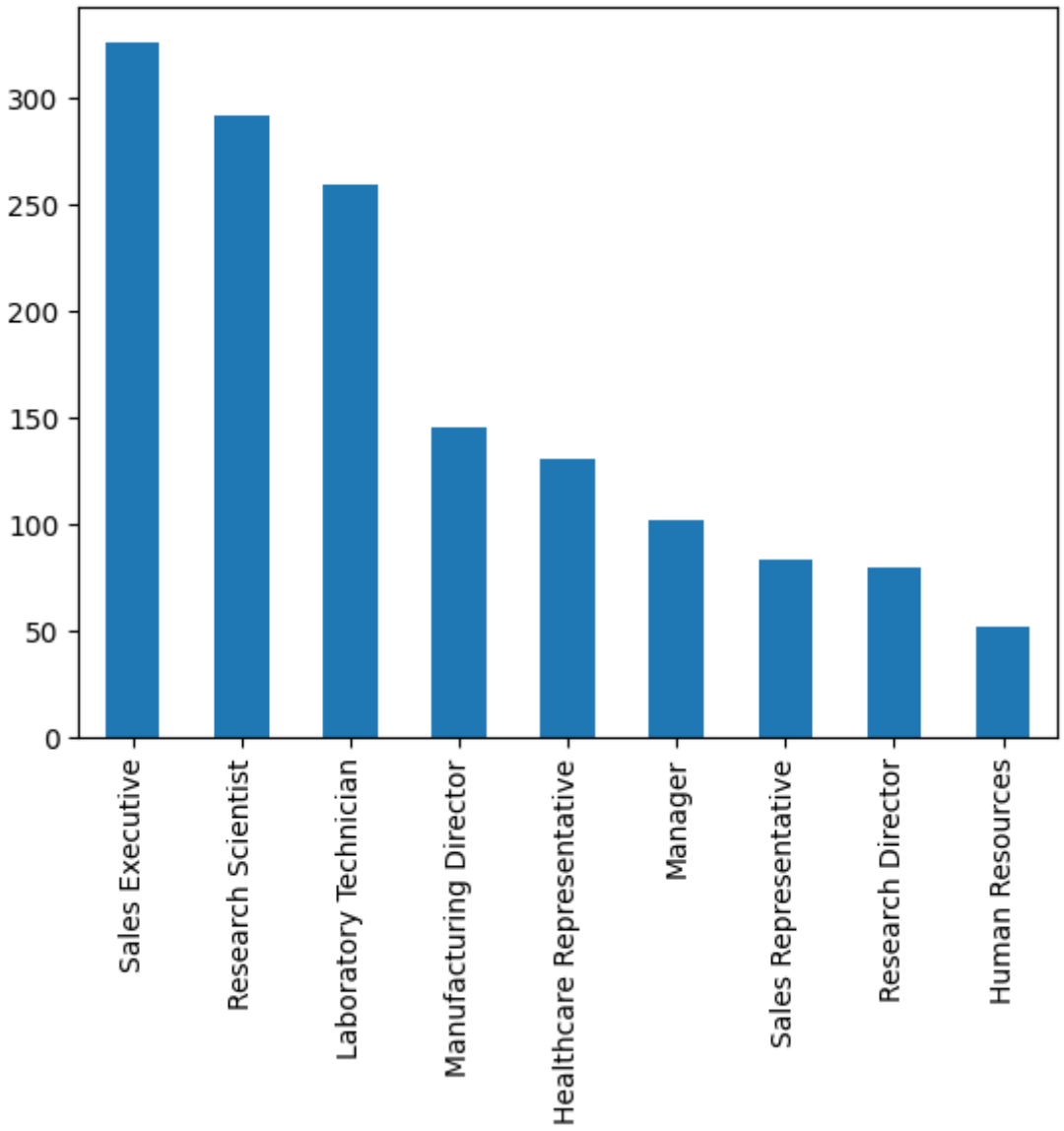
In [15]:
```
df.Gender.value_counts().plot.pie()
```

Out[15]: <Axes: ylabel='Gender'>



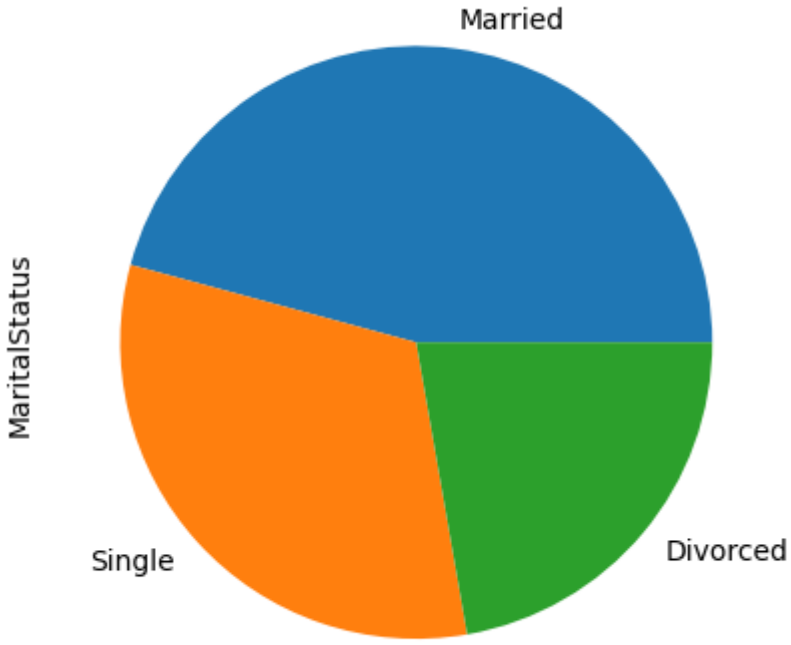
In [16]:
```
df.JobRole.value_counts().plot.bar()
```

Out[16]: <Axes: >



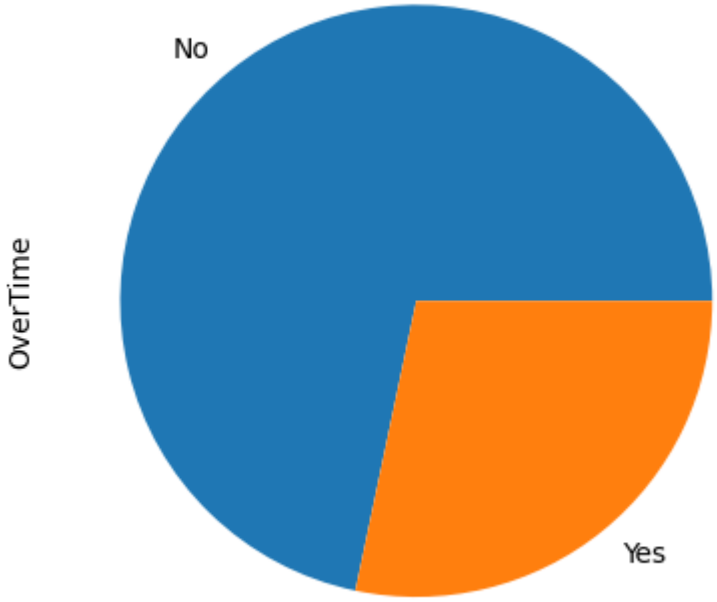
In [17]:
```
df.MaritalStatus.value_counts().plot.pie()
```

Out[17]: <Axes: ylabel='MaritalStatus'>



In [18]:
```
df.OverTime.value_counts().plot.pie()
```

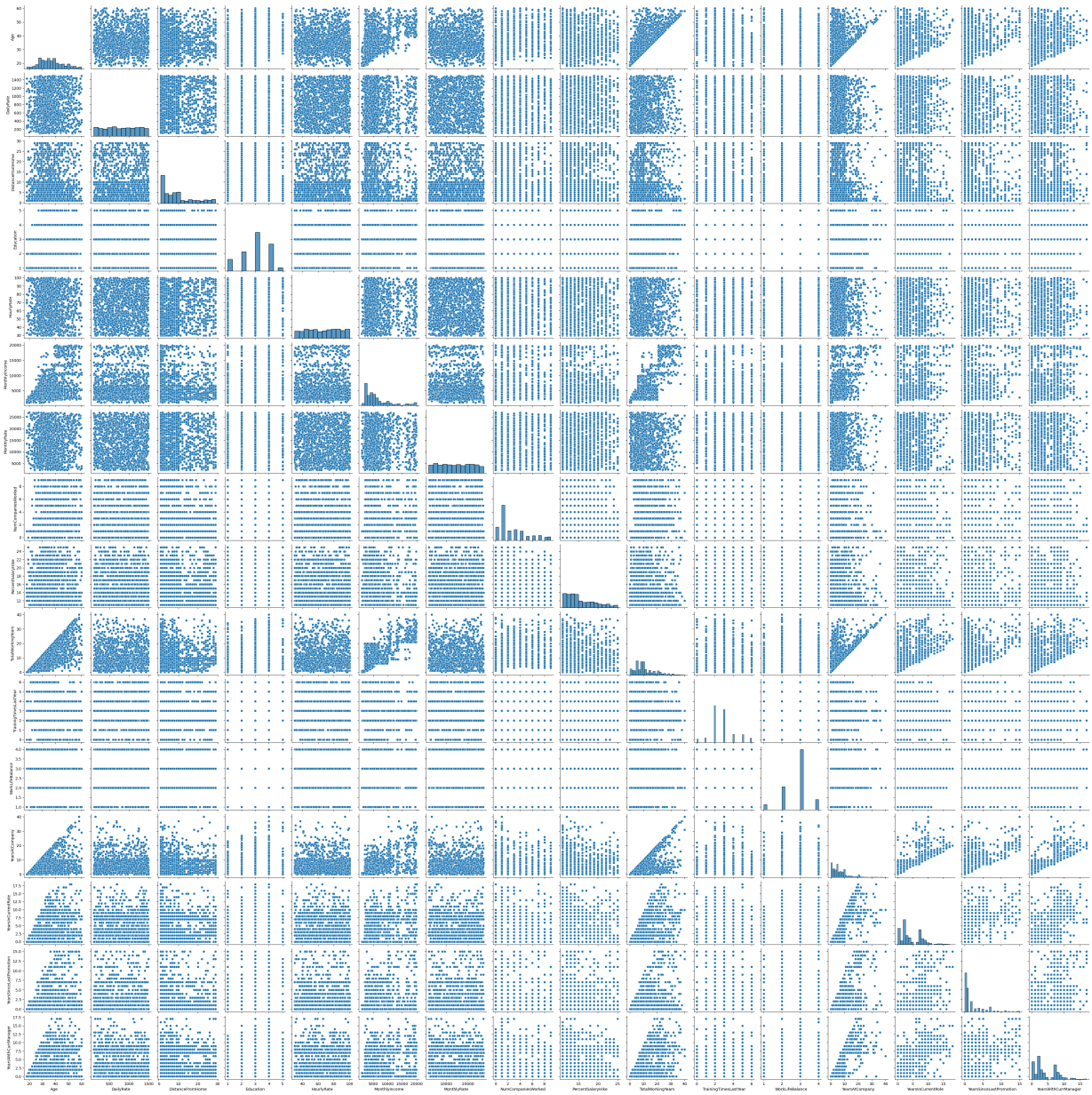
Out[18]: <Axes: ylabel='OverTime'>



Bivariate Analysis

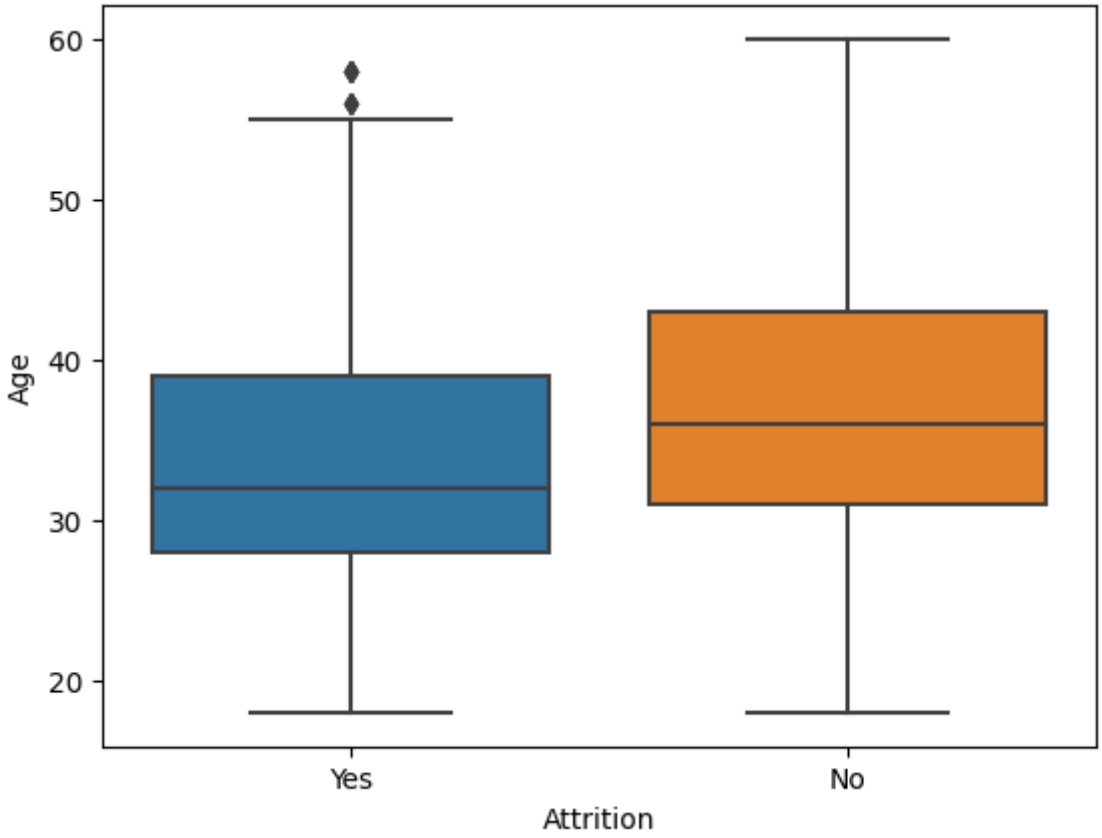
In [19]:
```
sns.pairplot(data = df, vars = ['Age', 'DailyRate', 'DistanceFromHome', 'Education',
```

Out[19]: <seaborn.axisgrid.PairGrid at 0x7b4251af4ee0>



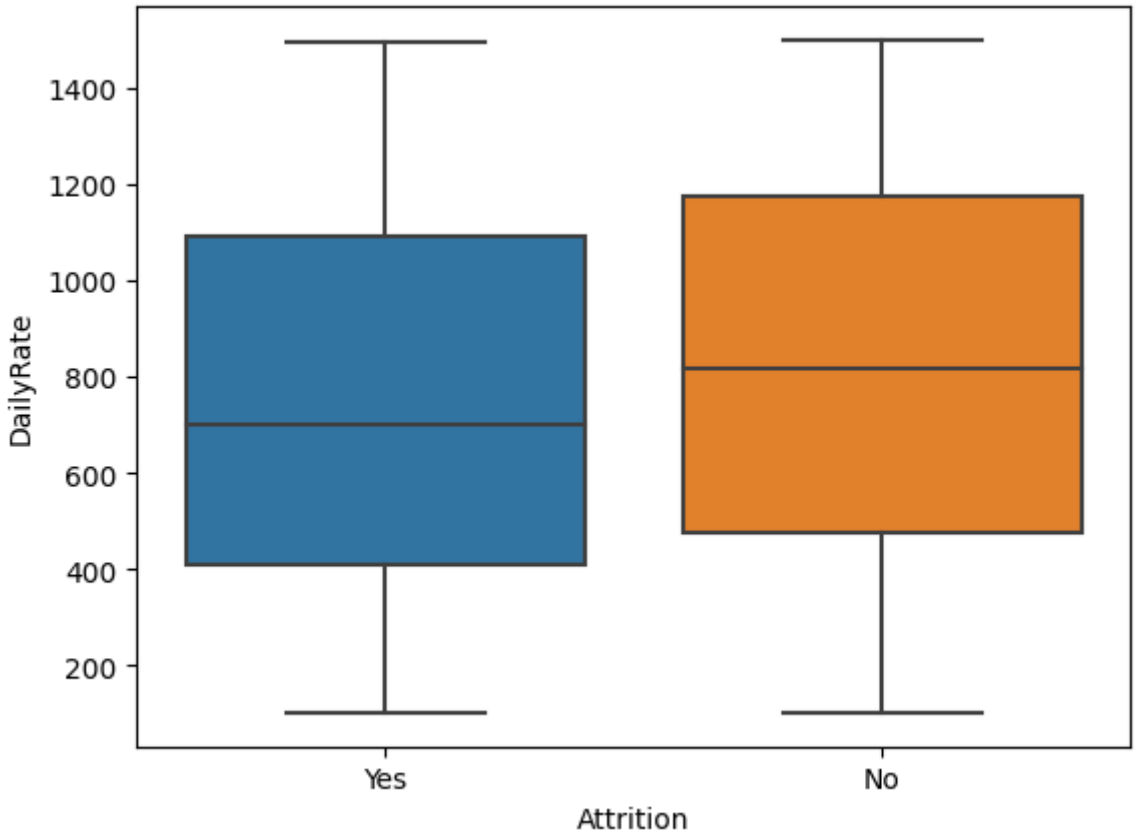
In [20]:

```
sns.boxplot(data = df, x = "Attrition", y = "Age")
plt.show()
```



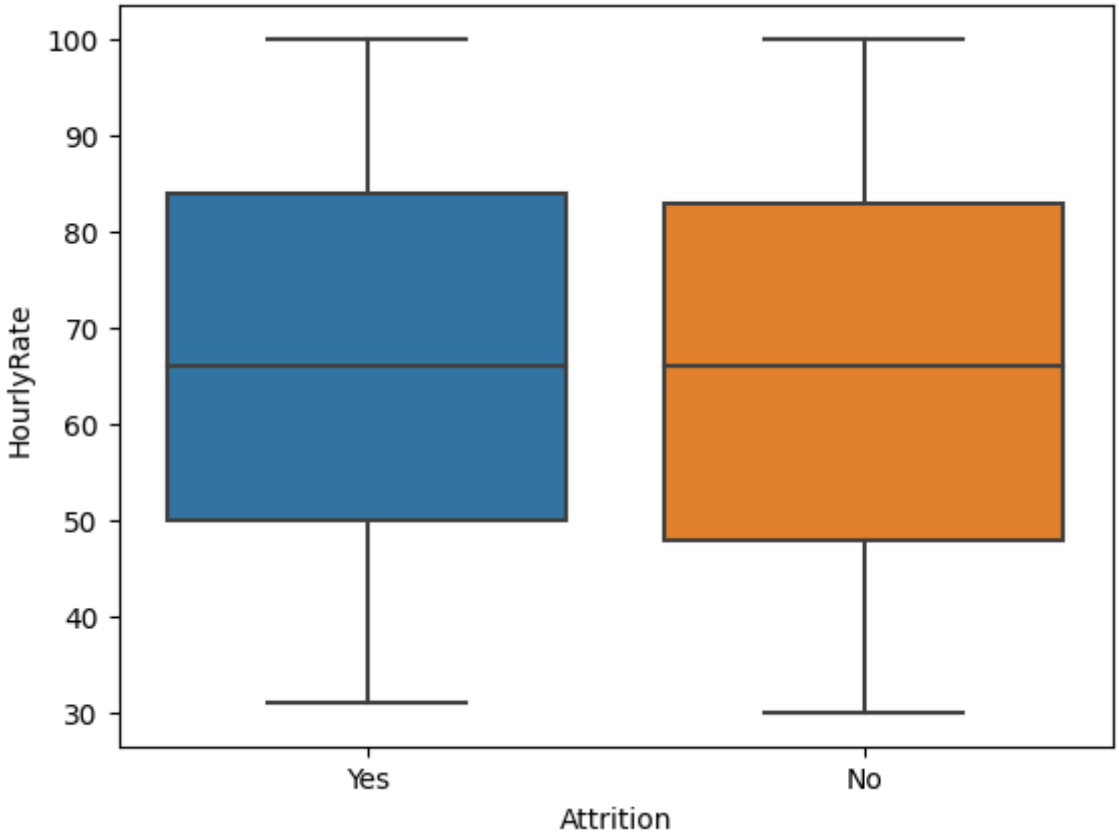
In [21]:

```
sns.boxplot(data = df, x = "Attrition", y = "DailyRate")
plt.show()
```



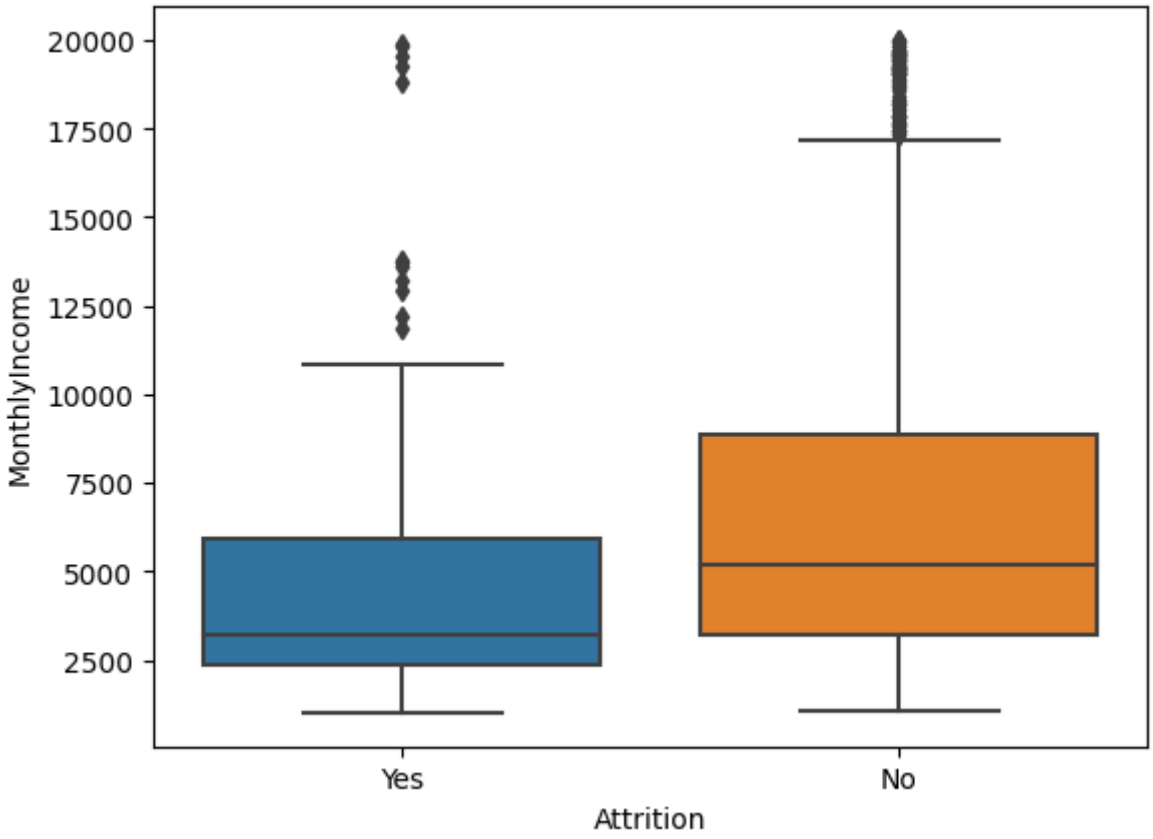
In [22]:

```
sns.boxplot(data = df, x = "Attrition", y = "HourlyRate")
plt.show()
```



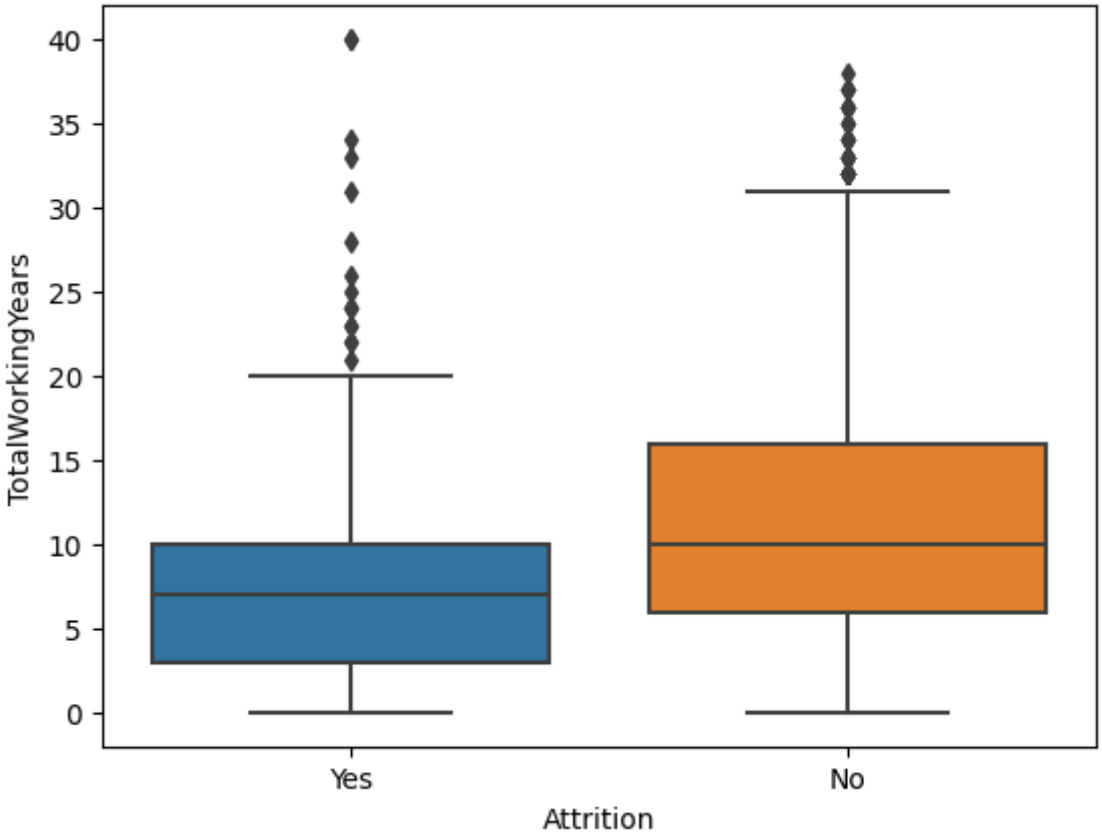
In [23]:

```
sns.boxplot(data = df, x = "Attrition", y = "MonthlyIncome")
plt.show()
```



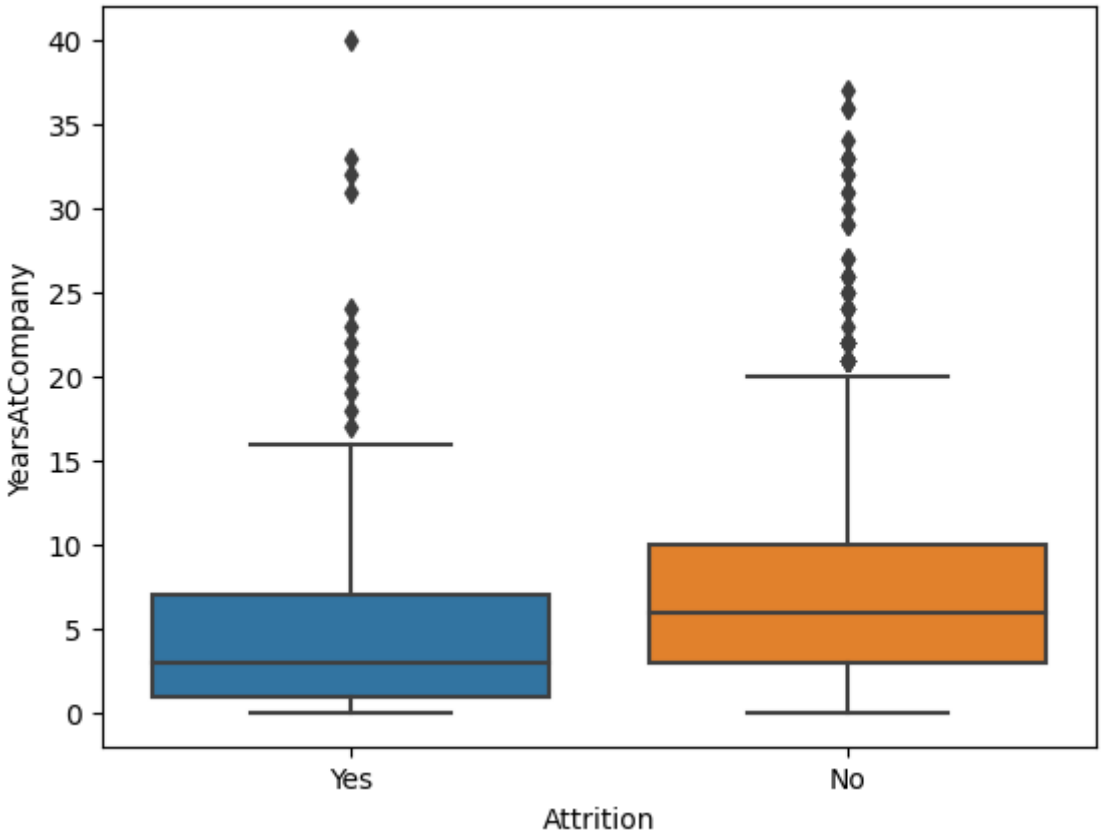
In [24]:

```
sns.boxplot(data = df, x = "Attrition", y = "TotalWorkingYears")
plt.show()
```



In [25]:

```
sns.boxplot(data = df, x = "Attrition", y = "YearsAtCompany")
plt.show()
```



In [26]:

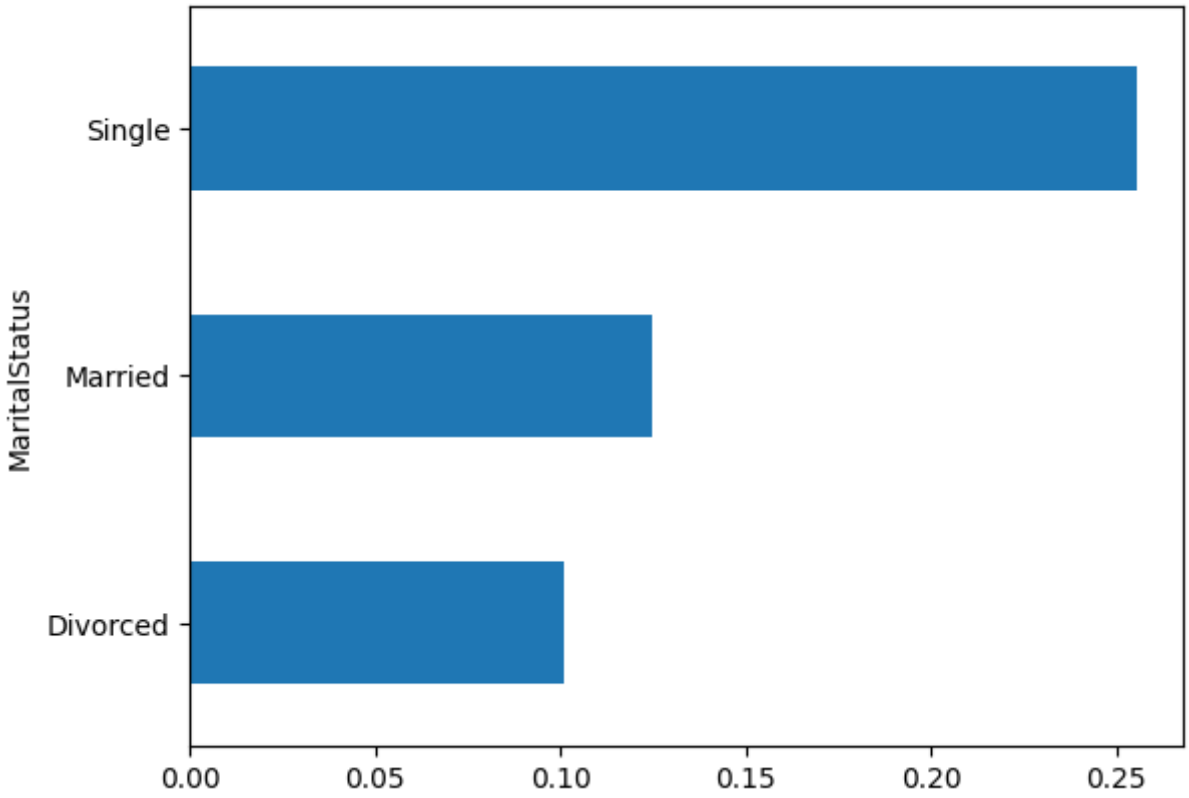
```
copy = df.copy()
```

In [27]:

```
copy['attrition_flag'] = np.where(copy.Attrition == "Yes", 1, 0)
```

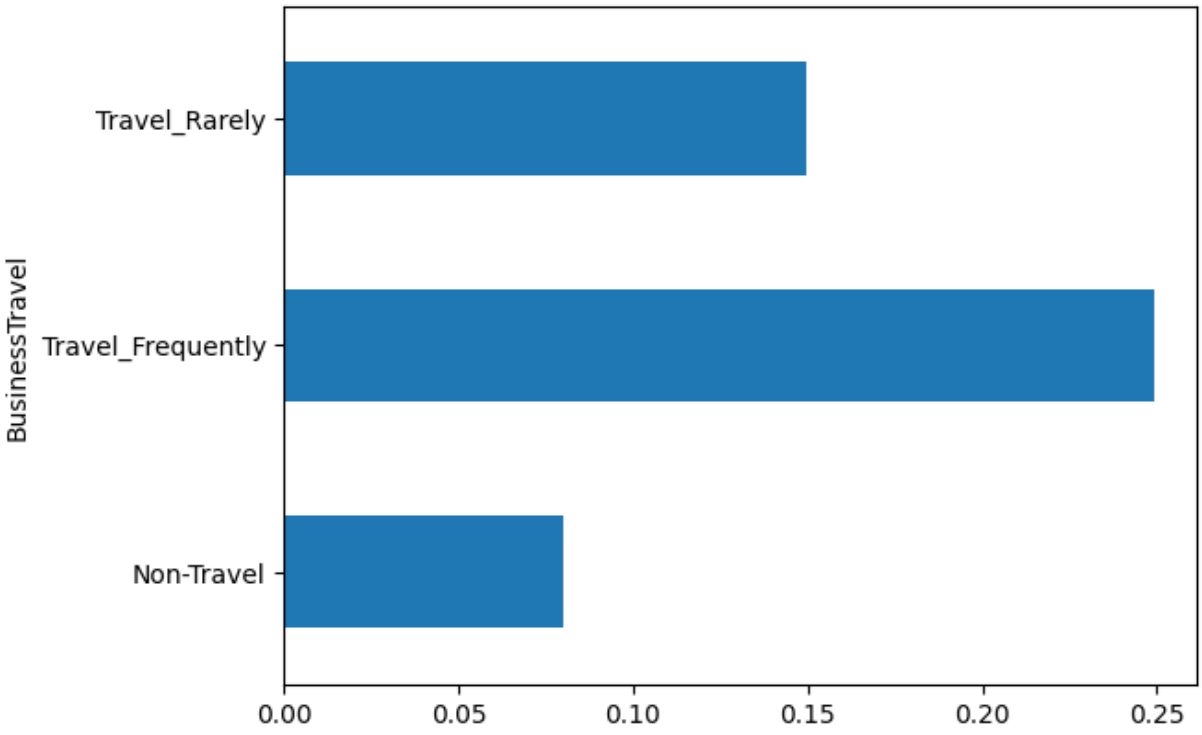
In [28]:
```
copy.groupby(['MaritalStatus'])['attrition_flag'].mean().plot.barh()
```

Out[28]: <Axes: ylabel='MaritalStatus'>



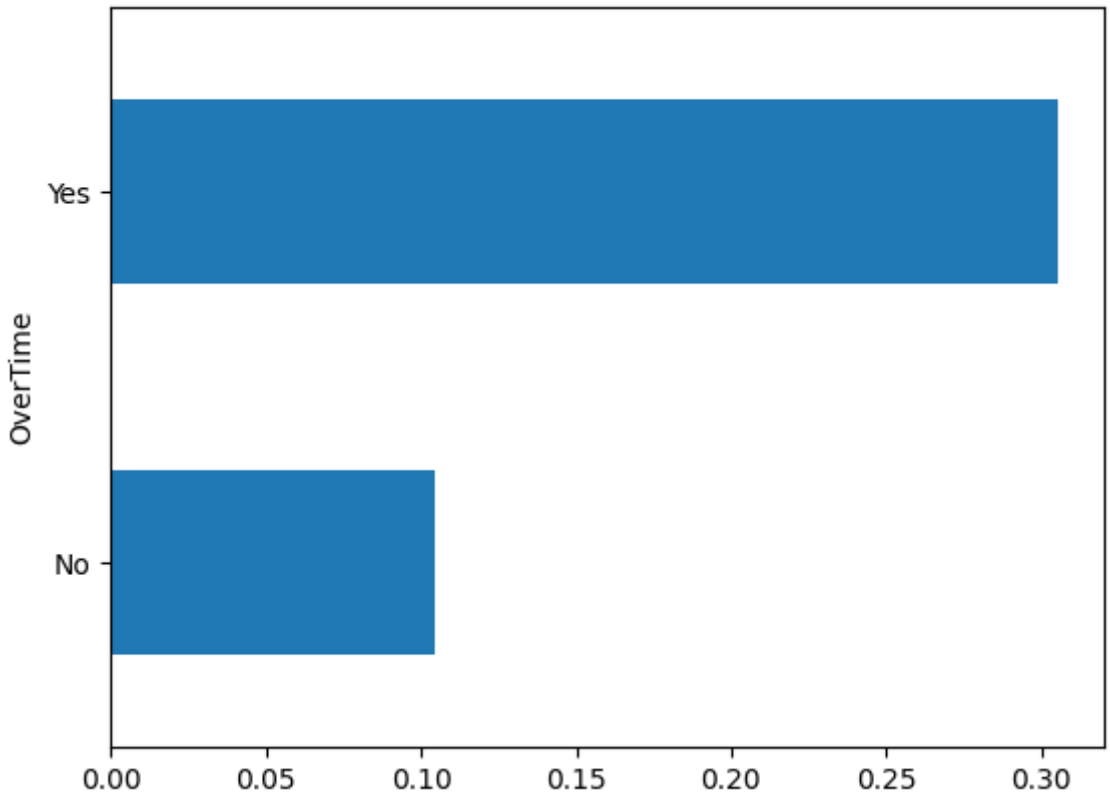
In [29]:
```
copy.groupby(['BusinessTravel'])['attrition_flag'].mean().plot.barh()
```

Out[29]: <Axes: ylabel='BusinessTravel'>



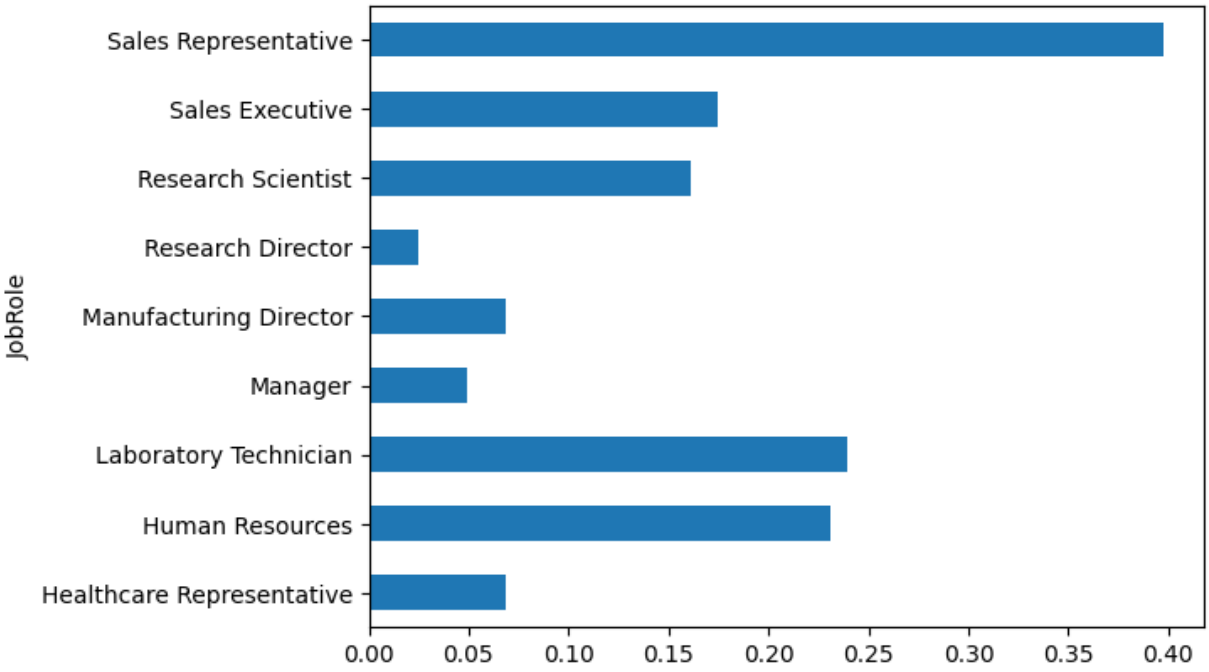
In [30]:
```
copy.groupby(['OverTime'])['attrition_flag'].mean().plot.barh()
```

Out[30]: <Axes: ylabel='OverTime'>



In [31]:
```
copy.groupby(['JobRole'])['attrition_flag'].mean().plot.barh()
```

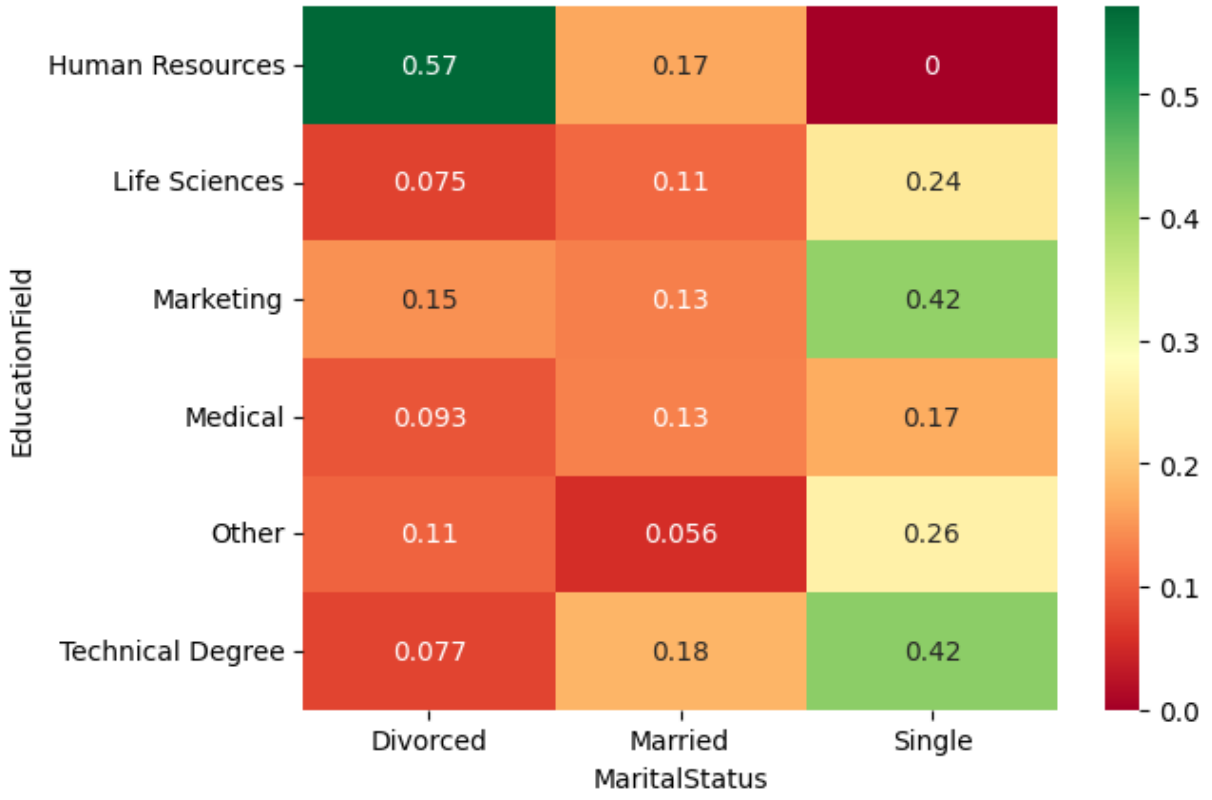
Out[31]: <Axes: ylabel='JobRole'>



Multivariate Analysis

In [32]:
```
p1 = pd.pivot_table(data = copy, index = 'EducationField', columns = 'MaritalStatus'
sns.heatmap(p1, annot = True, cmap = 'RdYlGn')
```

Out[32]: <Axes: xlabel='MaritalStatus', ylabel='EducationField'>

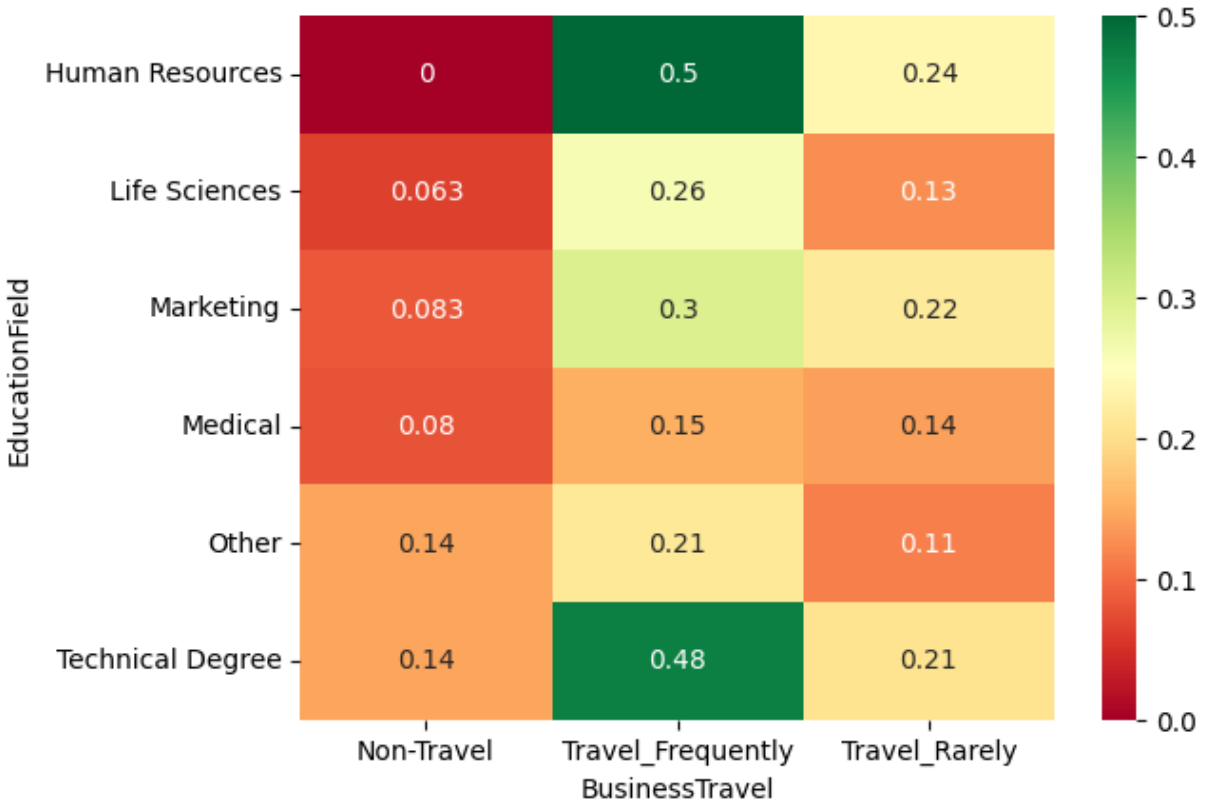


In [33]:
```
p2 = pd.pivot_table(data = copy, index = 'JobRole', columns = 'MaritalStatus', value
sns.heatmap(p2, annot = True, cmap = 'RdYlGn')
```

Out[33]: <Axes: xlabel='MaritalStatus', ylabel='JobRole'>

In [34]:
```
p3 = pd.pivot_table(data = copy, index = 'EducationField', columns = 'BusinessTravel
sns.heatmap(p3, annot = True, cmap = 'RdYlGn')
```

Out[34]: <Axes: xlabel='BusinessTravel', ylabel='EducationField'>



Binary categorical variables like "Yes" or "No" and "Male" or "Female" ought to be converted to 0, 1.

In [35]:
```
copy.drop(['attrition_flag'], axis = 1, inplace = True)
```

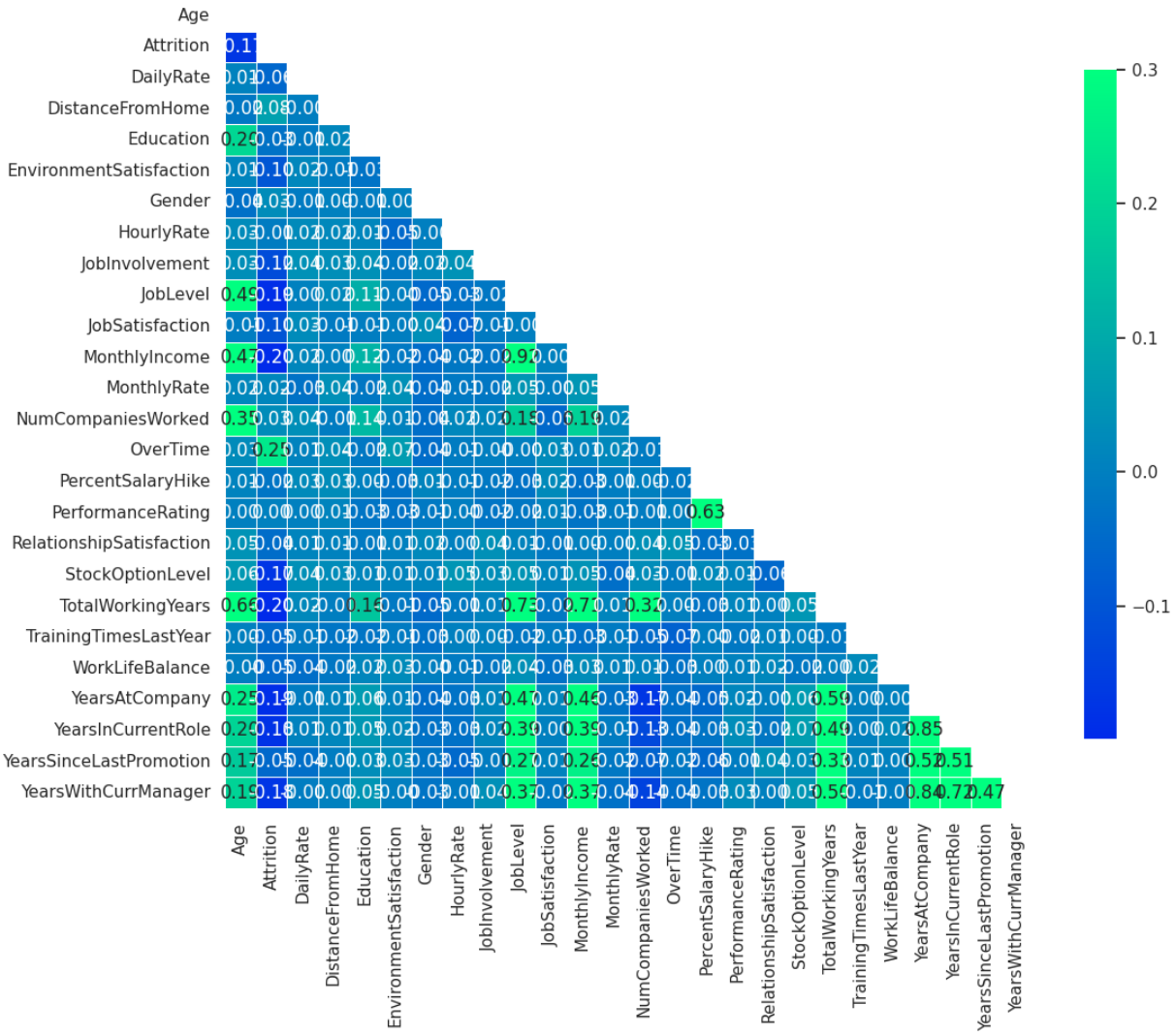
Encoding unordered categorical variables with more than 2 distinct values.

In [36]:
```
df['Attrition'] = df['Attrition'].map({'Yes': 1, 'No': 0})
df['OverTime'] = df['OverTime'].map({'Yes': 1, 'No': 0})
df['Gender'] = df['Gender'].map({'Male': 1, 'Female': 0})
```

In [37]:
```
copy = pd.get_dummies(copy, columns = ['BusinessTravel', 'Department', 'EducationFie
```

In [38]:
```
dfi = df.select_dtypes('int64')
```

In [39]:
```
corr = dfi.corr(method = "spearman")
sns.set(style = "white")
mask = np.triu(np.ones_like(corr, dtype = bool))
plt.figure(figsize=(12, 10), dpi = 100)
sns.heatmap(corr, mask = mask, cmap = "winter", vmax = .3, center = 0, square = True
plt.show()
```



Encoding categorical variabes with multiple values.

In [40]:

```
DF = df.copy()

DF['Attrition'] = DF['Attrition'].replace('Yes',2)
DF['Attrition'] = DF['Attrition'].replace('No',3)

DF['BusinessTravel'] = DF['BusinessTravel'].replace('Travel_Rarely',2)
DF['BusinessTravel'] = DF['BusinessTravel'].replace('Travel_Frequently',3)
DF['BusinessTravel'] = DF['BusinessTravel'].replace('Non-Travel',4)

DF['Department'] = DF['Department'].replace('Sales',2)
DF['Department'] = DF['Department'].replace('Human Resources',3)
DF['Department'] = DF['Department'].replace('Research & Development',4)

DF['EducationField'] = DF['EducationField'].replace('Life Sciences',2)
DF['EducationField'] = DF['EducationField'].replace('Medical',3)
DF['EducationField'] = DF['EducationField'].replace('Marketing',4)
DF['EducationField'] = DF['EducationField'].replace('Technical Degree',2)
DF['EducationField'] = DF['EducationField'].replace('Human Resources',3)
DF['EducationField'] = DF['EducationField'].replace('Other',4)

DF['Gender'] = DF['Gender'].replace('Male',2)
DF['Gender'] = DF['Gender'].replace('Female',3)

DF['JobRole'] = DF['JobRole'].replace('Sales Executive',2)
DF['JobRole'] = DF['JobRole'].replace('Manufacturing Director',3)
DF['JobRole'] = DF['JobRole'].replace('Healthcare Representative',4)
DF['JobRole'] = DF['JobRole'].replace('Manager',2)
```

```
DF['JobRole'] = DF['JobRole'].replace('Research Director',3)
DF['JobRole'] = DF['JobRole'].replace('Laboratory Technician',4)
DF['JobRole'] = DF['JobRole'].replace('Sales Representative',2)
DF['JobRole'] = DF['JobRole'].replace('Research Scientist',3)
DF['JobRole'] = DF['JobRole'].replace('Human Resources',4)

DF['MaritalStatus'] = DF['MaritalStatus'].replace('Single', 2)
DF['MaritalStatus'] = DF['MaritalStatus'].replace('Married', 3)
DF['MaritalStatus'] = DF['MaritalStatus'].replace('Divorced', 4)

DF['OverTime'] = DF['OverTime'].replace('Yes',2)
DF['OverTime'] = DF['OverTime'].replace('No',3)
```

From cell 40, clearly, some combinations of variables have high correlation and therefore I will remove them.

In [41]:
```
DF.drop(['MonthlyIncome' ,'YearsInCurrentRole' , 'YearsAtCompany', 'YearsWithCurrMan
```

Data Preprocessing

In [42]:
```
scale = MinMaxScaler(feature_range = (0, 1))
DFx = DF.drop(columns = ['Attrition'])
norm = scale.fit_transform(DF)
N = pd.DataFrame(norm, columns = DF.columns)
```

In [43]:
```
X = pd.DataFrame(N.drop(columns = 'Attrition'))
y = pd.DataFrame(N.Attrition).values.reshape(-1, 1)
```

In [44]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_st
```

As observed in cell 11 of this notebook, this dataset has a data imabalance problem. Therefore, I will be importing and using the SMOTE module. SMOTE is specifically designed to tackle imbalanced datasets by generating synthetic samples for the minority class.

In [45]:
```
from imblearn.over_sampling import SMOTE
oversampler = SMOTE(random_state = 0)
smote_train, smote_target = oversampler.fit_resample(X_train, y_train)
```

In [46]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_st
```

Random Forest Classifier

In [47]:
```
rfc = RandomForestClassifier()
rfc = rfc.fit(smote_train , smote_target)


y_pred = rfc.predict(X_test)

print ('accuracy', accuracy_score(y_test, y_pred))
```

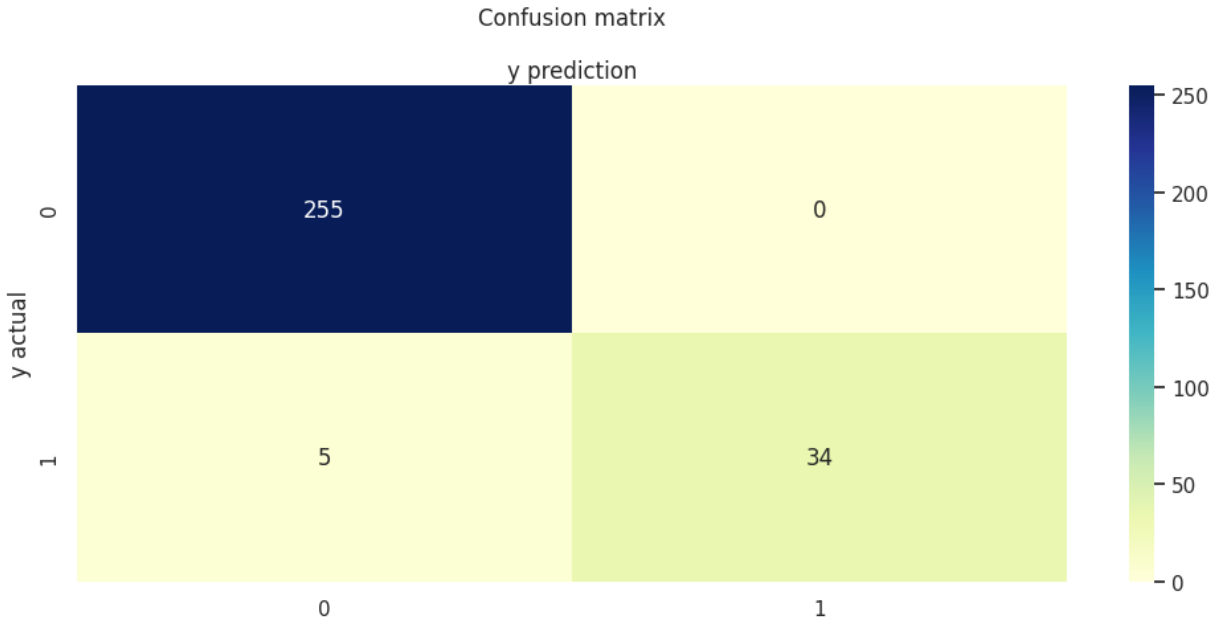
```
accuracy 0.9829931972789115
```

Model Evaluation

In [48]:

```
fig, ax = plt.subplots(figsize = (10,5))
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(confusion_matrix(y_test, y_pred), annot = True, cmap = "YlGnBu" ,fmt = '
plt.title('Confusion matrix', y = 1.1)
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.xlabel('y prediction')
plt.ylabel('y actual')
plt.show()

print(classification_report(y_test, y_pred))
```



	precision	recall	f1-score	support
0.0	0.98	1.00	0.99	255
1.0	1.00	0.87	0.93	39
accuracy			0.98	294
macro avg	0.99	0.94	0.96	294
weighted avg	0.98	0.98	0.98	294