HR EMPLOYEE ATTRITION

Dissertation submitted in fulfilment of the requirements for the Degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

MANISH

KUMAR

DAS

1201595

Supervisor

AJAY SHARMA

School of Computer Science and Engineering

Lovely Professional

University Phagwara,

Punjab(India), April 2024

ACKNOWLEDGEMENT

Primarily I would like to thank God for being able to learn a new technology. Then I would like to express my special thanks of gratitude to the teacher and instructor of the course Exploratory Data Analysis Project who provided me with the golden opportunity to learn a new technology.

I would like to also thank my own college, Lovely Professional University, for offering sucha course which not only improve my programming skill but also taught me other new technology.

Then I would like to thank my parents and friends who have helped me with their valuable suggestions and guidance in choosing this course.

Finally, I would like to thank everyone who has helped me a lot.

Dated: 30TH June 2024

TABLE OF CONTENTS

INTRODUCTION

In the dynamic landscape of modern business, where the war for talent rages on, organizations face a formidable challenge: retaining skilled employees. Human Resource (HR) departments grapple with the complexities of understanding why employees leave, often referred to as employee attrition, and how to mitigate this costly phenomenon. In this pursuit, Exploratory Data Analysis (EDA) emerges as a powerful tool, offering a lens through which to decipher the intricate patterns and factors contributing to attrition within an organization.

This report delves into the realm of HR employee attrition through the prism of exploratory data analysis. It aims to illuminate the intricate interplay of variables, shedding light on the underlying causes and potential solutions for mitigating attrition. By leveraging data-driven insights, organizations can proactively address attrition, fostering a work environment conducive to employee satisfaction, engagement, and ultimately, retention.

DOMAIN KNOWLEDGE

Employee attrition represents more than just a numerical metric; it embodies a multifaceted phenomenon influenced by a myriad of internal and external variables. From organizational culture and leadership effectiveness to job satisfaction and career growth opportunities, the drivers of attrition are as diverse as they are complex. Through EDA, we embark on a journey to dissect these factors, discerning the pivotal drivers that precipitate attrition and identifying opportunities for intervention. In an era characterized by unprecedented mobility and competition for skilled labor, the stakes of employee retention have never been higher. Beyond the tangible costs associated with recruitment and onboarding, attrition inflicts intangible wounds upon organizational morale and productivity. Consequently, organizations must embrace a proactive stance towards retention, leveraging EDA as a compass to navigate the tumultuous waters of talent management.

As we embark on this exploratory journey into HR employee attrition, armed with the tools of data analysis and a commitment to organizational excellence, we are poised to uncover the insights that will shape the future of talent retention. Through meticulous examination of data, thoughtful interpretation of findings, and strategic implementation of solutions, we endeavor to transform attrition from a looming threat into an opportunity for organizational growth and resilience.

REASON OF CHOOSING THIS DATASET

In this era of heightened competition for talent, effective decision-making regarding HR employee attrition is paramount. Attrition not only incurs significant costs in recruitment and training but also disrupts organizational stability and productivity. Moreover, in a digital age where employee experiences are shared instantaneously, high attrition rates can tarnish employer branding, deterring top talent from joining the organization. Proactive measures to understand and mitigate attrition through data-driven insights not only enhance employee retention but also foster a positive workplace culture, positioning the organization as an employer of choice in an increasingly competitive talent landscape.

LIBRARY USED AND APPROACHES

I have used pandas, numpy, matplotlib.pyplot, seaborn for this project. To start, pandas is used to load and manipulate the data, allowing for easy cleaning and preprocessing. NumPy is then employed for numerical operations and calculations, providing a foundation for statistical analysis. Seaborn and matplotlib.pyplot are utilized for data visualization, enabling the creation of informative plots such as histograms, scatter plots, and box plots. These visualizations help in understanding the distribution of data, identifying outliers, and exploring relationships between variables. Through this iterative process of data manipulation and visualization, EDA allows for the identification of patterns, trends, and anomalies within the dataset, ultimately informing decision-making and hypothesis generation for further analysis.

DATA DESCRIPTION

Columns: 35

Rows :1470

Employee no: Unique id of the employee

Age: age of the workers

Years in company

Years in current role: not yet promoted workers

Years in same post

Standard hours: time each employee works

Total working hours: monthly or yearly working hours of the employee.

EXPLORATORY DATA ANALYSIS

IMPORTING REQUIRED LIBRARIES:

```
#import packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.stats as stats
%matplotlib inline
```

First, we need to import the pandas and NumPy as pd and np respectively is used to perform a wide variety of mathematical operations on arrays.

Matplotlib.pyplot and seaborn are used to display different types of charts which will help in further decision making.

And Pandas is used because it is an open-source library. It can be used to perform data manipulation and analysis.

KNOWING AND CHECKING FIRST 5 ROWS OF THE DATASET:

head() is used to find the first elements of first 5 rows.

```
hr_data = pd.read_csv("/content/HR-Employee-Attrition.csv")  #importing the dataset
hr_data.head()  #to print first 5 rows
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	...
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	...
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	...
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	...
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	...
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	...

5 rows × 35 columns

describe() is used to view some basic statistical details like percentile, mean, std etc. of a data frame.

```
hr_data.describe()
```

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	JobInvolvement	JobLevel	...
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	...
mean	36.923810	802.485714	9.192517	2.912925	1.0	1024.865306	2.721769	65.891156	2.729932	2.063946	...
std	9.135373	403.509100	8.106864	1.024165	0.0	602.024335	1.093082	20.329428	0.711561	1.106940	...
min	18.000000	102.000000	1.000000	1.000000	1.0	1.000000	1.000000	30.000000	1.000000	1.000000	...
25%	30.000000	465.000000	2.000000	2.000000	1.0	491.250000	2.000000	48.000000	2.000000	1.000000	...
50%	36.000000	802.000000	7.000000	3.000000	1.0	1020.500000	3.000000	66.000000	3.000000	2.000000	...
75%	43.000000	1157.000000	14.000000	4.000000	1.0	1555.750000	4.000000	83.750000	3.000000	3.000000	...
max	60.000000	1499.000000	29.000000	5.000000	1.0	2068.000000	4.000000	100.000000	4.000000	5.000000	...

8 rows × 26 columns

FINDING THE DATATYPES USING info() METHOD:

This step helps us to find the different types of Datatypes available in the given data set.

```
hr_data.info()  #this will print information about the dataframe
```

```
<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
```

CHECKING THE NULL VALUES:

BY determining the source of missingness in our datasets, we can more accurately decide how to handle missing values to improve model quality.

```
#Missing values check
hr_data.isnull().sum()
```

```
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
```

TO FIND DATATYPES USING dtypes() METHOD:

```
hr_data.dtypes
#Check the structure of dataset
#data type object (dtype) informs us about the layout of the array. This means it gives us information about:
#Type of the data (integer, float, Python object, etc.) Size of the data(number of bytes)
```

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

CHECKING THE COLUMNS USING .columns:

```
hr_data.columns
```

```
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'],
      dtype='object')
```

KNOWING THE DATATYPES OF THE COLUMN:

```
#Dimension of dataset
print("Dataset is of ", hr_data.ndim, " dimension.")
#Rows and column of dataset
print("Dataset has ", hr_data.shape[0], " rows.", "\nDataset has ", hr_data.shape[1])
#Knowing the data Types of the columns
print("Data Types:")
print(hr_data.dtypes)
```

```
Dataset is of  2  dimension.
Dataset has  1470  rows.
Dataset has  35
Data Types:
Age                         int64
Attrition                  object
BusinessTravel             object
DailyRate                   int64
Department                 object
DistanceFromHome            int64
Education                   int64
EducationField             object
EmployeeCount               int64
EmployeeNumber              int64
EnvironmentSatisfaction     int64
Gender                     object
HourlyRate                  int64
JobInvolvement              int64
JobLevel                    int64
JobRole                    object
JobSatisfaction             int64
MaritalStatus              object
MonthlyIncome               int64
MonthlyRate                 int64
NumCompaniesWorked          int64
Over18                     object
OverTime                   object
PercentSalaryHike           int64
PerformanceRating           int64
RelationshipSatisfaction    int64
StandardHours               int64
StockOptionLevel            int64
TotalWorkingYears           int64
TrainingTimesLastYear       int64
WorkLifeBalance             int64
YearsAtCompany              int64
YearsInCurrentRole          int64
```

CHECKING THE DUPLICATE ROWS AND COLUMNS:

```
hr_data.duplicated()
```

```
0       False
1       False
2       False
3       False
4       False
        ...
1465    False
1466    False
1467    False
1468    False
1469    False
Length: 1470, dtype: bool
```

TO KNOW TOTAL NUMBER OF DUPLICATES:

```
hr_data.duplicated().sum()
```

```
0
```

REMOVING THE DUPLICATE ROWS AND COLUMNS:

```
hr_data.drop_duplicates()
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	...
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	...
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	...
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	...
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	...
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	...
...	...	...	...	...	...	...	...	...	...	...	...
1465	36	No	Travel_Frequently	884	Research & Development	23	2	Medical	1	2061	...
1466	39	No	Travel_Rarely	613	Research & Development	6	1	Medical	1	2062	...
1467	27	No	Travel_Rarely	155	Research & Development	4	3	Life Sciences	1	2064	...
1468	49	No	Travel_Frequently	1023	Sales	2	3	Medical	1	2065	...
1469	34	No	Travel_Rarely	628	Research & Development	8	3	Medical	1	2068	...

1470 rows × 35 columns

Removing duplicate values plays an important role in the cleansing process. Duplicate data takes up unnecessary storage space and slows down calculations at a minimum.

UNIVARIATE DATA ANALYSIS

CALCULATE MEAN OF THE POINTS:

```
#Measures the center and spread of values.
#1
#calculate mean of "points"
hr_data['DailyRate'].mean()
```

```
802.4857142857143
```

CALCULATE MODE OF THE POINTS:

```
#calculate mode of "points"
hr_data['DailyRate'].mode()
```

```
0    691
Name: DailyRate, dtype: int64
```

CALCULATE STANDARD DEVIATION:

A standard deviation is a measure of how dispersed the data is in relation to the mean.

```
hr_data['DailyRate'].std()
```

```
403.50909994352816
```

CREATING FREQUENCY TABLE OF THE POINTS:

```
hr_data['DailyRate'].value_counts()
```

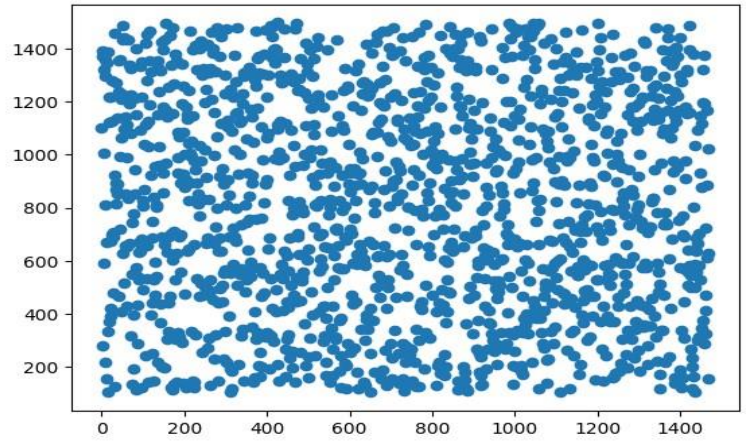
```
691     6
408     5
530     5
1329    5
1082    5
       ..
650     1
279     1
316     1
314     1
628     1
Name: DailyRate, Length: 886, dtype: int64
```

PLOTTING A SCATTER PLOT

A scatter plot is used to represent the values for two variables in a two-dimensional data-set.

The below graph of scatterplot shows the daily rate of the employ.

```
plt.scatter(hr_data.index, hr_data['DailyRate'])
plt.show()
```

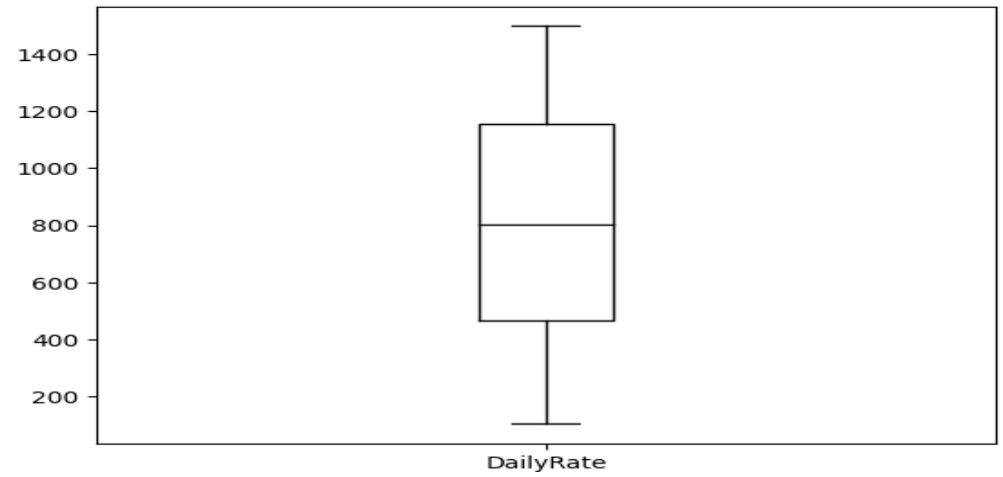


PLOTTING A BOXPLOT:

Box plots are used to show distributions of numeric data values, especially when you want to compare them between multiple groups.

```
hr_data.boxplot(column=['DailyRate'], grid=False, color='black')
```

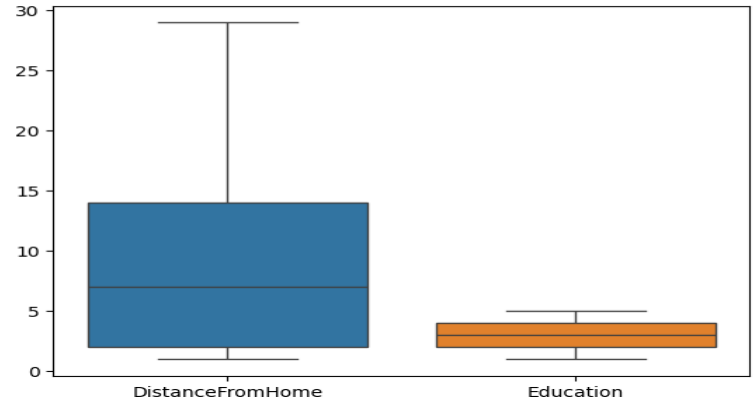
```
<Axes: >
```



PLOTTING BOXPLOT FOR 2 COLUMNS:

```
sns.boxplot(data = hr_data.loc[:, ['DistanceFromHome', 'Education']])
```

```
<Axes: >
```



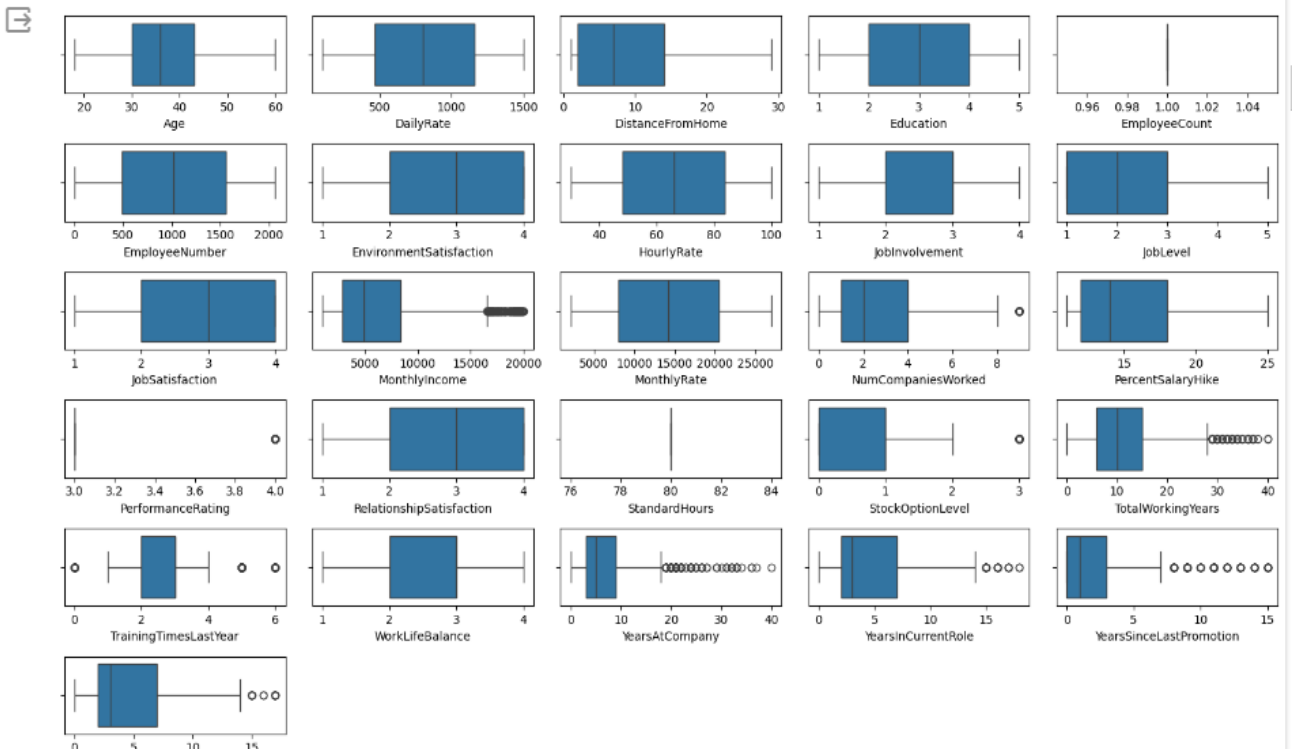
Observation:
From the above graph we can tell that number of employees who travel more distance are more in count and most of them are well educated up to graduation or masters level.

BOXPLOT FOR NUM COLUMNS:

```
cols = 5
rows = 6
num_cols = hr_data.select_dtypes(exclude='object').columns
fig = plt.figure(figsize = (15,9))

for i, col in enumerate(num_cols):
  ax = fig.add_subplot(rows, cols, i+1)
  sns.boxplot(x = hr_data[col], ax = ax)

fig.tight_layout()
plt.show()
#boxplot for num cols
```



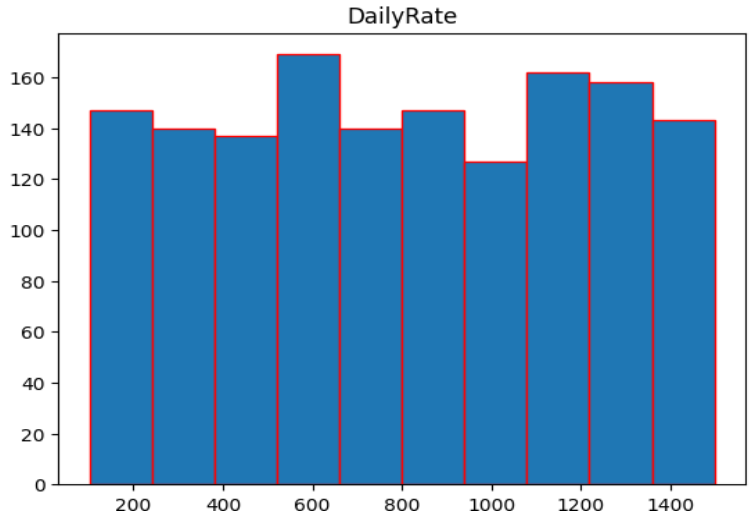
Observation:
1.The age range of the employees in the company is from 30-40 years.
2.Employee rate of the company is from 500-1500.
3.Enviroment satisfaction is not bad for most of the employees.
4.Working hour rate of employees is from 50-85 hours per week.
5.Job involvement of the employees is not much.
6.But when it comes to Job Satisfaction most of the employees are satisfied with it.
7.Monthly income range of the employees from 2.5k to 7.5k.
8.Most of the employees are experienced as they have already worked in to 1-4 company before joining this department
9.Most of the employees work for a period of 4-8 years.
10.The range of years for current job roles of employees and the range for years with current manager is same and it is 2.4 to 5.4 years

PLOTTING HISTPLOT:

Histograms are visual representation of a data set which show how often each value in the data set occurs. The values are grouped in to bins along the x-axis. The height of the bar indicates how many values of the data set fall in to the bin.

```
hr_data.hist(column = 'DailyRate', grid = False, edgecolor = 'red')
```

```
array([[<Axes: title={'center': 'DailyRate'}>]], dtype=object)
```

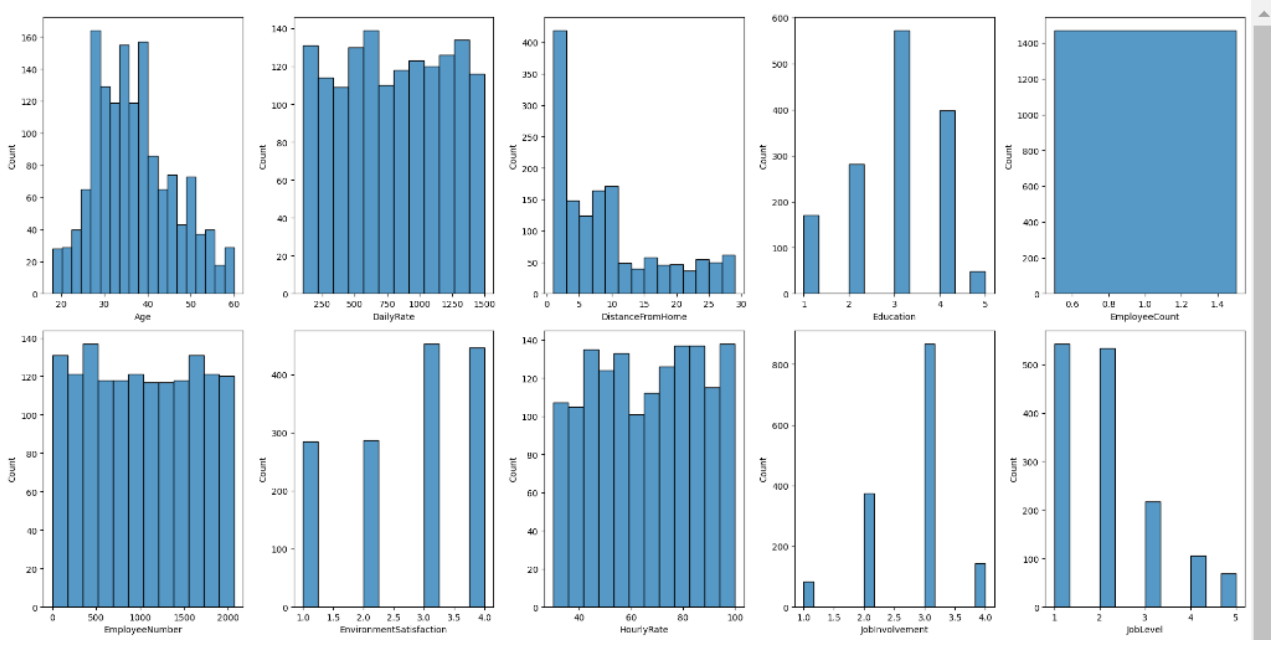


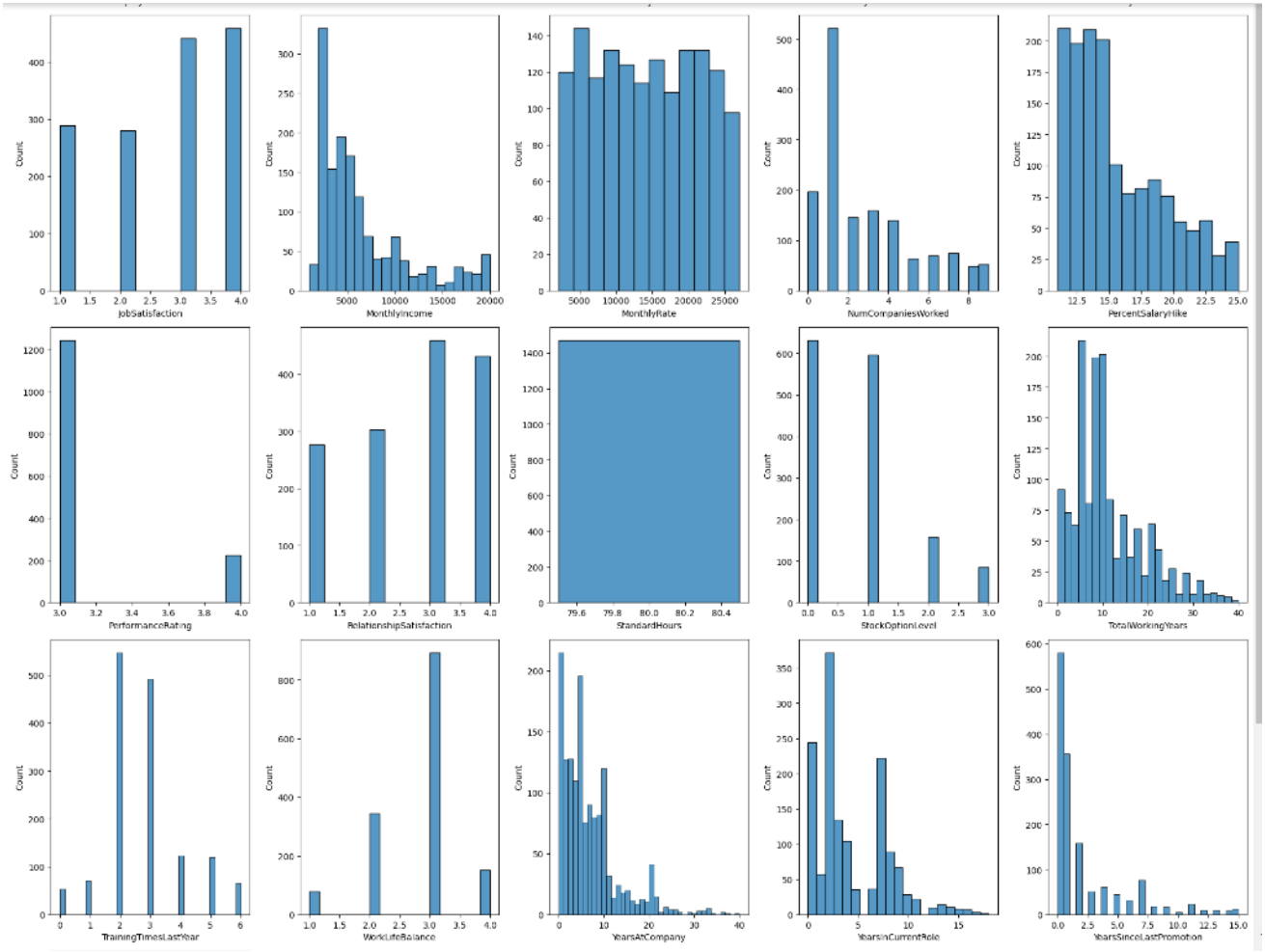
HISTPLOT FOR NUMERIC VALUES:

```
cols = 5
rows = 6
num_cols = hr_data.select_dtypes(exclude = 'object').columns  #excluding dtypes object
fig = plt.figure(figsize = (cols*4, rows*5))

for i, col in enumerate(num_cols):
  ax = fig.add_subplot(rows, cols, i+1)
  sns.histplot(x = hr_data[col], ax = ax)

fig.tight_layout()
plt.show()
#histplot for numeric values
```

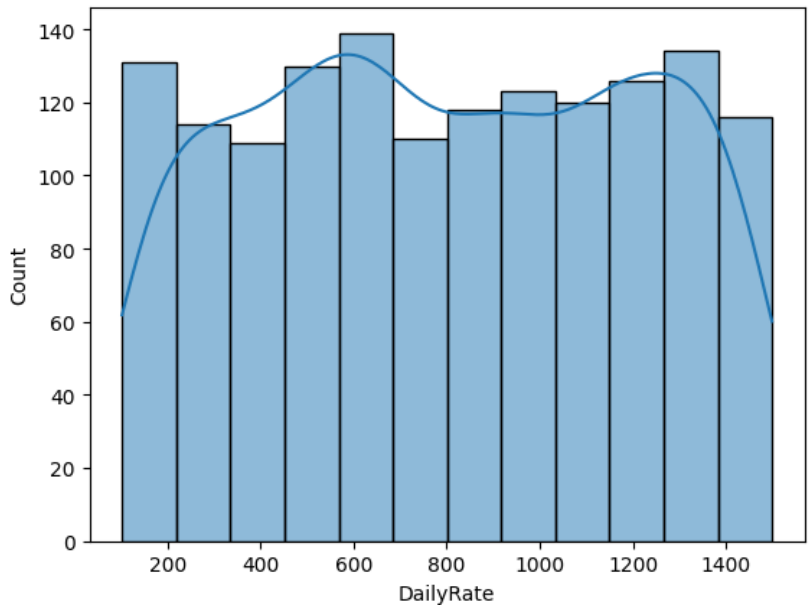




HISTPLOT AND KDEPLOT:

```
sns.histplot(x = 'DailyRate', data = hr_data, kde = True)
```

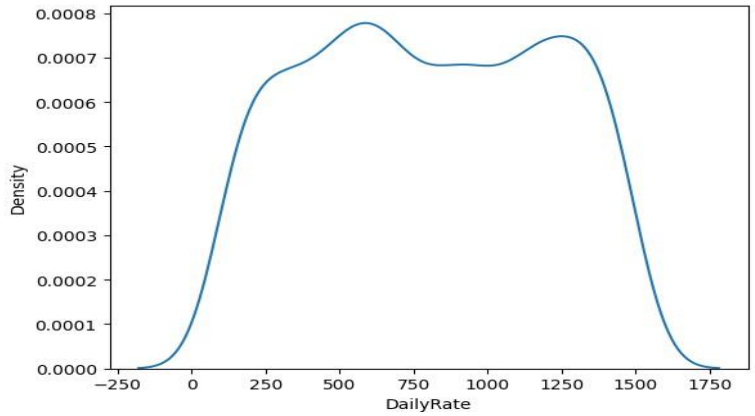
```
<Axes: xlabel='DailyRate', ylabel='Count'>
```



KDE PLOT:

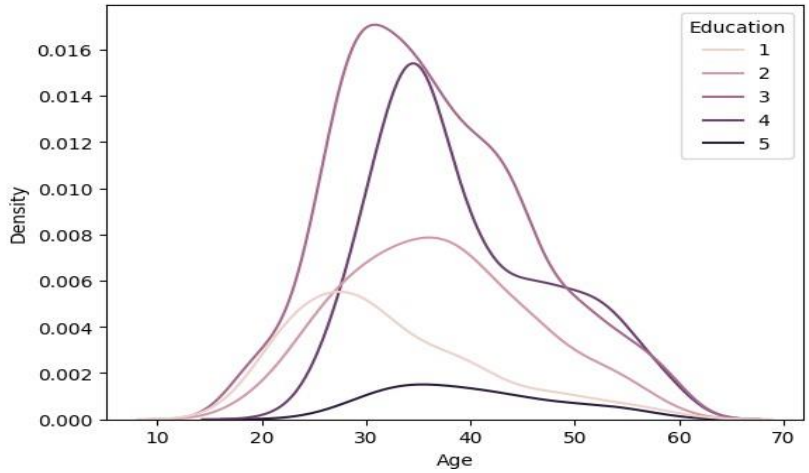
```
sns.kdeplot(hr_data['DailyRate'])
```

```
<Axes: xlabel='DailyRate', ylabel='Density'>
```




```
sns.kdeplot(x = 'Age', data = hr_data, hue = 'Education')
```

```
<Axes: xlabel='Age', ylabel='Density'>
```

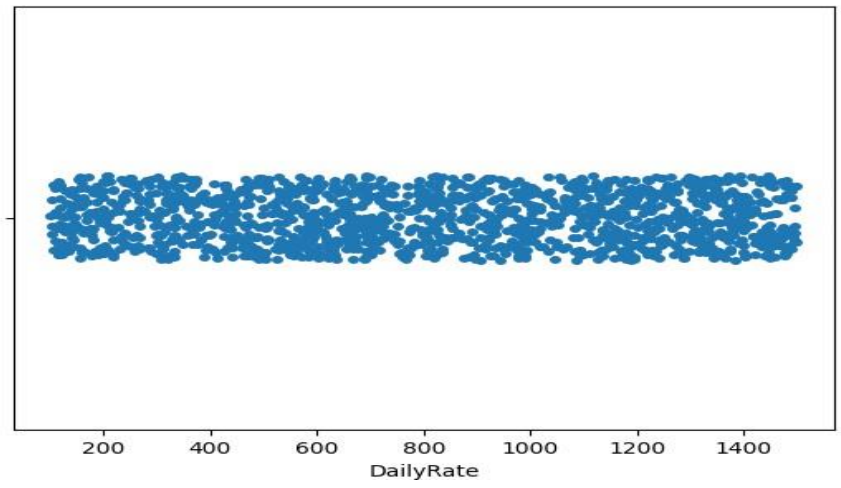


This graph shows us the combination of age and the education level of employees. By this graph we can say that most of the employee are in the age of 30 years and are having 3rd rank of education.

STRIP PLOT:

```
sns.stripplot(x = hr_data['DailyRate'])
```

```
<Axes: xlabel='DailyRate'>
```

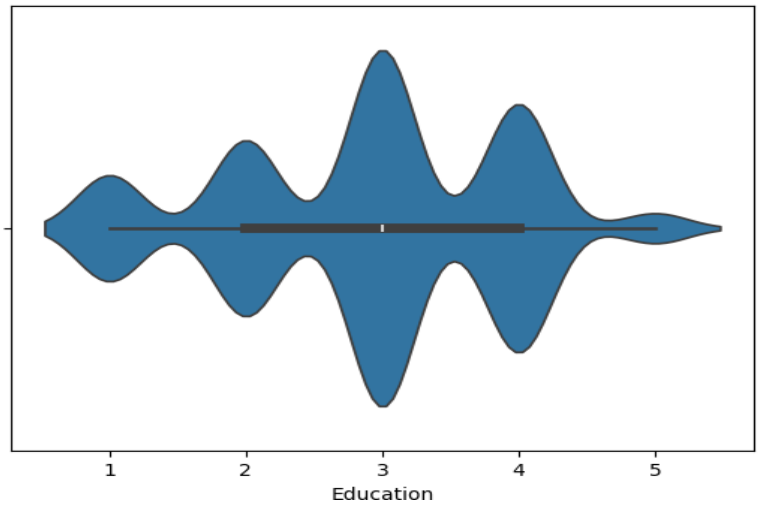


A strip plot is a single-axis scatter plot that is used visualize the distribution of many individual one-dimensional values. The values are plotted as dots along one unique axis.

VIOLON PLOT:

```
sns.violinplot(x = hr_data['Education'])
```

```
<Axes: xlabel='Education'>
```



Violin plots are used when you want to observe the distribution of numeric data, and are especially useful when you want to make a comparison of distributions between multiple groups. The peaks, valleys, and tails of each group's density curves can be compared to see where groups are similar or different.

Observation:

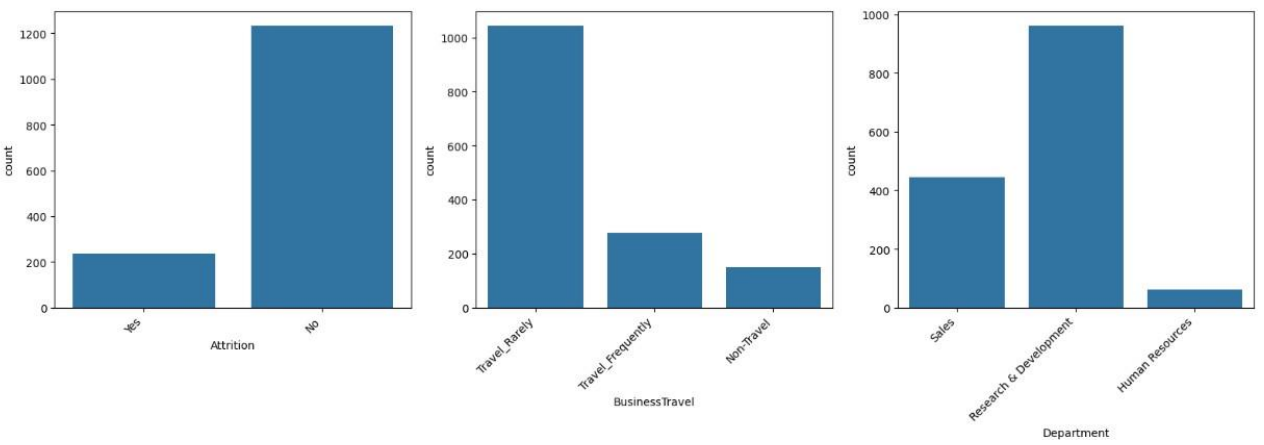
Most of the employees are having 3rd rank education.

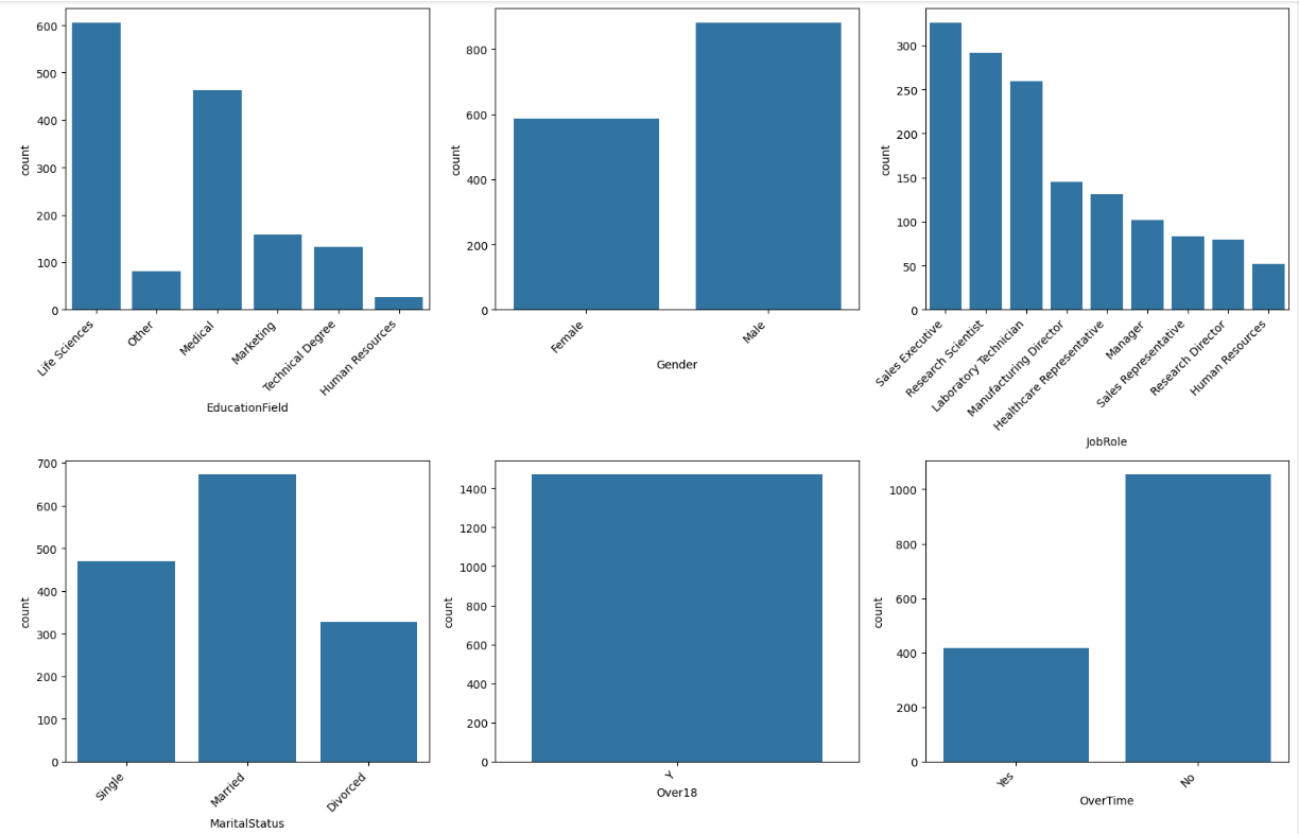
BAR CHART:

```
cols = 3
rows = 3
fig = plt.figure(figsize= (16,16))
all_categs = hr_data.select_dtypes(include = 'object')
cat_cols = all_categs.columns[all_categs.nunique() < 10]

for i, col in enumerate(cat_cols):
  ax = fig.add_subplot(rows, cols, i+1)
  sns.countplot(x = hr_data[col], ax = ax)
  plt.xticks(rotation = 45, ha = 'right')

fig.tight_layout()
plt.show()
#A count plot compares different classes of a categorical feature
#A bar chart with the bar height showing number of times each class occurs in the dataset.
```





Observation:

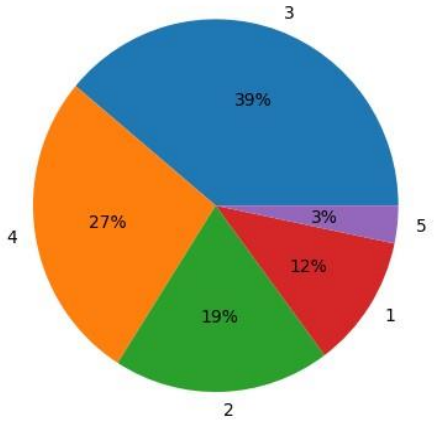
1. Most of the workers work peacefully.
2. Most of the employee travel rarely when it comes to business travels.
3. Research and development has the most investment of the company.
4. Most of the employees have life Sciences as their Education field.
5. There are more Male workers when compare to Female workers.
6. When it comes to Marital Status the most of the employees are married and then next comes are single and finally are divorced.
7. All the employees are over 18(age).
8. Most of the employees do not prefer over time. only 400 workers work overtime.

PIE CHART:

Education indifferent fields represented in form of a pie chart.
Blue: life sciences
Purple: technical degree
Red: marketing
Green: medical
Orange: other

```
plt.pie(df, labels= df.index, autopct="%.0f%%")
#A pie chart displays the percentage distribution of a categorical variable in the dataset.
#create the pie chart using plt.pie()
```

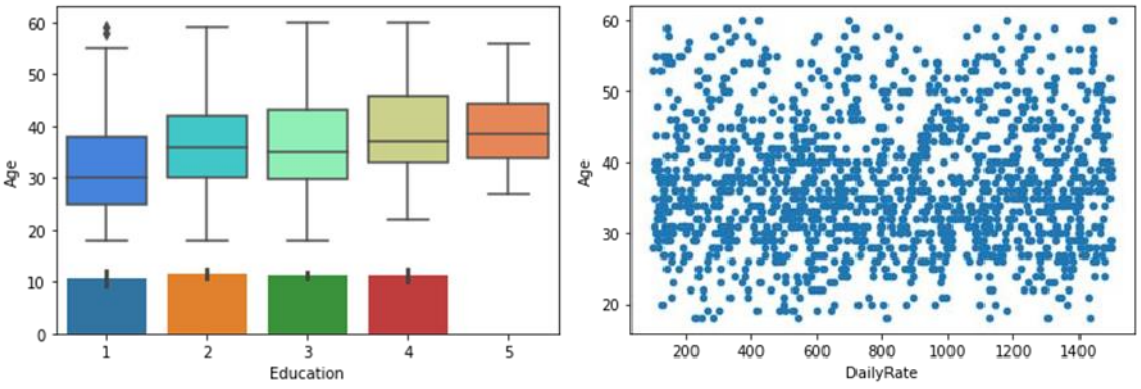
```
([<matplotlib.patches.Wedge at 0x7a4b64cd0ca0>,
  <matplotlib.patches.Wedge at 0x7a4b64cd39a0>,
  <matplotlib.patches.Wedge at 0x7a4b64925d80>,
  <matplotlib.patches.Wedge at 0x7a4b6478c3a0>,
  <matplotlib.patches.Wedge at 0x7a4b6478e860>],
 [Text(0.3754857506198448, 1.0339296161158418, '3'),
  Text(-1.0870032443661248, -0.16859402936497775, '4'),
  Text(0.03995548572036188, -1.09927410556278, '2'),
  Text(0.9269929359739639, -0.592185863267919, '1'),
  Text(1.0942172941140829, -0.11264330100656154, '5')],
 [Text(0.20481040942900622, 0.5639616087904591, '39%'),
  Text(-0.5929108605633407, -0.09196037965362422, '27%'),
  Text(0.021793901302015566, -0.5996040575796981, '19%'),
  Text(0.505632510531253, -0.32301047087341034, '12%'),
  Text(0.5968457967894997, -0.06144180054903356, '3%')])
```



BIVARIATE DATA ANALYSIS

Bivariate analysis is preformed to find the relationship between each variable in the dataset and the target variable of interest or using 2 variables and finding the relationship between them.

Categorical vs continuous (numerical) columns: Boxplot, Bar plot
Continuous vs continuous columns: scatter plot
Categorical vs Categorical columns: Group By (Sum, Count, Value Count)



Box plot between Education vs Age.
Where employees are also compared with their age and working rate.

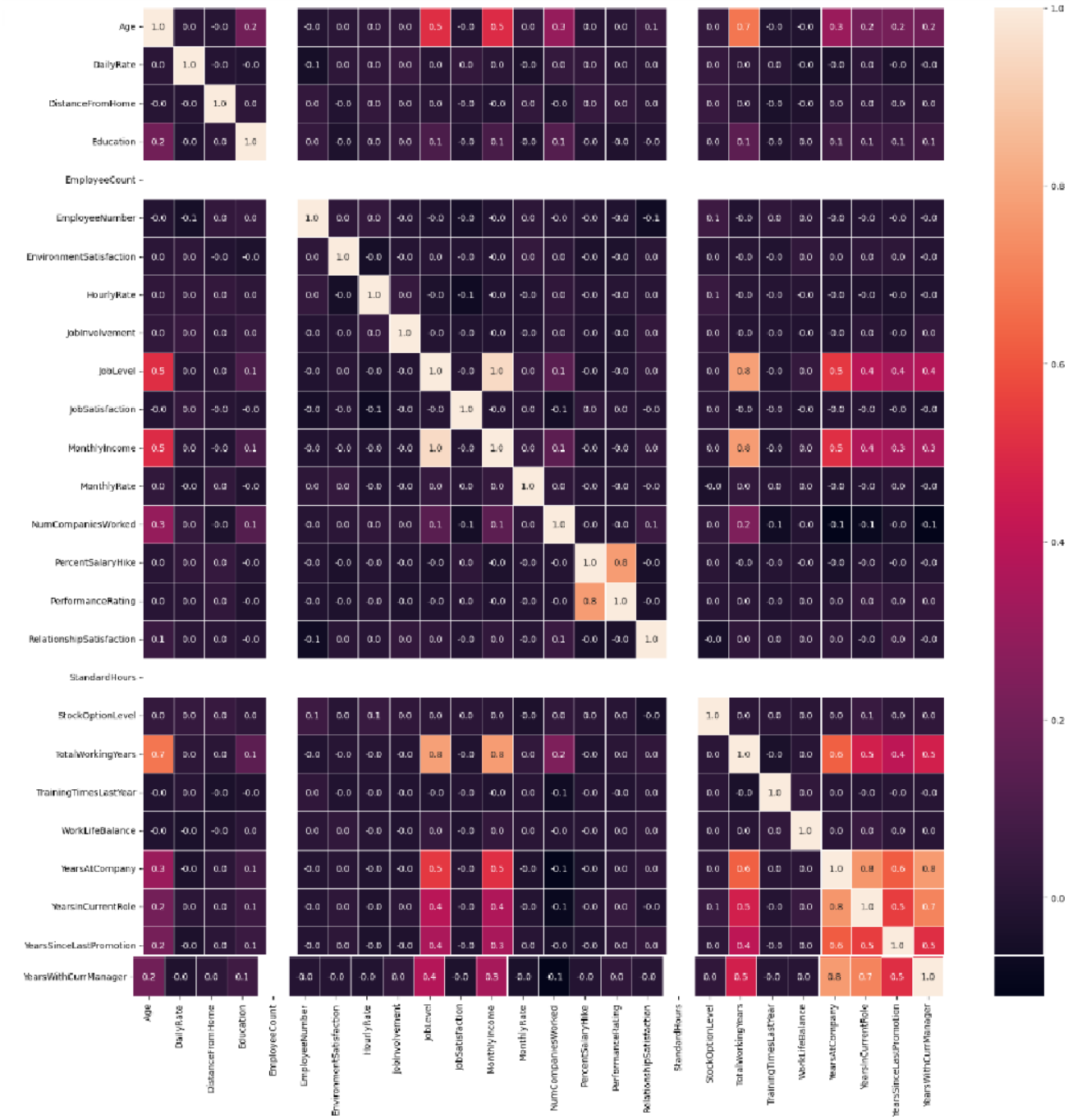
MULTIVARIATE DATA ANALYSIS

Multivariate analysis is used to describe analyses of data where there are multiple variables or observations for each unit or individual.

HEATMAP:

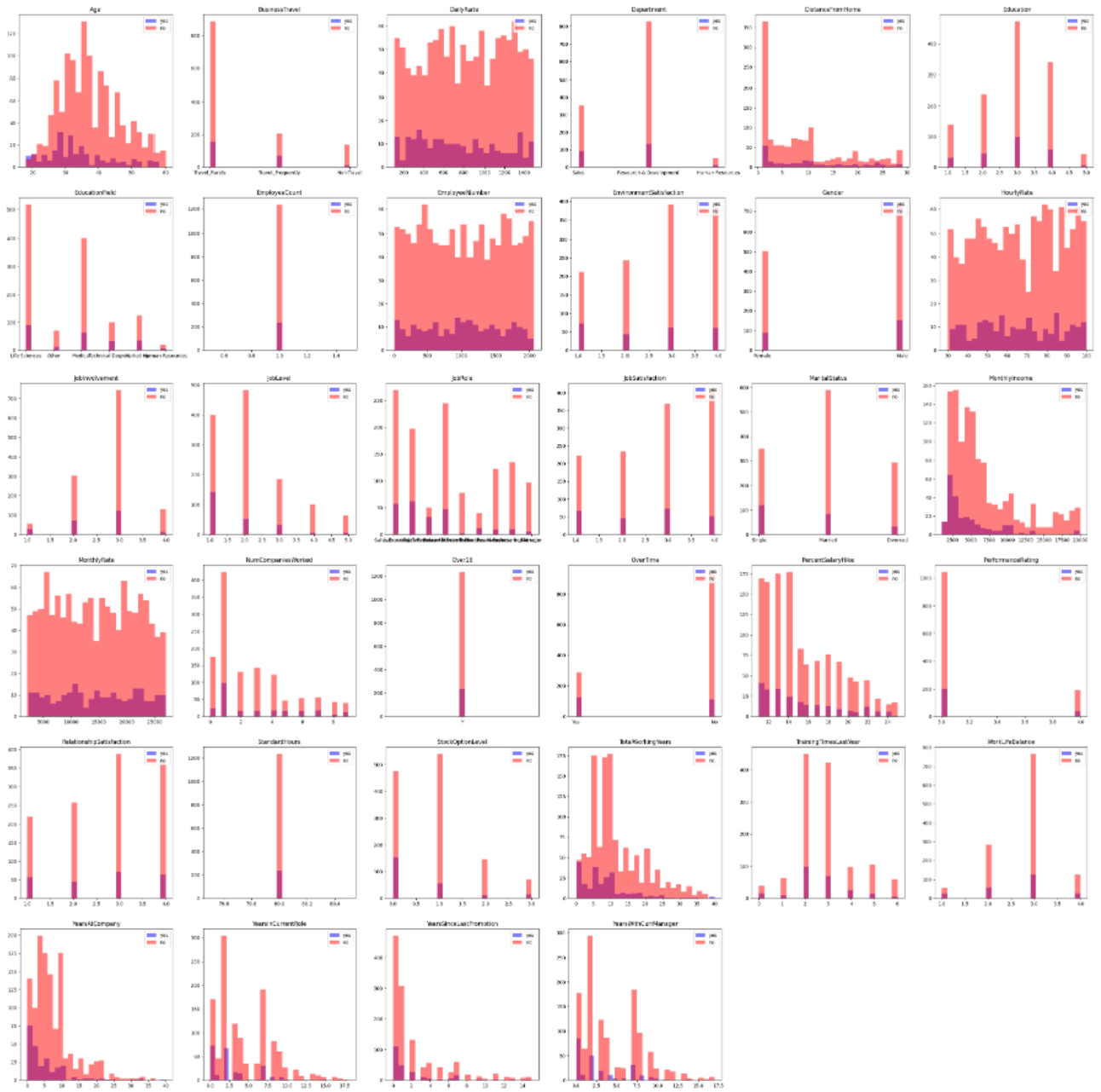
A heatmap is used as a visual representation of data where values in a matrix are represented as colors. It provides a quick and intuitive way to understand the distribution, patterns, and relationships within the data. Heatmaps are particularly useful for identifying correlations between variables in a dataset.

```
f,ax=plt.subplots(figsize=(20,20))
sns.heatmap(hr_data.corr(),annot=True,linewidth=.5,fmt='.1f')
```



PLOT DISTRIBUTION:

```
#plot distributions
k=1
plt.figure(figsize=(40, 40))
for col in hr_data:
  if col=="Attrition":
    continue
  yes = hr_data[hr_data['Attrition'] == 'Yes'][col]
  no = hr_data[hr_data['Attrition'] == 'No'][col]
  plt.subplot(6, 6, k)
  plt.hist(yes, bins=25, alpha=0.5, label='yes', color='b')
  plt.hist(no, bins=25, alpha=0.5, label='no', color='r')
  plt.legend(loc='upper right')
  plt.title(col)
  k+=1
```



USING AGE COLUMN:

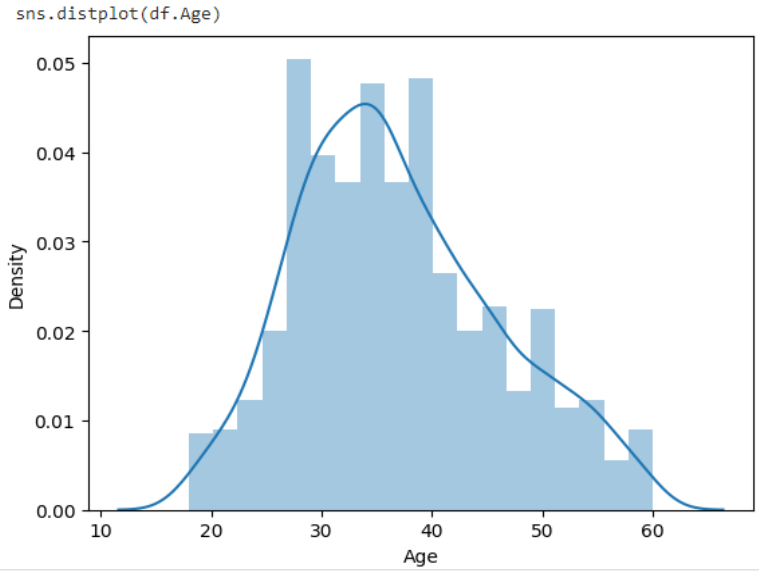
```
df = hr_data[['Age']]
df
```

	Age
0	41
1	49
2	37
3	33
4	27
...	...
1465	36
1466	39
1467	27
1468	49
1469	34

1470 rows × 1 columns

DISPLOT:

```
sns.distplot(df.Age)
plt.show()
```



Observations:

1.Age group of 26-28 years are most active.
2.Most of the workers are from the age 26-40.
3.Mostly employees from age 30-40 are promoted.

MEAN AND STANDARD DEVIATION OF THE AGE COLUMN:

```
df.Age.mean()
```

```
36.923809523809524
```

```
df.Age.std()
```

```
9.135373489136732
```

CALCULATING MEAN BY SAMPLE SIZE:

```
samp_size = 30
df.Age.sample(samp_size).mean()
```

```
33.63333333333333
```

CONCLUSION:

In conclusion, our journey through exploratory data analysis (EDA) in the realm of HR employee attrition has illuminated the intricate dynamics underlying this critical organizational challenge. By harnessing the power of tools like pandas, NumPy, seaborn, and matplotlib.pyplot, we embarked on a comprehensive exploration of the dataset, unraveling hidden patterns and insights. Additionally,

1.Most of the employees left their jobs after getting a promotion.
2.Probably after 4-8 years of time employee quits the job.
3.sales representative job roles have the highest attrition rate.

REFERENCES:

- **https://jovian.ml/learn/data-analysis-with-python-zero-to-pandas**
- **https://matplotlib.org/3.1.1/index.html**
- **https://pandas.pydata.org/pandas-docs/stable/index.html**
- **https://www.geeksforgeeks.org/**
- **https://seaborn.pydata.org/examples/index.html**

PROJECT CODE:
https://drive.google.com/drive/folders/17OvyGEzCQI_XpsAVgrZdARRZkWfaAWcC?usp=drive_link