Attrition : Company losing its Employee Base

Prepared by: *Biswarup Das*

Employee attrition is the gradual reduction in employee numbers. Employee attrition happens when the size of your workforce diminishes over time. This means that employees are leaving faster than they are hired. Employee attrition happens when employees retire, resign, or simply aren't replaced.

Calculating Employee Attrition Rate: The Formula

- Attrition Rate = (No. of employees resigned/No. of employees at the start of the month + no. of employees joined - no. of employees resigned) x 100
- Suppose an organization has 100 employees working. In a particular month, 50 new employees join, and subsequently, 30 employees leave the company.
- Plugging the values in the formula (30/100+50-30)x100 = 25%. This is a very high attrition rate. Ideally, attrition rate should be less than 10%.
- And what the top factors which lead to employee attrition? Let's find out from the data.

Dataset Link https://drive.google.com/file/d/1t1tC7y_PqeH-i-kMCOSEC77LmgX8jtIm/view

Description about the data

- **Age:** A period of employee life, measured by years from birth.
- **Attrition:** The departure of employees from the organization.
- **BusinessTravel:** Did the employee travel on a business trip or not.
- **DailyRate:** Employee salary for the period is divided by the amount of calendar days in the period.
- **Department:** In which department the Employee working.
- **DistanceFromHome:** How far the Employee live from the office location.
- **Education:** In education 1 means 'Below College', 2 means 'College', 3 means 'Bachelor', 4 means 'Master', 5 means 'Doctor'
- **EducationField:** In which field Employee complete his education.
- **EmployeeCount:** How many employee working in a department
- **EmployeeNumber:** An Employee Number is a unique number that has been assigned to each current and former State employee and elected official in the Position and Personnel DataBase (PPDB).
- **Job involvement:** Is the degree to which an employee identifies with their work and actively participates in it where 1 means 'Low', 2 means 'Medium', 3 means 'High', 4 means 'Very High'
- **JobLevel:** Job levels, also known as job grades and classifications, set the responsibility level and expectations of roles at your organization. They may be further defined by impact, seniority, knowledge, skills, or job title, and are often associated with a pay band. The way you structure your job levels should be dictated by the needs of your unique organization and teams.
- **JobRole:** What is the jobrole of an employee.
- **JobSatisfaction:** Employee job satisfaction rate where, 1 means 'Low', 2 means 'Medium', 3 means 'High', 4 means 'Very High'
- **MaritalStatus:** Marital status of the employee.
- **MonthlyIncome:** total monetary value paid by the organization to an employee.
- **MonthlyRate:** The per-day wage of the employee.
- **NumCompaniesWorked:** Before joining this organization how many organizations employee worked.
- **Over18:** Is the employee age over than 18 or not.
- **OverTime:** A Employee works more than 9 hours in any day or for more than 48 hours in any week.
- **PercentSalaryHike:**
- **PerformanceRating:** 1 'Low' 2 'Good' 3 'Excellent' 4 'Outstanding'
- **EnvironmentSatisfaction:** 1 'Low' 2 'Medium' 3 'High' 4 'Very High'
- **RelationshipSatisfaction:** 1 'Low' 2 'Medium' 3 'High' 4 'Very High'
- **StandardHours:** Is the number of hours of production time that should have been used during an working period.
- **StockOptionLevel:** Employee stock options, also known as ESOs, are stock options in the company's stock

StockOptionLevel: Employee stock options, also known as ESOs, are stock options in the company's stock granted by an employer to certain employees. Typically they are granted to those in management or officer-level positions. Stock options give the employee the right to buy a certain amount of stock at a specific price, during a specific period of time. Options typically have expiration dates as well, by which the options must have been exercised, otherwise they will become worthless.

- **TotalWorkingYears:** Total years the employee working in any organization
- **TrainingTimesLastYear:** Last year how many times employee took training session.
- **WorkLifeBalance:** 1 'Bad' 2 'Good' 3 'Better' 4 'Best'
- **YearsAtCompany:** How many years the employee working in the current organization
- **YearsInCurrentRole:** How many years the employee working in the current position
- **YearsSinceLastPromotion:** How many years the employee working in the current position after promotion
- **YearsWithCurrManager:** How many years the employee working under the current manager

Getting the Data

In [1]:

```
import pandas as pd
employee=pd.read_csv('WA_Fn-UseC_-HR-Employee-Attrition.csv')
```

1. Analysing Employee Data:

Before moving forward let's check for any data missing or null values present in the dataset or any features that is unimportant for this analysis.

In [2]:

```
# display all the features from the dataset
pd.set_option('display.max_columns',None)
```

In [3]:

```
employee.head(3)
```

Out[3]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1

Data Information

In [4]:

```
print(employee.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
None
```

In [5]:

```
employee.describe()
```

Out[5]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfactio
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	1470.000000	1470.00000
mean	36.923810	802.485714	9.192517	2.912925	1.0	1024.865306	2.72176
std	9.135373	403.509100	8.106864	1.024165	0.0	602.024335	1.09308
min	18.000000	102.000000	1.000000	1.000000	1.0	1.000000	1.00000
25%	30.000000	465.000000	2.000000	2.000000	1.0	491.250000	2.00000
50%	36.000000	802.000000	7.000000	3.000000	1.0	1020.500000	3.00000
75%	43.000000	1157.000000	14.000000	4.000000	1.0	1555.750000	4.00000
max	60.000000	1499.000000	29.000000	5.000000	1.0	2068.000000	4.00000

Observations

- **We can see that there are 9 categorical values (Object) present in the dataset and rest are integer type values.**
- **It's very good that we are having a complete dataset, there is no missing values in dataset.**

Checking Duplicates Values

In [6]:

```
print(employee.duplicated().value_counts())
employee.drop_duplicates(inplace = True)
print(len(employee))
```

```
False    1470
dtype: int64
1470
```

Unique Values

In [7]:

```
# Let's see all unique categorical values at a glance
import numpy as np
print("Attrition      : ",np.unique(employee['Attrition']))
print('BusinessTravel: ',np.unique(employee['BusinessTravel']))
print('Department     : ',np.unique(employee['Department']))
print('EducationField: ',np.unique(employee['EducationField']))
print('Gender         : ',np.unique(employee['Gender']))
print('JobRole        : ',np.unique(employee['JobRole']))
print('MaritalStatus : ',np.unique(employee['MaritalStatus']))
print('Over18         : ',np.unique(employee['Over18']))
print('OverTime       : ',np.unique(employee['OverTime']))
```

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

From the above result we can see many entities in each categorical column except "Over18" column. So in the part of Data Filteration we will remove "Over18" column.

In [8]:

```
# Analysing some other numerical values
print('StandardHours : ',np.unique(employee['StandardHours']))
print('EmployeeCount : ',np.unique(employee['EmployeeCount']))
print('EmployeeNumber: ',np.unique(employee['EmployeeNumber']))
```

```
StandardHours :  [80]
EmployeeCount :  [1]
EmployeeNumber:  [   1    2    4 ... 2064 2065 2068]
```

In the above, 3 columns we can see that the column "StandardHour" & "EmployeeCount" have fixed value and surely it's not going impact our futher analysis. In the case of "EmployeeNumber" column, it is a continuous value and only associated with individual employee so we can also remove this column along with previous 2 columns from our Dataset.

2. Data Processing (Filteration):

Replacing all the categorical values to numerical values.

Attrition

- :No = 0
- :Yes = 1

BusinessTravel

- : Non-Travel = 0
- : Travel_Rarely = 1
- : Travel_Frequently = 2

Department

- : Human Resources = 0
- : Research & Development = 1
- : Sales = 2

EducationField

- : Other = 0
- : Life Sciences = 1
- : Marketing = 2
- : Medical = 3
- : Technical Degree= 4
- : Human Resources = 5

Gender

- : Female = 0
- : Male = 1

JobRole

- : Healthcare Representative = 0
- : Human Resources = 1
- : Laboratory Technician = 2
- : Manager = 3
- : Manufacturing Director = 4
- : Research Director = 5
- : Research Scientist = 6
- : Sales Executive = 7
- : Sales Representative = 8

MaritalStatus

- : Divorced = 0
- : Married = 2
- : Single = 1

OverTime

- : No = 0
- : Yes = 1

In [9]:

```
employee['Attrition']      =employee['Attrition'].map({'Yes':1,'No':0})
employee['BusinessTravel']=employee['BusinessTravel'].map({'Non-Travel':0, 'Travel_Freque
ntly':2, 'Travel_Rarely':1})
employee['Department']     =employee['Department'].map({'Human Resources':0, 'Research &
Development':1, 'Sales':2})
employee['EducationField']=employee['EducationField'].map({'Human Resources':5, 'Life Sci
ences':1, 'Marketing':2,
                                                              'Medical':3,'Other':0, 'Tech
nical Degree':4})
employee['Gender']         =employee['Gender'].map({'Female':0, 'Male':1})
employee['JobRole']        =employee['JobRole'].map({'Healthcare Representative':0, 'Huma
n Resources':1,
                                                    'Laboratory Technician':2,'Manager':3,
'Manufacturing Director':4,
```

```
                                                'Research Director':5, 'Research Scien
tist':6, 'Sales Executive':7,
                                                'Sales Representative':8})
employee['MaritalStatus'] =employee['MaritalStatus'].map({'Divorced':0, 'Married':2, 'Sin
gle':1})
employee['OverTime']      =employee['OverTime'].map({'No':0, 'Yes':1})
```

Dropping unnessesary columns

Reasons:

- **StandardHours : Have fixed values for all the rows.**
- **EmployeeCount : Have fixed values for all the rows.**
- **EmployeeNumber: Have no significance with our goal.**
- **Over18 : Have fixed values for all the rows.**

In [10]:

```
employee=employee.drop(['StandardHours','EmployeeCount','EmployeeNumber','Over18'],axis=1
)
```

In [11]:

```
employee.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1470 entries, 0 to 1469
Data columns (total 31 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Age                       1470 non-null   int64
 1   Attrition                 1470 non-null   int64
 2   BusinessTravel            1470 non-null   int64
 3   DailyRate                 1470 non-null   int64
 4   Department                1470 non-null   int64
 5   DistanceFromHome          1470 non-null   int64
 6   Education                 1470 non-null   int64
 7   EducationField            1470 non-null   int64
 8   EnvironmentSatisfaction   1470 non-null   int64
 9   Gender                    1470 non-null   int64
 10  HourlyRate                1470 non-null   int64
 11  JobInvolvement            1470 non-null   int64
 12  JobLevel                  1470 non-null   int64
 13  JobRole                   1470 non-null   int64
 14  JobSatisfaction           1470 non-null   int64
 15  MaritalStatus             1470 non-null   int64
 16  MonthlyIncome             1470 non-null   int64
 17  MonthlyRate               1470 non-null   int64
 18  NumCompaniesWorked        1470 non-null   int64
 19  OverTime                  1470 non-null   int64
 20  PercentSalaryHike         1470 non-null   int64
 21  PerformanceRating         1470 non-null   int64
 22  RelationshipSatisfaction  1470 non-null   int64
 23  StockOptionLevel          1470 non-null   int64
 24  TotalWorkingYears         1470 non-null   int64
 25  TrainingTimesLastYear     1470 non-null   int64
 26  WorkLifeBalance           1470 non-null   int64
 27  YearsAtCompany            1470 non-null   int64
 28  YearsInCurrentRole        1470 non-null   int64
 29  YearsSinceLastPromotion   1470 non-null   int64
 30  YearsWithCurrManager      1470 non-null   int64
dtypes: int64(31)
memory usage: 367.5 KB
```

Checking Missing Values

In [12]:

```
print('Data columns with null values:\n',
     employee.isnull().sum())
```

```
Data columns with null values:
 Age                         0
Attrition                   0
BusinessTravel              0
DailyRate                   0
Department                  0
DistanceFromHome            0
Education                   0
EducationField              0
EnvironmentSatisfaction     0
Gender                      0
HourlyRate                  0
JobInvolvement              0
JobLevel                    0
JobRole                     0
JobSatisfaction             0
MaritalStatus               0
MonthlyIncome               0
MonthlyRate                 0
NumCompaniesWorked          0
OverTime                    0
PercentSalaryHike           0
PerformanceRating           0
RelationshipSatisfaction    0
StockOptionLevel            0
TotalWorkingYears           0
TrainingTimesLastYear       0
WorkLifeBalance             0
YearsAtCompany              0
YearsInCurrentRole          0
YearsSinceLastPromotion     0
YearsWithCurrManager        0
dtype: int64
```

In [13]:

```
employee.head(5)
```

Out[13]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EnvironmentSatisfact
0	41	1	1	1102	2	1	2	1	
1	49	0	2	279	1	8	1	1	
2	37	1	1	1373	1	2	2	0	
3	33	0	2	1392	1	3	4	1	
4	27	0	1	591	1	2	1	3	

Now all values in the dataset are in integer format and no float values as well

Correlation

Why do we need correlation in machine learning?

- **Correlation is a highly applied technique in machine learning during data analysis and data mining. It can extract key problems from a given set of features, which can later cause significant damage during the fitting model.**

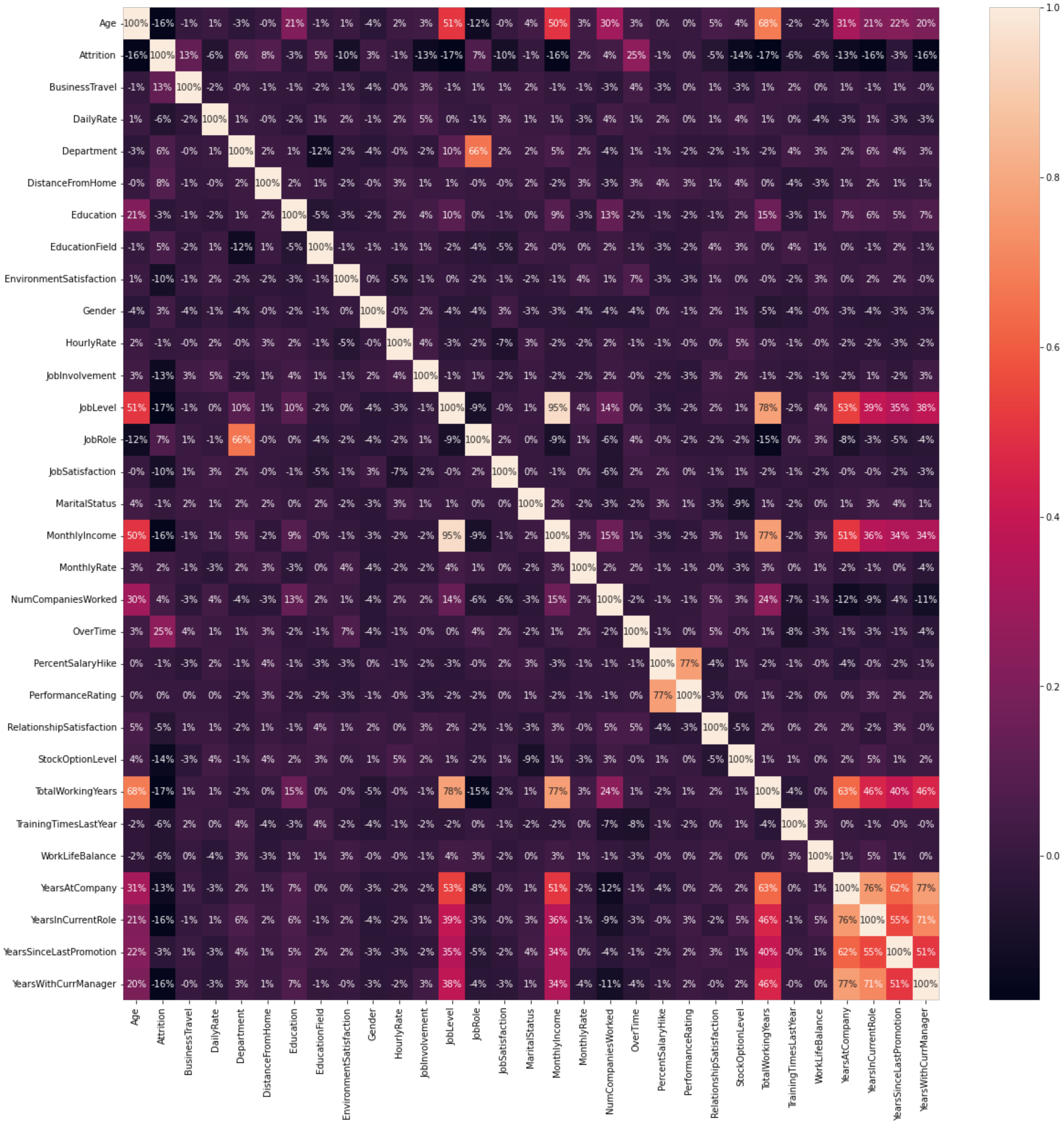
In [14]:

```
import matplotlib.pyplot as plt
```

```python
import matplotlib.pyplot as plt
import seaborn as sns
correlation=round(employee.corr(),2)
plt.figure(figsize=(20,20))
sns.heatmap(correlation,annot=True, fmt='.0%')
```

Out[14]:

<AxesSubplot:>



After analysing the correlation matrix I have figured out some key features to look on:

- **Positive Correlation Percentage(+):**
- **Considering percentage: >50%**

```
JobLevel            + MonthlyIncome        = 95%
JobLevel            + TotalWorkingYears    = 78%
MonthlyIncome       + TotalWorkingYears    = 77%
PercentSalaryHike   + PerformanceRating    = 77%
YearsAtCompany      + YearsWithCurrManager = 77%
YearsAtCompany      + YearsInCurrentRole   = 76%
```

```
YearsInCurrentRole       + YearsWithCurrManager    = 71%
Age                      + TotalWorkingYears       = 68%
Department               + JobRole                 = 66%
TotalWorkingYears        + YearsAtCompany          = 63%
YearsAtCompany           + YearsSinceLastPromotion = 62%
YearsInCurrentRole       + YearsSinceLastPromotion = 55%
JobLevel                 + YearsAtCompany          = 53%
JobLevel                 + Age                     = 51%
MonthlyIncome            + YearsAtCompany          = 51%
YearsSinceLastPromotion + YearsWithCurrManager    = 51%
```

- **Negetive Correlation Percentage(-):**
- **Considering percentage: <-10%**

```
Attrition                + JobLevel                = -17%
Attrition                + TotalWorkingYears       = -17%
Attrition                + Age                     = -16%
Attrition                + YearsWithCurrManager    = -16%
Attrition                + YearsInCurrentRole      = -16%
Attrition                + MonthlyIncome           = -16%
Attrition                + YearsInCurrentRole      = -16%
Attrition                + StockOptionLevel        = -14%
Attrition                + YearsAtCompany          = -13%
Attrition                + JobInvironment          = -13%
Department               + EducationField          = -12%
JobRole                  + TotalWorkingYears       = -15%
JobRole                  + Age                     = -12%
NumCompaniesWorked       + YearsAtCompany          = -12%
NumCompaniesWorked       + YearsWithCurrManager    = -11%
```

Therefore, from the above correlation, wheather it is negetive or positive I can easily filter the most important 17 out of 30 features which are directly reponsible for Employee Attrition.

- **Those are:**

```
1 .Age
2 .Attrition
3 .Department
4 .EducationField
5 .JobInvolvement
6 .JobRole
7 .JobLevel
8 .PercentSalaryHike
9 .PerformanceRating
10.StockOptionLevel
11.MonthlyIncome
12.NumCompaniesWorked
13.TotalWorkingYears
14.YearsAtCompany
15.YearsInCurrentRole
16.YearsSinceLastPromotion
17.YearsWithCurrManager
```

3. Data Visualization

In [15]:

```
attrition_count = pd.DataFrame(employee['Attrition'].value_counts())
plt.pie(attrition_count['Attrition'] , labels = ['No' , 'Yes'] , explode = (0.2,0))
print(attrition_count)
```

```
   Attrition
0       1233
1        237
```

Out of 1470 employees 237 left the company and 1233 employees still remains. It actually 16% of the entire company resources.

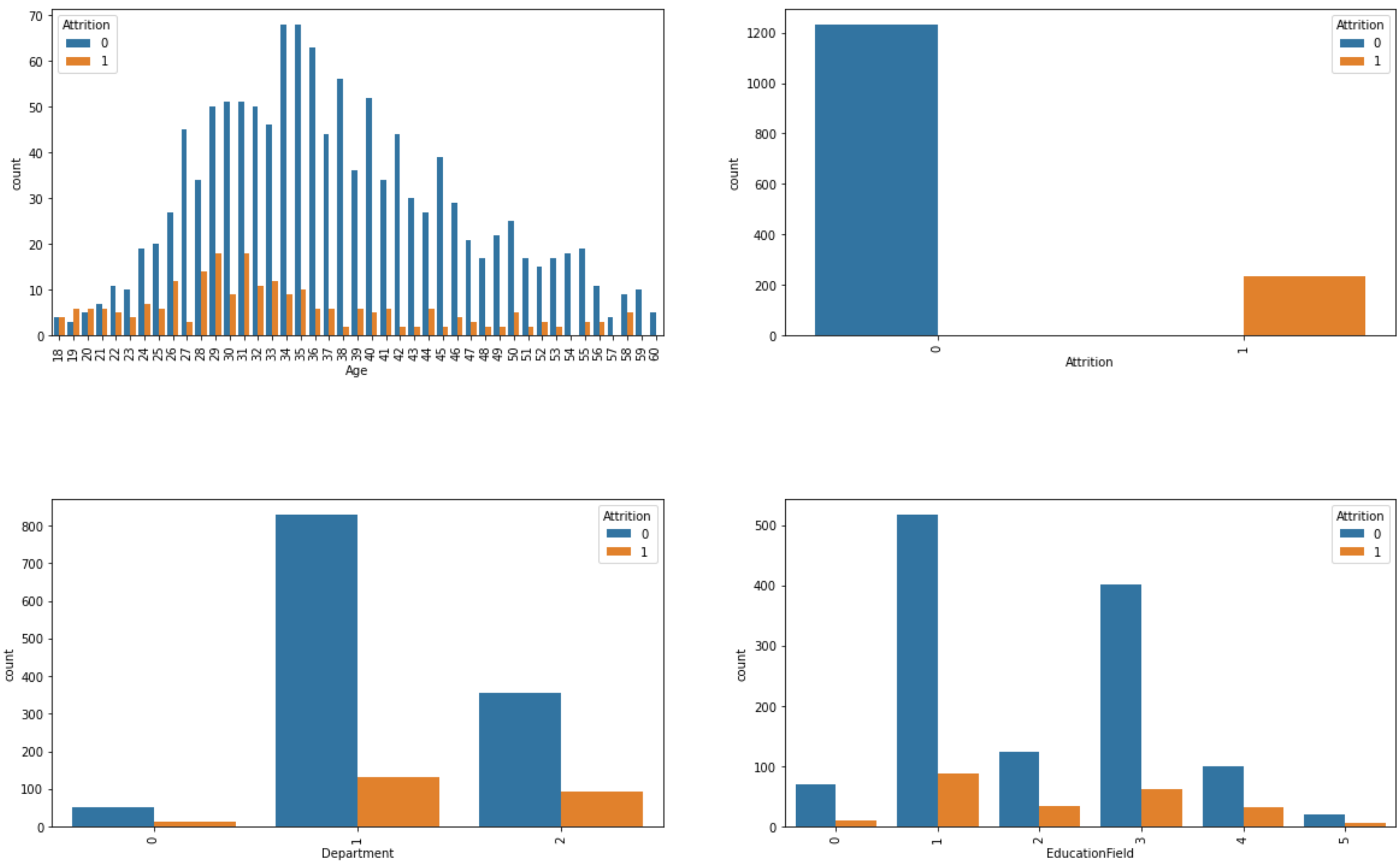
Visualizing other features

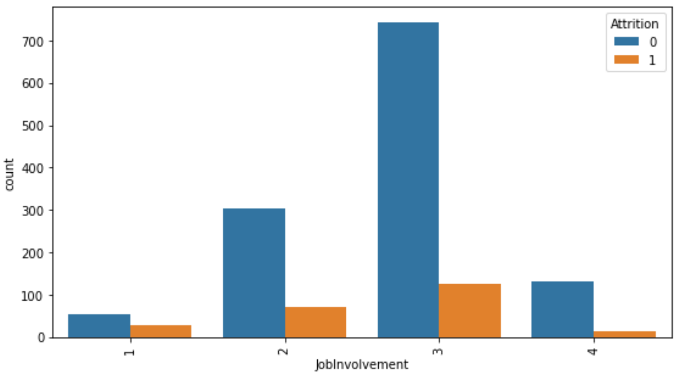
In [16]:

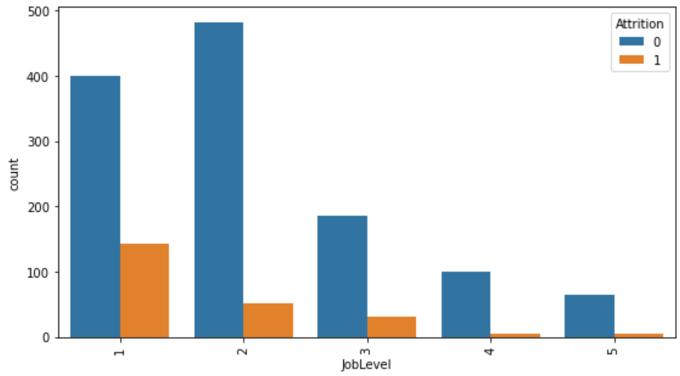
```python
feature = ['Age','Attrition','Department','EducationField','JobInvolvement','JobLevel','
JobRole','MonthlyIncome',
           'StockOptionLevel','NumCompaniesWorked','PercentSalaryHike','PerformanceRating
','TotalWorkingYears',
           'YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCur
rManager']
```

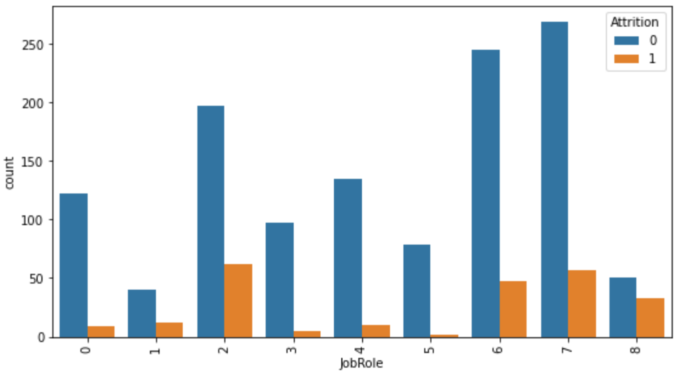
In [17]:

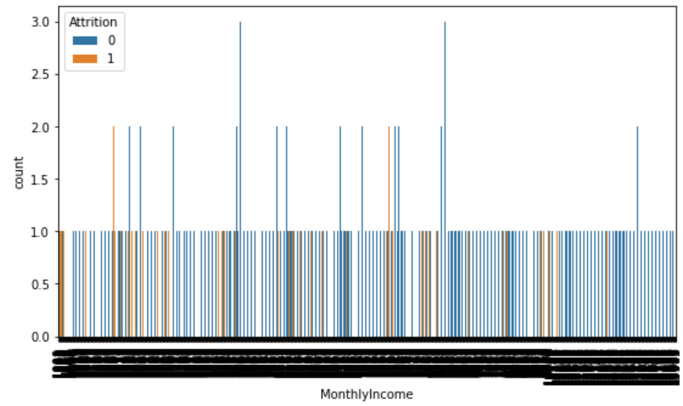
```python
fig = plt.subplots(figsize=(20,65))
for p, q in enumerate(feature):
    plt.subplot(9,2,p+1)
    plt.subplots_adjust(hspace=0.5)
    sns.countplot(x=q,data=employee, hue='Attrition')
    plt.xticks(rotation=90)
```

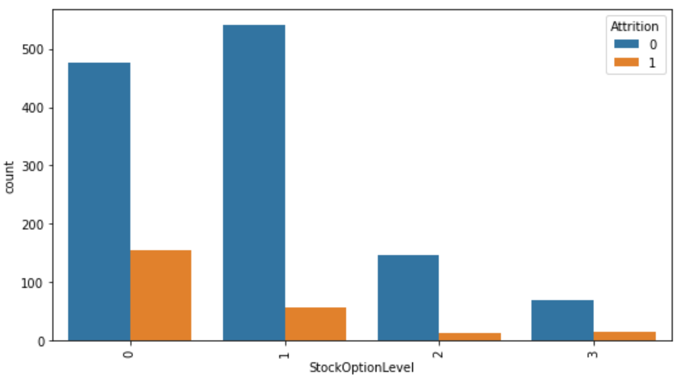


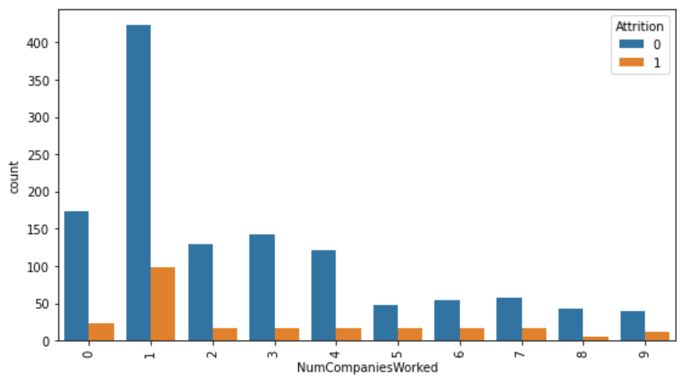

Attrition
0
1
count
0
100
200
300
400
500
600
700
1
2
3
4
JobInvolvement

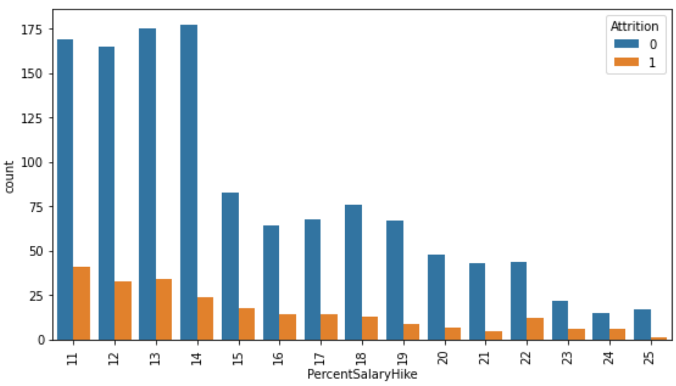

Attrition
0
1
count
0
100
200
300
400
500
1
2
3
4
5
JobLevel

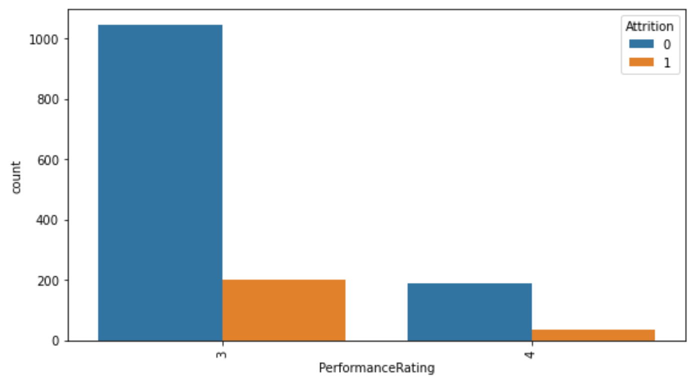

Attrition
0
1
count
0
50
100
150
200
250
0
1
2
3
4
5
6
7
8
JobRole


Attrition
0
1
count
0.0
0.5
1.0
1.5
2.0
2.5
3.0
MonthlyIncome


Attrition
0
1
count
0
100
200
300
400
500
0
1
2
3
StockOptionLevel

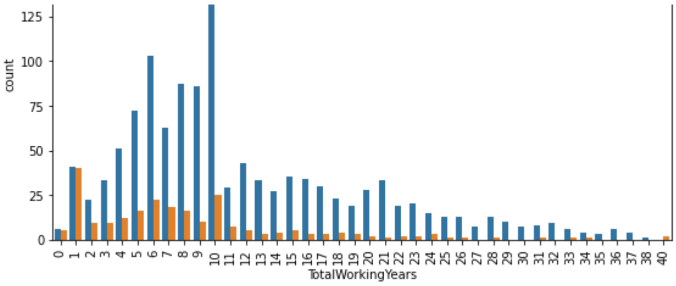

Attrition
0
1
count
0
50
100
150
200
250
300
350
400
0
1
2
3
4
5
6
7
8
9
NumCompaniesWorked

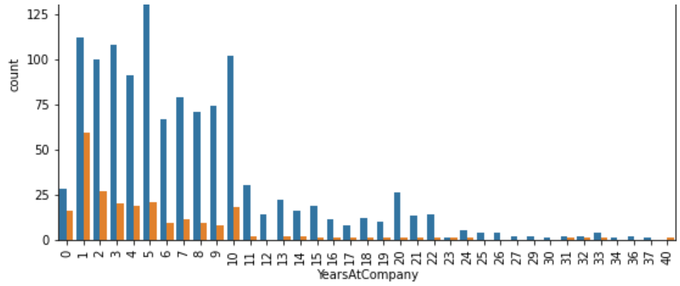

Attrition
0
1
count
0
25
50
75
100
125
150
175
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
PercentSalaryHike

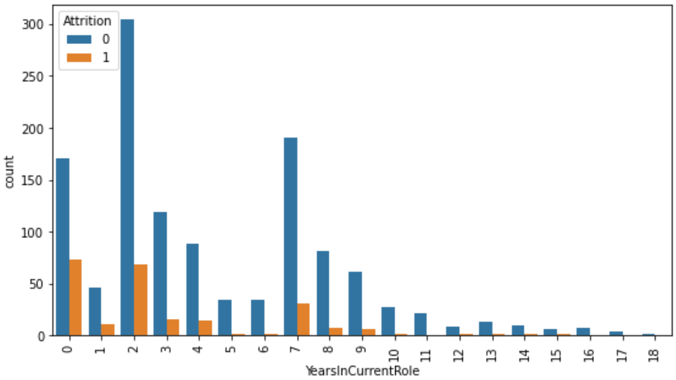

Attrition
0
1
count
0
200
400
600
800
1000
3
4
PerformanceRating

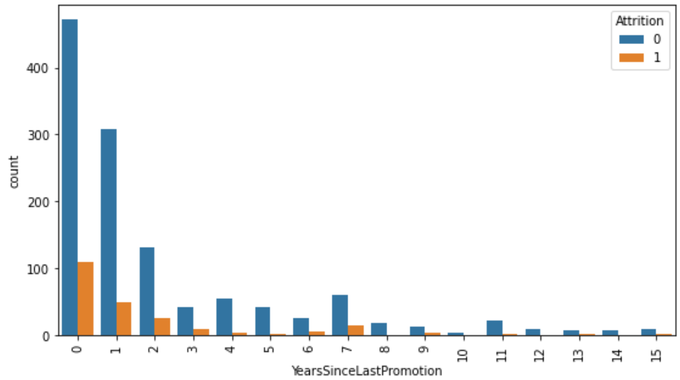

Attrition
0
1
150
175

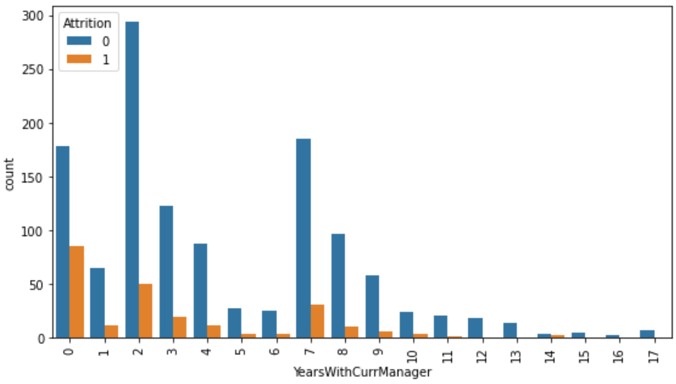

Attrition
0
1
150
175











Feature wise Analysis: Color ORANGE denotes employees leaving the company and BLUE shows employees still with the company.

Age:

- **We can clearly visualize from the age group of 18 to 25, there is a very high churn rate and it goes even higher on the age group of 26 to all the way upto 37. It shows that when an employee start his/her professional career they are more prone to switch companies and it happens even high with employees having ample amount of exprerience. To understand deeply we have to analyse others features.**

Department:

- **Human Resources :0**
- **Research & Development:1**
- **Sales :2**
- **Here in Department R&D and Sales, churning of employees are much higher than HR department. There must be someother factors involved in it, let's find out.**

EducationField:

- **Other :0**
- **Life Sciences :1**
- **Marketing :2**
- **Medical :3**
- **Technical Degree:4**
- **Human Resources :5**

- It was seen that employees from LifeScience and Medical background have very high churn tendency. After them people from Marketing and Technical background also have significant high churn rate.

JobInvolvement:

- level 1
- level 2
- level 3
- level 4
- Here we can see that when the job involvement increases the churn rate is also gone high upto level 3 starting starting from level 1. It clearly shows people with extra burden of work are more prone to leave company.

JobLevel:

- level 1
- level 2
- level 3
- level 4
- level 5
- Interestingly, with low level job employees are more prone to churning out of company and it goes completely opposite when employees have high level of job. So the visuals are pointing out that over the time employees get frustrated as in low level job there is very less to explore opportunities and that frustration push them to make a move and look for other opportunities. It also shows with high level job employees are more satisfied and they have lot of things to explore and work on that basis, which make them less prone to churning out of company.

JobRole:

- Healthcare Representative:0
- Human Resources :1
- Laboratory Technician :2
- Manager :3
- Manufacturing Director :4
- Research Director :5
- Research Scientist :6
- Sales Executive :7
- Sales Representative :8
- Here we can see that Laboratory Technician have the most high attrition rate then Sales Executive, Research Scientist and lastly Sales Representatives have high attrition rate as compared to the others.

MonthlyIncome:

- There are huge data points in monthly income so we can skip it

StockOptionLevel:

- level 0
- level 1
- level 2
- level 3
- StockOption: Employee stock options (ESOs) are a form of equity compensation granted by companies to their employees. ESOs give employees the right to purchase a certain number of shares of the company's stock at a fixed price (the "strike price") for a certain period of time.So in the above visual of StockOptionLevel we can see that employee holding low stock are not caring about the company so chrurning is actually easy for them but when employees having high stock it create dependencies , that's why high stock holding employees are actually staying in the company hoping for a better market value in future.

NumCompaniesWorked:

- When employees having atleast one year of experience, the data shows there is a high chance of churning out of company but when they have more than one year of exprience the churn rate is gone down in a linear manner. So from the visuals we can say that authorities of the company need to have a close watch on the employeesatleast for one to one and half years.

PercentSalaryHike:

- It is very clear by the visuals that low salary hike equals to high churn rate and high salary hike makes employee stay.

PerformanceRating:

- rating 3
- rating 4
- As we know in every MNCs rating matters the most, with high rating employees can have increment, promotion and many other benefits will come. Some in majority every employee work hard to achieve high rating from their manager so employees getting low rating creates dissatisfaction which make them to churn their existing company. So here is no exception, we can clearly see that low rated (3) employees churn rate is high as compared to employees who achieved high rating (4).

TotalWorkingYears:

- From 0 to 10 years of work span we can see there is a trend of leaving the company and after that the habit of leaving is drastically gone down. From 0 to 6 years of work span we can see there is a high curvy and churn rate touches its peak when employee having 1 year or more than 5 years of experience. After that we can see the curvy goes down untill it reaches to 10 (TotalWorkingYears). So people with more than 9 years of experience are also have high chance to churn.

YearsAtCompany:

- Like the above one, here also we can see a trend of chrun from employees of 0 to 10 years and it reaches its peak when employee spend atleast one year with the company.

YearsInCurrentRole:

- Visuals shows there is a high change of churn when an employee having experience of less than one year in a particular role, there might be n number of reasons, role is not suitable or irrelevant job profile, it can be any thing. But surprisingly employees with 2 years of experience in current role are also churning the most.

YearsSinceLastPromotion:

- Employees those who are newly promoted, there is a high change of churn out without serving their position for at least one year and the curve goes down when such employees spend some more time some extra years on their current position. So promotion comes with more responsibilities and extra work load and that is reflecting here. Those employees survives that situation stays for upcoming years.

YearsWithCurrManager:

- Visuals says us employees having less than one year and upto 2 years with the current manager are about to churn more.

That is all we can gather from the visuals.

Here I am going to use LOGISTIC REGRESSION, DECISION TREE & RANDOM FOREST CLASSIFIER in search of best model with higher accuracy.

4.A. Logistic Regression

In [18]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.preprocessing import StandardScaler
```

In [19]:

```
# Further dropping unnessesary columns
employee=employee.drop(['BusinessTravel','DailyRate','DistanceFromHome','Education',
                        'EnvironmentSatisfaction','Gender','HourlyRate','JobSatisfaction
','MaritalStatus',
                       'MonthlyRate','OverTime','RelationshipSatisfaction','TrainingTime
sLastYear',
                       'WorkLifeBalance'],axis=1)
```

In [20]:

```
# printing column names after making a list of them
employee_columns_list=list(employee.columns)
print(employee_columns_list)
```

```
['Age', 'Attrition', 'Department', 'EducationField', 'JobInvolvement', 'JobLevel', 'JobRo
le', 'MonthlyIncome', 'NumCompaniesWorked', 'PercentSalaryHike', 'PerformanceRating', 'St
ockOptionLevel', 'TotalWorkingYears', 'YearsAtCompany', 'YearsInCurrentRole', 'YearsSince
LastPromotion', 'YearsWithCurrManager']
```

In [21]:

```
#features:separate features based on input and output
features=list(set(employee_columns_list)-set(['Attrition'])) #features is our input varia
ble
y1=employee['Attrition'].values # output variable
x1=employee[features].values  # input variable
```

In [22]:

```
# now dividing input data and output data into training and test data
train_x,test_x,train_y,test_y=train_test_split(x1,y1,test_size=0.3,random_state=50)
```

In [23]:

```
# data scaling or normalization
scaler=StandardScaler()
# Fit on training set only
scaler.fit(train_x) # this will bring out mean and variance of the data or store it in me
mory
```

Out[23]:

```
StandardScaler()
```

In [24]:

```
# Apply transform to both the remaining set and test set
train_x=scaler.transform(train_x) # now this with take mean and variance to normalize th
e data
test_x=scaler.transform(test_x) # now this with take mean and variance to normailize the
data
```

In [25]:

```
LRM = LogisticRegression()
```

In [26]:

```
# Fitting the values for x and y
LRM.fit(train_x,train_y)
```

Out[26]:

```
LogisticRegression()
```

In [27]:

```
# Prediction from test data
prediction = LRM.predict(test_x)
prediction
```

Out[27]:

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

In [28]:

```
# Confusion matrix
confusion_matrix = confusion_matrix(prediction,test_y)
confusion_matrix
```

Out[28]:

```
array([[369,  68],
       [  1,   3]])
```

Model Accuracy

In [29]:

```
# Calculating the accuracy
accuracy_score = accuracy_score(prediction,test_y)
atd2=LRM.score(train_x, train_y)
print('Accuracy of Trained Data:',atd2)
print('Model Accuracy Score    :',accuracy_score)
```

```
Accuracy of Trained Data: 0.8464528668610302
Model Accuracy Score    : 0.8435374149659864
```

4.B. Decision Tree Classifier

In [30]:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.tree import DecisionTreeClassifier
```

In [31]:

```
employee_columns_list2=employee_columns_list.copy()
employee_columns_list2
```

Out[31]:

```
['Age',
 'Attrition',
 'Department',
```

```
 'EducationField',
 'JobInvolvement',
 'JobLevel',
 'JobRole',
 'MonthlyIncome',
 'NumCompaniesWorked',
 'PercentSalaryHike',
 'PerformanceRating',
 'StockOptionLevel',
 'TotalWorkingYears',
 'YearsAtCompany',
 'YearsInCurrentRole',
 'YearsSinceLastPromotion',
 'YearsWithCurrManager']
```

In [32]:

```
#features:separate features based on input and output
features1=list(set(employee_columns_list2)-set(['Attrition'])) #features is our input var
iable
y1=employee['Attrition'].values # output variable
x1=employee[features1].values   # input variable
```

In [33]:

```
# Splitting the dataset into training and test set.

# now dividing input data and output data into training and test data
train_x,test_x,train_y,test_y=train_test_split(x1,y1,test_size=0.3,random_state=3)
```

In [34]:

```
#feature Scaling

st_x= StandardScaler()
train_x= st_x.fit_transform(train_x)
test_x= st_x.transform(test_x)
```

In [35]:

```
#Fitting Decision Tree classifier to the training set

DT= DecisionTreeClassifier(criterion='entropy', random_state=0)
DT.fit(train_x, train_y)
```

Out[35]:

```
DecisionTreeClassifier(criterion='entropy', random_state=0)
```

In [36]:

```
#Predicting the test set result
y_pred= DT.predict(test_x)
```

In [37]:

```
#Creating the Confusion matrix
confusion_matrix(test_y, y_pred)
```

Out[37]:

```
array([[305,  61],
       [ 44,  31]])
```

Model Accuracy

In [38]:

```
# calculating the accuracy score
accuracy_score1 = accuracy_score(test_y,y_pred)
```

```
atd1=DT.score(train_x, train_y)
print('Accuracy of Trained Data:',atd1)
print('Model Accuracy Score    :',accuracy_score1)
```

```
Accuracy of Trained Data: 1.0
Model Accuracy Score     : 0.7619047619047619
```

4.C. Random Forest Classifier

In [39]:

```
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error
from sklearn.metrics import accuracy_score, confusion_matrix
```

In [40]:

```
employee_columns_list3=employee_columns_list.copy()
employee_columns_list3
```

Out[40]:

```
['Age',
 'Attrition',
 'Department',
 'EducationField',
 'JobInvolvement',
 'JobLevel',
 'JobRole',
 'MonthlyIncome',
 'NumCompaniesWorked',
 'PercentSalaryHike',
 'PerformanceRating',
 'StockOptionLevel',
 'TotalWorkingYears',
 'YearsAtCompany',
 'YearsInCurrentRole',
 'YearsSinceLastPromotion',
 'YearsWithCurrManager']
```

In [41]:

```
# Segrigating data
features2 = list(set(employee.columns)-set(['Attrition']))
target    = list(['Attrition'])
print(features,'\n',target)
```

```
['YearsInCurrentRole', 'MonthlyIncome', 'YearsSinceLastPromotion', 'JobLevel', 'PercentSa
laryHike', 'YearsAtCompany', 'YearsWithCurrManager', 'StockOptionLevel', 'NumCompaniesWor
ked', 'PerformanceRating', 'Department', 'Age', 'EducationField', 'TotalWorkingYears', 'J
obInvolvement', 'JobRole']
 ['Attrition']
```

In [42]:

```
# Separating out of the feature
x2 = employee.loc[:,features2].values
y2 = employee.loc[:,target].values
```

In [43]:

```
train_x,test_x,train_y,test_y=train_test_split(x2,y2,test_size=0.3,random_state=1)
```

In [44]:

```
clf = RandomForestClassifier(n_estimators = 500, max_depth = 4, max_features = 3, bootst
```

```
rap = True, random_state = 18).fit(train_x, train_y.ravel())
```

In [45]:

```
# Create our predictions
prediction = clf.predict(test_x)
```

In [46]:

```
confusion_matrix(test_y, prediction)
```

Out[46]:

```
array([[361,   3],
       [ 70,   7]])
```

Model Accuracy

In [47]:

```
# calculating the accuracy score
accuracy_score2 = accuracy_score(test_y, prediction)
atd=clf.score(train_x, train_y)
print('Accuracy of Trained Data:',atd)
print('Model Accuracy Score    :',accuracy_score2)
```

```
Accuracy of Trained Data: 0.8610301263362488
Model Accuracy Score    : 0.8344671201814059
```

Feature Importance

In [48]:

```
# Return the feature importances (the higher, the more important the feature).
importances = pd.DataFrame({'features':employee.iloc[:, 1:employee.shape[1]].columns,'im
portance':np.round(clf.feature_importances_,3)}) #Note: The target column is at position
0
importances = importances.sort_values('importance',ascending=False).set_index('features'
)
importances
```

Out[48]:

	importance
features	
Department	0.163
TotalWorkingYears	0.116
YearsInCurrentRole	0.099
NumCompaniesWorked	0.097
JobRole	0.087
MonthlyIncome	0.063
YearsWithCurrManager	0.061
YearsSinceLastPromotion	0.053
Attrition	0.047
JobInvolvement	0.042
PercentSalaryHike	0.041
YearsAtCompany	0.036
JobLevel	0.034
EducationField	0.032

StockOptionLevel	0.027
PerformanceRating	0.004

Above are the important parameters starting from high to low in importance, by which an employee attrition can be determined.

Choosing Best Model

By using different machine learning algorithm we found different accurcy score:

- **Logistic Regression**

```
Accuracy of Trained Data: 0.8464528668610302
Model Accuracy Score    : 0.8435374149659864
```

- **Decision Tree Classifier**

```
Accuracy of Trained Data: 1.0
Model Accuracy Score    : 0.7687074829931972
```

- **Random Forest Classifier**

```
Accuracy of Trained Data: 0.8620019436345967
Model Accuracy Score    : 0.8344671201814059
```

Here we can consider Random Forest Classifier as this model is giving us maximum accuracy with low bias and low variance.

5. Verdict

11 Tips to Improve Employee Attrition Rate

With all that in mind, what can we do to keep high performers and contributors with business? Much employee arrtition is preventable, and small changes in career development opportunities, work-life balance, manager relationships, compensation and overall wellbeing can make a big difference.

1. Hire the right people

- **Some of the blame for poor hires falls on recruiting. Recruiters must be clear about the organization's culture upfront, telling the candidate not what they think the person wants to hear, but how the company actually operates. But a big part of hiring the right person is making sure that recruiting is looking for the right person from the beginning. Less than half of workers believe that job descriptions reflect actual job responsibilities.**
- **One way many organizations have improved their success rate with new hires is by allowing peers in that person's role to make the hiring decisions. Organizations should also invest time into getting to know the candidate by whatever means available. In-person visits to the office and opportunities to see how the person reacts and interacts with potential co-workers is ideal, but can sometimes be accomplished via video, as well. If possible, considering making certain roles remote to increase the pool of available candidates and boost the chances you find the ideal fit.**

2. Keep up with the market rate and offer competitive salaries and total compensation

- **Pay and benefits are key reasons people take jobs and show up for work every day. It's also a top reason why professionals change jobs. It's therefore no surprise that higher pay tops the list of what would convince workers to stay, followed by time off and benefits.**
- **Companies should start by offering an appropriate starting salary that will attract qualified and talented candidates. They should also offer regular raises and monitor what other companies pay for similar roles, especially when it comes to hard-to-fill jobs. Organizations should expect to pay more for those with in-demand skills, and more are offering bonuses that are tied to project completion. Establishing talent**

management processes that identify top performers and correcting pay imbalances by conducting racial and gender pay equity analyses can also limit compensation-related turnover.

3. Train Middle Managers

- People leave their bosses, not their job. Statistics said 92% of employees leave their job due to unapologetic and rude bosses. Middle managers and supervisors should be properly trained to handle their subordinates to reduce attrition.Conducting sessions for middle managers with the human resource management team to develop people skills.

4. Standardize performance reviews

- Another not-so-surprising turnover predictor are unproductive or infrequent performance reviews. The traditional performance review — a static, annual or biannual event consisting of reviewing an Excel spreadsheet with static goals doesn't exactly inspire. In fact, it may do more harm than good. Data shows that employees who felt criticized or unmotivated after a performance review started to look for a new job.
- Making the performance review a collaborative, dynamic and continuous process that works to improve the relationship between an employee and a manager, rather than put up walls between them, is the way to go. For instance, functionality in human capital management (HCM) or human resources management system (HRMS) software reimagines the performance review as a process that aligns the manager and employee on goal setting, offers an opportunity to reflect on the progress and provides rewards in response to high performance. Tying goals to actionable metrics and viewing them through performance management dashboards helps managers easily automatically updates goals in real time.

5. Focus on onboarding

- Onboarding is often a new employee's first introduction to the culture of an organization. It's tough to recover from a bad onboarding experience. Employees who have negative new hire onboarding experiences are twice as likely to explore new opportunities early on in their tenure.
- But small improvements in the process have the ability to leave positive first impressions that last. Indeed, employees are more likely to stay with the company for several years after a good onboarding experience. Better onboarding — and longer onboarding, in particular — leads to faster time to productivity. The best onboarding processes don't park employees in a room for eight hours and call it a day. They pair new employees with mentors and facilitate connections with people in different departments. And they continually check in to see how things are going, providing support and resources along the way.

6. Change Of Departments

- The most important factor of employee attrition is the fact that employees want a change in their career.Data shows us in the very initial stage of onboarding, employees leave their job due to a desire for change in career. Having an option of a change of department in the company itself gives employees lots of freedom. This should reduce some amount of attrition rates.Having a clear and structured program for any employees who want to change departments goes a long way. Human resource management should be involved in this process to facilitate a smooth transition of departments.

7. Analyze previous and current turnover to find issues

- The most concrete way to go about reducing employee attrition rate is to collate and analyze the data related to turnover. This will give insights into the reason for employees departing and can help rectify the issue and save company from losing the best talents.

8. Optimize workforce utilization

- According to data employees have left their job due to burnout. Overutilization can put employees under immense pressure and can contribute to employee attrition. At the same time, underutilization can lead to disengagement and low morale. Thus, optimizing employees' utilization is critical to leverage their skills at maximum potential and retain them.
- Managers need to keep in mind that effective utilization is not just about working too many hours. Productivity must go hand in hand with utilization. They must therefore ascertain that employees' maximum time is booked for strategic/billable work. Spending time on mundane admin tasks or BAU activities will neither put their skillset to the right use nor generate profits for the firm.
- Employers can make adequate use of dashboards to measure and get a comprehensive view of employee utilization levels.

9. Minimize bench time

9. Minimize bench time

- **Once a project gets over and if resources are not scheduled for another project, they will spend bench-time until they are allocated a new project. Extended bench time leads to significant issues such as lesser ROI [ROI = (Net Benefits of training / Costs of Training) x 100] as the resources are not generating any revenue for the organization. It can lead to planned attrition which affects firm's reputation as well as unplanned attrition when employees begin to look for other job opportunities for growth and development.**
- **For effective bench-management and to reduce unplanned attrition, managers can employ an effective resource management tool which will predict resources that will end up on the bench in advance. Project vacancy reports can be used to quickly assign them to billable or strategic work before they land-up on bench. Moreover, advanced planning on pipeline projects will help allocate them better.**

10. Plan training & development programs

- **Providing training and development programs displays the commitment given by the company. A resource manager can help the resources by projecting a career path, thereby giving a purpose and setting direction. Managers can implement an Individual Development Plan or IDP to help employees reach short and long-term career goals and improve current job performance. Training facilitates self-growth and will allow the resources to contribute better. They can take up more responsibility in the team or even be eligible for higher roles.**
- **Managers can track the project's progress and gauge the employee's key strengths and weaknesses based on the way they perform the tasks. Based on this, they can motivate them to learn new skills and practice on the job. When the workforce feels that their goals and objectives are being taken care of, they are likely to stay with the firm for a longer duration.**

11. Identify key performers

- **Every business needs a set of worker bees who are diligent in their work. It is expected of employees to show up promptly on time and get the job rightly done and keep the flow of work going. To effectively grow company, we need to nurture and reward the top performers to keep up the employee morale of those who put a little extra into their work.**

Thank You

In [ ]: