In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```
employee = pd.read_csv("Employee.csv")
```

In [3]:
```
employee
```

Out[3]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	E
0	41	Yes	Travel_Rarely	1102	Sales	1	2	
1	49	No	Travel_Frequently	279	Research & Development	8	1	
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	
4	27	No	Travel_Rarely	591	Research & Development	2	1	
...	...	...	...	...	...	...	...	...
1465	36	No	Travel_Frequently	884	Research & Development	23	2	
1466	39	No	Travel_Rarely	613	Research & Development	6	1	
1467	27	No	Travel_Rarely	155	Research & Development	4	3	
1468	49	No	Travel_Frequently	1023	Sales	2	3	
1469	34	No	Travel_Rarely	628	Research & Development	8	3	

1470 rows × 35 columns

In [4]:
```
pd.set_option('display.max_columns', None)
employee.head(5)
```

Out[4]:

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

In [5]:
```
employee.tail(5)
```

Out[5]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	E
1465	36	No	Travel_Frequently	884	Research & Development	23	2	
1466	39	No	Travel_Rarely	613	Research & Development	6	1	
1467	27	No	Travel_Rarely	155	Research & Development	4	3	
1468	49	No	Travel_Frequently	1023	Sales	2	3	
1469	34	No	Travel_Rarely	628	Research & Development	8	3	

In [6]:
```
employee.describe().T
```

Out[6]:

	count	mean	std	min	25%	50%	75%
Age	1470.0	36.923810	9.135373	18.0	30.00	36.0	43.00
DailyRate	1470.0	802.485714	403.509100	102.0	465.00	802.0	1157.00
DistanceFromHome	1470.0	9.192517	8.106864	1.0	2.00	7.0	14.00
Education	1470.0	2.912925	1.024165	1.0	2.00	3.0	4.00
EmployeeCount	1470.0	1.000000	0.000000	1.0	1.00	1.0	1.00
EmployeeNumber	1470.0	1024.865306	602.024335	1.0	491.25	1020.5	1555.75
EnvironmentSatisfaction	1470.0	2.721769	1.093082	1.0	2.00	3.0	4.00
HourlyRate	1470.0	65.891156	20.329428	30.0	48.00	66.0	83.75
JobInvolvement	1470.0	2.729932	0.711561	1.0	2.00	3.0	3.00
JobLevel	1470.0	2.063946	1.106940	1.0	1.00	2.0	3.00
JobSatisfaction	1470.0	2.728571	1.102846	1.0	2.00	3.0	4.00
MonthlyIncome	1470.0	6502.931293	4707.956783	1009.0	2911.00	4919.0	8379.00
MonthlyRate	1470.0	14313.103401	7117.786044	2094.0	8047.00	14235.5	20461.50
NumCompaniesWorked	1470.0	2.693197	2.498009	0.0	1.00	2.0	4.00
PercentSalaryHike	1470.0	15.209524	3.659938	11.0	12.00	14.0	18.00
PerformanceRating	1470.0	3.153741	0.360824	3.0	3.00	3.0	3.00
RelationshipSatisfaction	1470.0	2.712245	1.081209	1.0	2.00	3.0	4.00
StandardHours	1470.0	80.000000	0.000000	80.0	80.00	80.0	80.00
StockOptionLevel	1470.0	0.793878	0.852077	0.0	0.00	1.0	1.00
TotalWorkingYears	1470.0	11.279592	7.780782	0.0	6.00	10.0	15.00
TrainingTimesLastYear	1470.0	2.799320	1.289271	0.0	2.00	3.0	3.00
WorkLifeBalance	1470.0	2.761224	0.706476	1.0	2.00	3.0	3.00
YearsAtCompany	1470.0	7.008163	6.126525	0.0	3.00	5.0	9.00
YearsInCurrentRole	1470.0	4.229252	3.623137	0.0	2.00	3.0	7.00
YearsSinceLastPromotion	1470.0	2.187755	3.222430	0.0	0.00	1.0	3.00
YearsWithCurrManager	1470.0	4.123129	3.568136	0.0	2.00	3.0	7.00

In [7]:
```
employee.info
```

```
2                    2          2          Other             1
3                    3          4  Life Sciences             1
4                    2          1        Medical             1
...                ...        ...            ...           ...
1465                23          2        Medical             1
1466                 6          1        Medical             1
1467                 4          3  Life Sciences             1
1468                 2          3        Medical             1
1469                 8          3        Medical             1

      EmployeeNumber  EnvironmentSatisfaction  Gender  HourlyRate  \
0                  1                        2  Female          94
1                  2                        3    Male          61
2                  4                        4    Male          92
3                  5                        4  Female          56
4                  7                        1    Male          40
...              ...                      ...     ...         ...
1465            2061                        3    Male          41
1466            2062                        4    Male          42
1467            2064                        2    Male          87
```

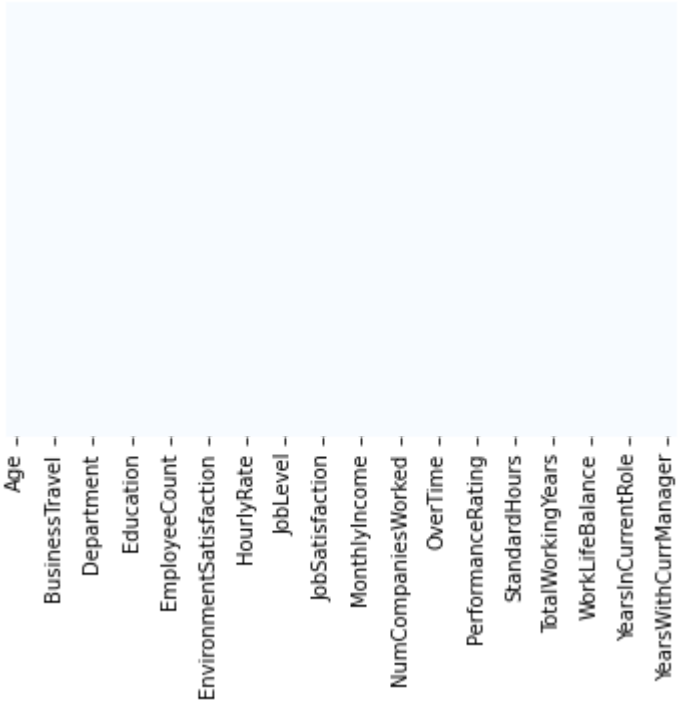
In [8]:
```
employee['Attrition'] = employee['Attrition'].apply(lambda x:1 if x == 'Yes' e
employee['OverTime'] = employee['OverTime'].apply(lambda x:1 if x == 'Yes' els
employee['Over18'] = employee['Over18'].apply(lambda x:1 if x == 'Yes' else 0)
```

In [9]:
```
employee['Attrition']
```

Out[9]:
```
0       1
1       0
2       1
3       0
4       0
       ..
1465    0
1466    0
1467    0
1468    0
1469    0
Name: Attrition, Length: 1470, dtype: int64
```

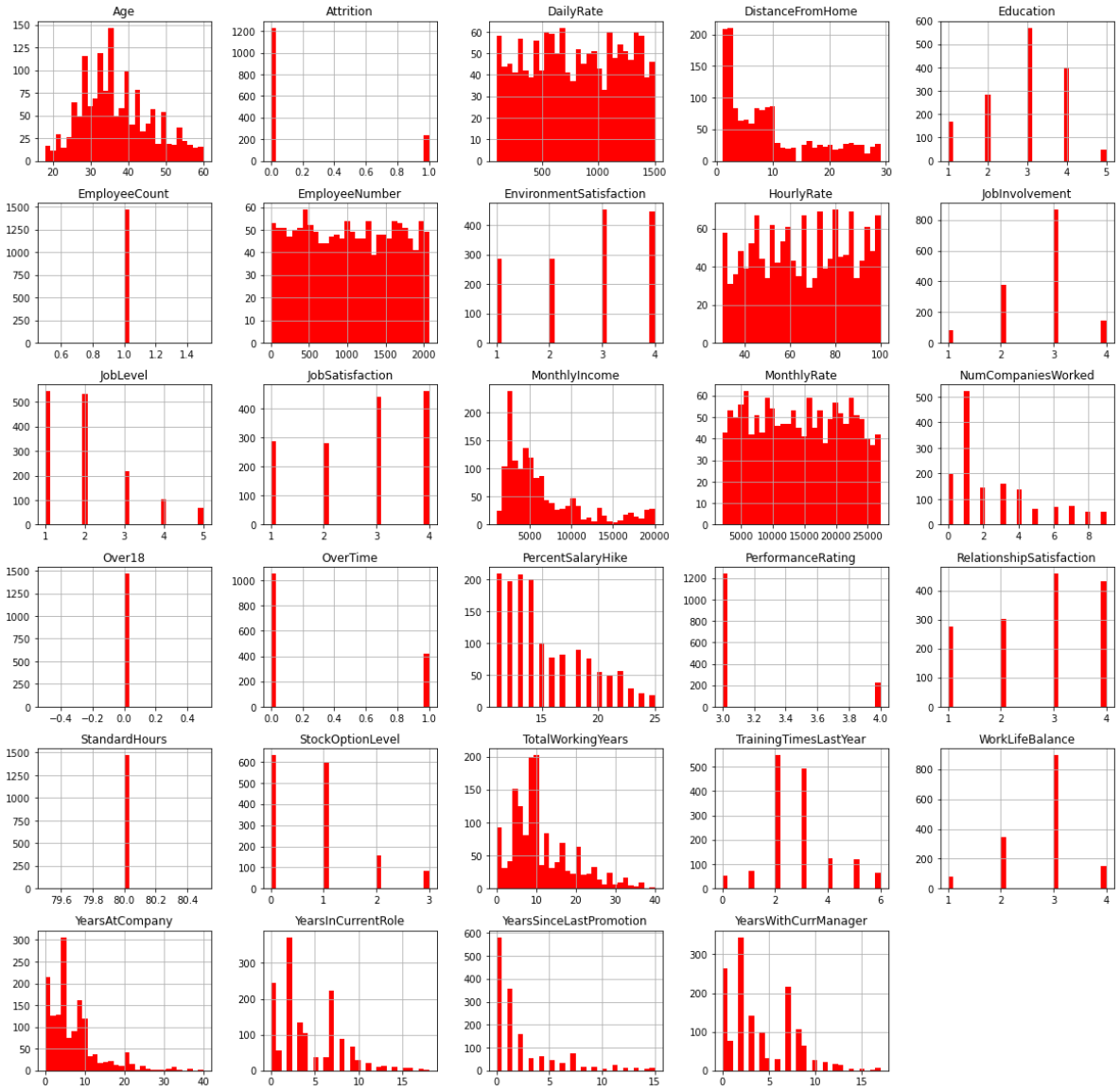
In [10]:
```
sns.heatmap(employee.isnull(), yticklabels = False, cbar = False, cmap = 'Blue
```

Out[10]: <AxesSubplot:>



In [11]:
```
employee.hist(bins = 30, figsize = (20,20), color = 'r')
```

Out[11]:
```
array([[<AxesSubplot:title={'center':'Age'}>,
        <AxesSubplot:title={'center':'Attrition'}>,
        <AxesSubplot:title={'center':'DailyRate'}>,
        <AxesSubplot:title={'center':'DistanceFromHome'}>,
        <AxesSubplot:title={'center':'Education'}>],
       [<AxesSubplot:title={'center':'EmployeeCount'}>,
        <AxesSubplot:title={'center':'EmployeeNumber'}>,
        <AxesSubplot:title={'center':'EnvironmentSatisfaction'}>,
        <AxesSubplot:title={'center':'HourlyRate'}>,
        <AxesSubplot:title={'center':'JobInvolvement'}>],
       [<AxesSubplot:title={'center':'JobLevel'}>,
        <AxesSubplot:title={'center':'JobSatisfaction'}>,
        <AxesSubplot:title={'center':'MonthlyIncome'}>,
        <AxesSubplot:title={'center':'MonthlyRate'}>,
        <AxesSubplot:title={'center':'NumCompaniesWorked'}>],
       [<AxesSubplot:title={'center':'Over18'}>,
        <AxesSubplot:title={'center':'OverTime'}>,
        <AxesSubplot:title={'center':'PercentSalaryHike'}>,
        <AxesSubplot:title={'center':'PerformanceRating'}>,
        <AxesSubplot:title={'center':'RelationshipSatisfaction'}>],
       [<AxesSubplot:title={'center':'StandardHours'}>,
        <AxesSubplot:title={'center':'StockOptionLevel'}>,
        <AxesSubplot:title={'center':'TotalWorkingYears'}>,
        <AxesSubplot:title={'center':'TrainingTimesLastYear'}>,
        <AxesSubplot:title={'center':'WorkLifeBalance'}>],
       [<AxesSubplot:title={'center':'YearsAtCompany'}>,
        <AxesSubplot:title={'center':'YearsInCurrentRole'}>,
        <AxesSubplot:title={'center':'YearsSinceLastPromotion'}>,
        <AxesSubplot:title={'center':'YearsWithCurrManager'}>,
        <AxesSubplot:>]], dtype=object)
```



In [12]:
```
employee.drop(['EmployeeCount', 'StandardHours', 'Over18', 'EmployeeNumber'],
```

In [13]:
```
left = employee[employee['Attrition'] == 1]
stayed = employee[employee['Attrition'] == 0]
```

In [14]:
```
print("Number of employee left", len(left))
print("Number of employee who stayed", len(stayed))
print("Number of emmployee who did not leave company", len(stayed))
print("Percentage of employees who did not leave the company", 1.*len(stayed)/
```

```
Number of employee left 237
Number of employee who stayed 1233
Number of emmployee who did not leave company 1233
Percentage of employees who did not leave the company 83.87755102040816 %
```

In [15]:
```
left.describe()
```

Out[15]:

	Age	Attrition	DailyRate	DistanceFromHome	Education	EnvironmentSatisfactic
count	237.000000	237.0	237.000000	237.000000	237.000000	237.0000(
mean	33.607595	1.0	750.362869	10.632911	2.839662	2.46413
std	9.689350	0.0	401.899519	8.452525	1.008244	1.16979
min	18.000000	1.0	103.000000	1.000000	1.000000	1.0000(
25%	28.000000	1.0	408.000000	3.000000	2.000000	1.0000(
50%	32.000000	1.0	699.000000	9.000000	3.000000	3.0000(
75%	39.000000	1.0	1092.000000	17.000000	4.000000	4.0000(
max	58.000000	1.0	1496.000000	29.000000	5.000000	4.0000(

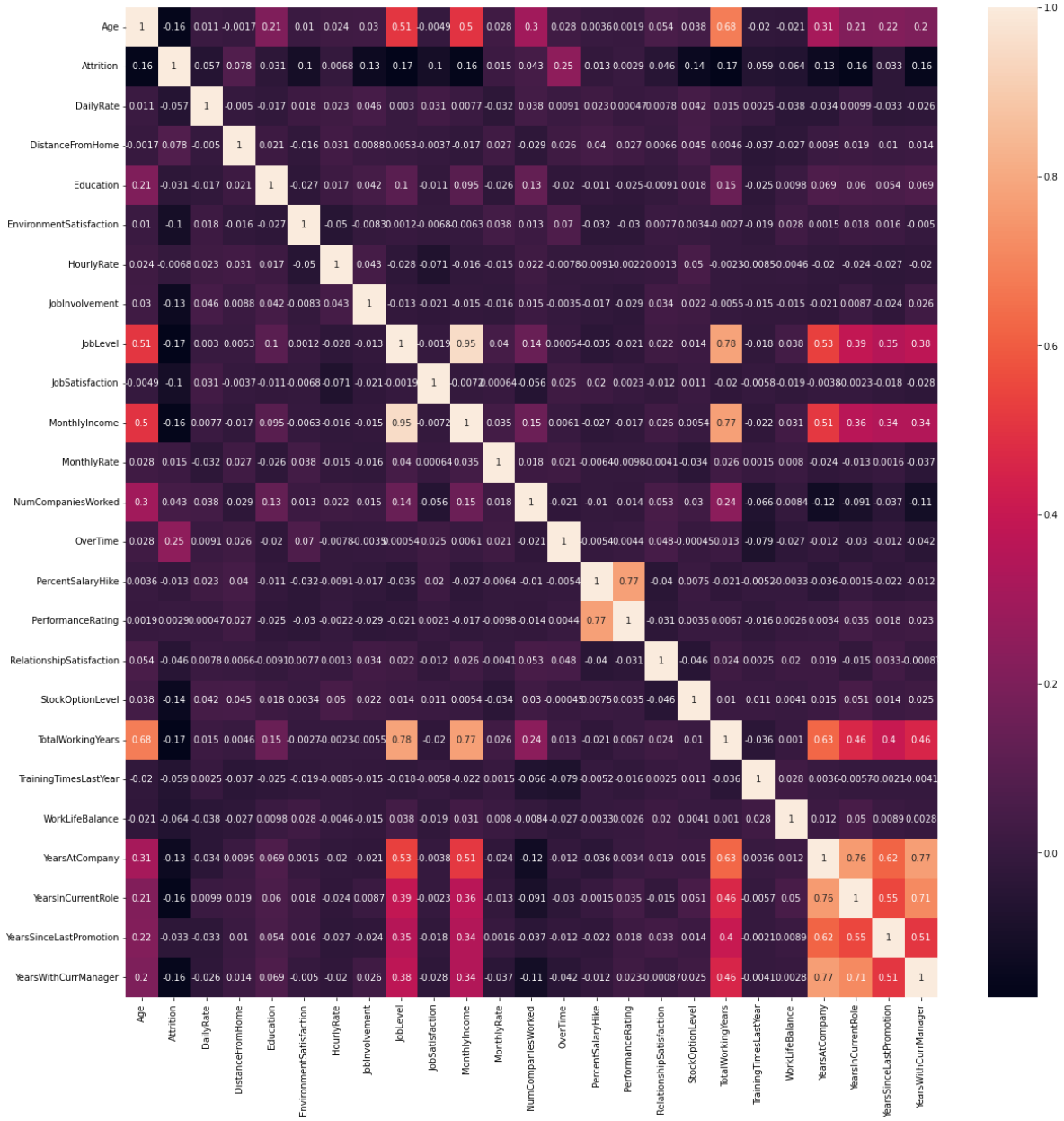
In [16]:
```
stayed.describe()
```

Out[16]:

	Age	Attrition	DailyRate	DistanceFromHome	Education	EnvironmentSatisfac
count	1233.000000	1233.0	1233.000000	1233.000000	1233.000000	1233.00(
mean	37.561233	0.0	812.504461	8.915653	2.927007	2.77
std	8.888360	0.0	403.208379	8.012633	1.027002	1.07
min	18.000000	0.0	102.000000	1.000000	1.000000	1.00(
25%	31.000000	0.0	477.000000	2.000000	2.000000	2.00(
50%	36.000000	0.0	817.000000	7.000000	3.000000	3.00(
75%	43.000000	0.0	1176.000000	13.000000	4.000000	4.00(
max	60.000000	0.0	1499.000000	29.000000	5.000000	4.00(

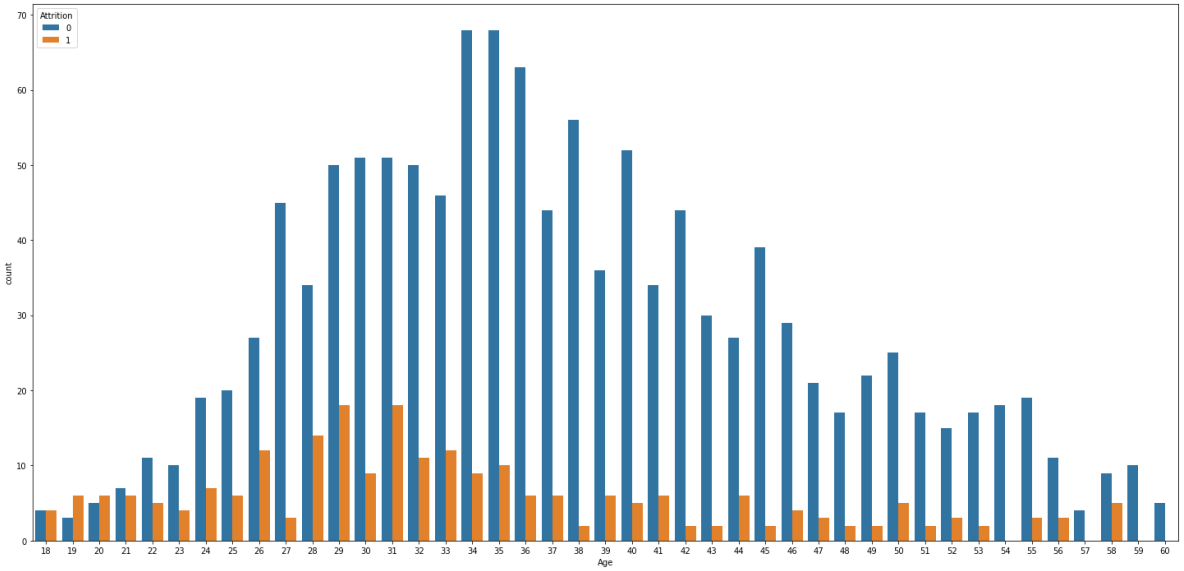
In [17]:
```
correlations = employee.corr()
f, ax = plt.subplots(figsize = (20,20))
sns.heatmap(correlations, annot = True)
```

Out[17]: <AxesSubplot:>



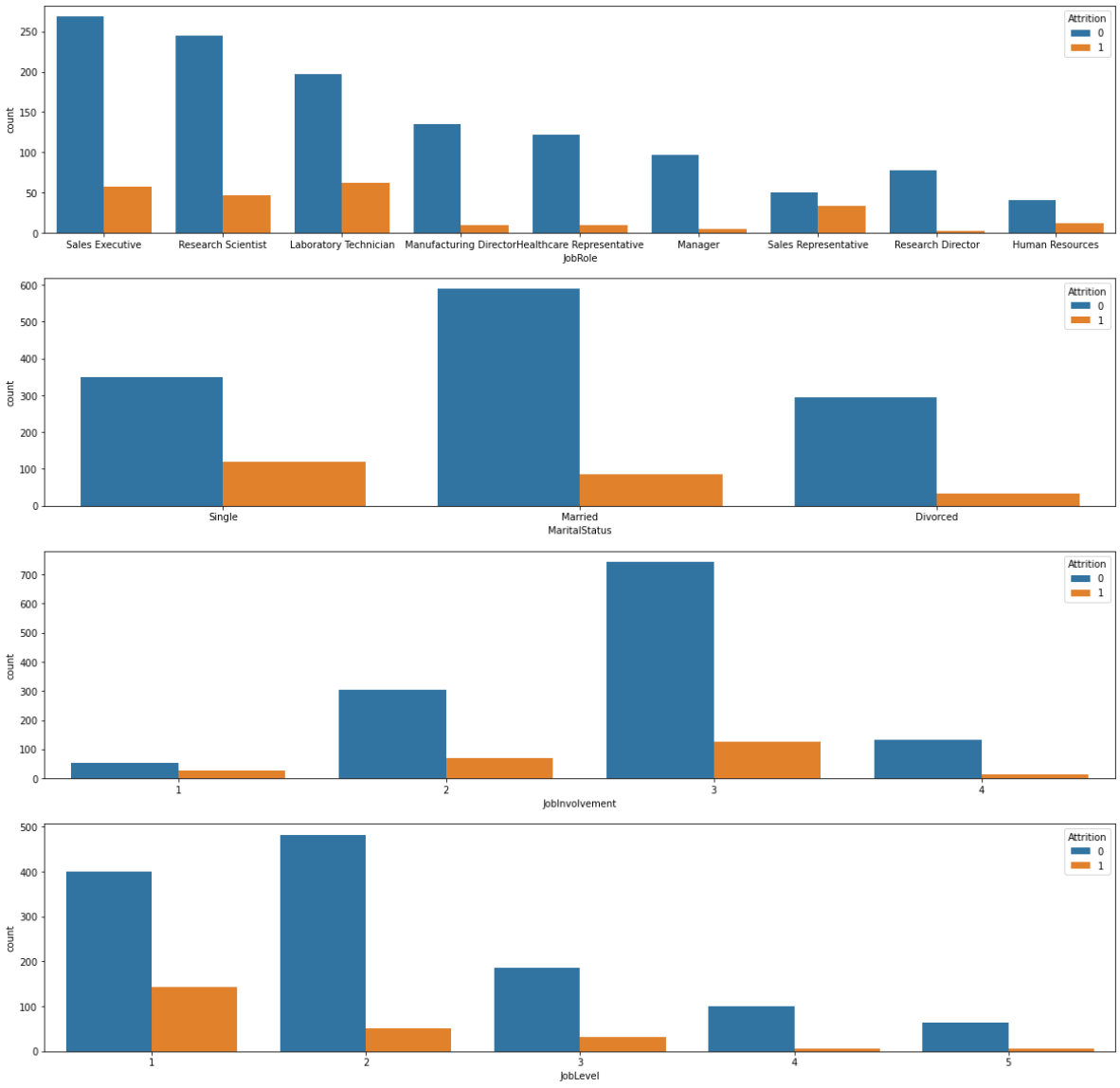
In [18]:
```
plt.figure(figsize = [25,12])
sns.countplot(x = 'Age', hue = 'Attrition', data = employee)
```

Out[18]:
```
<AxesSubplot:xlabel='Age', ylabel='count'>
```



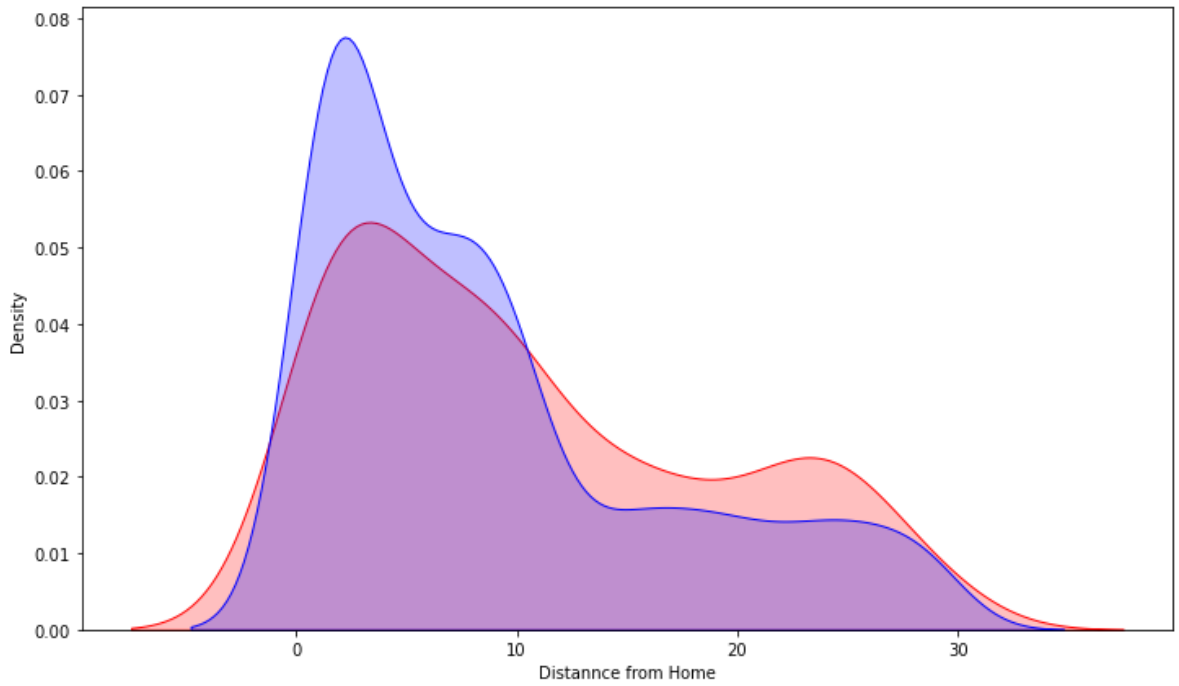
In [19]:
```
plt.figure(figsize = [20,20])
plt.subplot(411)
sns.countplot(x = 'JobRole', hue = "Attrition", data = employee)
plt.subplot(412)
sns.countplot(x = 'MaritalStatus', hue = 'Attrition', data = employee)
plt.subplot(413)
sns.countplot(x = 'JobInvolvement', hue = 'Attrition', data = employee)
plt.subplot(414)
sns.countplot(x = 'JobLevel', hue = 'Attrition', data = employee)
```

Out[19]: <AxesSubplot:xlabel='JobLevel', ylabel='count'>



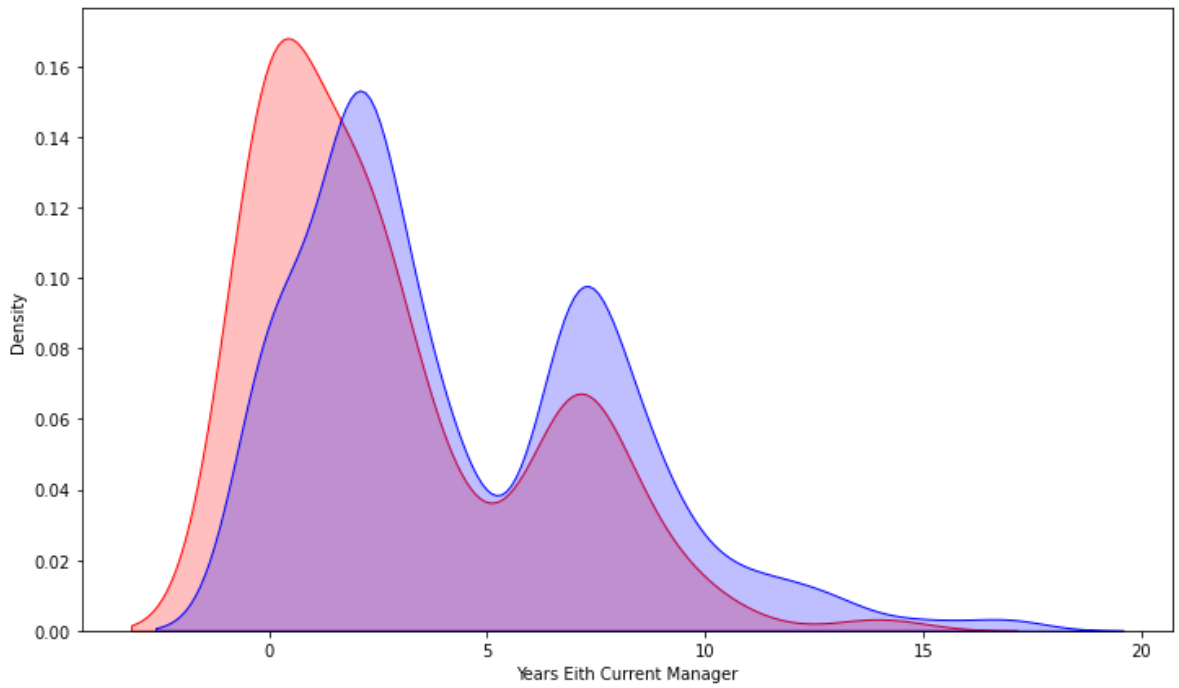
In [20]:
```
plt.figure(figsize = (12,7))
sns.kdeplot(left['DistanceFromHome'], label = "Employee who Left", shade = Tru
sns.kdeplot(stayed['DistanceFromHome'], label = "Employee who Left", shade = T
plt.xlabel("Distannce from Home")
```

Out[20]:
```
Text(0.5, 0, 'Distannce from Home')
```



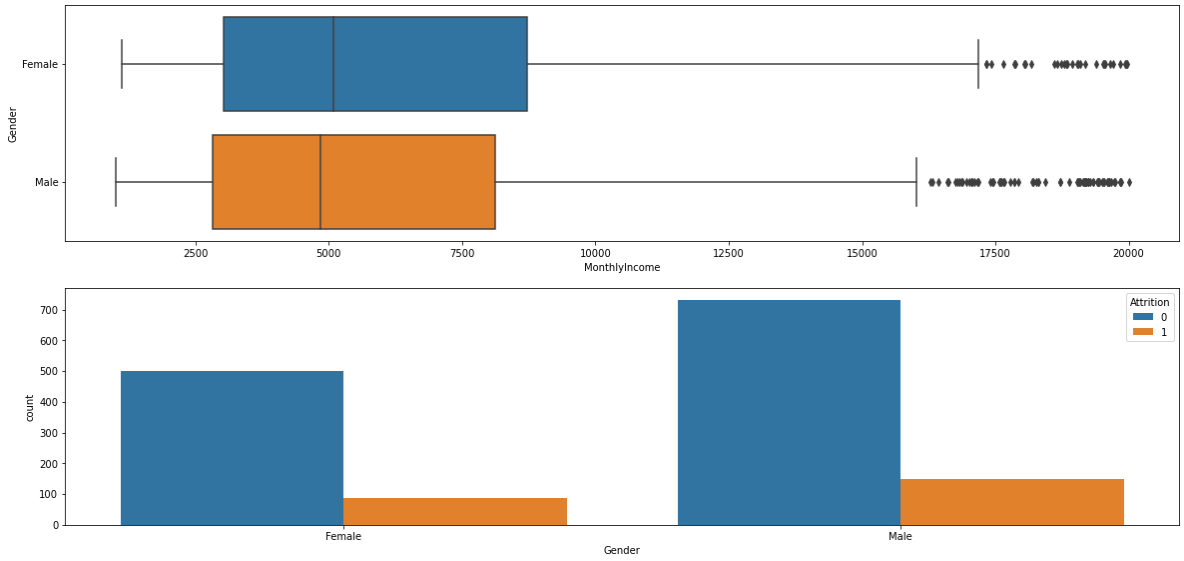
In [21]:
```
plt.figure(figsize = (12,7))
sns.kdeplot(left["YearsWithCurrManager"], label = 'Employee who Left', shade =
sns.kdeplot(stayed["YearsWithCurrManager"], label = 'Employee who stayed', sha
plt.xlabel('Years Eith Current Manager')
```

Out[21]:
```
Text(0.5, 0, 'Years Eith Current Manager')
```



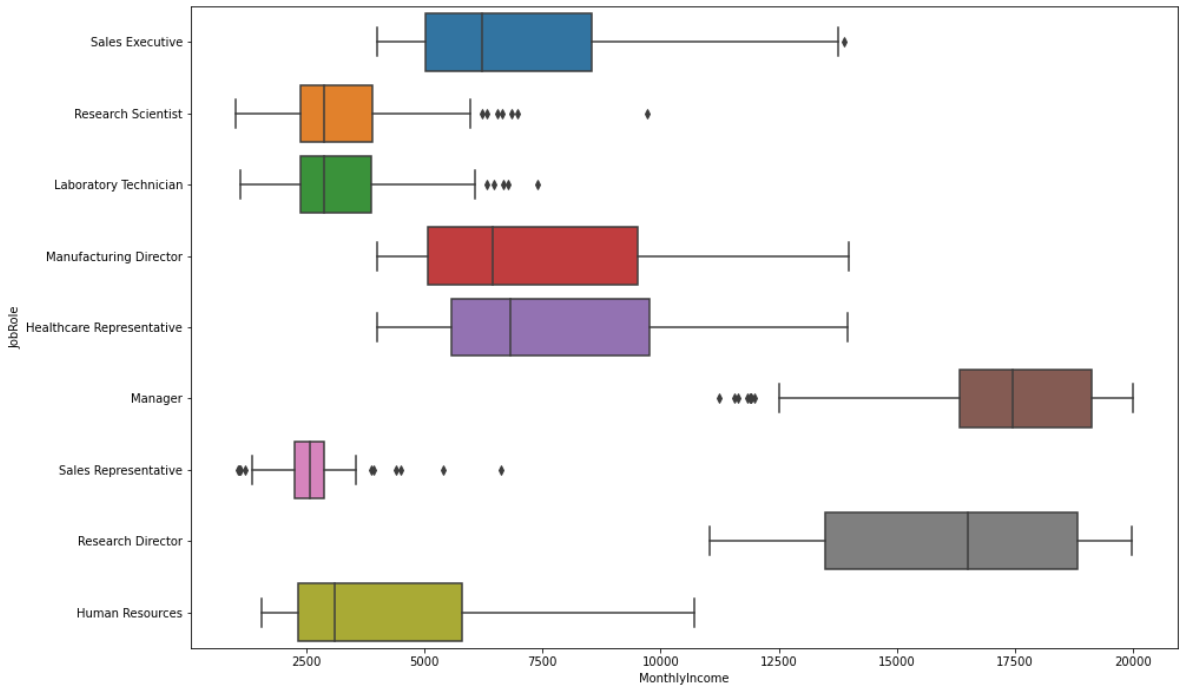
In [22]:
```
plt.figure(figsize = [20,20])
plt.subplot(411)
sns.boxplot(x = 'MonthlyIncome', y = "Gender", data = employee)
plt.subplot(412)
sns.countplot(x = 'Gender', hue = 'Attrition', data = employee)
```

Out[22]: <AxesSubplot:xlabel='Gender', ylabel='count'>

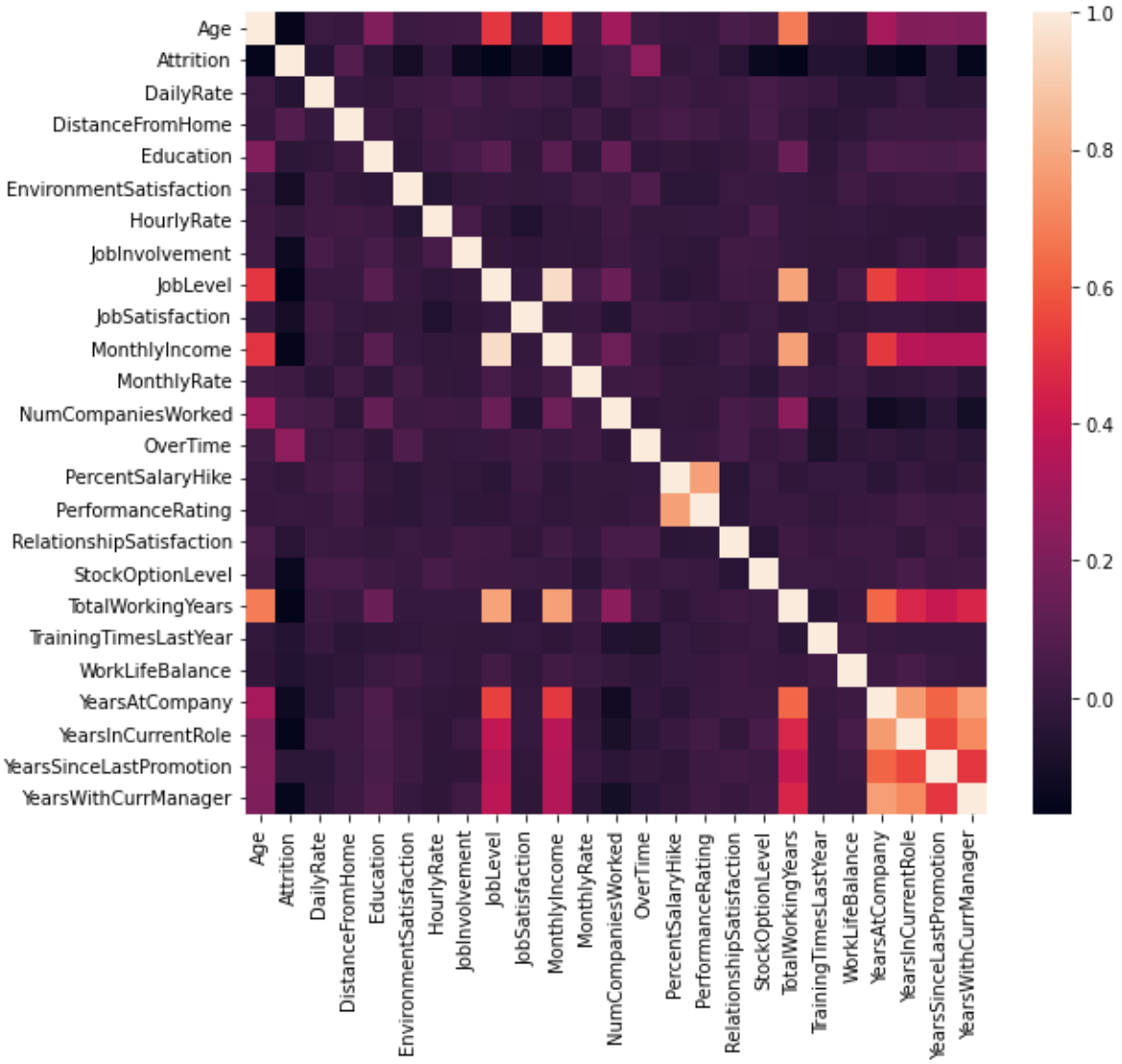


In [23]:
```
plt.figure(figsize = (15,10))
sns.boxplot(x = 'MonthlyIncome', y = 'JobRole', data = employee)
```

Out[23]: <AxesSubplot:xlabel='MonthlyIncome', ylabel='JobRole'>



In [24]:
```
corr = employee.corr()
fig,ax = plt.subplots(1,1, figsize = (9,8))
ax = sns.heatmap(corr,
                 xticklabels = corr.columns.values,
                 yticklabels = corr.columns.values)
```



In [26]:
```
to_drop_cols = [
    'PerformanceRating',
    'YearsInCurrentRole',
    'YearsWithCurrManager',
]
employee.drop(to_drop_cols, axis = 1, inplace = True)
```

In [27]:
```
employee.head()
```

Out[27]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	Educ
0	41	1	Travel_Rarely	1102	Sales	1	2	Lif
1	49	0	Travel_Frequently	279	Research & Development	8	1	Lif
2	37	1	Travel_Rarely	1373	Research & Development	2	2	
3	33	0	Travel_Frequently	1392	Research & Development	3	4	Lif
4	27	0	Travel_Rarely	591	Research & Development	2	1	

In [28]:
```
X_cat = employee.select_dtypes(exclude = ["number"])
X_cat
```

Out[28]:

	BusinessTravel	Department	EducationField	Gender	JobRole	MaritalStatus
0	Travel_Rarely	Sales	Life Sciences	Female	Sales Executive	Single
1	Travel_Frequently	Research & Development	Life Sciences	Male	Research Scientist	Married
2	Travel_Rarely	Research & Development	Other	Male	Laboratory Technician	Single
3	Travel_Frequently	Research & Development	Life Sciences	Female	Research Scientist	Married
4	Travel_Rarely	Research & Development	Medical	Male	Laboratory Technician	Married
...	...	...	...	...	...	...
1465	Travel_Frequently	Research & Development	Medical	Male	Laboratory Technician	Married
1466	Travel_Rarely	Research & Development	Medical	Male	Healthcare Representative	Married
1467	Travel_Rarely	Research & Development	Life Sciences	Male	Manufacturing Director	Married
1468	Travel_Frequently	Sales	Medical	Male	Sales Executive	Married
1469	Travel_Rarely	Research & Development	Medical	Male	Laboratory Technician	Married

1470 rows × 6 columns

In [29]:
```
from sklearn.preprocessing import OneHotEncoder
ohc = OneHotEncoder()
X_cat = ohc.fit_transform(X_cat).toarray()
```

In [30]:
```
X_cat = pd.DataFrame(X_cat)
X_cat
```

Out[30]:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	1
0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
1	0.0	1.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.
2	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.
3	0.0	1.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
4	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
1465	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.
1466	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.
1467	0.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.
1468	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.
1469	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.

1470 rows × 26 columns

In [31]:
```
X_numeric = employee.select_dtypes(include = "number").drop(columns = ["Attrit
X_numeric
```

Out[31]:

	Age	DailyRate	DistanceFromHome	Education	EnvironmentSatisfaction	HourlyRate	JobIr
0	41	1102	1	2	2	94	
1	49	279	8	1	3	61	
2	37	1373	2	2	4	92	
3	33	1392	3	4	4	56	
4	27	591	2	1	1	40	
...	...	...	...	...	...	...	
1465	36	884	23	2	3	41	
1466	39	613	6	1	4	42	
1467	27	155	4	3	2	87	
1468	49	1023	2	3	4	63	
1469	34	628	8	3	2	82	

1470 rows × 21 columns

In [32]:
```
X_complete  = pd.concat([X_cat, X_numeric], axis = 1)
X_complete
```

Out[32]:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	1
0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
1	0.0	1.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.
2	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.
3	0.0	1.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.
4	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
1465	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.
1466	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.
1467	0.0	0.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.
1468	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.
1469	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.

1470 rows × 47 columns

In [33]:
```
X_complete_copy = X_complete.copy()
X_complete_copy.columns = X_complete_copy.columns.astype(str)
```

In [34]:
```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X = scaler.fit_transform(X_complete_copy)
```

In [35]:
```
X
```

Out[35]:
```
array([[0.        , 0.        , 1.        , ..., 0.        , 0.15      ,
        0.        ],
       [0.        , 1.        , 0.        , ..., 0.66666667, 0.25      ,
        0.06666667],
       [0.        , 0.        , 1.        , ..., 0.66666667, 0.        ,
        0.        ],
       ...,
       [0.        , 0.        , 1.        , ..., 0.66666667, 0.15      ,
        0.        ],
       [0.        , 1.        , 0.        , ..., 0.33333333, 0.225     ,
        0.        ],
       [0.        , 0.        , 1.        , ..., 1.        , 0.1       ,
        0.06666667]])
```

In [36]:
```
y = employee["Attrition"]
```

In [37]:
```
X.dtype
```

Out[37]:
```
dtype('float64')
```

In [47]:
```
import numpy as np
def sigmoid(x):
    return 1/(1+np.exp(-x))
class LogisticRegression():
    def __init__(self, lr = 0.01, n_iters = 1000):
        self.lr = lr
        self.n_iters = n_iters
        self.weights = None
        self.bias = Nonefit
    def fit(self, X, y):
        n_samples, n_features = X.shape
        self.weights = np.zeros(n_features)
        self.bias = 0

        for _ in range(self.n_iters):
            linear_pred = np.dit(X, self.weights) + self.bias
            predictions = sigmoid(linear_pred)

            dw = (1/n_samples) * np.dot(X.T, (predictions - y))
            db = (1/n_samples) * np.sum(predictions-y)
            self.weights = self.weights - self.lr * dw
            self.bias = self.bias - self.lr*db

    def predict(self, X):
        linear_pred = np.dot(X, self.weights) + self.bias
        y_pred = sigmoid(linear_pred)
        class_pred = [0 if y<=0.5 else 1 for y in y_pred]
        return class_pred
```

In [ ]: