In [99]:

```
# Employee Attrition using Machine Learning
```

In [78]:

```
# Importing Libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
from sklearn import linear_model
from sklearn.metrics import confusion_matrix
from sklearn.ensemble import RandomForestClassifier
```

In [101]:

```
df=pd.read_csv("C:/Users/ISHITA GUPTA/Downloads//employee.csv")
```

In [100]:

```
df.head()
```

Out[100]:

	DailyRate	DistanceFromHome	Education	EnvironmentSatisfaction	Gender	HourlyRate	Job
0	1102	1	2	2	0	94	
1	279	8	1	3	1	61	
2	1373	2	2	4	1	92	
3	1392	3	4	4	0	56	
4	591	2	1	1	1	40	

5 rows × 48 columns

In [102]:

```
df.tail()
```

Out[102]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education
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

5 rows × 35 columns

In [103]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 35 columns):
Age                         1470 non-null int64
Attrition                   1470 non-null object
BusinessTravel              1470 non-null object
DailyRate                   1470 non-null int64
Department                  1470 non-null object
DistanceFromHome            1470 non-null int64
Education                   1470 non-null int64
EducationField              1470 non-null object
EmployeeCount               1470 non-null int64
EmployeeNumber              1470 non-null int64
EnvironmentSatisfaction     1470 non-null int64
Gender                      1470 non-null object
HourlyRate                  1470 non-null int64
JobInvolvement              1470 non-null int64
JobLevel                    1470 non-null int64
JobRole                     1470 non-null object
JobSatisfaction             1470 non-null int64
MaritalStatus               1470 non-null object
MonthlyIncome               1470 non-null int64
MonthlyRate                 1470 non-null int64
NumCompaniesWorked          1470 non-null int64
Over18                      1470 non-null object
OverTime                    1470 non-null object
PercentSalaryHike           1470 non-null int64
PerformanceRating           1470 non-null int64
RelationshipSatisfaction    1470 non-null int64
StandardHours               1470 non-null int64
StockOptionLevel            1470 non-null int64
TotalWorkingYears           1470 non-null int64
TrainingTimesLastYear       1470 non-null int64
WorkLifeBalance             1470 non-null int64
YearsAtCompany              1470 non-null int64
YearsInCurrentRole          1470 non-null int64
YearsSinceLastPromotion     1470 non-null int64
YearsWithCurrManager        1470 non-null int64
dtypes: int64(26), object(9)
memory usage: 402.0+ KB
```

In [3]:

```
df.shape
```

Out[3]:

```
(1470, 35)
```

In [4]:

```
df.describe()
```

Out[4]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeN
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	1470.0
mean	36.923810	802.485714	9.192517	2.912925	1.0	1024.8
std	9.135373	403.509100	8.106864	1.024165	0.0	602.0
min	18.000000	102.000000	1.000000	1.000000	1.0	1.0
25%	30.000000	465.000000	2.000000	2.000000	1.0	491.2
50%	36.000000	802.000000	7.000000	3.000000	1.0	1020.5
75%	43.000000	1157.000000	14.000000	4.000000	1.0	1555.7
max	60.000000	1499.000000	29.000000	5.000000	1.0	2068.0

8 rows × 26 columns

In [5]:

```
## To check number of null values
df.isnull().sum()
```

```
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
```

In [8]:

```
df['Department'].unique()
```

Out[8]:

```
array(['Sales', 'Research & Development', 'Human Resources'], dtype=object)
```

In [6]:

```
attrition_count = pd.DataFrame(df['Attrition'].value_counts())
attrition_count
```

Out[6]:

	Attrition
No	1233
Yes	237

In [7]:

```
## Dropping irrelevant data
df.drop(['EmployeeCount' , 'EmployeeNumber'] , axis = 1)
```

Out[7]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationFielc
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Othe
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medica
5	32	No	Travel_Frequently	1005	Research & Development	2	2	Life Sciences
					Research &			

In [12]:

```
df.groupby('Department').sum()
```

Out[12]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumb
Department						
Human Resources	2382	47347	548	187	63	759
Research & Development	35598	775384	8788	2786	961	9672
Sales	16298	356923	4177	1309	446	4633

3 rows × 26 columns

In [15]:

```
df.corr()
```

Out[15]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount
Age	1.000000	0.010661	-0.001686	0.208034	NaN
DailyRate	0.010661	1.000000	-0.004985	-0.016806	NaN
DistanceFromHome	-0.001686	-0.004985	1.000000	0.021042	NaN
Education	0.208034	-0.016806	0.021042	1.000000	NaN
EmployeeCount	NaN	NaN	NaN	NaN	NaN
EmployeeNumber	-0.010145	-0.050990	0.032916	0.042070	NaN
EnvironmentSatisfaction	0.010146	0.018355	-0.016075	-0.027128	NaN
HourlyRate	0.024287	0.023381	0.031131	0.016775	NaN
JobInvolvement	0.029820	0.046135	0.008783	0.042438	NaN
JobLevel	0.509604	0.002966	0.005303	0.101589	NaN
JobSatisfaction	-0.004892	0.030571	-0.003669	-0.011296	NaN
MonthlyIncome	0.497855	0.007707	-0.017014	0.094961	NaN
MonthlyRate	0.028051	-0.032182	0.027473	-0.026084	NaN
NumCompaniesWorked	0.299635	0.038153	-0.029251	0.126317	NaN
PercentSalaryHike	0.003634	0.022704	0.040235	-0.011111	NaN
PerformanceRating	0.001904	0.000473	0.027110	-0.024539	NaN
RelationshipSatisfaction	0.053535	0.007846	0.006557	-0.009118	NaN
StandardHours	NaN	NaN	NaN	NaN	NaN
StockOptionLevel	0.037510	0.042143	0.044872	0.018422	NaN
TotalWorkingYears	0.680381	0.014515	0.004628	0.148280	NaN
TrainingTimesLastYear	-0.019621	0.002453	-0.036942	-0.025100	NaN
WorkLifeBalance	-0.021490	-0.037848	-0.026556	0.009819	NaN
YearsAtCompany	0.311309	-0.034055	0.009508	0.069114	NaN
YearsInCurrentRole	0.212901	0.009932	0.018845	0.060236	NaN
YearsSinceLastPromotion	0.216513	-0.033229	0.010029	0.054254	NaN
YearsWithCurrManager	0.202089	-0.026363	0.014406	0.069065	NaN

26 rows × 26 columns

In [ ]:

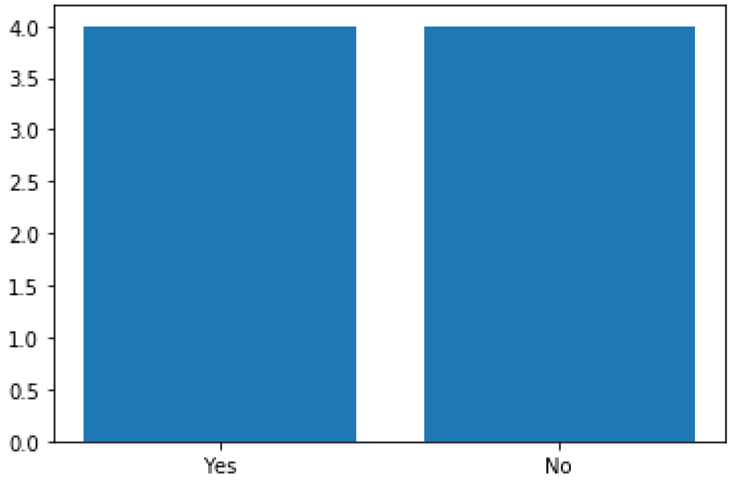
```
# Plotting Data
```

In [16]:

```
plt.bar(x=df['Attrition'],height=df['JobSatisfaction'])
```

Out[16]:

```
<BarContainer object of 1470 artists>
```

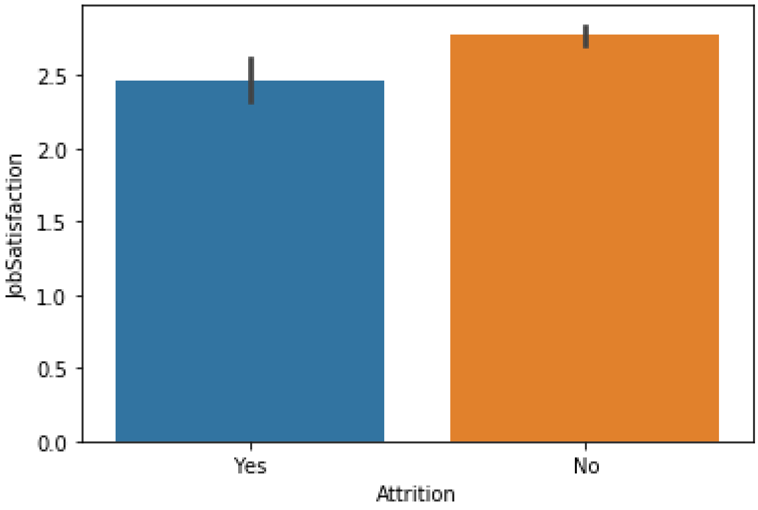


In [17]:

```
sns.barplot(x='Attrition',y='JobSatisfaction',data=df)
```

Out[17]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x26f8c78c8d0>
```

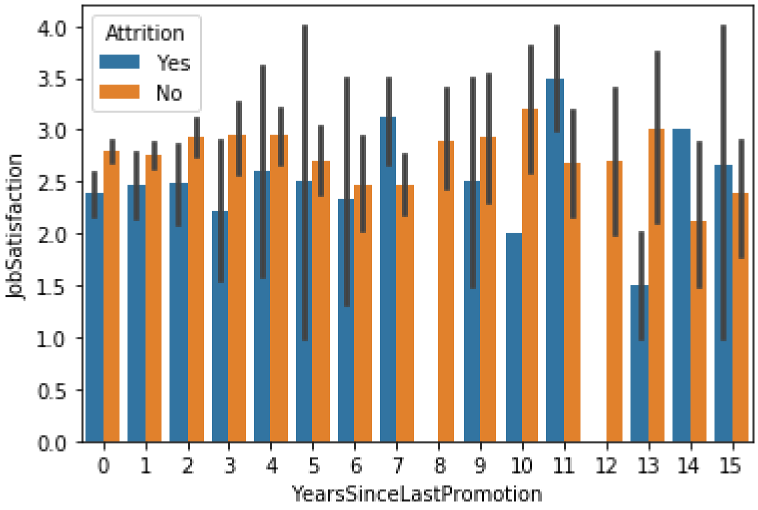


In [22]:

```
sns.barplot(x='YearsSinceLastPromotion',y='JobSatisfaction',data=df,hue='Attrition')
```

Out[22]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x26fca838eb8>
```



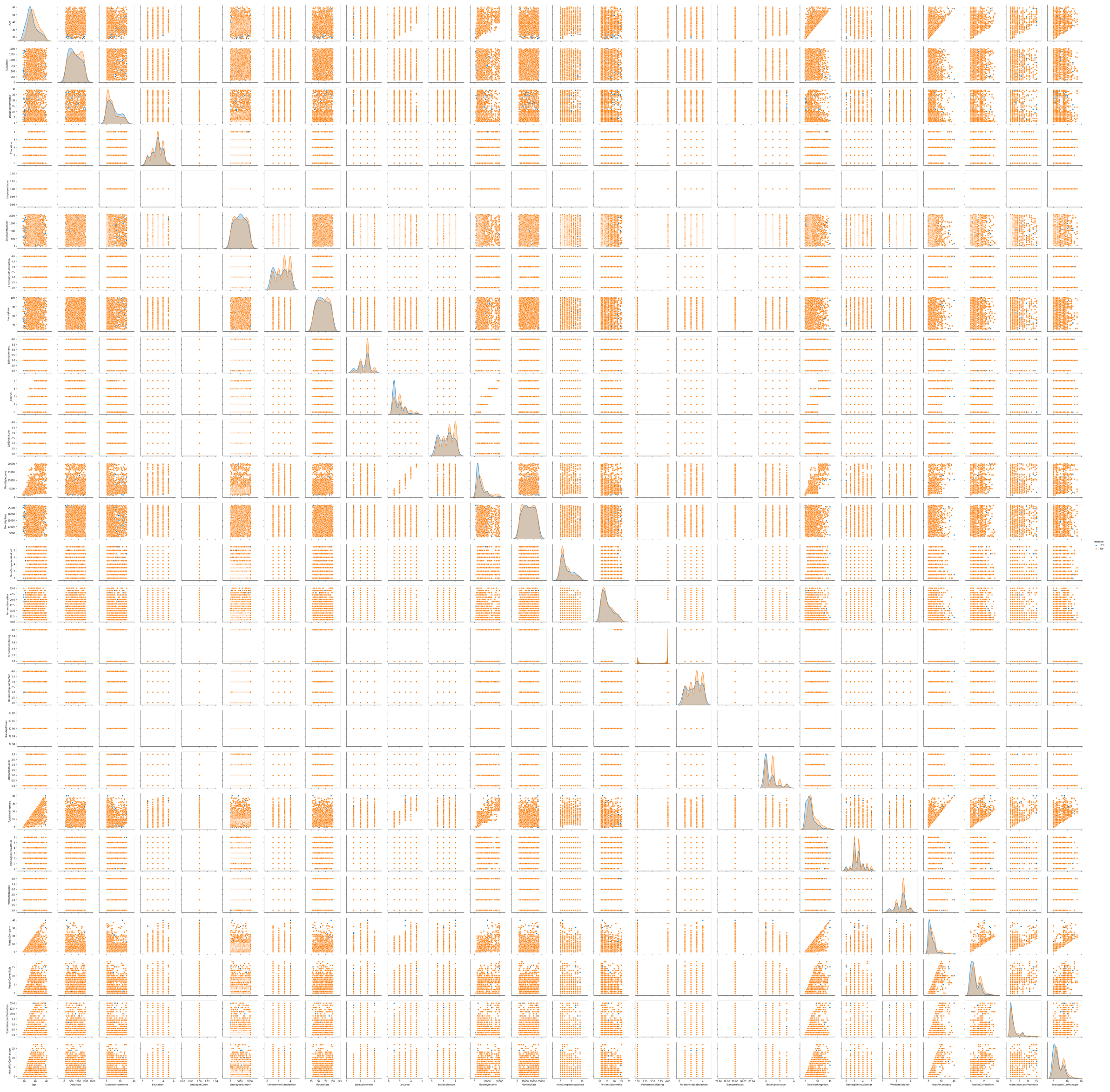
In [30]:

```
sns.pairplot(df,hue='Attrition')
```

```
C:\Users\ISHITA GUPTA\Anaconda3\lib\site-packages\statsmodels\nonparametric
\kde.py:487: RuntimeWarning: invalid value encountered in true_divide
  binned = fast_linbin(X, a, b, gridsize) / (delta * nobs)
C:\Users\ISHITA GUPTA\Anaconda3\lib\site-packages\statsmodels\nonparametric
\kdetools.py:34: RuntimeWarning: invalid value encountered in double_scalars
  FAC1 = 2*(np.pi*bw/RANGE)**2
```

Out[30]:

```
<seaborn.axisgrid.PairGrid at 0x26fcc61e9b0>
```



In [63]:

```
# Taking out the target variable
Y = df['Attrition']
```

In [64]:

```
df = df.drop(columns = ['Attrition','EmployeeNumber','EmployeeCount','StandardHours'])
```

In [66]:

```
# Feature Encoding
le = LabelEncoder()
Y = le.fit_transform(Y)
```

In [68]:

```
# Testing and Training data Split
Train_x, Test_x, Train_y, Test_y = train_test_split(df, Y, test_size=0.35, random_state= 40
```

In [69]:

```
# Model training
reg = linear_model.LogisticRegression()
reg.fit(Train_x,Train_y)
```

```
C:\Users\ISHITA GUPTA\Anaconda3\lib\site-packages\sklearn\linear_model\logis
tic.py:432: FutureWarning: Default solver will be changed to 'lbfgs' in 0.2
2. Specify a solver to silence this warning.
  FutureWarning)
```

Out[69]:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='warn', n_jobs=None, penalty='l2',
                   random_state=None, solver='warn', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [70]:

```
# Making Predictions
predict = reg.predict(Test_x)
```

In [71]:

```
# Checking Accuracy
accuracy_score(Test_y, predict)
```

Out[71]:

```
0.8757281553398059
```

In [77]:

```
# Checking Confusion Matrix
confu_mat = confusion_matrix(Test_y, predict)
print(confu_mat)
```

```
[[430   2]
 [ 62  21]]
```

In [73]:

```
# Checking the Random Forest Classifier
lm2 = RandomForestClassifier(n_estimators= 300,random_state= 20).fit(Train_x,Train_y)
forest_pred = lm2.predict(Test_x)
```

In [74]:

```
## Accuracy of Random Forest
accuracy_score(Test_y, forest_pred)
```

Out[74]:

```
0.8504854368932039
```

In [75]:

```
## Confusion Matrix
confu_mat = confusion_matrix(Test_y, forest_pred)
print(confu_mat)
```

```
[[432   0]
 [ 77   6]]
```

In [86]:

```
# Separating continuous and catagorical data
con=[]
cat=[]
for i in df.columns:
    if df[i].dtypes=='int64':
        con.append(i)
    elif df[i].dtypes=='object':
        cat.append(i)
```

In [88]:

```
con
```

Out[88]:

```
['DailyRate',
 'DistanceFromHome',
 'Education',
 'EnvironmentSatisfaction',
 'Gender',
 'HourlyRate',
 'JobInvolvement',
 'JobSatisfaction',
 'MonthlyIncome',
 'MonthlyRate',
 'NumCompaniesWorked',
 'Over18',
 'OverTime',
 'PercentSalaryHike',
 'PerformanceRating',
 'RelationshipSatisfaction',
 'StockOptionLevel',
 'TotalWorkingYears',
 'TrainingTimesLastYear',
 'WorkLifeBalance',
 'YearsAtCompany',
 'YearsInCurrentRole',
 'YearsSinceLastPromotion',
 'YearsWithCurrManager']
```

In [89]:

```
cat
```

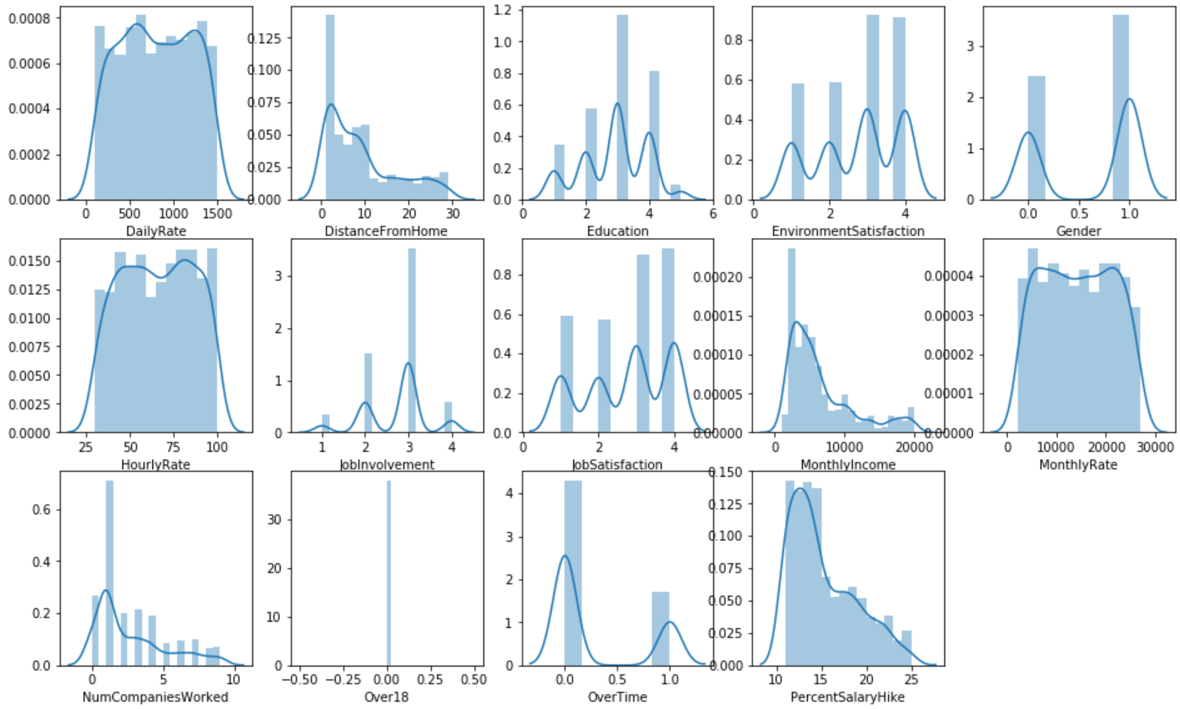
Out[89]:

```
[]
```

In [90]:

```
# Checking the distribution of Values in continous features
plt.figure(figsize = (16,20))
for i in range(0,14):
    plt.subplot(6,5,i+1)
    sns.distplot(df[con[i]])
```

```
C:\Users\ISHITA GUPTA\Anaconda3\lib\site-packages\statsmodels\nonparametric
\kde.py:487: RuntimeWarning: invalid value encountered in true_divide
  binned = fast_linbin(X, a, b, gridsize) / (delta * nobs)
C:\Users\ISHITA GUPTA\Anaconda3\lib\site-packages\statsmodels\nonparametric
\kdetools.py:34: RuntimeWarning: invalid value encountered in double_scalars
  FAC1 = 2*(np.pi*bw/RANGE)**2
```


DailyRate
DistanceFromHome
Education
EnvironmentSatisfaction
Gender
HourlyRate
JobInvolvement
JobSatisfaction
MonthlyIncome
MonthlyRate
NumCompaniesWorked
Over18
OverTime
PercentSalaryHike