In [165…

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
```

Data-Importing And Pre-processing

In [57]:

```
df=pd.read_csv("C:/Users/Dell/Downloads/archive/WA_Fn-UseC_-HR-Employee-Attrition.
```

In [58]:

```
df.describe()
```

Out[58]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNum
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	1470.000
mean	36.923810	802.485714	9.192517	2.912925	1.0	1024.865
std	9.135373	403.509100	8.106864	1.024165	0.0	602.024
min	18.000000	102.000000	1.000000	1.000000	1.0	1.000
25%	30.000000	465.000000	2.000000	2.000000	1.0	491.250
50%	36.000000	802.000000	7.000000	3.000000	1.0	1020.500
75%	43.000000	1157.000000	14.000000	4.000000	1.0	1555.750
max	60.000000	1499.000000	29.000000	5.000000	1.0	2068.000

8 rows × 26 columns

In [68]:

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

Out[68]:

```
Age                         0
Attrition                   0
BusinessTravel              0
DailyRate                   0
Department                  0
DistanceFromHome            0
Education                   0
EducationField              0
EmployeeCount               0
EmployeeNumber              0
EnvironmentSatisfaction     0
Gender                      0
HourlyRate                  0
JobInvolvement              0
JobLevel                    0
JobRole                     0
JobSatisfaction             0
MaritalStatus               0
MonthlyIncome               0
MonthlyRate                 0
NumCompaniesWorked          0
Over18                      0
OverTime                    0
PercentSalaryHike           0
PerformanceRating           0
RelationshipSatisfaction    0
StandardHours               0
StockOptionLevel            0
TotalWorkingYears           0
TrainingTimesLastYear       0
WorkLifeBalance             0
YearsAtCompany              0
YearsInCurrentRole          0
YearsSinceLastPromotion     0
YearsWithCurrManager        0
dtype: int64
```

In [60]:

```
df.head(5)
```

Out[60]:

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

5 rows × 35 columns

In [61]:

```
df.dtypes
```

Out[61]:

```
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
```

In [69]:

```
# removingUnwanatedAttributes
df=df.drop(["BusinessTravel","Department","EducationField","Over18","OverTime","Jol
```

In [70]:

```
df=df.replace({"Gender":{"Male":1,"Female":0},"MaritalStatus":{"Married":0,"Single
```

In [75]:

```
x=df.drop("Attrition",axis=1)
y=df["Attrition"]
```

splliting data

In [166…

```
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.6,stratify=y)
```

In [167…

```
scaler = MinMaxScaler()
x_train= scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)
```

LogisticRegression Model and Training and Valadition

```
In [168…  model=LogisticRegression(C=12)
          model.fit(x_train,y_train)
```

```
Out[168]:  ▾ LogisticRegression
           LogisticRegression(C=12)
```

```
In [169…  y_pred=model.predict(x_test)
```

```
In [170…  accuracy_score(y_pred,y_test)
```

```
Out[170]:  0.8503401360544217
```

DecisionTree training And Validation

```
In [171…  model1=DecisionTreeClassifier()
          model1.fit(x_train,y_train)
```

```
Out[171]:  ▾ DecisionTreeClassifier
           DecisionTreeClassifier()
```

```
In [172…  y_pred1=model1.predict(x_test)
```

```
In [173…  accuracy_score(y_pred1,y_test)
```

```
Out[173]:  0.7568027210884354
```

```
In [164…  accuracy_scores.mean()
```

```
Out[164]:  0.8416666666666667
```

```
In [ ]:  # by comparing above tw0 logistic regressino is best option
```

prediction Using Logistic Regression

```
In [191…  model.predict(x_train[879].reshape(1,-1))
```

```
Out[191]:  array([0], dtype=int64)
```

```
In [ ]:
```

```
In [ ]:
```