Name: DHARMANA GNANA SAI

Reg No: 21BCE7400

Assignment-4

• Data Preprocessing.

```
o       Import the Libraries.
o       Importing the dataset.
o       Checking for Null Values.
o       Data Visualization.
o       Outlier Detection
o       Splitting Dependent and Independent variables
o-      Encoding
o       Feature Scaling.
o       Splitting Data into Train and Test.
```

Data Collection. o Collect the dataset or Create the dataset • Data Preprocessing. o Import the Libraries. o Importing the dataset. o Checking for Null Values. o Data Visualization. o Outlier Detection o Splitting Dependent and Independent variables o-Encoding o Feature Scaling. o Splitting Data into Train and Test. • Model Building o Import the model building Libraries o Initializing the model o Training and testing the model o Evaluation of Model o Save the Model

In [1]:

```python
#Import the Libraries.
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:

```python
#Importing the dataset.
df = pd.read_csv('Employee-Attrition.csv')
df
```

Out[2]:

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

1470 rows × 35 columns

In [3]:

```
df.head()
```

Out[3]:

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

5 rows × 35 columns

In [4]:

```
df.shape
```

Out[4]:

```
(1470, 35)
```

```
In [5]: df.DailyRate.value_counts()
```

```
Out[5]: 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
```

```
In [6]: df.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
```

In [7]:
```
df.describe()
```

Out[7]:

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

8 rows × 26 columns

In [8]:
```
#Checking for Null Values.
df.isnull().any()
```

```
Out[8]: Age                         False
        Attrition                   False
        BusinessTravel              False
        DailyRate                   False
        Department                  False
        DistanceFromHome            False
        Education                   False
        EducationField              False
        EmployeeCount               False
        EmployeeNumber              False
        EnvironmentSatisfaction     False
        Gender                      False
        HourlyRate                  False
        JobInvolvement              False
        JobLevel                    False
        JobRole                     False
        JobSatisfaction             False
        MaritalStatus               False
        MonthlyIncome               False
        MonthlyRate                 False
        NumCompaniesWorked          False
        Over18                      False
        OverTime                    False
        PercentSalaryHike           False
        PerformanceRating           False
        RelationshipSatisfaction    False
        StandardHours               False
        StockOptionLevel            False
        TotalWorkingYears           False
        TrainingTimesLastYear       False
        WorkLifeBalance             False
        YearsAtCompany              False
        YearsInCurrentRole          False
        YearsSinceLastPromotion     False
        YearsWithCurrManager        False
        dtype: bool
```

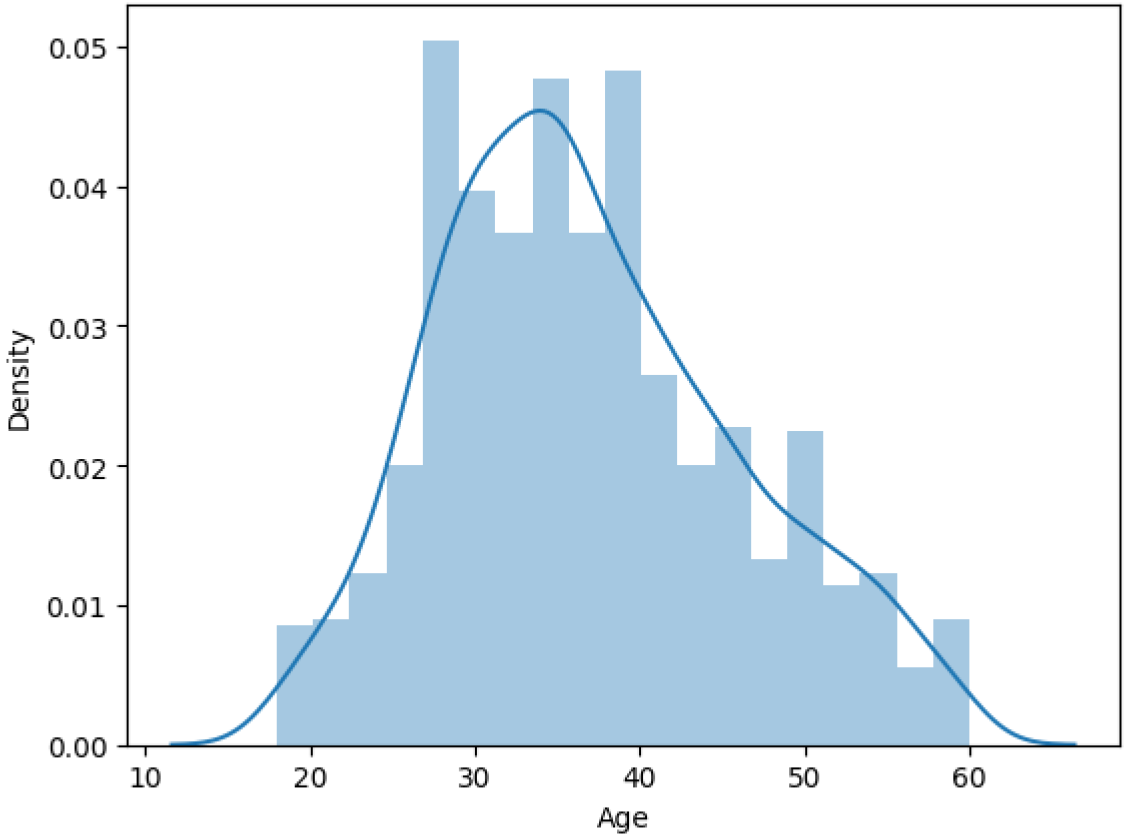
```
In [9]: df.isnull().sum()
```

```
Out[9]:  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 [3]:  #Data Visualization.
         sns.distplot(df["Age"])
```

```
C:\Users\dharm\AppData\Local\Temp\ipykernel_14632\2400079689.py:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df["Age"])
```

```
Out[3]:  <Axes: xlabel='Age', ylabel='Density'>
```



In [11]:
```
df.corr()
```

```
C:\Users\chatu\AppData\Local\Temp\ipykernel_8416\1134722465.py:1: FutureWarning:
The default value of numeric_only in DataFrame.corr is deprecated. In a future ve
rsion, it will default to False. Select only valid columns or specify the value o
f numeric_only to silence this warning.
  df.corr()
```

Out[11]:

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

26 rows × 26 columns

In [12]:

```
df.head()
```

Out[12]:

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

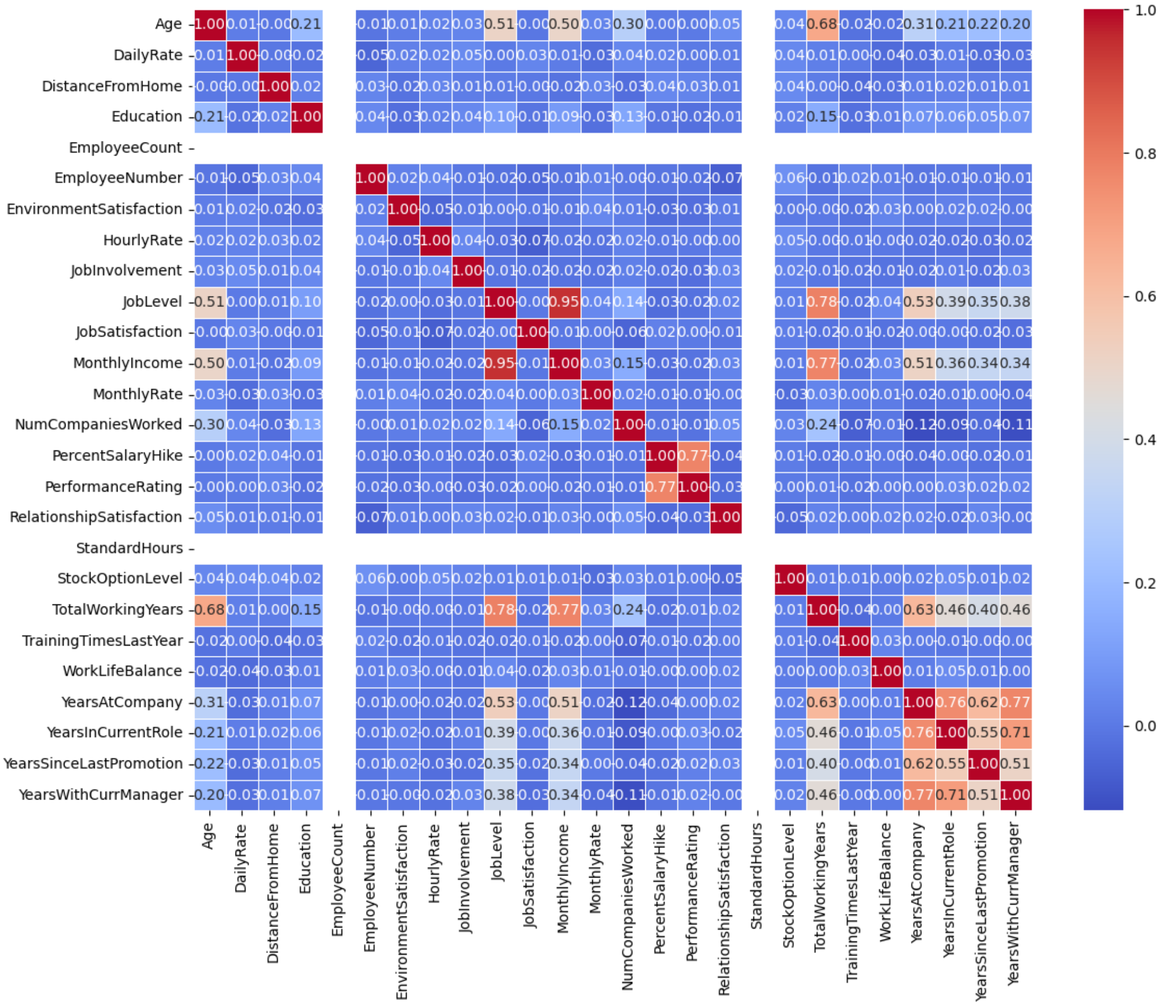
5 rows × 35 columns

In [13]:

```
plt.figure(figsize=(13, 10))
sns.heatmap(df.corr(),annot=True,cmap='coolwarm', fmt='.2f', linewidths=0.5)
```

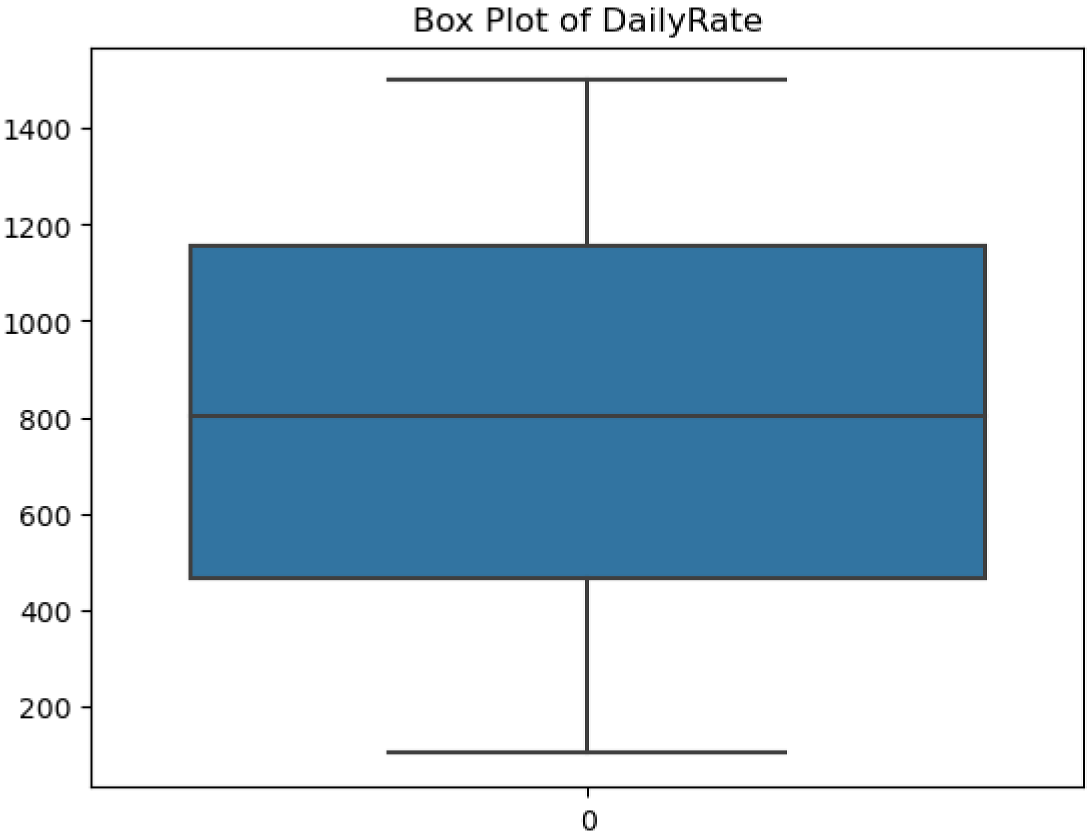
```
C:\Users\chatu\AppData\Local\Temp\ipykernel_8416\195898717.py:2: FutureWarning: T
he default value of numeric_only in DataFrame.corr is deprecated. In a future ver
sion, it will default to False. Select only valid columns or specify the value of
numeric_only to silence this warning.
  sns.heatmap(df.corr(),annot=True,cmap='coolwarm', fmt='.2f', linewidths=0.5)
```

Out[13]: <Axes: >



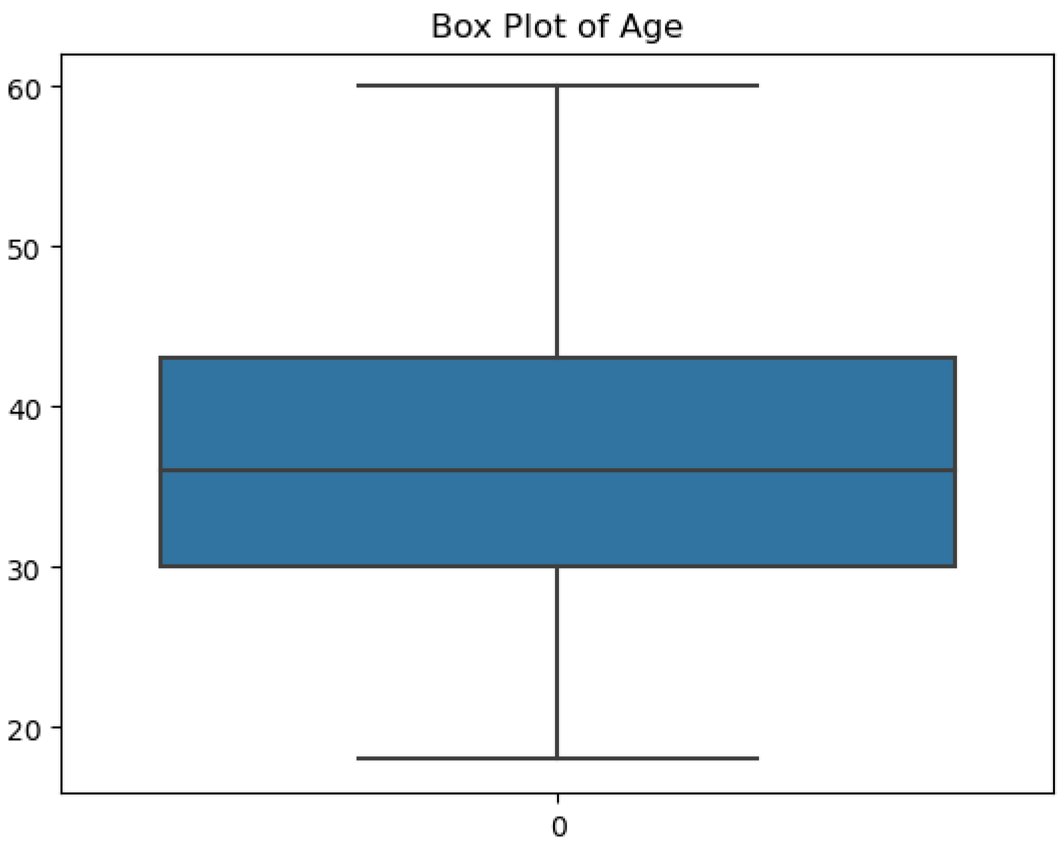
In [14]:
```
plt.title("Box Plot of DailyRate")
sns.boxplot(df.DailyRate)
```

Out[14]:
```
<Axes: title={'center': 'Box Plot of DailyRate'}>
```



In [17]:
```
plt.title("Box Plot of Age")
sns.boxplot(df.Age)
```

Out[17]:
```
<Axes: title={'center': 'Box Plot of Age'}>
```



In [18]:
```
df.head()
```

Out[18]:

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

5 rows × 35 columns

In [21]:
```
#Splitting Dependent and Independent variables
x=df.iloc[:,1:7]
x.head()
```

Out[21]:

	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education
0	Yes	Travel_Rarely	1102	Sales	1	2
1	No	Travel_Frequently	279	Research & Development	8	1
2	Yes	Travel_Rarely	1373	Research & Development	2	2
3	No	Travel_Frequently	1392	Research & Development	3	4
4	No	Travel_Rarely	591	Research & Development	2	1

In [22]:

```
y=df.DistanceFromHome
y.head()
```

Out[22]:

```
0    1
1    8
2    2
3    3
4    2
Name: DistanceFromHome, dtype: int64
```

In [23]:

```
#label encoding
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
x['DailyRate']=le.fit_transform(x['DailyRate'])
x.head()
```

Out[23]:

	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education
0	Yes	Travel_Rarely	624	Sales	1	2
1	No	Travel_Frequently	113	Research & Development	8	1
2	Yes	Travel_Rarely	805	Research & Development	2	2
3	No	Travel_Frequently	820	Research & Development	3	4
4	No	Travel_Rarely	312	Research & Development	2	1

In [24]:

```
#feature scaling
non_numeric_columns = x.select_dtypes(exclude=['number']).columns
# Drop non-numeric columns
x_numeric = x.drop(columns=non_numeric_columns)
x_encoded = pd.get_dummies(x, columns=non_numeric_columns)

from sklearn.preprocessing import MinMaxScaler
ms=MinMaxScaler()
x_scaled = pd.DataFrame(ms.fit_transform(x_numeric), columns=x_numeric.columns)
```

In [25]:

```
x_scaled
```

Out[25]:

	DailyRate	DistanceFromHome	Education
0	0.705085	0.000000	0.25
1	0.127684	0.250000	0.00
2	0.909605	0.035714	0.25
3	0.926554	0.071429	0.75
4	0.352542	0.035714	0.00
...	...	...	...
1465	0.558192	0.785714	0.25
1466	0.369492	0.178571	0.00
1467	0.044068	0.107143	0.50
1468	0.654237	0.035714	0.50
1469	0.379661	0.250000	0.50

1470 rows × 3 columns

In [26]:

```
#Splitting Data into Train and Test.
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x_scaled,y,test_size=0.2,random_s
```

In [27]:

```
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

Out[27]:

```
((1176, 3), (294, 3), (1176,), (294,))
```

In [28]:

```
x_train.head()
```

Out[28]:

	DailyRate	DistanceFromHome	Education
1374	0.364972	0.714286	0.50
1092	0.605650	0.964286	0.50
768	0.141243	0.892857	0.50
569	0.954802	0.250000	0.75
911	0.358192	0.821429	0.00

• Model Building

```
o       Import the model building Libraries
o       Initializing the model
o       Training and testing the model
o       Evaluation of Model
o       Save the Model
```

In [29]:

```
from sklearn.tree import DecisionTreeClassifier
```

```
dtc=DecisionTreeClassifier()
```

In [30]:
```
dtc.fit(x_train,y_train)
```

Out[30]:
```
▾ DecisionTreeClassifier

DecisionTreeClassifier()
```

In [31]:
```
pred=dtc.predict(x_test)
```

In [32]:
```
pred
```

Out[32]:
```
array([10, 25, 18, 20, 24,  3, 24,  2,  1, 14, 20, 23,  2,  1, 20,  2,  3,
        3, 10, 15,  9,  6,  3, 15, 23,  8, 25,  2,  5,  2,  2,  2, 18,  7,
       10,  3, 24, 12,  9, 18, 29, 13, 10,  3,  1,  5,  2,  2, 10,  1, 15,
       20,  9,  1,  2,  3,  4,  8,  8,  1,  2, 18,  2,  1, 26,  3,  2, 29,
        9,  9, 25, 23, 20, 19,  4, 19,  9,  2,  2,  2,  4,  8,  9,  1, 20,
        5, 14,  1,  3, 29, 14, 27, 16,  7, 10, 16, 25,  2, 10,  2, 13, 29,
       22, 14, 10,  2,  6, 10, 29,  1, 16,  6,  1, 10,  9, 10,  1,  6,  4,
        3, 29,  2,  5, 24,  1,  2,  2,  8,  7,  5, 12,  9,  2,  7,  7,  6,
        5,  6,  7,  2,  6, 10,  8,  1,  6,  1,  2,  4,  3,  3,  1, 17,  1,
        2, 18,  8, 29,  2, 21,  1, 12,  1,  9, 25,  1, 17,  1, 18, 20,  3,
       26,  2,  8, 11, 12,  1, 10,  1,  7,  9, 20,  9,  2, 24, 11,  2, 16,
        1, 23,  2, 28,  8, 15,  1,  7,  2,  2, 17, 12,  2,  7,  8,  9,  5,
       27, 20,  1,  1,  7,  3, 10, 28,  8,  1,  8, 10,  5, 10,  9,  1, 12,
        2, 26,  8,  3, 13,  8,  2,  9,  6,  7,  2,  1, 10,  6,  8,  4, 18,
       19, 23,  1,  4,  2,  9,  1, 10, 11,  2,  7,  7, 19,  2, 29, 14,  4,
        7, 18, 10, 16,  9, 10,  1, 22,  7,  2,  9,  8,  7, 13,  6,  1,  7,
       26, 16,  7, 11, 18,  2, 19,  6,  4,  1,  4,  2,  1,  6,  2, 22,  3,
        3,  1,  6,  2,  8], dtype=int64)
```

In [33]:
```
y_test
```

Out[33]:
```
442     10
1091    25
981     18
785     20
1332    24
        ..
1439     3
481      1
124      6
198      2
1229     8
Name: DistanceFromHome, Length: 294, dtype: int64
```

In [34]:
```
df
```

Out[34]:

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

1470 rows × 35 columns

In [35]:

```
dtc.predict(ms.transform([[1,20,18000]]))
```

```
C:\Users\chatu\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X do
es not have valid feature names, but MinMaxScaler was fitted with feature names
  warnings.warn(
C:\Users\chatu\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X do
es not have valid feature names, but DecisionTreeClassifier was fitted with featu
re names
  warnings.warn(
```

Out[35]:

```
array([20], dtype=int64)
```

Evaluation of classification model

In [36]:

```
#Accuracy score
from sklearn.metrics import accuracy_score,confusion_matrix,classification_repor
```

In [37]:

```
accuracy_score(y_test,pred)
```

Out[37]:

```
1.0
```

In [38]:

```
confusion_matrix(y_test,pred)
```

```
Out[38]: array([[37,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0, 45,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0, 17,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0, 10,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  8,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0, 14,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0, 18,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0, 17,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0, 18,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0, 20,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  4,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  6,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  4,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  5,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  4,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  6,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  3,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  9,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  5,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  9,  0,  0,  0,  0,  0,  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,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  3,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  5,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  5,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  5,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  4,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  2,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  2,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
                  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  8]], dtype=int64)
```

In [39]:

```
pd.crosstab(y_test,pred)
```

Out[39]:

col_0	1	2	3	4	5	6	7	8	9	10	...	20	21	22	23	24	2
DistanceFromHome																	
1	37	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
2	0	45	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
3	0	0	17	0	0	0	0	0	0	0	...	0	0	0	0	0	
4	0	0	0	10	0	0	0	0	0	0	...	0	0	0	0	0	
5	0	0	0	0	8	0	0	0	0	0	...	0	0	0	0	0	
6	0	0	0	0	0	14	0	0	0	0	...	0	0	0	0	0	
7	0	0	0	0	0	0	18	0	0	0	...	0	0	0	0	0	
8	0	0	0	0	0	0	0	17	0	0	...	0	0	0	0	0	
9	0	0	0	0	0	0	0	0	18	0	...	0	0	0	0	0	
10	0	0	0	0	0	0	0	0	0	20	...	0	0	0	0	0	
11	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
12	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
13	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
14	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
15	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
16	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
17	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
18	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
19	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
20	0	0	0	0	0	0	0	0	0	0	...	9	0	0	0	0	
21	0	0	0	0	0	0	0	0	0	0	...	0	1	0	0	0	
22	0	0	0	0	0	0	0	0	0	0	...	0	0	3	0	0	
23	0	0	0	0	0	0	0	0	0	0	...	0	0	0	5	0	
24	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	5	
25	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
26	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
27	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
28	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	
29	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	

29 rows × 29 columns

```
                predicted no     predicted yes
```

Actual No 58=TN 0=FP Actual yes 6=FN 16=TP

In [41]:
```
(58+16)/80 #accuracy
```

Out[41]:
```
0.925
```

In [42]:
```
print(classification_report(y_test,pred))
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00        37
           2       1.00      1.00      1.00        45
           3       1.00      1.00      1.00        17
           4       1.00      1.00      1.00        10
           5       1.00      1.00      1.00         8
           6       1.00      1.00      1.00        14
           7       1.00      1.00      1.00        18
           8       1.00      1.00      1.00        17
           9       1.00      1.00      1.00        18
          10       1.00      1.00      1.00        20
          11       1.00      1.00      1.00         4
          12       1.00      1.00      1.00         6
          13       1.00      1.00      1.00         4
          14       1.00      1.00      1.00         5
          15       1.00      1.00      1.00         4
          16       1.00      1.00      1.00         6
          17       1.00      1.00      1.00         3
          18       1.00      1.00      1.00         9
          19       1.00      1.00      1.00         5
          20       1.00      1.00      1.00         9
          21       1.00      1.00      1.00         1
          22       1.00      1.00      1.00         3
          23       1.00      1.00      1.00         5
          24       1.00      1.00      1.00         5
          25       1.00      1.00      1.00         5
          26       1.00      1.00      1.00         4
          27       1.00      1.00      1.00         2
          28       1.00      1.00      1.00         2
          29       1.00      1.00      1.00         8

    accuracy                           1.00       294
   macro avg       1.00      1.00      1.00       294
weighted avg       1.00      1.00      1.00       294
```

Roc-AUC curve

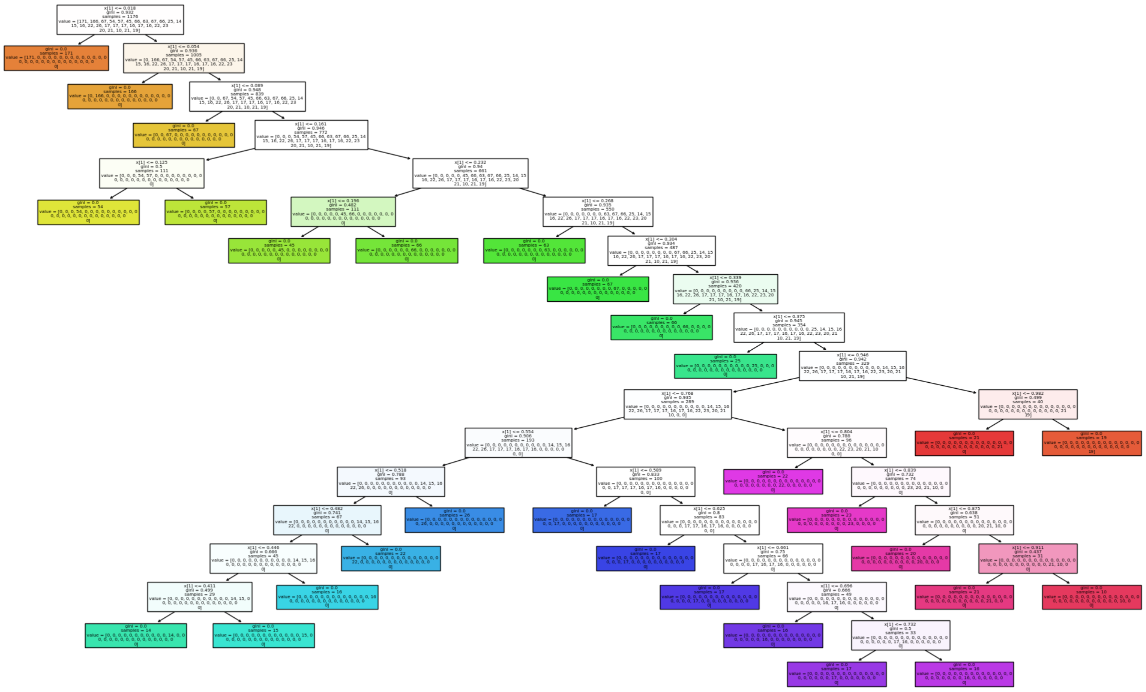
In [43]:
```
probability=dtc.predict_proba(x_test)[:,1]
```

In [44]:
```
probability
```

```
Out[44]: array([0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 1., 0., 0., 1., 0.,
                0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 1., 1., 1., 0., 0.,
                0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0.,
                0., 0., 0., 1., 0., 0., 0., 0., 0., 1., 0., 1., 0., 0., 0., 1., 0.,
                0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 1., 1., 0., 0., 0., 0., 0.,
                0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 1., 0., 0.,
                0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
                0., 0., 1., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 1., 0., 0., 0.,
                0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.,
                1., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
                0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 1., 0.,
                0., 0., 1., 0., 0., 0., 0., 0., 1., 1., 0., 0., 1., 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., 1., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0.,
                0., 0., 0., 0., 1., 0., 0., 0., 0., 1., 0., 0., 0., 1., 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., 1., 0., 0., 0., 0., 0., 1., 0., 0., 1., 0., 0.,
                0., 0., 0., 1., 0.])
```

```
In [45]: from sklearn import tree
         plt.figure(figsize=(25,15))
         tree.plot_tree(dtc,filled=True)
```

```
Out[45]: [Text(0.1095890410958904, 0.9722222222222222, 'x[1] <= 0.018\ngini = 0.932\nsam
         ples = 1176\nvalue = [171, 166, 67, 54, 57, 45, 66, 63, 67, 66, 25, 14\n15, 16,
         22, 26, 17, 17, 17, 16, 17, 16, 22, 23\n20, 21, 10, 21, 19]'),
          Text(0.0547945205479452, 0.9166666666666666, 'gini = 0.0\nsamples = 171\nvalue
         = [171, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.1643835616438356, 0.9166666666666666, 'x[1] <= 0.054\ngini = 0.936\nsam
         ples = 1005\nvalue = [0, 166, 67, 54, 57, 45, 66, 63, 67, 66, 25, 14\n15, 16, 2
         2, 26, 17, 17, 17, 16, 17, 16, 22, 23\n20, 21, 10, 21, 19]'),
          Text(0.1095890410958904, 0.8611111111111112, 'gini = 0.0\nsamples = 166\nvalue
         = [0, 166, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.2191780821917808, 0.8611111111111112, 'x[1] <= 0.089\ngini = 0.948\nsam
         ples = 839\nvalue = [0, 0, 67, 54, 57, 45, 66, 63, 67, 66, 25, 14\n15, 16, 22,
         26, 17, 17, 17, 16, 17, 16, 22, 23\n20, 21, 10, 21, 19]'),
          Text(0.1643835616438356, 0.8055555555555556, 'gini = 0.0\nsamples = 67\nvalue
         = [0, 0, 67, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.273972602739726, 0.8055555555555556, 'x[1] <= 0.161\ngini = 0.946\nsamp
         les = 772\nvalue = [0, 0, 0, 54, 57, 45, 66, 63, 67, 66, 25, 14\n15, 16, 22, 2
         6, 17, 17, 17, 16, 17, 16, 22, 23\n20, 21, 10, 21, 19]'),
          Text(0.136986301369863, 0.75, 'x[1] <= 0.125\ngini = 0.5\nsamples = 111\nvalue
         = [0, 0, 0, 54, 57, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.0821917808219178, 0.6944444444444444, 'gini = 0.0\nsamples = 54\nvalue
         = [0, 0, 0, 54, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.1917808219178082, 0.6944444444444444, 'gini = 0.0\nsamples = 57\nvalue
         = [0, 0, 0, 0, 57, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.410958904109589, 0.75, 'x[1] <= 0.232\ngini = 0.94\nsamples = 661\nvalu
         e = [0, 0, 0, 0, 0, 45, 66, 63, 67, 66, 25, 14, 15\n16, 22, 26, 17, 17, 17, 16,
         17, 16, 22, 23, 20\n21, 10, 21, 19]'),
          Text(0.3013698630136986, 0.6944444444444444, 'x[1] <= 0.196\ngini = 0.482\nsam
         ples = 111\nvalue = [0, 0, 0, 0, 0, 45, 66, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0\n0]'),
          Text(0.2465753424657534, 0.6388888888888888, 'gini = 0.0\nsamples = 45\nvalue
         = [0, 0, 0, 0, 0, 45, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.3561643835616438, 0.6388888888888888, 'gini = 0.0\nsamples = 66\nvalue
         = [0, 0, 0, 0, 0, 0, 66, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.5205479452054794, 0.6944444444444444, 'x[1] <= 0.268\ngini = 0.935\nsam
         ples = 550\nvalue = [0, 0, 0, 0, 0, 0, 0, 63, 67, 66, 25, 14, 15\n16, 22, 26, 1
         7, 17, 17, 16, 17, 16, 22, 23, 20\n21, 10, 21, 19]'),
          Text(0.4657534246575342, 0.6388888888888888, 'gini = 0.0\nsamples = 63\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 63, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.5753424657534246, 0.6388888888888888, 'x[1] <= 0.304\ngini = 0.934\nsam
         ples = 487\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 67, 66, 25, 14, 15\n16, 22, 26, 1
         7, 17, 17, 16, 17, 16, 22, 23, 20\n21, 10, 21, 19]'),
          Text(0.5205479452054794, 0.5833333333333334, 'gini = 0.0\nsamples = 67\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 67, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.6301369863013698, 0.5833333333333334, 'x[1] <= 0.339\ngini = 0.936\nsam
         ples = 420\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 66, 25, 14, 15\n16, 22, 26, 17,
         17, 17, 16, 17, 16, 22, 23, 20\n21, 10, 21, 19]'),
          Text(0.5753424657534246, 0.5277777777777778, 'gini = 0.0\nsamples = 66\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 66, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.684931506849315, 0.5277777777777778, 'x[1] <= 0.375\ngini = 0.945\nsamp
         les = 354\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 25, 14, 15, 16\n22, 26, 17, 1
         7, 17, 16, 17, 16, 22, 23, 20, 21\n10, 21, 19]'),
          Text(0.6301369863013698, 0.4722222222222222, 'gini = 0.0\nsamples = 25\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 25, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.7397260273972602, 0.4722222222222222, 'x[1] <= 0.946\ngini = 0.942\nsam
         ples = 329\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 15, 16\n22, 26, 17, 1
         7, 17, 16, 17, 16, 22, 23, 20, 21\n10, 21, 19]'),
          Text(0.589041095890411, 0.4166666666666667, 'x[1] <= 0.768\ngini = 0.935\nsamp
         les = 289\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 15, 16\n22, 26, 17, 1
         7, 17, 16, 17, 16, 22, 23, 20, 21\n10, 0, 0]'),
          Text(0.4520547945205479, 0.3611111111111111, 'x[1] <= 0.554\ngini = 0.906\nsam
         ples = 193\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 15, 16\n22, 26, 17, 1
         7, 17, 16, 17, 16, 0, 0, 0, 0, 0\n0, 0]'),
          Text(0.3424657534246575, 0.3055555555555556, 'x[1] <= 0.518\ngini = 0.788\nsam
         ples = 93\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 15, 16\n22, 26, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0\n0]'),
          Text(0.2876712328767123, 0.25, 'x[1] <= 0.482\ngini = 0.741\nsamples = 67\nval
         ue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 15, 16\n22, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0, 0, 0\n0]'),
          Text(0.2328767123287671, 0.19444444444444445, 'x[1] <= 0.446\ngini = 0.666\nsa
         mples = 45\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 15, 16\n0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0\n0]'),
          Text(0.1780821917808219, 0.1388888888888889, 'x[1] <= 0.411\ngini = 0.499\nsam
         ples = 29\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 15, 0\n0, 0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0\n0]'),
          Text(0.1232876712328767, 0.08333333333333333, 'gini = 0.0\nsamples = 14\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.2328767123287671, 0.08333333333333333, 'gini = 0.0\nsamples = 15\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 15, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.2876712328767123, 0.1388888888888889, 'gini = 0.0\nsamples = 16\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 16\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.3424657534246575, 0.19444444444444445, 'gini = 0.0\nsamples = 22\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n22, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.3972602739726027, 0.25, 'gini = 0.0\nsamples = 26\nvalue = [0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 26, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0]'),
          Text(0.5616438356164384, 0.3055555555555556, 'x[1] <= 0.589\ngini = 0.833\nsam
         ples = 100\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 17, 17, 1
         7, 16, 17, 16, 0, 0, 0, 0, 0\n0, 0]'),
          Text(0.5068493150684932, 0.25, 'gini = 0.0\nsamples = 17\nvalue = [0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 17, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0]'),
          Text(0.6164383561643836, 0.25, 'x[1] <= 0.625\ngini = 0.8\nsamples = 83\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 17, 17, 16, 17, 16, 0, 0,
         0, 0, 0\n0, 0]'),
          Text(0.5616438356164384, 0.19444444444444445, 'gini = 0.0\nsamples = 17\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 17, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.6712328767123288, 0.19444444444444445, 'x[1] <= 0.661\ngini = 0.75\nsam
         ples = 66\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 17, 1
         6, 17, 16, 0, 0, 0, 0, 0, 0\n0]'),
          Text(0.6164383561643836, 0.1388888888888889, 'gini = 0.0\nsamples = 17\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 17, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.726027397260274, 0.1388888888888889, 'x[1] <= 0.696\ngini = 0.666\nsamp
         les = 49\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 16,
         17, 16, 0, 0, 0, 0, 0, 0\n0]'),
          Text(0.6712328767123288, 0.08333333333333333, 'gini = 0.0\nsamples = 16\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 16, 0, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.7808219178082192, 0.08333333333333333, 'x[1] <= 0.732\ngini = 0.5\nsamp
         les = 33\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
         17, 16, 0, 0, 0, 0, 0, 0\n0]'),
          Text(0.726027397260274, 0.027777777777777776, 'gini = 0.0\nsamples = 17\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 17, 0, 0, 0, 0,
         0, 0, 0\n0]'),
          Text(0.8356164383561644, 0.027777777777777776, 'gini = 0.0\nsamples = 16\nvalu
         e = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 16, 0, 0,
         0, 0, 0, 0\n0]'),
          Text(0.726027397260274, 0.3611111111111111, 'x[1] <= 0.804\ngini = 0.788\nsamp
         les = 96\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
         0, 0, 22, 23, 20, 21, 10\n0, 0]'),
          Text(0.6712328767123288, 0.3055555555555556, 'gini = 0.0\nsamples = 22\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 22, 0, 0,
         0, 0, 0\n0]'),
          Text(0.7808219178082192, 0.3055555555555556, 'x[1] <= 0.839\ngini = 0.732\nsam
         ples = 74\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
         0, 0, 0, 23, 20, 21, 10, 0\n0]'),
          Text(0.726027397260274, 0.25, 'gini = 0.0\nsamples = 23\nvalue = [0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 23, 0, 0, 0, 0\n0]'),
          Text(0.8356164383561644, 0.25, 'x[1] <= 0.875\ngini = 0.638\nsamples = 51\nval
         ue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 2
         0, 21, 10, 0\n0]'),
          Text(0.7808219178082192, 0.19444444444444445, 'gini = 0.0\nsamples = 20\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 20,
         0, 0, 0\n0]'),
          Text(0.8904109589041096, 0.19444444444444445, 'x[1] <= 0.911\ngini = 0.437\nsa
         mples = 31\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 21, 10, 0\n0]'),
          Text(0.8356164383561644, 0.1388888888888889, 'gini = 0.0\nsamples = 21\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2
         1, 0, 0\n0]'),
          Text(0.9452054794520548, 0.1388888888888889, 'gini = 0.0\nsamples = 10\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 10, 0\n0]'),
          Text(0.8904109589041096, 0.4166666666666667, 'x[1] <= 0.982\ngini = 0.499\nsam
         ples = 40\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 21\n19]'),
          Text(0.8356164383561644, 0.3611111111111111, 'gini = 0.0\nsamples = 21\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 21\n0]'),
          Text(0.9452054794520548, 0.3611111111111111, 'gini = 0.0\nsamples = 19\nvalue
         = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0\n19]')]
```



In [46]:
```
from sklearn.model_selection import GridSearchCV
parameter={
 'criterion':['gini','entropy'],
  'splitter':['best','random'],
  'max_depth':[1,2,3,4,5],
  'max_features':['auto', 'sqrt', 'log2']

}
```

In [47]:
```
grid_search=GridSearchCV(estimator=dtc,param_grid=parameter,cv=5,scoring="accura
```

In [48]:
```
grid_search.fit(x_train,y_train)
```

```
C:\Users\chatu\anaconda3\Lib\site-packages\sklearn\model_selection\_validation.p
y:425: FitFailedWarning:
100 fits failed out of a total of 300.
The score on these train-test partitions for these parameters will be set to nan.
If these failures are not expected, you can try to debug them by setting error_sc
ore='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
100 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\chatu\anaconda3\Lib\site-packages\sklearn\model_selection\_valid
ation.py", line 732, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\chatu\anaconda3\Lib\site-packages\sklearn\base.py", line 1144, i
n wrapper
    estimator._validate_params()
  File "C:\Users\chatu\anaconda3\Lib\site-packages\sklearn\base.py", line 637, in
_validate_params
    validate_parameter_constraints(
  File "C:\Users\chatu\anaconda3\Lib\site-packages\sklearn\utils\_param_validatio
n.py", line 95, in validate_parameter_constraints
    raise InvalidParameterError(
sklearn.utils._param_validation.InvalidParameterError: The 'max_features' paramet
er of DecisionTreeClassifier must be an int in the range [1, inf), a float in the
range (0.0, 1.0], a str among {'sqrt', 'log2'} or None. Got 'auto' instead.

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\chatu\anaconda3\Lib\site-packages\sklearn\model_selection\_search.py:97
6: UserWarning: One or more of the test scores are non-finite: [        nan
nan 0.23390912 0.1641291  0.26443924 0.16072845
        nan        nan 0.19476019 0.16327804 0.22870537 0.17773891
        nan        nan 0.2041291  0.1981392  0.29338262 0.19558601
        nan        nan 0.30954922 0.16496574 0.26795889 0.26097367
        nan        nan 0.33679409 0.29338983 0.30106022 0.25424089
        nan        nan 0.17860079 0.16156149 0.17860079 0.16326722
        nan        nan 0.15561125 0.15393076 0.17179228 0.20485034
        nan        nan 0.20491165 0.16751893 0.25003967 0.17263613
        nan        nan 0.26614497 0.24069239 0.24834115 0.24322395
        nan        nan 0.31804904 0.23473134 0.29600793 0.22699243]
  warnings.warn(
```

Out[48]:

```
▸ GridSearchCV
▸ estimator: DecisionTreeClassifier
    ▸ DecisionTreeClassifier
```

In [49]:

```
dtc_cv=DecisionTreeClassifier(criterion= 'entropy',
 max_depth=3,
 max_features='sqrt',
 splitter='best')
dtc_cv.fit(x_train,y_train)
```

Out[49]:

```
▾ DecisionTreeClassifier
DecisionTreeClassifier(criterion='entropy', max_depth=3, max_features
='sqrt')
```

In [50]:

```
pred=dtc_cv.predict(x_test)
```

In [51]:

```
print(classification_report(y_test,pred))
```

```
              precision    recall  f1-score   support

           1       0.30      0.86      0.45        37
           2       0.40      0.22      0.29        45
           3       0.00      0.00      0.00        17
           4       0.00      0.00      0.00        10
           5       0.00      0.00      0.00         8
           6       0.00      0.00      0.00        14
           7       0.21      0.94      0.35        18
           8       0.00      0.00      0.00        17
           9       0.25      0.06      0.09        18
          10       0.00      0.00      0.00        20
          11       0.00      0.00      0.00         4
          12       0.00      0.00      0.00         6
          13       0.00      0.00      0.00         4
          14       0.00      0.00      0.00         5
          15       0.00      0.00      0.00         4
          16       0.08      1.00      0.15         6
          17       0.00      0.00      0.00         3
          18       0.00      0.00      0.00         9
          19       0.00      0.00      0.00         5
          20       0.00      0.00      0.00         9
          21       0.00      0.00      0.00         1
          22       0.00      0.00      0.00         3
          23       0.00      0.00      0.00         5
          24       0.00      0.00      0.00         5
          25       0.00      0.00      0.00         5
          26       0.00      0.00      0.00         4
          27       0.00      0.00      0.00         2
          28       0.00      0.00      0.00         2
          29       0.00      0.00      0.00         8

    accuracy                           0.22       294
   macro avg       0.04      0.11      0.05       294
weighted avg       0.13      0.22      0.13       294
```

```
C:\Users\chatu\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:146
9: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to control
this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\chatu\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:146
9: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to control
this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\chatu\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:146
9: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to control
this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

Random Forest

```
In [52]: from sklearn.ensemble import RandomForestClassifier
         rfc=RandomForestClassifier()
```

```
In [53]: forest_params = [{'max_depth': list(range(10, 15)), 'max_features': list(range(
```

```
In [54]: rfc_cv= GridSearchCV(rfc,param_grid=forest_params,cv=10,scoring="accuracy")
```

```
In [55]: # Fit the GridSearchCV to your training data
         rfc_cv.fit(x_train, y_train)
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[55], line 2
      1 # Fit the GridSearchCV to your training data
----> 2 rfc_cv.fit(x_train, y_train)

File ~\anaconda3\Lib\site-packages\sklearn\base.py:1151, in _fit_context.<locals
>.decorator.<locals>.wrapper(estimator, *args, **kwargs)
   1144     estimator._validate_params()
   1146 with config_context(
   1147     skip_parameter_validation=(
   1148         prefer_skip_nested_validation or global_skip_validation
   1149     )
   1150 ):
-> 1151     return fit_method(estimator, *args, **kwargs)

File ~\anaconda3\Lib\site-packages\sklearn\model_selection\_search.py:898, in Bas
eSearchCV.fit(self, X, y, groups, **fit_params)
    892     results = self._format_results(
    893         all_candidate_params, n_splits, all_out, all_more_results
    894     )
    896     return results
--> 898 self._run_search(evaluate_candidates)
    900 # multimetric is determined here because in the case of a callable
    901 # self.scoring the return type is only known after calling
    902 first_test_score = all_out[0]["test_scores"]

File ~\anaconda3\Lib\site-packages\sklearn\model_selection\_search.py:1419, in Gr
idSearchCV._run_search(self, evaluate_candidates)
   1417 def _run_search(self, evaluate_candidates):
   1418     """Search all candidates in param_grid"""
-> 1419     evaluate_candidates(ParameterGrid(self.param_grid))

File ~\anaconda3\Lib\site-packages\sklearn\model_selection\_search.py:845, in Bas
eSearchCV.fit.<locals>.evaluate_candidates(candidate_params, cv, more_results)
    837 if self.verbose > 0:
    838     print(
    839         "Fitting {0} folds for each of {1} candidates,"
    840         " totalling {2} fits".format(
    841             n_splits, n_candidates, n_candidates * n_splits
    842         )
    843     )
--> 845 out = parallel(
    846     delayed(_fit_and_score)(
    847         clone(base_estimator),
    848         X,
    849         y,
    850         train=train,
    851         test=test,
    852         parameters=parameters,
    853         split_progress=(split_idx, n_splits),
    854         candidate_progress=(cand_idx, n_candidates),
    855         **fit_and_score_kwargs,
    856     )
    857     for (cand_idx, parameters), (split_idx, (train, test)) in product(
    858         enumerate(candidate_params), enumerate(cv.split(X, y, groups))
    859     )
    860 )
    862 if len(out) < 1:
    863     raise ValueError(
    864         "No fits were performed. "
    865         "Was the CV iterator empty? "
    866         "Were there no candidates?"
    867     )

File ~\anaconda3\Lib\site-packages\sklearn\utils\parallel.py:65, in Parallel.__call__(self, iterable)
     60 config = get_config()
     61 iterable_with_config = (
     62     (_with_config(delayed_func, config), args, kwargs)
     63     for delayed_func, args, kwargs in iterable
     64 )
---> 65 return super().__call__(iterable_with_config)

File ~\anaconda3\Lib\site-packages\joblib\parallel.py:1088, in Parallel.__call__(self, iterable)
   1085 if self.dispatch_one_batch(iterator):
   1086     self._iterating = self._original_iterator is not None
-> 1088 while self.dispatch_one_batch(iterator):
   1089     pass
   1091 if pre_dispatch == "all" or n_jobs == 1:
   1092     # The iterable was consumed all at once by the above for loop.
   1093     # No need to wait for async callbacks to trigger to
   1094     # consumption.

File ~\anaconda3\Lib\site-packages\joblib\parallel.py:901, in Parallel.dispatch_one_batch(self, iterator)
    899     return False
    900 else:
--> 901     self._dispatch(tasks)
    902     return True

File ~\anaconda3\Lib\site-packages\joblib\parallel.py:819, in Parallel._dispatch(self, batch)
    817 with self._lock:
    818     job_idx = len(self._jobs)
--> 819     job = self._backend.apply_async(batch, callback=cb)
    820     # A job can complete so quickly than its callback is
    821     # called before we get here, causing self._jobs to
    822     # grow. To ensure correct results ordering, .insert is
    823     # used (rather than .append) in the following line
    824     self._jobs.insert(job_idx, job)

File ~\anaconda3\Lib\site-packages\joblib\_parallel_backends.py:208, in SequentialBackend.apply_async(self, func, callback)
    206 def apply_async(self, func, callback=None):
    207     """Schedule a func to be run"""
--> 208     result = ImmediateResult(func)
    209     if callback:
    210         callback(result)

File ~\anaconda3\Lib\site-packages\joblib\_parallel_backends.py:597, in ImmediateResult.__init__(self, batch)
    594 def __init__(self, batch):
    595     # Don't delay the application, to avoid keeping the input
    596     # arguments in memory
--> 597     self.results = batch()

File ~\anaconda3\Lib\site-packages\joblib\parallel.py:288, in BatchedCalls.__call__(self)
```

```
    284 def __call__(self):
    285     # Set the default nested backend to self._backend but do not set the
    286     # change the default number of processes to -1
    287     with parallel_backend(self._backend, n_jobs=self._n_jobs):
--> 288         return [func(*args, **kwargs)
    289                 for func, args, kwargs in self.items]

File ~\anaconda3\Lib\site-packages\joblib\parallel.py:288, in <listcomp>(.0)
    284 def __call__(self):
    285     # Set the default nested backend to self._backend but do not set the
    286     # change the default number of processes to -1
    287     with parallel_backend(self._backend, n_jobs=self._n_jobs):
--> 288         return [func(*args, **kwargs)
    289                 for func, args, kwargs in self.items]

File ~\anaconda3\Lib\site-packages\sklearn\utils\parallel.py:127, in _FuncWrappe
r.__call__(self, *args, **kwargs)
    125     config = {}
    126 with config_context(**config):
--> 127     return self.function(*args, **kwargs)

File ~\anaconda3\Lib\site-packages\sklearn\model_selection\_validation.py:732, in
_fit_and_score(estimator, X, y, scorer, train, test, verbose, parameters, fit_par
ams, return_train_score, return_parameters, return_n_test_samples, return_times,
return_estimator, split_progress, candidate_progress, error_score)
    730         estimator.fit(X_train, **fit_params)
    731     else:
--> 732         estimator.fit(X_train, y_train, **fit_params)
    734 except Exception:
    735     # Note fit time as time until error
    736     fit_time = time.time() - start_time

File ~\anaconda3\Lib\site-packages\sklearn\base.py:1151, in _fit_context.<locals
>.decorator.<locals>.wrapper(estimator, *args, **kwargs)
   1144     estimator._validate_params()
   1146 with config_context(
   1147     skip_parameter_validation=(
   1148         prefer_skip_nested_validation or global_skip_validation
   1149     )
   1150 ):
-> 1151     return fit_method(estimator, *args, **kwargs)

File ~\anaconda3\Lib\site-packages\sklearn\ensemble\_forest.py:456, in BaseFores
t.fit(self, X, y, sample_weight)
    445 trees = [
    446     self._make_estimator(append=False, random_state=random_state)
    447     for i in range(n_more_estimators)
    448 ]
    450 # Parallel loop: we prefer the threading backend as the Cython code
    451 # for fitting the trees is internally releasing the Python GIL
    452 # making threading more efficient than multiprocessing in
    453 # that case. However, for joblib 0.12+ we respect any
    454 # parallel_backend contexts set at a higher level,
    455 # since correctness does not rely on using threads.
--> 456 trees = Parallel(
    457     n_jobs=self.n_jobs,
    458     verbose=self.verbose,
    459     prefer="threads",
    460 )(
    461     delayed(_parallel_build_trees)(
    462         t,
    463         self.bootstrap,
    464         X,
    465         y,
    466         sample_weight,
    467         i,
    468         len(trees),
    469         verbose=self.verbose,
    470         class_weight=self.class_weight,
    471         n_samples_bootstrap=n_samples_bootstrap,
    472     )
    473     for i, t in enumerate(trees)
    474 )
    476 # Collect newly grown trees
    477 self.estimators_.extend(trees)

File ~\anaconda3\Lib\site-packages\sklearn\utils\parallel.py:65, in Parallel.__ca
ll__(self, iterable)
     60 config = get_config()
     61 iterable_with_config = (
     62     (_with_config(delayed_func, config), args, kwargs)
     63     for delayed_func, args, kwargs in iterable
     64 )
---> 65 return super().__call__(iterable_with_config)

File ~\anaconda3\Lib\site-packages\joblib\parallel.py:1088, in Parallel.__call__
(self, iterable)
   1085 if self.dispatch_one_batch(iterator):
   1086     self._iterating = self._original_iterator is not None
-> 1088 while self.dispatch_one_batch(iterator):
   1089     pass
   1091 if pre_dispatch == "all" or n_jobs == 1:
   1092     # The iterable was consumed all at once by the above for loop.
   1093     # No need to wait for async callbacks to trigger to
   1094     # consumption.

File ~\anaconda3\Lib\site-packages\joblib\parallel.py:901, in Parallel.dispatch_o
ne_batch(self, iterator)
    899     return False
    900 else:
--> 901     self._dispatch(tasks)
    902     return True

File ~\anaconda3\Lib\site-packages\joblib\parallel.py:819, in Parallel._dispatch
(self, batch)
    817 with self._lock:
    818     job_idx = len(self._jobs)
--> 819     job = self._backend.apply_async(batch, callback=cb)
    820     # A job can complete so quickly than its callback is
    821     # called before we get here, causing self._jobs to
    822     # grow. To ensure correct results ordering, .insert is
    823     # used (rather than .append) in the following line
    824     self._jobs.insert(job_idx, job)

File ~\anaconda3\Lib\site-packages\joblib\_parallel_backends.py:208, in Sequentia
lBackend.apply_async(self, func, callback)
    206 def apply_async(self, func, callback=None):
    207     """Schedule a func to be run"""
--> 208     result = ImmediateResult(func)
    209     if callback:
```

```
    210         callback(result)

File ~\anaconda3\Lib\site-packages\joblib\_parallel_backends.py:597, in Immediate
Result.__init__(self, batch)
    594 def __init__(self, batch):
    595     # Don't delay the application, to avoid keeping the input
    596     # arguments in memory
--> 597     self.results = batch()

File ~\anaconda3\Lib\site-packages\joblib\parallel.py:288, in BatchedCalls.__call
__(self)
    284 def __call__(self):
    285     # Set the default nested backend to self._backend but do not set the
    286     # change the default number of processes to -1
    287     with parallel_backend(self._backend, n_jobs=self._n_jobs):
--> 288         return [func(*args, **kwargs)
    289                 for func, args, kwargs in self.items]

File ~\anaconda3\Lib\site-packages\joblib\parallel.py:288, in <listcomp>(.0)
    284 def __call__(self):
    285     # Set the default nested backend to self._backend but do not set the
    286     # change the default number of processes to -1
    287     with parallel_backend(self._backend, n_jobs=self._n_jobs):
--> 288         return [func(*args, **kwargs)
    289                 for func, args, kwargs in self.items]

File ~\anaconda3\Lib\site-packages\sklearn\utils\parallel.py:127, in _FuncWrappe
r.__call__(self, *args, **kwargs)
    125     config = {}
    126 with config_context(**config):
--> 127     return self.function(*args, **kwargs)

File ~\anaconda3\Lib\site-packages\sklearn\ensemble\_forest.py:188, in _parallel_
build_trees(tree, bootstrap, X, y, sample_weight, tree_idx, n_trees, verbose, cla
ss_weight, n_samples_bootstrap)
    185     elif class_weight == "balanced_subsample":
    186         curr_sample_weight *= compute_sample_weight("balanced", y, indice
s=indices)
--> 188     tree.fit(X, y, sample_weight=curr_sample_weight, check_input=False)
    189 else:
    190     tree.fit(X, y, sample_weight=sample_weight, check_input=False)

File ~\anaconda3\Lib\site-packages\sklearn\base.py:1151, in _fit_context.<locals
>.decorator.<locals>.wrapper(estimator, *args, **kwargs)
   1144     estimator._validate_params()
   1146 with config_context(
   1147     skip_parameter_validation=(
   1148         prefer_skip_nested_validation or global_skip_validation
   1149     )
   1150 ):
-> 1151     return fit_method(estimator, *args, **kwargs)

File ~\anaconda3\Lib\site-packages\sklearn\tree\_classes.py:959, in DecisionTreeC
lassifier.fit(self, X, y, sample_weight, check_input)
    928 @_fit_context(prefer_skip_nested_validation=True)
    929 def fit(self, X, y, sample_weight=None, check_input=True):
    930     """Build a decision tree classifier from the training set (X, y).
    931 
    932     Parameters
   (...)
```

```
    956         Fitted estimator.
    957     """
--> 959     super()._fit(
    960         X,
    961         y,
    962         sample_weight=sample_weight,
    963         check_input=check_input,
    964     )
    965     return self

File ~\anaconda3\Lib\site-packages\sklearn\tree\_classes.py:443, in BaseDecisionT
ree._fit(self, X, y, sample_weight, check_input, missing_values_in_feature_mask)
    432 else:
    433     builder = BestFirstTreeBuilder(
    434         splitter,
    435         min_samples_split,
   (...)
    440         self.min_impurity_decrease,
    441     )
--> 443 builder.build(self.tree_, X, y, sample_weight, missing_values_in_feature_
mask)
    445 if self.n_outputs_ == 1 and is_classifier(self):
    446     self.n_classes_ = self.n_classes_[0]

KeyboardInterrupt:
```

In [ ]:

```
pred = rfc_cv.predict(x_test)
```

In [56]:

```
print(classification_report(y_test,pred))
```

```
              precision    recall  f1-score   support

           1       0.30      0.86      0.45        37
           2       0.40      0.22      0.29        45
           3       0.00      0.00      0.00        17
           4       0.00      0.00      0.00        10
           5       0.00      0.00      0.00         8
           6       0.00      0.00      0.00        14
           7       0.21      0.94      0.35        18
           8       0.00      0.00      0.00        17
           9       0.25      0.06      0.09        18
          10       0.00      0.00      0.00        20
          11       0.00      0.00      0.00         4
          12       0.00      0.00      0.00         6
          13       0.00      0.00      0.00         4
          14       0.00      0.00      0.00         5
          15       0.00      0.00      0.00         4
          16       0.08      1.00      0.15         6
          17       0.00      0.00      0.00         3
          18       0.00      0.00      0.00         9
          19       0.00      0.00      0.00         5
          20       0.00      0.00      0.00         9
          21       0.00      0.00      0.00         1
          22       0.00      0.00      0.00         3
          23       0.00      0.00      0.00         5
          24       0.00      0.00      0.00         5
          25       0.00      0.00      0.00         5
          26       0.00      0.00      0.00         4
          27       0.00      0.00      0.00         2
          28       0.00      0.00      0.00         2
          29       0.00      0.00      0.00         8

    accuracy                           0.22       294
   macro avg       0.04      0.11      0.05       294
weighted avg       0.13      0.22      0.13       294
```

```
C:\Users\chatu\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:146
9: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to control
this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\chatu\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:146
9: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to control
this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\chatu\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:146
9: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to control
this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [58]:

```
from sklearn.ensemble import RandomForestClassifier
rf_model = RandomForestClassifier()
rf_model.fit(x_train, y_train)
```

```
Out[58]: ▾ RandomForestClassifier
         RandomForestClassifier()
```