In [2]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [3]:
```
file_path="C:\\Users\\Venkatesh\\TechnoHacks internship\\Data Files\\WA_Fn-Use
df=pd.read_csv(file_path)
df
```

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]:
```python
# To get first 5 rows of the data

df.head()
```

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	

5 rows × 35 columns

In [5]:
```
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 [6]:
```
# To get the number of rows and columns
df.shape
```

Out[6]:
```
(1470, 35)
```

In [7]:
```
# Get the column datatypes
df.dtypes
```

Out[7]:
```
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 [9]:
```python
df.select_dtypes('object')
```

Out[9]:

	Attrition	BusinessTravel	Department	EducationField	Gender	JobRole	MaritalStat
0	Yes	Travel_Rarely	Sales	Life Sciences	Female	Sales Executive	Sin
1	No	Travel_Frequently	Research & Development	Life Sciences	Male	Research Scientist	Marri
2	Yes	Travel_Rarely	Research & Development	Other	Male	Laboratory Technician	Sin
3	No	Travel_Frequently	Research & Development	Life Sciences	Female	Research Scientist	Marri
4	No	Travel_Rarely	Research & Development	Medical	Male	Laboratory Technician	Marri
...	...	...	...	...	...	...	
1465	No	Travel_Frequently	Research & Development	Medical	Male	Laboratory Technician	Marri
1466	No	Travel_Rarely	Research & Development	Medical	Male	Healthcare Representative	Marri
1467	No	Travel_Rarely	Research & Development	Life Sciences	Male	Manufacturing Director	Marri
1468	No	Travel_Frequently	Sales	Medical	Male	Sales Executive	Marri
1469	No	Travel_Rarely	Research & Development	Medical	Male	Laboratory Technician	Marri

1470 rows × 9 columns

In [10]:
```
df.select_dtypes('int64')
```

Out[10]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	Enviror
0	41	1102	1	2	1	1	
1	49	279	8	1	1	2	
2	37	1373	2	2	1	4	
3	33	1392	3	4	1	5	
4	27	591	2	1	1	7	
...	...	...	...	...	...	...	
1465	36	884	23	2	1	2061	
1466	39	613	6	1	1	2062	
1467	27	155	4	3	1	2064	
1468	49	1023	2	3	1	2065	
1469	34	628	8	3	1	2068	

1470 rows × 26 columns

In [18]:
```
# Get a count of the empty values for each column
df.isna().sum()
```

Out[18]:
```
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 [9]:
```
# check for any missing values in the data
df.isnull().values.any()
```

Out[9]:
```
False
```

In [10]:
```
df.describe()
```

Out[10]:
```
              Age     DailyRate  DistanceFromHome    Education  EmployeeCount  EmployeeNu
count  1470.000000  1470.000000       1470.000000  1470.000000         1470.0     1470.00
mean     36.923810   802.485714          9.192517     2.912925            1.0     1024.86
std       9.135373   403.509100          8.106864     1.024165            0.0      602.02
min      18.000000   102.000000          1.000000     1.000000            1.0        1.00
25%      30.000000   465.000000          2.000000     2.000000            1.0      491.25
50%      36.000000   802.000000          7.000000     3.000000            1.0     1020.50
75%      43.000000  1157.000000         14.000000     4.000000            1.0     1555.75
max      60.000000  1499.000000         29.000000     5.000000            1.0     2068.00

8 rows × 26 columns
```

In [16]:
```
df.select_dtypes('object').describe()
```

Out[16]:
```
        Attrition  BusinessTravel    Department  EducationField  Gender     JobRole  MaritalStatus
count        1470            1470          1470            1470    1470        1470           1470
unique          2               3             3               6       2           9              3
top            No   Travel_Rarely    Research &   Life Sciences    Male       Sales        Married
                                    Development                           Executive
freq         1233            1043           961             606     882         326            673
```

In [14]:
```
df.select_dtypes('int64').describe()
```

Out[14]:
```
              Age     DailyRate  DistanceFromHome    Education  EmployeeCount  EmployeeNu
count  1470.000000  1470.000000       1470.000000  1470.000000         1470.0     1470.00
mean     36.923810   802.485714          9.192517     2.912925            1.0     1024.86
std       9.135373   403.509100          8.106864     1.024165            0.0      602.02
min      18.000000   102.000000          1.000000     1.000000            1.0        1.00
25%      30.000000   465.000000          2.000000     2.000000            1.0      491.25
50%      36.000000   802.000000          7.000000     3.000000            1.0     1020.50
75%      43.000000  1157.000000         14.000000     4.000000            1.0     1555.75
max      60.000000  1499.000000         29.000000     5.000000            1.0     2068.00

8 rows × 26 columns
```

In [48]:
```
# Drop is used to remove
# axis : 1 means it will drop the column
# axis : 0 means it will drop the row
df1=df.drop(['Over18', 'EmployeeNumber','EmployeeCount','StandardHours'],axis=
df1
```

Out[48]:

	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 × 31 columns

In [22]:
```
# To get all the columns in the data
df.columns
```

Out[22]:
```
Index(['Age', 'Attrition', 'BusinessTravel', 'DailyRate', 'Department',
       'DistanceFromHome', 'Education', 'EducationField', 'EmployeeCount',
       'EmployeeNumber', 'EnvironmentSatisfaction', 'Gender', 'HourlyRate',
       'JobInvolvement', 'JobLevel', 'JobRole', 'JobSatisfaction',
       'MaritalStatus', 'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked',
       'Over18', 'OverTime', 'PercentSalaryHike', 'PerformanceRating',
       'RelationshipSatisfaction', 'StandardHours', 'StockOptionLevel',
       'TotalWorkingYears', 'TrainingTimesLastYear', 'WorkLifeBalance',
       'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion',
       'YearsWithCurrManager'],
      dtype='object')
```

In [25]:
```
# To get all the categorical column names in the data
df.select_dtypes('object').columns
```

Out[25]:
```
Index(['Attrition', 'BusinessTravel', 'Department', 'EducationField', 'Gende
r',
       'JobRole', 'MaritalStatus', 'Over18', 'OverTime'],
      dtype='object')
```

In [26]:
```
# To get all the numerical column names in the data
df.select_dtypes('int64').columns
```

Out[26]:
```
Index(['Age', 'DailyRate', 'DistanceFromHome', 'Education', 'EmployeeCount',
       'EmployeeNumber', 'EnvironmentSatisfaction', 'HourlyRate',
       'JobInvolvement', 'JobLevel', 'JobSatisfaction', 'MonthlyIncome',
       'MonthlyRate', 'NumCompaniesWorked', 'PercentSalaryHike',
       'PerformanceRating', 'RelationshipSatisfaction', 'StandardHours',
       'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear',
       'WorkLifeBalance', 'YearsAtCompany', 'YearsInCurrentRole',
       'YearsSinceLastPromotion', 'YearsWithCurrManager'],
      dtype='object')
```

In [27]:
```
# get a count of the number of employees that stayed and left the company
df['Attrition'].value_counts()
```

Out[27]:
```
Attrition
No     1233
Yes     237
Name: count, dtype: int64
```

In [28]:
```
count=df['Attrition'].value_counts().keys()
values=df['Attrition'].value_counts().to_list()
Attrition_df=pd.DataFrame(zip(count,values),columns=['Attrition','count'])
Attrition_df
```

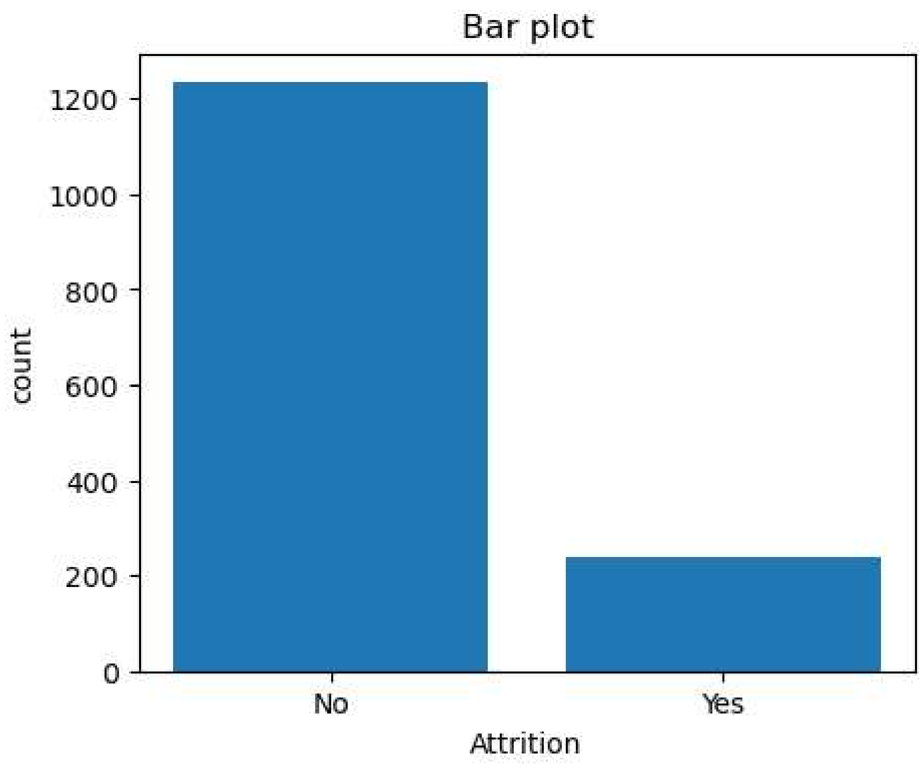
Out[28]:

	Attrition	count
0	No	1233
1	Yes	237

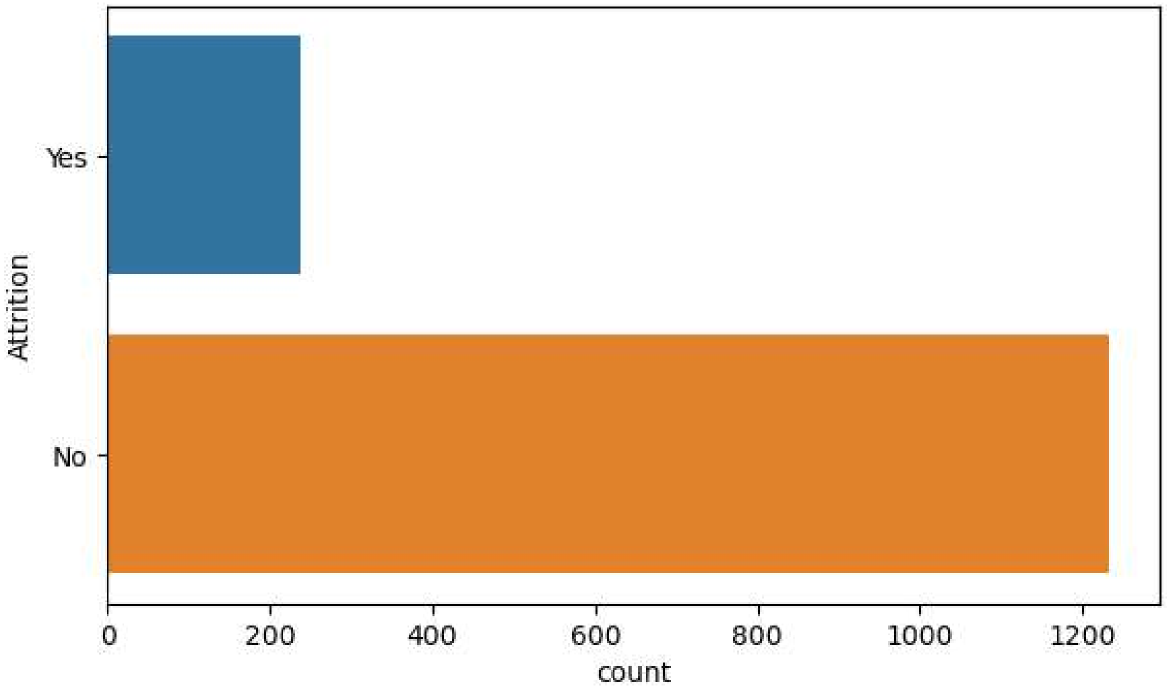
In [29]:
```
for i in df:
    if dict(df.dtypes)[i]=='object':
        count=df[i].value_counts().keys()
        values=df[i].value_counts().to_list()
        print(pd.DataFrame(zip(count,values),columns=[i,'count']))
        print("-----------------------------------")
```

```
  Attrition  count
0        No   1233
1       Yes    237
-----------------------------------
      BusinessTravel  count
0      Travel_Rarely   1043
1  Travel_Frequently    277
2         Non-Travel    150
-----------------------------------
               Department  count
0  Research & Development    961
1                   Sales    446
2         Human Resources     63
-----------------------------------
     EducationField  count
0     Life Sciences    606
1           Medical    464
2         Marketing    159
3  Technical Degree    132
4             Other     82
5   Human Resources     27
-----------------------------------
   Gender  count
0    Male    882
1  Female    588
-----------------------------------
                     JobRole  count
0            Sales Executive    326
1         Research Scientist    292
2      Laboratory Technician    259
3     Manufacturing Director    145
4  Healthcare Representative    131
5                    Manager    102
6       Sales Representative     83
7          Research Director     80
8            Human Resources     52
-----------------------------------
  MaritalStatus  count
0       Married    673
1        Single    470
2      Divorced    327
-----------------------------------
  Over18  count
0      Y   1470
-----------------------------------
  OverTime  count
0       No   1054
1      Yes    416
-----------------------------------
```

In [13]:
```
plt.figure(figsize=(5,4))
plt.title('Bar plot')
plt.xlabel('Attrition')
plt.ylabel('count')
plt.bar('Attrition','count',data=Attrition_df)

plt.show()
```



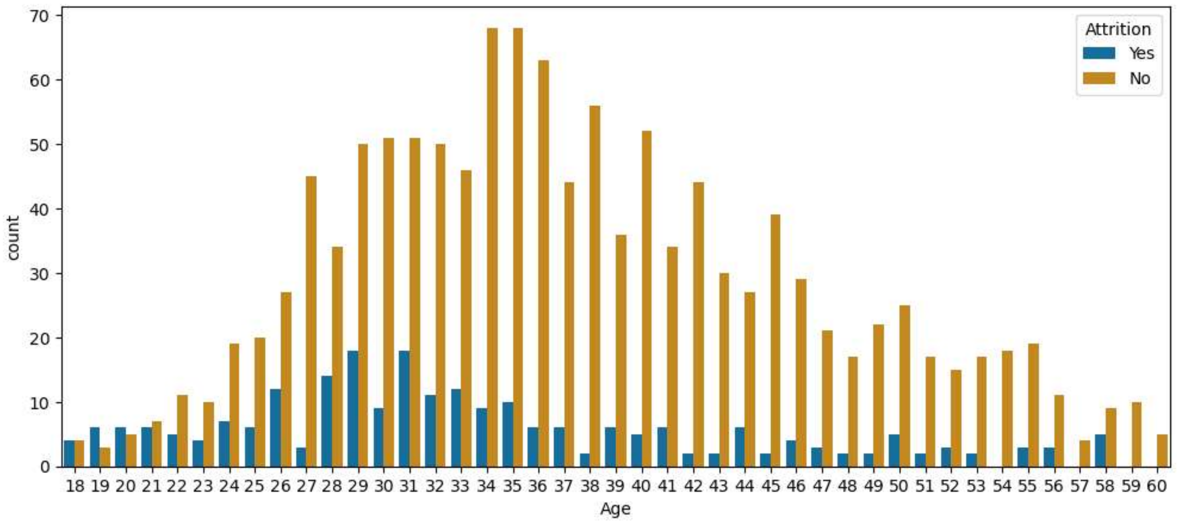
In [14]:
```
plt.figure(figsize=(7,4))
sns.countplot(y='Attrition',data=df)

plt.show()
```



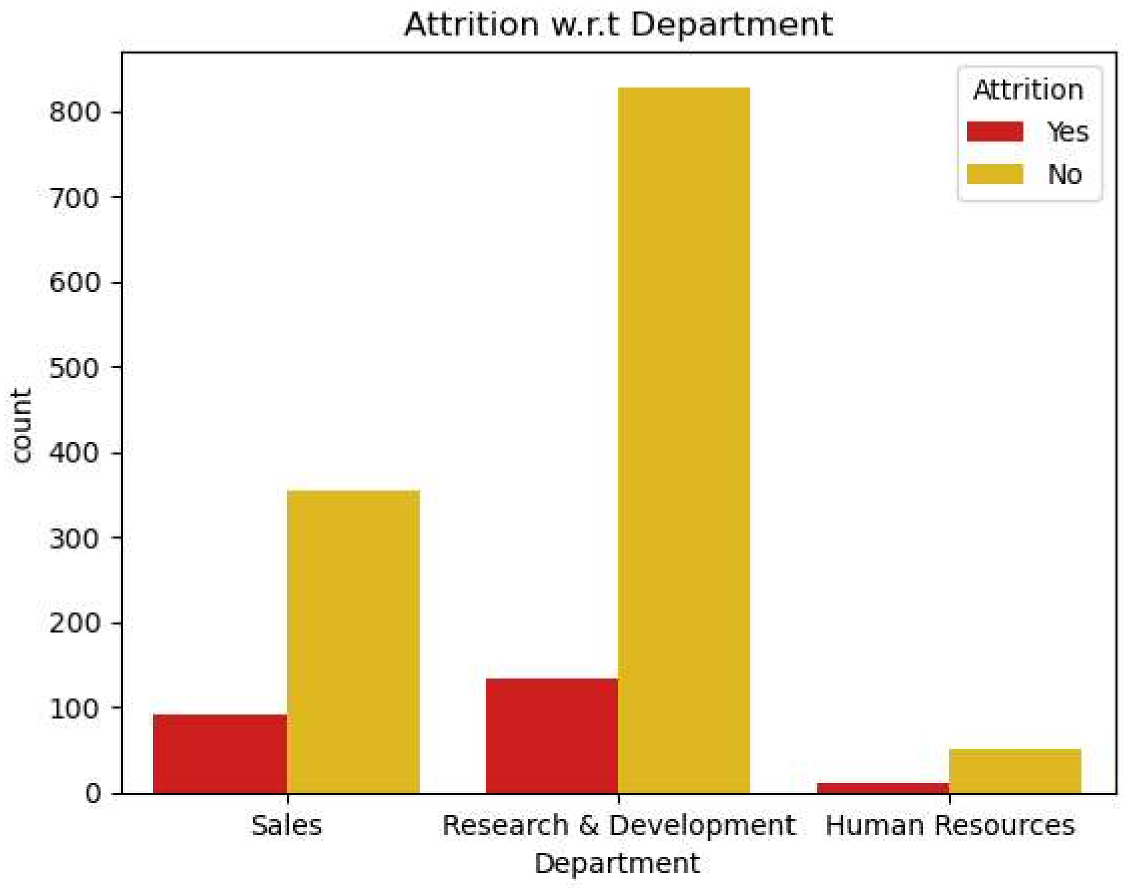
In [15]:
```
(1233 - 237) / 1233
```

Out[15]:
```
0.8077858880778589
```

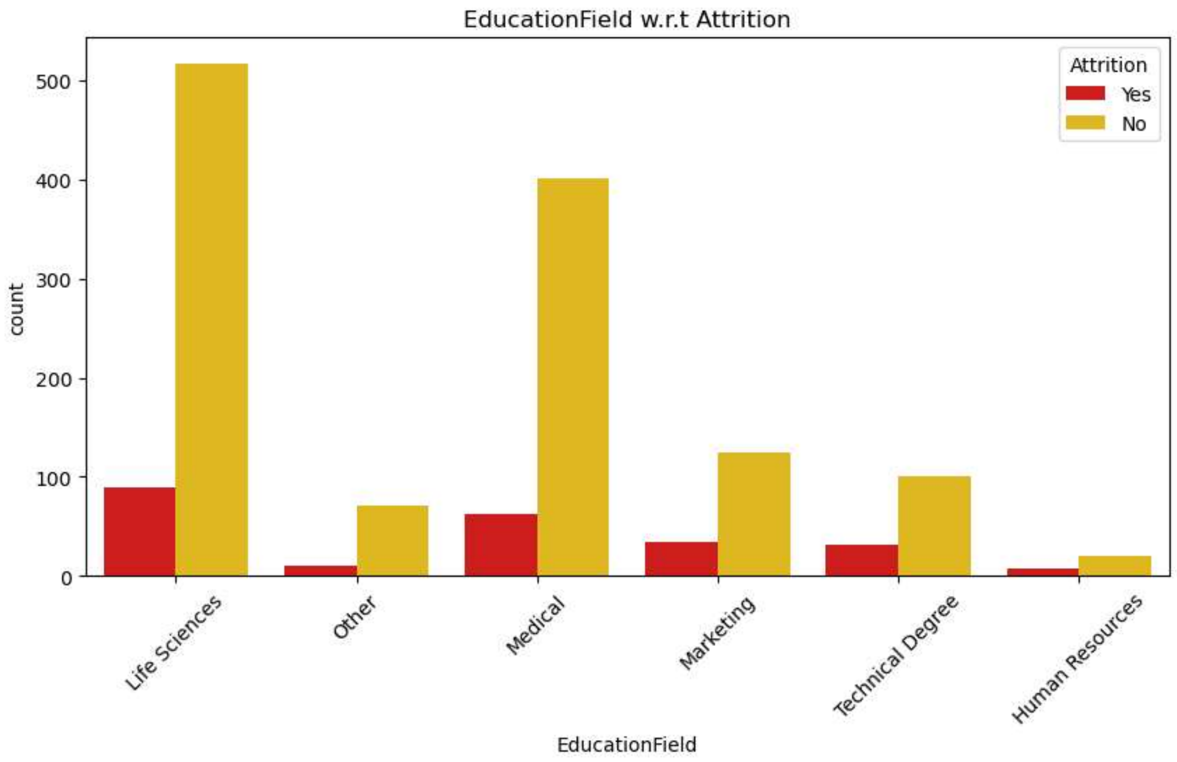
In [30]:
```
# show the number of employees that left and stayed by age
plt.figure(figsize=(12,5))
sns.countplot(x='Age',hue='Attrition',data=df,palette='colorblind')

plt.show()
```



In [32]:
```
plt.title('Attrition w.r.t Department')
sns.countplot(x='Department',hue='Attrition',data=df,palette='hot')

plt.show()
```



In [35]:
```
plt.figure(figsize=(10,5))
plt.title('EducationField w.r.t Attrition')
sns.countplot(x='EducationField',hue='Attrition',data=df,palette='hot')
plt.xticks(rotation=45)
plt.show()
```

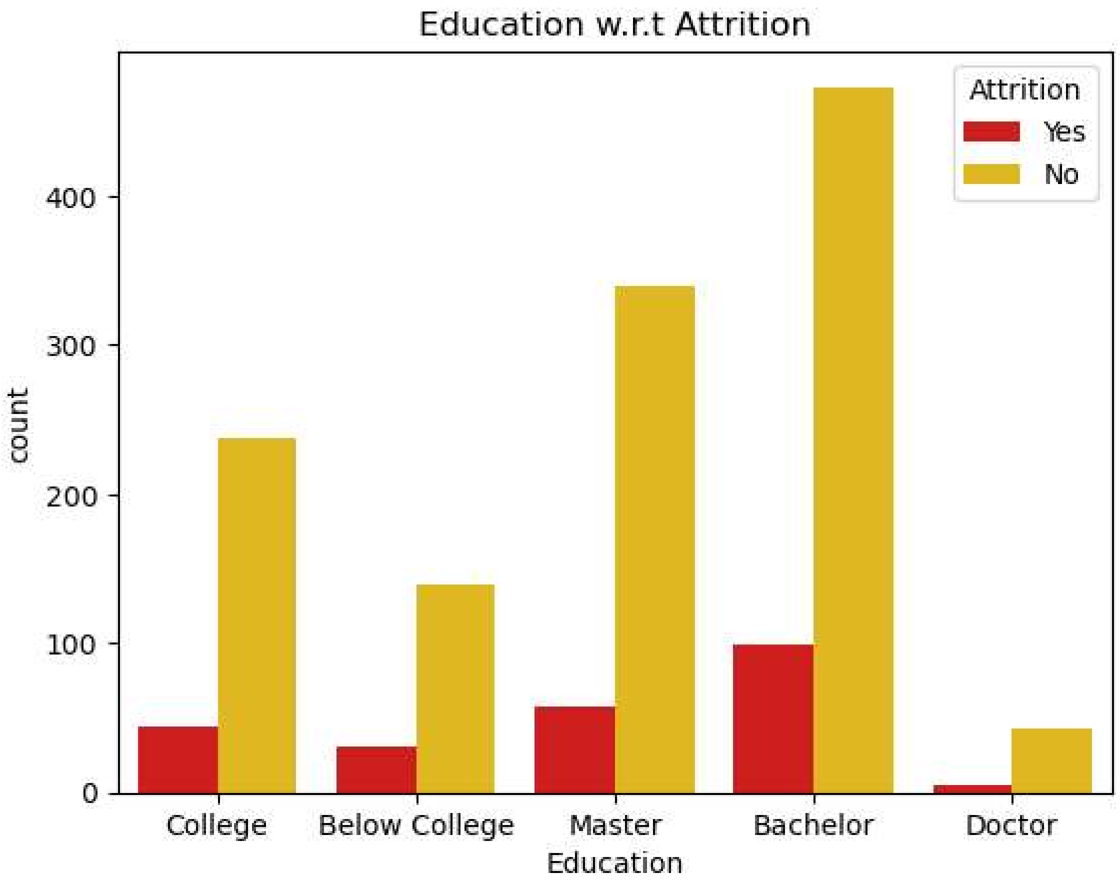


In [186]:
```
ordinal_features = ['Education','EnvironmentSatisfaction',
                    'JobInvolvement','JobSatisfaction',
                    'PerformanceRating','RelationshipSatisfaction',
                    'WorkLifeBalance']
df[ordinal_features].head()
```

Out[186]:

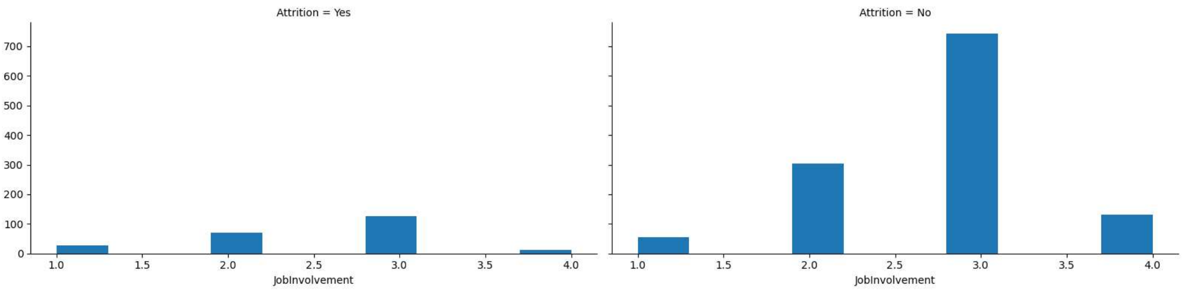
	Education	EnvironmentSatisfaction	JobInvolvement	JobSatisfaction	PerformanceRating	Rela
0	2	2	3	4	3	
1	1	3	2	2	4	
2	2	4	2	3	3	
3	4	4	3	3	3	
4	1	1	3	2	3	

In [50]:
```
edu_map={1:'Below College',2:'College',3:'Bachelor',4:'Master',5:'Doctor'}
plt.title('Education w.r.t Attrition')
sns.countplot(x=df['Education'].map(edu_map),hue='Attrition',data=df,palette='

plt.show()
```



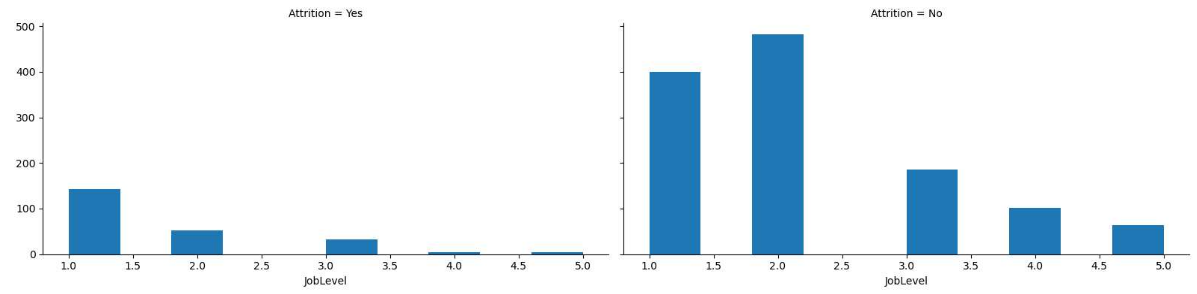
In [57]:
```
n=sns.FacetGrid(df1, col='Attrition', height=4, aspect=2)
n.map(plt.hist,'JobInvolvement')

plt.show()
```

```
C:\Users\Venkatesh\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserW
arning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



In [58]:
```
n= sns.FacetGrid(df1, col='Attrition', height=4, aspect=2)
n.map(plt.hist, 'JobLevel')

plt.show()
```

```
C:\Users\Venkatesh\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserW
arning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



percentile-quantile

In [81]:
```
# you can find mean value using pandas
mean_rate=df['DailyRate'].mean()
round(mean_rate,2)
```

Out[81]:
```
802.49
```

In [99]:
```
# you can also find mean by using numpy
mean_rate=np.mean(df['DailyRate'])
median_rate=np.median(df['DailyRate'])
min_rate=np.min(df['DailyRate'])
max_rate=np.max(df['DailyRate'])
std_rate=np.std(df['DailyRate'])

list1=[mean_rate,median_rate,min_rate,max_rate,std_rate]
index=['Mean','Median','Min','Max','Std']
pd.DataFrame(list1,columns=['DailyRate'],index=index)
```

Out[99]:

	DailyRate
Mean	802.485714
Median	802.000000
Min	102.000000
Max	1499.000000
Std	403.371829

In [94]:
```
per_25=np.percentile(df['DailyRate'],25)
per_50=np.percentile(df['DailyRate'],50)
per_75=np.percentile(df['DailyRate'],75)
print(per_25,per_50,per_75)
```

```
465.0 802.0 1157.0
```

In [96]:
```
round(np.quantile(df['DailyRate'],0.50),2)
```

Out[96]:
```
802.0
```

In [98]:
```
mean_rate=np.mean(df['DailyRate'])
median_rate=np.median(df['DailyRate'])
min_rate=np.min(df['DailyRate'])
max_rate=np.max(df['DailyRate'])
std_rate=np.std(df['DailyRate'])

list1=[mean_rate,median_rate,min_rate,max_rate,std_rate,per_25,per_50,per_75]
index=['Mean','Median','Min','Max','Std','25%','50%','75%']
pd.DataFrame(list1,columns=['DailyRate'],index=index)
```

Out[98]:

	DailyRate
Mean	802.485714
Median	802.000000
Min	102.000000
Max	1499.000000
Std	403.371829
25%	465.000000
50%	802.000000
75%	1157.000000

emperical rule

In [101]:
```
################### u-1*sigma to u+1*sigma #################
val_minus_1_sigma=mean_rate-1*std_rate
val_plus_1_sigma=mean_rate+1*std_rate

################### u-2*sigma to u+2*sigma #################
val_minus_2_sigma=mean_rate-2*std_rate
val_plus_2_sigma=mean_rate+2*std_rate

################### u-3*sigma to u+3*sigma #################
val_minus_3_sigma=mean_rate-3*std_rate
val_plus_3_sigma=mean_rate+3*std_rate
```

In [102]:
```
df['DailyRate']
```

Out[102]:
```
0       1102
1        279
2       1373
3       1392
4        591
        ... 
1465     884
1466     613
1467     155
1468    1023
1469     628
Name: DailyRate, Length: 1470, dtype: int64
```

In [103]:
```
df['DailyRate']<465
```

Out[103]:
```
0       False
1        True
2       False
3       False
4       False
        ...  
1465    False
1466    False
1467     True
1468    False
1469    False
Name: DailyRate, Length: 1470, dtype: bool
```

In [105]:
```
# if you want to get only data has DailyRate has 465
# i want to retrieve a true values
cond=df['DailyRate']<465
df[cond]
```

Out[105]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	Educ
1	49	No	Travel_Frequently	279	Research & Development	8	1	Lif
8	38	No	Travel_Frequently	216	Research & Development	23	3	Lif
11	29	No	Travel_Rarely	153	Research & Development	15	2	Lif
14	28	Yes	Travel_Rarely	103	Research & Development	24	3	Lif
16	32	No	Travel_Rarely	334	Research & Development	5	2	Lif
...	...	...	...	...	...	...	...	
454	45	No	Travel_Rarely	374	Sales	20	3	Lif
458	35	No	Travel_Rarely	287	Research & Development	1	4	Lif
461	50	Yes	Travel_Rarely	410	Sales	28	3	
463	31	No	Non-Travel	325	Research & Development	5	3	
467	27	No	Travel_Rarely	155	Research & Development	4	3	Lif

5 rows × 35 columns

In [117]:
```
val_minus_1_sigma,val_plus_1_sigma
```

Out[117]:
```
(399.1138856848022, 1205.8575428866263)
```

In [110]:
```
cond1=df['DailyRate']>val_minus_1_sigma
cond2=df['DailyRate']<val_plus_1_sigma
len(df[cond1&cond2])
```

Out[110]:
```
850
```

In [118]:
```
val_minus_2_sigma,val_plus_2_sigma
```

Out[118]:
```
(-4.257942916109869, 1609.2293714875384)
```

In [116]:
```
cond1=df['DailyRate']>val_minus_2_sigma
cond2=df['DailyRate']<val_plus_2_sigma
len(df[cond1&cond2])
```

Out[116]:
```
1470
```

In [119]:
```
val_minus_3_sigma,val_plus_3_sigma
```

Out[119]:
```
(-407.62977151702194, 2012.6012000884505)
```

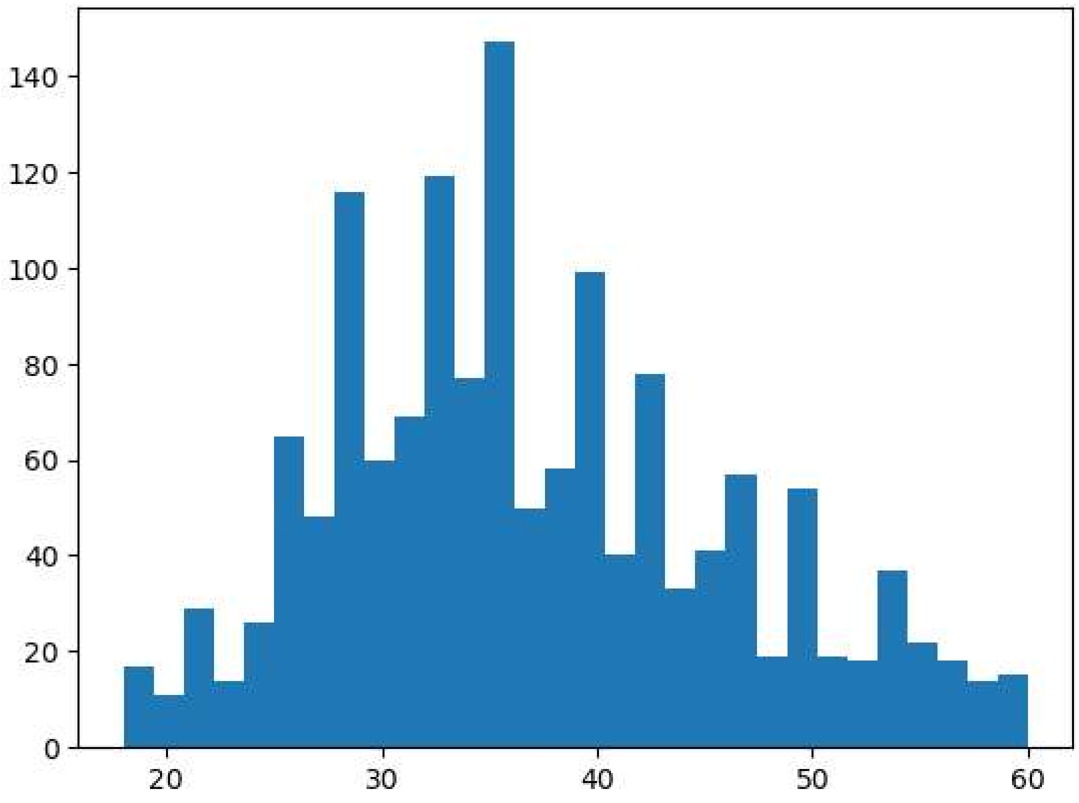
In [121]:
```
cond1=df['DailyRate']>val_minus_3_sigma
cond2=df['DailyRate']<val_plus_3_sigma
len(df[cond1&cond2])
```

Out[121]:
```
1470
```

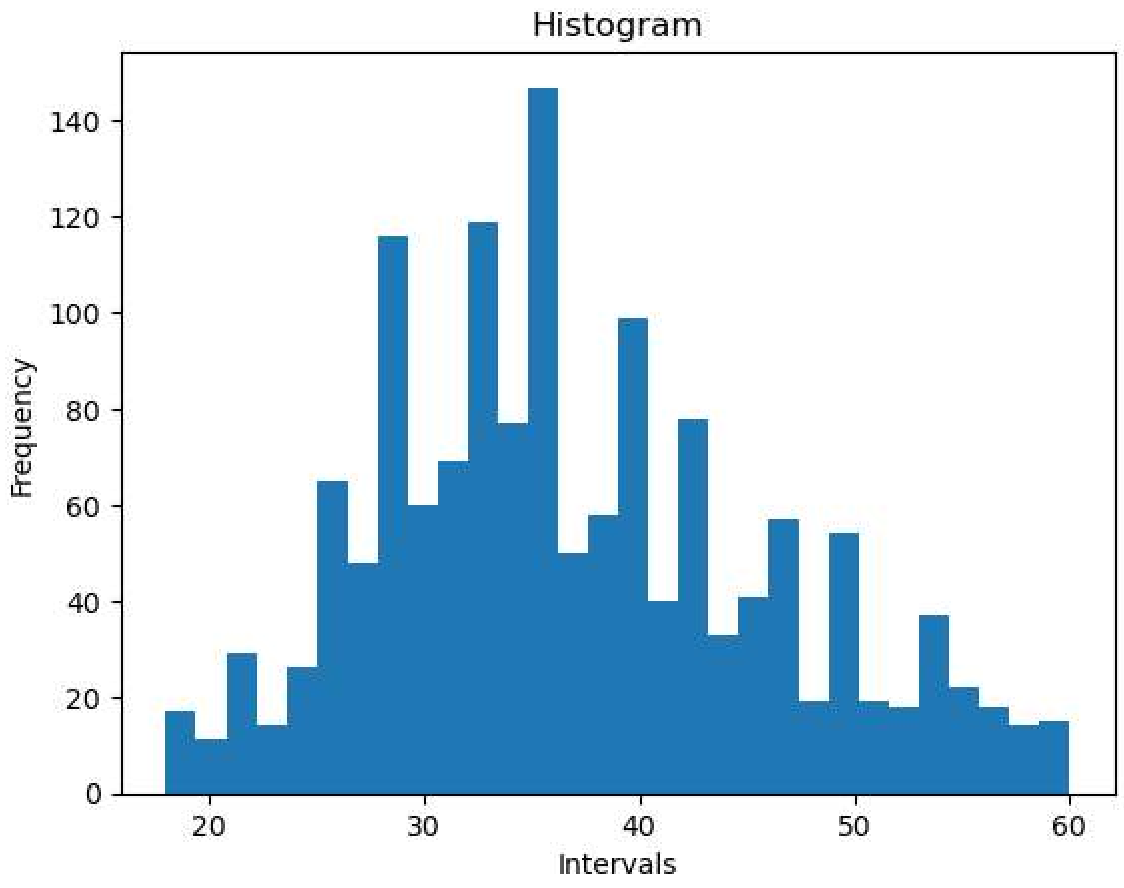
histogram

In [150]:
```
data=df['Age']
plt.hist(data,bins=30)
```

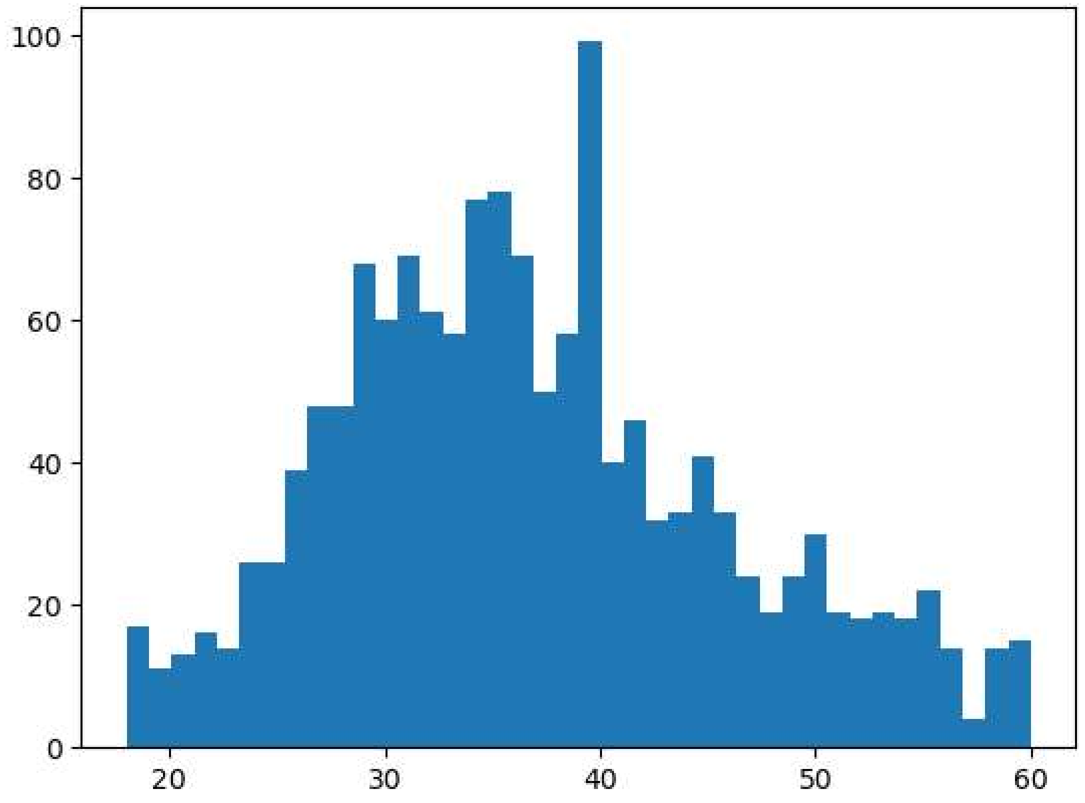
Out[150]:
```
(array([ 17.,  11.,  29.,  14.,  26.,  65.,  48., 116.,  60.,  69., 119.,
         77., 147.,  50.,  58.,  99.,  40.,  78.,  33.,  41.,  57.,  19.,
         54.,  19.,  18.,  37.,  22.,  18.,  14.,  15.]),
 array([18. , 19.4, 20.8, 22.2, 23.6, 25. , 26.4, 27.8, 29.2, 30.6, 32. ,
        33.4, 34.8, 36.2, 37.6, 39. , 40.4, 41.8, 43.2, 44.6, 46. , 47.4,
        48.8, 50.2, 51.6, 53. , 54.4, 55.8, 57.2, 58.6, 60. ]),
 <BarContainer object of 30 artists>)
```



In [151]:
```
data=df['Age']
plt.hist(data,bins=30)
plt.title('Histogram')
plt.xlabel('Intervals')
plt.ylabel('Frequency')
plt.show()
```



In [152]:
```
frequency,interval,n=plt.hist(data,bins=40)

# returning 3 values

# print(frequency)
# print(interval)
# print(n)
```

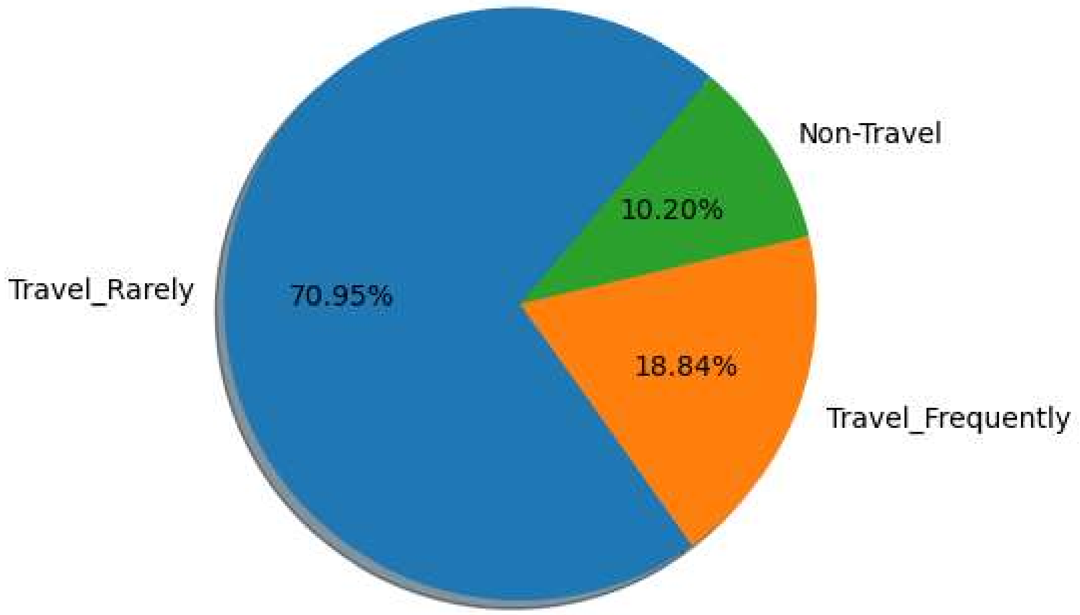


pie-chart

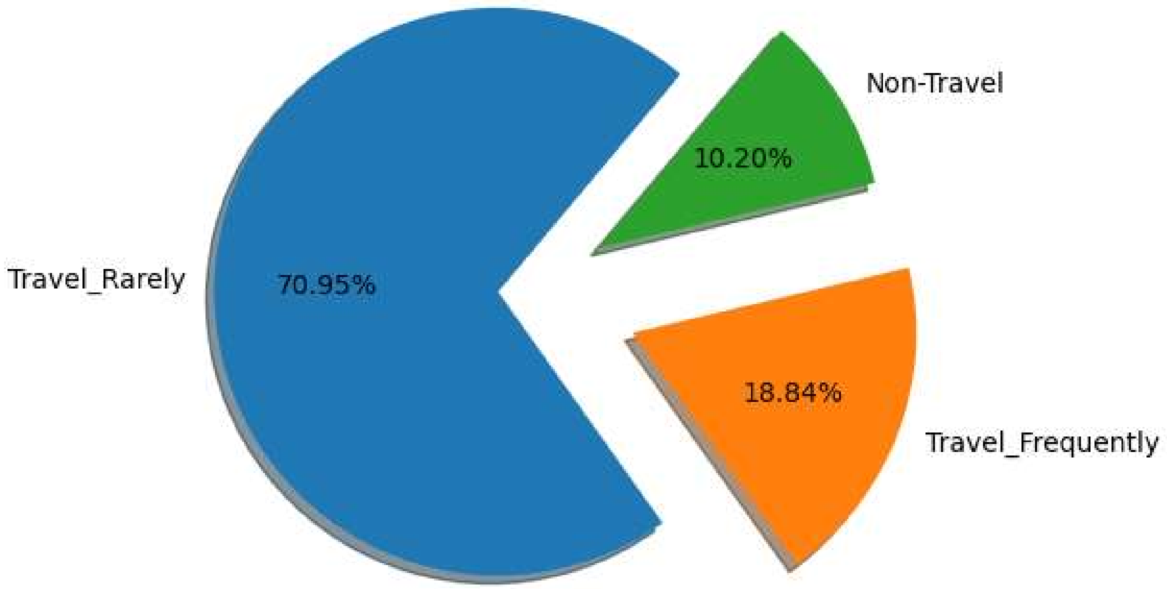
In [178]:
```
values=df['BusinessTravel'].value_counts().values.tolist()
names=df['BusinessTravel'].value_counts().keys().tolist()
values,names
```

Out[178]:
```
([1043, 277, 150], ['Travel_Rarely', 'Travel_Frequently', 'Non-Travel'])
```

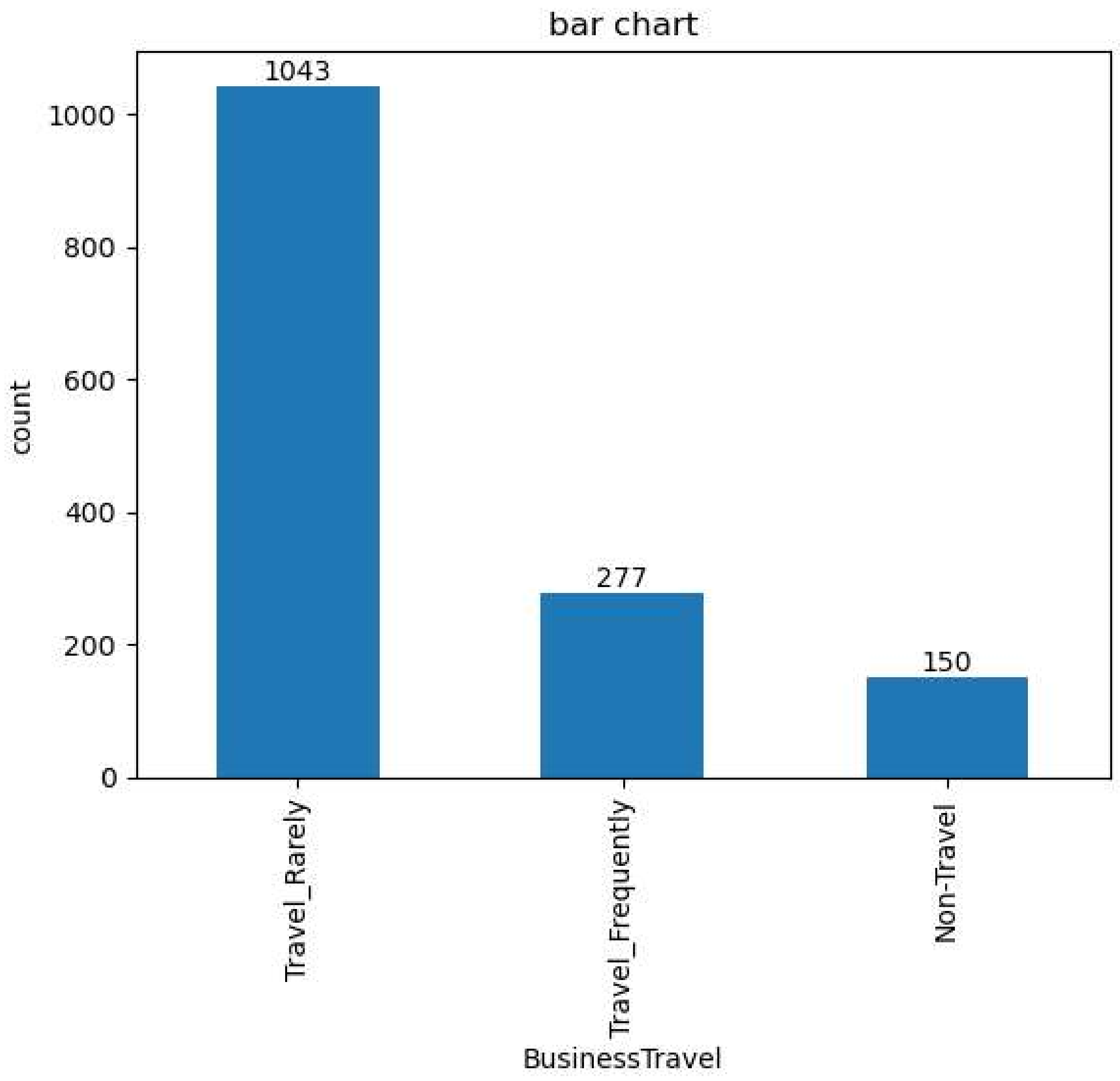
In [183]:
```
plt.pie(x=values,
        labels=names,
        autopct="%0.2f%%",
        shadow=True,
        startangle=50)

plt.show()
```



In [184]:
```
plt.pie(x=values,
        labels=names,
        autopct="%0.2f%%",
        shadow=True,
        startangle=50,
        explode=[0.1,0.4,0.3])

plt.show()
```



In [185]:
```
value=df['BusinessTravel'].value_counts()
ax=value.plot(kind='bar')
ax.bar_label(ax.containers[0])
plt.title('bar chart')
plt.ylabel('count')
plt.show()
```



In [ ]:
```
```