In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
from patsy import dmatrices
import sklearn
import seaborn as sns
```

In [2]:
```
#import dataset
df_IBM = pd.read_csv("IBM Attrition Data.csv")
```

In [3]:
```
df_IBM.head(5)
```

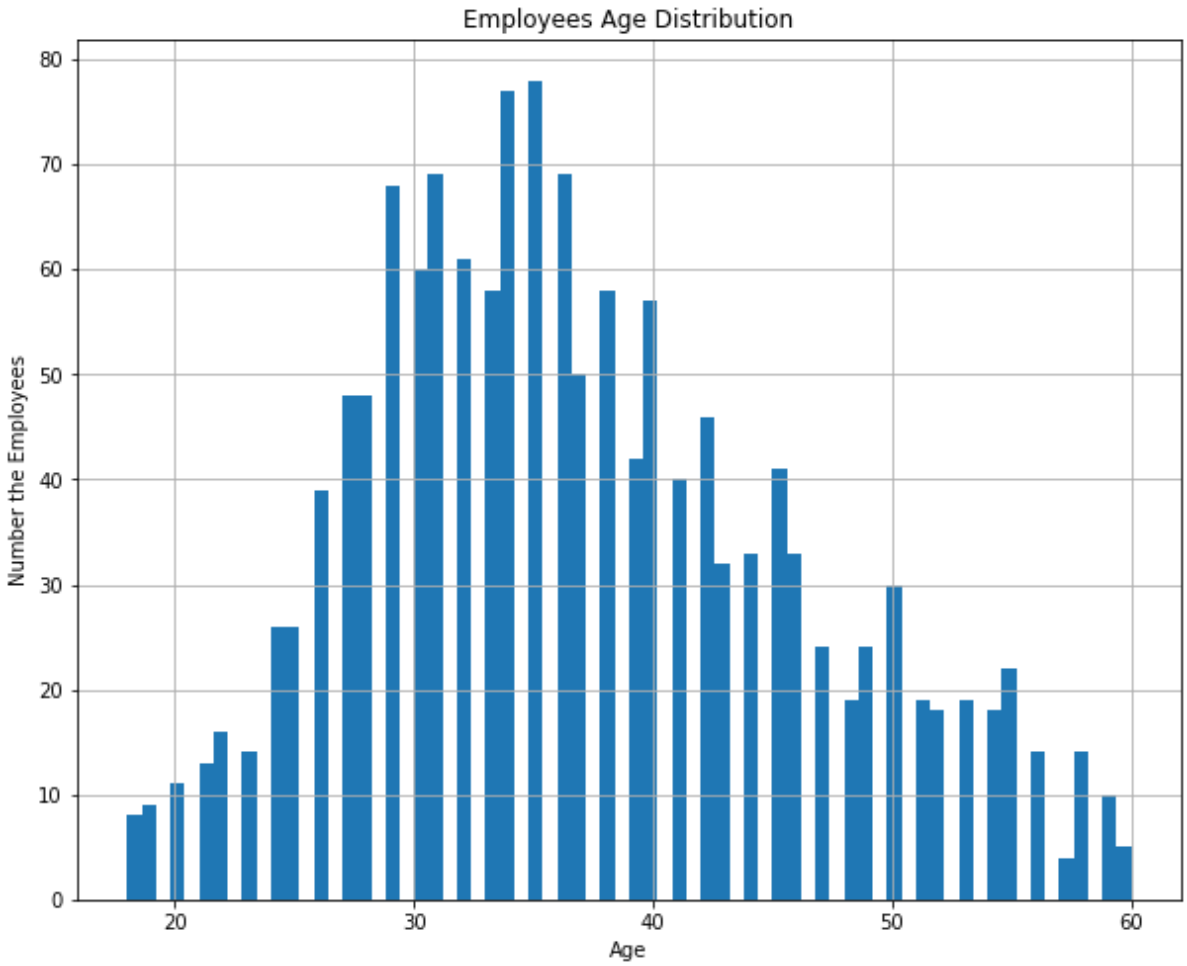
Out[3]:

	Age	Attrition	Department	DistanceFromHome	Education	EducationField	EnvironmentSatisf
0	41	Yes	Sales	1	2	Life Sciences	
1	49	No	Research & Development	8	1	Life Sciences	
2	37	Yes	Research & Development	2	2	Other	
3	33	No	Research & Development	3	4	Life Sciences	
4	27	No	Research & Development	2	1	Medical	

In [4]:
```
names = df_IBM.columns.values
print(names)
```

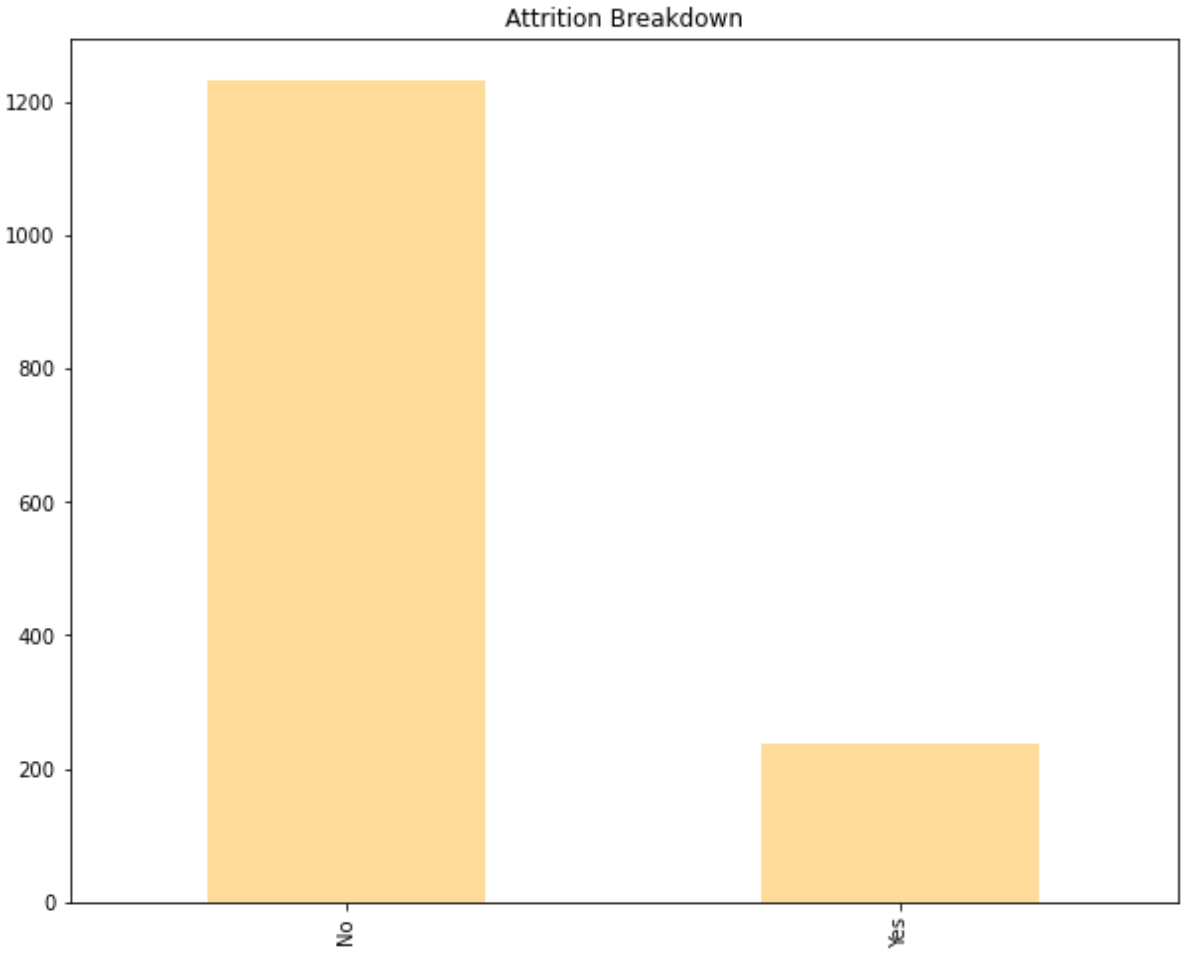
```
['Age' 'Attrition' 'Department' 'DistanceFromHome' 'Education'
 'EducationField' 'EnvironmentSatisfaction' 'JobSatisfaction'
 'MaritalStatus' 'MonthlyIncome' 'NumCompaniesWorked' 'WorkLifeBalance'
 'YearsAtCompany']
```

In [5]:
```
# histogram Age distribution
plt.figure(figsize= (10,8))
df_IBM['Age'].hist(bins=70)
plt.title("Employees Age Distribution")
plt.xlabel("Age")
plt.ylabel("Number the Employees")
plt.show()
```

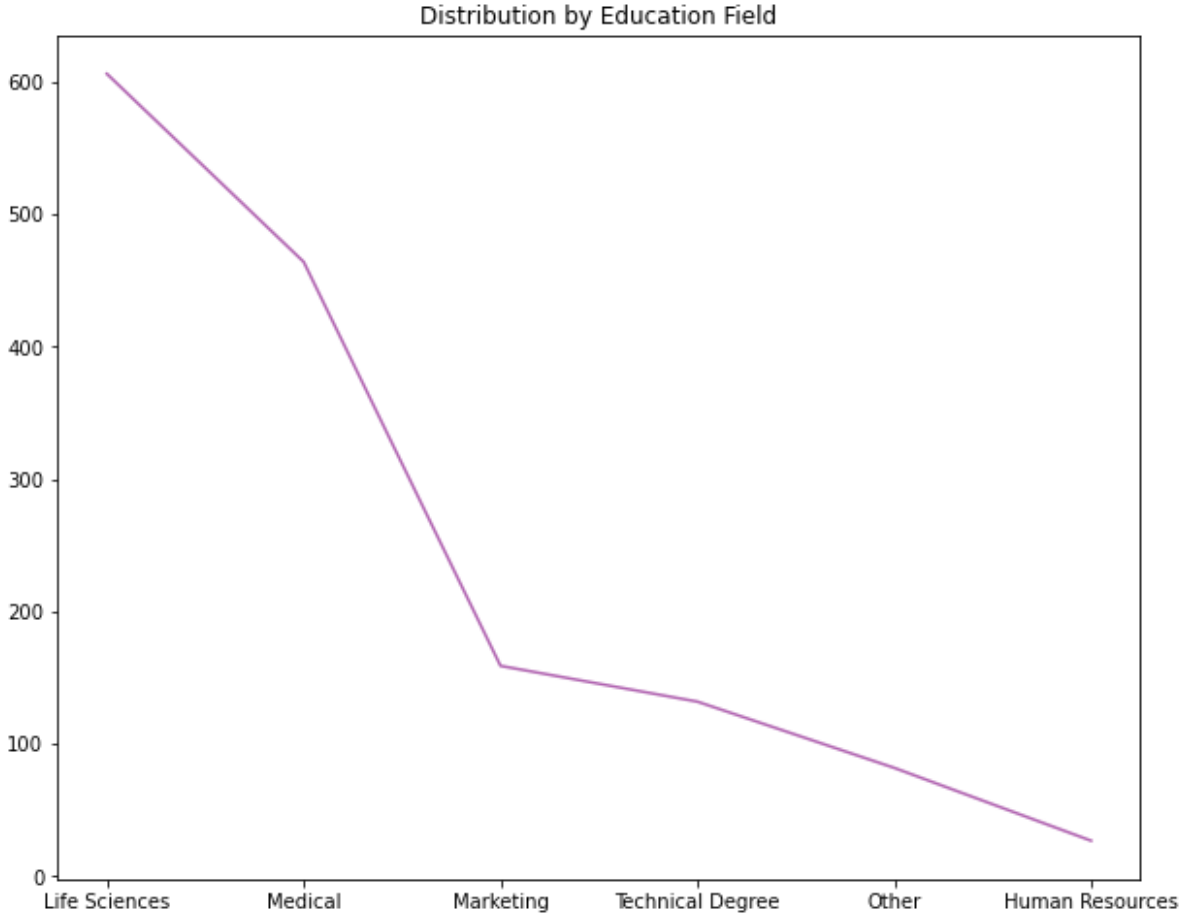


In [6]:
```
# Explore attrition by age (alpha =transparency)
plt.figure(figsize=(12,8))
plt.scatter(df_IBM.Attrition,df_IBM.Age,alpha=.4)
plt.title("Attrition by Age")
plt.ylabel("Age")
plt.grid(b = True, which='major',axis='y')
plt.show()
```

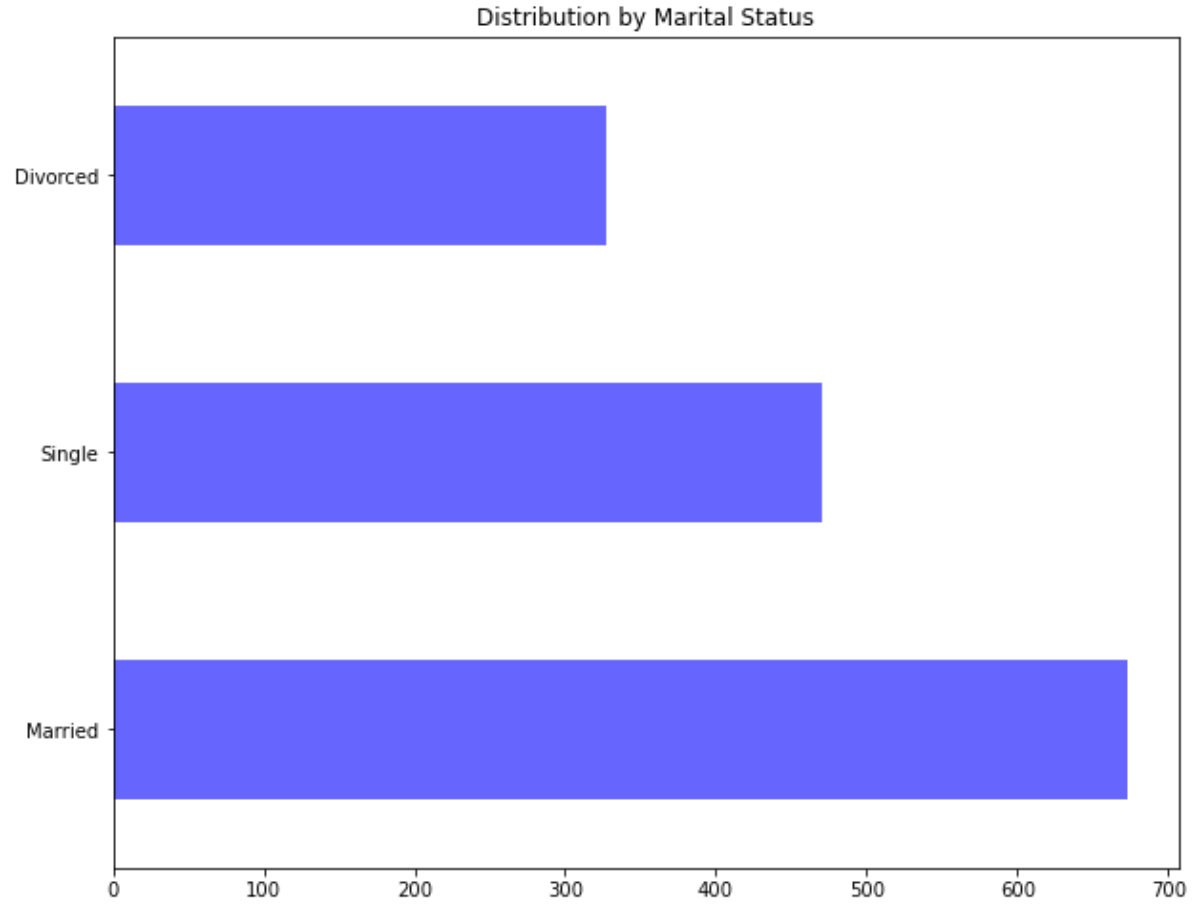
In [7]:
```
#Explore data for Left employees
plt.figure(figsize=(10,8))
df_IBM.Attrition.value_counts().plot(kind='bar',color ='orange',alpha=.4)
plt.title("Attrition Breakdown")
plt.show()
```



In [8]:
```
#Find out the distribution of employees by the education field
plt.figure(figsize=(10,8))
df_IBM.EducationField.value_counts().plot(kind='line',color ='purple',alpha=.6
)
plt.title("Distribution by Education Field")
plt.show()
```



In [9]:
```
#Give a bar chart for the number of married and unmarried employees
plt.figure(figsize=(10,8))
df_IBM.MaritalStatus.value_counts().plot(kind='barh',color ='blue',alpha=.6)
plt.title("Distribution by Marital Status")
plt.show()
```



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

Out[10]:

	Age	DistanceFromHome	Education	EnvironmentSatisfaction	JobSatisfaction	M
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	
mean	36.923810	9.192517	2.912925	2.721769	2.728571	
std	9.135373	8.106864	1.024165	1.093082	1.102846	
min	18.000000	1.000000	1.000000	1.000000	1.000000	
25%	30.000000	2.000000	2.000000	2.000000	2.000000	
50%	36.000000	7.000000	3.000000	3.000000	3.000000	
75%	43.000000	14.000000	4.000000	4.000000	4.000000	
max	60.000000	29.000000	5.000000	4.000000	4.000000	

In [11]:
```
df_IBM.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 13 columns):
 #   Column                   Non-Null Count  Dtype 
---  ------                   --------------  ----- 
 0   Age                      1470 non-null   int64 
 1   Attrition                1470 non-null   object
 2   Department               1470 non-null   object
 3   DistanceFromHome         1470 non-null   int64 
 4   Education                1470 non-null   int64 
 5   EducationField           1470 non-null   object
 6   EnvironmentSatisfaction  1470 non-null   int64 
 7   JobSatisfaction          1470 non-null   int64 
 8   MaritalStatus            1470 non-null   object
 9   MonthlyIncome            1470 non-null   int64 
 10  NumCompaniesWorked       1470 non-null   int64 
 11  WorkLifeBalance          1470 non-null   int64 
 12  YearsAtCompany           1470 non-null   int64 
dtypes: int64(9), object(4)
memory usage: 149.4+ KB
```

In [12]:
```
df_IBM.columns
```

Out[12]:
```
Index(['Age', 'Attrition', 'Department', 'DistanceFromHome', 'Education',
       'EducationField', 'EnvironmentSatisfaction', 'JobSatisfaction',
       'MaritalStatus', 'MonthlyIncome', 'NumCompaniesWorked',
       'WorkLifeBalance', 'YearsAtCompany'],
      dtype='object')
```

In [13]:
```
df_IBM.std()
```

Out[13]:
```
Age                          9.135373
DistanceFromHome             8.106864
Education                    1.024165
EnvironmentSatisfaction      1.093082
JobSatisfaction              1.102846
MonthlyIncome             4707.956783
NumCompaniesWorked           2.498009
WorkLifeBalance              0.706476
YearsAtCompany               6.126525
dtype: float64
```

In [14]:
```
#Find value counts for Attrition Values
df_IBM['Attrition'].value_counts()
```

Out[14]:
```
No     1233
Yes     237
Name: Attrition, dtype: int64
```

In [15]:
```
df_IBM['Attrition'].dtypes
```

Out[15]:
```
dtype('O')
```

In [16]:
```
df_IBM['Attrition'].replace('Yes',1,inplace=True)
df_IBM['Attrition'].replace('No',0,inplace=True)
```

In [17]:
```
df_IBM.head(5)
```

Out[17]:

	Age	Attrition	Department	DistanceFromHome	Education	EducationField	EnvironmentSatisf
0	41	1	Sales	1	2	Life Sciences	
1	49	0	Research & Development	8	1	Life Sciences	
2	37	1	Research & Development	2	2	Other	
3	33	0	Research & Development	3	4	Life Sciences	
4	27	0	Research & Development	2	1	Medical	

In [18]:
```
#Build up a logistic regression model to predict which employees are likely to attrite
X =df_IBM.drop(['Attrition'],axis=1)
X.head()
Y= df_IBM['Attrition']
Y.head()
```

Out[18]:
```
0    1
1    0
2    1
3    0
4    0
Name: Attrition, dtype: int64
```

In [19]:
```
df_IBM['EducationField'],replace('Life Sciences',1, inplace)
df_IBM['EducationField'],replace('Medical',2, inplace)
df_IBM['EducationField'],replace('Marketing',3, inplace)
df_IBM['EducationField'],replace('Other',4, inplace)
df_IBM['EducationField'],replace('Technical Degree',5, inplace)
df_IBM['EducationField'],replace('Human Resources',6, inplace)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-19-f447d13ed981> in <module>
----> 1 df_IBM['EducationField'],replace('Life Sciences',1, inplace)
      2 df_IBM['EducationField'],replace('Medical',2, inplace)
      3 df_IBM['EducationField'],replace('Marketing',3, inplace)
      4 df_IBM['EducationField'],replace('Other',4, inplace)
      5 df_IBM['EducationField'],replace('Technical Degree',5, inplace)

NameError: name 'replace' is not defined
```

In [20]:
```
df_IBM['EducationField'].value_counts()
```

Out[20]:
```
Life Sciences       606
Medical             464
Marketing           159
Technical Degree    132
Other                82
Human Resources      27
Name: EducationField, dtype: int64
```

In [22]:
```
df_IBM['Department'].value_counts()
```

Out[22]:
```
Research & Development    961
Sales                     446
Human Resources            63
Name: Department, dtype: int64
```

In [23]:
```
df_IBM['Department'],replace('Research & Development',1, inplace=True)
df_IBM['Department'],replace('Sales',2, inplace=True)
df_IBM['Department'],replace('Human Resources',3, inplace=True)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-23-a5c34ca945f2> in <module>
----> 1 df_IBM['Department'],replace('Research & Development',1, inplace=True
)
      2 df_IBM['Department'],replace('Sales',2, inplace=True)
      3 df_IBM['Department'],replace('Human Resources',3, inplace=True)

NameError: name 'replace' is not defined
```

In [24]:
```
df_IBM['Department'].value_counts()
```

Out[24]:
```
Research & Development    961
Sales                     446
Human Resources            63
Name: Department, dtype: int64
```

In [25]:
```
df_IBM['MaritalStatus'].value_counts()
```

Out[25]:
```
Married     673
Single      470
Divorced    327
Name: MaritalStatus, dtype: int64
```

In [ ]:
```
df_IBM['MaritalStatus'],replace('Married',1, inplace=True
df_IBM['MaritalStatus'],replace('Single',2, inplace=True
df_IBM['MaritalStatus'],replace('Divorced',3, inplace=True
```

In [26]:
```
df_IBM['MaritalStatus'].value_counts()
```

Out[26]:
```
Married     673
Single      470
Divorced    327
Name: MaritalStatus, dtype: int64
```

In [27]:
```
x=df_IBM.select_dtypes(include=['int64'])
x.dtypes
```

Out[27]:
```
Age                       int64
Attrition                 int64
DistanceFromHome          int64
Education                 int64
EnvironmentSatisfaction   int64
JobSatisfaction           int64
MonthlyIncome             int64
NumCompaniesWorked        int64
WorkLifeBalance           int64
YearsAtCompany            int64
dtype: object
```

In [28]:
```
x.columns
```

Out[28]:
```
Index(['Age', 'Attrition', 'DistanceFromHome', 'Education',
       'EnvironmentSatisfaction', 'JobSatisfaction', 'MonthlyIncome',
       'NumCompaniesWorked', 'WorkLifeBalance', 'YearsAtCompany'],
      dtype='object')
```

In [31]:
```
y=df_IBM['Attrition']
```

In [32]:
```
y.head()
```

Out[32]:
```
0    1
1    0
2    1
3    0
4    0
Name: Attrition, dtype: int64
```

In [44]:
```
y, x = dmatrices("Attrition ~ Age + DistanceFromHome + Education + Environment
Satisfaction + JobSatisfaction + MonthlyIncome + NumCompaniesWorked + WorkLife
Balance + YearsAtCompany", df_IBM, return_type="dataframe")
print (x.columns)
```

```
Index(['Intercept', 'Age', 'DistanceFromHome', 'Education',
       'EnvironmentSatisfaction', 'JobSatisfaction', 'MonthlyIncome',
       'NumCompaniesWorked', 'WorkLifeBalance', 'YearsAtCompany'],
      dtype='object')
```

In [45]:
```
#ravel() Return a contiguous flattened array
y = np.ravel(y)
```

In [46]:
```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model = model.fit(x,y)

#Check accuracy on the training set
model.score(x,y)
```

```
C:\Users\ctoqu\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:
762: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion
  n_iter_i = _check_optimize_result(
```

Out[46]: 0.8394557823129252

In [47]:
```
y.mean()
```

Out[47]: 0.16122448979591836

In [50]:
```
X_train, X_test, y_train, y_test=sklearn.model_selection.train_test_split(x,y)
model2=LogisticRegression()
model2.fit(X_train, y_train)
```

Out[50]: LogisticRegression()

In [51]:
```
predicted=model.predict(X_test)
print(predicted)
```

```
[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. 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. 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. 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. 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. 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. 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. 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. 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. 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. 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. 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. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0.]
```

In [52]:
```
probs = model2.predict_proba(X_test)
print(probs)
```

```
[[0.77710745 0.22289255]
 [0.89946267 0.10053733]
 [0.88812442 0.11187558]
 [0.66294731 0.33705269]
 [0.70803577 0.29196423]
 [0.93546321 0.06453679]
 [0.74919647 0.25080353]
 [0.75747813 0.24252187]
 [0.81452112 0.18547888]
 [0.88627987 0.11372013]
 [0.79803931 0.20196069]
 [0.91332106 0.08667894]
 [0.90852212 0.09147788]
 [0.79417341 0.20582659]
 [0.79797373 0.20202627]
 [0.96197734 0.03802266]
 [0.98211518 0.01788482]
 [0.90323497 0.09676503]
 [0.89289402 0.10710598]
 [0.66085666 0.33914334]
 [0.92738559 0.07261441]
 [0.916757   0.083243  ]
 [0.90069111 0.09930889]
 [0.81842562 0.18157438]
 [0.93213234 0.06786766]
 [0.9217071  0.0782929 ]
 [0.89911148 0.10088852]
 [0.84322463 0.15677537]
 [0.80601202 0.19398798]
 [0.79566466 0.20433534]
 [0.66057174 0.33942826]
 [0.81360249 0.18639751]
 [0.81835208 0.18164792]
 [0.8203218  0.1796782 ]
 [0.81653987 0.18346013]
 [0.92460463 0.07539537]
 [0.93119874 0.06880126]
 [0.97876957 0.02123043]
 [0.79956245 0.20043755]
 [0.8946298  0.1053702 ]
 [0.90085979 0.09914021]
 [0.87787653 0.12212347]
 [0.79334656 0.20665344]
 [0.73375767 0.26624233]
 [0.88212799 0.11787201]
 [0.87925515 0.12074485]
 [0.86986189 0.13013811]
 [0.84751056 0.15248944]
 [0.81003853 0.18996147]
 [0.82319508 0.17680492]
 [0.72824544 0.27175456]
 [0.74796916 0.25203084]
 [0.87139212 0.12860788]
 [0.97823662 0.02176338]
 [0.81019269 0.18980731]
 [0.88063548 0.11936452]
 [0.96216362 0.03783638]
```

```
[0.92195312 0.07804688]
[0.80501565 0.19498435]
[0.94263637 0.05736363]
[0.87455037 0.12544963]
[0.87370529 0.12629471]
[0.84684792 0.15315208]
[0.84231135 0.15768865]
[0.87272074 0.12727926]
[0.83869464 0.16130536]
[0.8299045  0.1700955 ]
[0.81836013 0.18163987]
[0.93390396 0.06609604]
[0.83630931 0.16369069]
[0.86161991 0.13838009]
[0.8252765  0.1747235 ]
[0.80645832 0.19354168]
[0.7137266  0.2862734 ]
[0.8570247  0.1429753 ]
[0.71053871 0.28946129]
[0.94964386 0.05035614]
[0.78891259 0.21108741]
[0.98602709 0.01397291]
[0.93437571 0.06562429]
[0.96035271 0.03964729]
[0.87068409 0.12931591]
[0.85751731 0.14248269]
[0.75086255 0.24913745]
[0.92026102 0.07973898]
[0.79720595 0.20279405]
[0.88639459 0.11360541]
[0.89525797 0.10474203]
[0.87362877 0.12637123]
[0.85002928 0.14997072]
[0.88404754 0.11595246]
[0.91897983 0.08102017]
[0.87297251 0.12702749]
[0.77775419 0.22224581]
[0.73151202 0.26848798]
[0.81103972 0.18896028]
[0.80475948 0.19524052]
[0.87338528 0.12661472]
[0.90492194 0.09507806]
[0.81651398 0.18348602]
[0.80242541 0.19757459]
[0.79975034 0.20024966]
[0.94941809 0.05058191]
[0.91971145 0.08028855]
[0.66461729 0.33538271]
[0.84774488 0.15225512]
[0.77051443 0.22948557]
[0.92859149 0.07140851]
[0.87801727 0.12198273]
[0.79815907 0.20184093]
[0.82054374 0.17945626]
[0.72173355 0.27826645]
[0.87990575 0.12009425]
[0.96221351 0.03778649]
```

```
[0.8152809  0.1847191 ]
[0.7523303  0.2476697 ]
[0.8371228  0.1628772 ]
[0.9697573  0.0302427 ]
[0.73482565 0.26517435]
[0.93499277 0.06500723]
[0.88797051 0.11202949]
[0.82755256 0.17244744]
[0.7881831  0.2118169 ]
[0.96959617 0.03040383]
[0.92531297 0.07468703]
[0.95750334 0.04249666]
[0.88923553 0.11076447]
[0.85874762 0.14125238]
[0.67868099 0.32131901]
[0.88662128 0.11337872]
[0.96648217 0.03351783]
[0.81175999 0.18824001]
[0.79854273 0.20145727]
[0.91911997 0.08088003]
[0.78154104 0.21845896]
[0.98867917 0.01132083]
[0.83757629 0.16242371]
[0.87924431 0.12075569]
[0.88864301 0.11135699]
[0.95606403 0.04393597]
[0.83941442 0.16058558]
[0.84886821 0.15113179]
[0.89745532 0.10254468]
[0.83789044 0.16210956]
[0.89642123 0.10357877]
[0.67297699 0.32702301]
[0.82574026 0.17425974]
[0.81909555 0.18090445]
[0.92040841 0.07959159]
[0.75068282 0.24931718]
[0.93494693 0.06505307]
[0.74590813 0.25409187]
[0.8656122  0.1343878 ]
[0.75505792 0.24494208]
[0.75002874 0.24997126]
[0.80032732 0.19967268]
[0.86394282 0.13605718]
[0.80970167 0.19029833]
[0.80610845 0.19389155]
[0.8665262  0.1334738 ]
[0.97229463 0.02770537]
[0.66516072 0.33483928]
[0.91894647 0.08105353]
[0.92180516 0.07819484]
[0.86638786 0.13361214]
[0.91151294 0.08848706]
[0.92053895 0.07946105]
[0.8308813  0.1691187 ]
[0.79233761 0.20766239]
[0.82840209 0.17159791]
[0.73709926 0.26290074]
```

```
[0.96343988 0.03656012]
[0.79921541 0.20078459]
[0.86269811 0.13730189]
[0.75018581 0.24981419]
[0.95587445 0.04412555]
[0.81914564 0.18085436]
[0.86895885 0.13104115]
[0.67884021 0.32115979]
[0.95893418 0.04106582]
[0.83710765 0.16289235]
[0.90285241 0.09714759]
[0.88866556 0.11133444]
[0.79317273 0.20682727]
[0.65549443 0.34450557]
[0.83294528 0.16705472]
[0.81607752 0.18392248]
[0.8589175  0.1410825 ]
[0.77744951 0.22255049]
[0.77918789 0.22081211]
[0.93952389 0.06047611]
[0.94829037 0.05170963]
[0.93491412 0.06508588]
[0.78039903 0.21960097]
[0.85468338 0.14531662]
[0.93809988 0.06190012]
[0.92303554 0.07696446]
[0.86103986 0.13896014]
[0.87930075 0.12069925]
[0.67487009 0.32512991]
[0.58192565 0.41807435]
[0.96847546 0.03152454]
[0.85993529 0.14006471]
[0.78578618 0.21421382]
[0.82195553 0.17804447]
[0.79108039 0.20891961]
[0.90664545 0.09335455]
[0.59014403 0.40985597]
[0.9788148  0.0211852 ]
[0.89530798 0.10469202]
[0.80700657 0.19299343]
[0.82194121 0.17805879]
[0.83309419 0.16690581]
[0.8617545  0.1382455 ]
[0.84005856 0.15994144]
[0.86948878 0.13051122]
[0.73320741 0.26679259]
[0.69024947 0.30975053]
[0.68940803 0.31059197]
[0.81474638 0.18525362]
[0.81923368 0.18076632]
[0.8145164  0.1854836 ]
[0.97065838 0.02934162]
[0.80627974 0.19372026]
[0.94835774 0.05164226]
[0.97728632 0.02271368]
[0.60873269 0.39126731]
[0.84339283 0.15660717]
```

```
[0.78395589 0.21604411]
[0.79019839 0.20980161]
[0.90591731 0.09408269]
[0.87240545 0.12759455]
[0.81652976 0.18347024]
[0.97781328 0.02218672]
[0.85327719 0.14672281]
[0.79911998 0.20088002]
[0.79592136 0.20407864]
[0.85179822 0.14820178]
[0.76479138 0.23520862]
[0.93043741 0.06956259]
[0.7999634  0.2000366 ]
[0.93590582 0.06409418]
[0.88693789 0.11306211]
[0.92135388 0.07864612]
[0.71468685 0.28531315]
[0.77611657 0.22388343]
[0.90193879 0.09806121]
[0.79849651 0.20150349]
[0.76273657 0.23726343]
[0.87205565 0.12794435]
[0.84355474 0.15644526]
[0.66209444 0.33790556]
[0.89650928 0.10349072]
[0.89596415 0.10403585]
[0.85362309 0.14637691]
[0.97413523 0.02586477]
[0.83407865 0.16592135]
[0.8177222  0.1822778 ]
[0.93748967 0.06251033]
[0.73666135 0.26333865]
[0.7024331  0.2975669 ]
[0.89455299 0.10544701]
[0.74122609 0.25877391]
[0.89475377 0.10524623]
[0.82341637 0.17658363]
[0.80188454 0.19811546]
[0.85648539 0.14351461]
[0.77004946 0.22995054]
[0.77180118 0.22819882]
[0.87664312 0.12335688]
[0.95978427 0.04021573]
[0.79984416 0.20015584]
[0.59165276 0.40834724]
[0.68863373 0.31136627]
[0.76461128 0.23538872]
[0.84260256 0.15739744]
[0.87653303 0.12346697]
[0.76696623 0.23303377]
[0.84337885 0.15662115]
[0.93217496 0.06782504]
[0.81912786 0.18087214]
[0.76565225 0.23434775]
[0.88879497 0.11120503]
[0.81754411 0.18245589]
[0.89292551 0.10707449]
```

```
[0.77533783 0.22466217]
[0.95738517 0.04261483]
[0.85626669 0.14373331]
[0.80133802 0.19866198]
[0.95125661 0.04874339]
[0.80220282 0.19779718]
[0.76156257 0.23843743]
[0.83753287 0.16246713]
[0.94251245 0.05748755]
[0.83645104 0.16354896]
[0.77650328 0.22349672]
[0.88851976 0.11148024]
[0.85980684 0.14019316]
[0.87838513 0.12161487]
[0.91573905 0.08426095]
[0.84760213 0.15239787]
[0.87749696 0.12250304]
[0.8240145  0.1759855 ]
[0.85400182 0.14599818]
[0.85625459 0.14374541]
[0.8472115  0.1527885 ]
[0.81926311 0.18073689]
[0.75185732 0.24814268]
[0.80440904 0.19559096]
[0.85742385 0.14257615]
[0.95003179 0.04996821]
[0.8351476  0.1648524 ]
[0.91299379 0.08700621]
[0.8839913  0.1160087 ]
[0.86052426 0.13947574]
[0.8755804  0.1244196 ]
[0.83505994 0.16494006]
[0.82200508 0.17799492]
[0.56169903 0.43830097]
[0.94554041 0.05445959]
[0.83779537 0.16220463]
[0.91477268 0.08522732]
[0.81852304 0.18147696]
[0.82036609 0.17963391]
[0.77788929 0.22211071]
[0.95556073 0.04443927]
[0.75507162 0.24492838]
[0.8636001  0.1363999 ]
[0.88308534 0.11691466]
[0.79432261 0.20567739]
[0.86325316 0.13674684]
[0.89488962 0.10511038]
[0.98048164 0.01951836]
[0.88758584 0.11241416]
[0.78094466 0.21905534]
[0.90265768 0.09734232]
[0.77188848 0.22811152]
[0.80364369 0.19635631]
[0.80979414 0.19020586]
[0.80161903 0.19838097]
[0.89676827 0.10323173]
[0.91230231 0.08769769]
```

```
 [0.78851939 0.21148061]
 [0.72222663 0.27777337]
 [0.80976145 0.19023855]
 [0.93949777 0.06050223]
 [0.8572034  0.1427966 ]
 [0.90941341 0.09058659]
 [0.8264128  0.1735872 ]
 [0.72257337 0.27742663]
 [0.86206259 0.13793741]
 [0.6586441  0.3413559 ]
 [0.91696799 0.08303201]
 [0.78559817 0.21440183]
 [0.89776109 0.10223891]
 [0.86841874 0.13158126]
 [0.77461268 0.22538732]
 [0.79660869 0.20339131]
 [0.77973746 0.22026254]
 [0.81406893 0.18593107]
 [0.86363485 0.13636515]
 [0.90356788 0.09643212]
 [0.77630101 0.22369899]
 [0.69451293 0.30548707]
 [0.85678114 0.14321886]
 [0.87293429 0.12706571]
 [0.83275312 0.16724688]
 [0.90204284 0.09795716]]
```

In [53]:
```
from sklearn import metrics
print(metrics.accuracy_score(y_test, predicted))
print(metrics.roc_auc_score(y_test,probs[:,1]))
```

```
0.8369565217391305
0.7485930735930736
```

In [54]:
```
print (metrics.confusion_matrix(y_test, predicted))
print (metrics.classification_report(y_test,predicted))
```

```
[[308   0]
 [ 60   0]]
              precision    recall  f1-score   support

         0.0       0.84      1.00      0.91       308
         1.0       0.00      0.00      0.00        60

    accuracy                           0.84       368
   macro avg       0.42      0.50      0.46       368
weighted avg       0.70      0.84      0.76       368


C:\Users\ctoqu\anaconda3\lib\site-packages\sklearn\metrics\_classification.p
y:1221: UndefinedMetricWarning: Precision and F-score are ill-defined and bei
ng set to 0.0 in labels with no predicted samples. Use `zero_division` parame
ter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [55]:
```
print(X_train)
```

```
      Intercept   Age  DistanceFromHome  Education  EnvironmentSatisfaction
\
753         1.0  39.0              22.0        3.0                      4.0
160         1.0  22.0              19.0        1.0                      3.0
684         1.0  40.0              10.0        4.0                      1.0
1242        1.0  40.0               7.0        4.0                      2.0
772         1.0  56.0               9.0        3.0                      1.0
...         ...   ...               ...        ...                      ...
1268        1.0  53.0               1.0        4.0                      1.0
78          1.0  37.0               7.0        4.0                      1.0
1099        1.0  45.0               1.0        4.0                      1.0
1377        1.0  49.0               2.0        1.0                      2.0
1373        1.0  38.0               8.0        3.0                      4.0

      JobSatisfaction  MonthlyIncome  NumCompaniesWorked  WorkLifeBalance  \
753               1.0        10880.0                 1.0              3.0
160               4.0         2323.0                 1.0              3.0
684               2.0         9705.0                 2.0              2.0
1242              2.0        19833.0                 1.0              2.0
772               3.0         2942.0                 2.0              3.0
...               ...            ...                 ...              ...
1268              3.0        12965.0                 4.0              2.0
78                3.0        13664.0                 4.0              4.0
1099              2.0         7441.0                 1.0              3.0
1377              4.0        19161.0                 3.0              3.0
1373              2.0         2133.0                 1.0              3.0

      YearsAtCompany
753             21.0
160              2.0
684              1.0
1242            21.0
772              5.0
...              ...
1268             3.0
78               5.0
1099            10.0
1377             5.0
1373            20.0

[1102 rows x 10 columns]
```

In [57]:
```
##add random values to KK according to the parameters mentioned above to check
the proabily of attrition of the employee
kk=[[1.0, 23.0, 1.0, 500.0, 3.0, 24.0, 1.0, 4.0, 10.0, 1.0]]
print(model.predict_proba(kk))
```

```
[[0.0051449 0.9948551]]
```

In [ ]:
```

```