IBM_Employee_Attrition_Prediction

September 21, 2021

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

[2]:
```
dataframe=pd.read_csv("IBM Attrition Data.csv")
```

[3]:
```
dataframe.head()
```

[3]:
```
   Age Attrition              Department  DistanceFromHome  Education  \
0   41       Yes                   Sales                 1          2
1   49        No  Research & Development                 8          1
2   37       Yes  Research & Development                 2          2
3   33        No  Research & Development                 3          4
4   27        No  Research & Development                 2          1

  EducationField  EnvironmentSatisfaction  JobSatisfaction MaritalStatus  \
0  Life Sciences                        2                4        Single
1  Life Sciences                        3                2       Married
2          Other                        4                3        Single
3  Life Sciences                        4                3       Married
4        Medical                        1                2       Married

   MonthlyIncome  NumCompaniesWorked  WorkLifeBalance  YearsAtCompany
0           5993                   8                1               6
1           5130                   1                3              10
2           2090                   6                3               0
3           2909                   1                3               8
4           3468                   9                3               2
```

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

```
['Age' 'Attrition' 'Department' 'DistanceFromHome' 'Education'
```

```
'EducationField' 'EnvironmentSatisfaction' 'JobSatisfaction'
'MaritalStatus' 'MonthlyIncome' 'NumCompaniesWorked' 'WorkLifeBalance'
'YearsAtCompany']
```

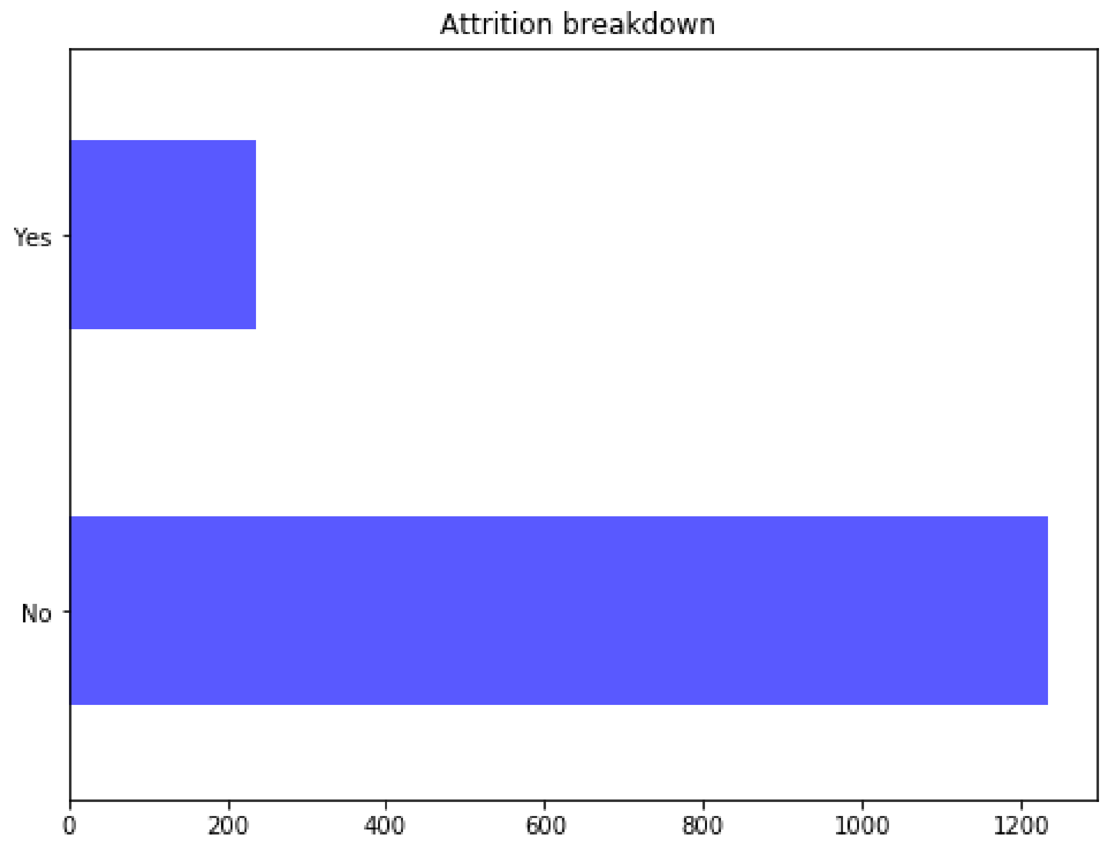
[5]:

```python
# histogram for age
plt.figure(figsize=(10,8))
dataframe['Age'].hist(bins=70)
plt.title("Age distribution of Employees")
plt.xlabel("Age")
plt.ylabel("# of Employees")
plt.show()
```

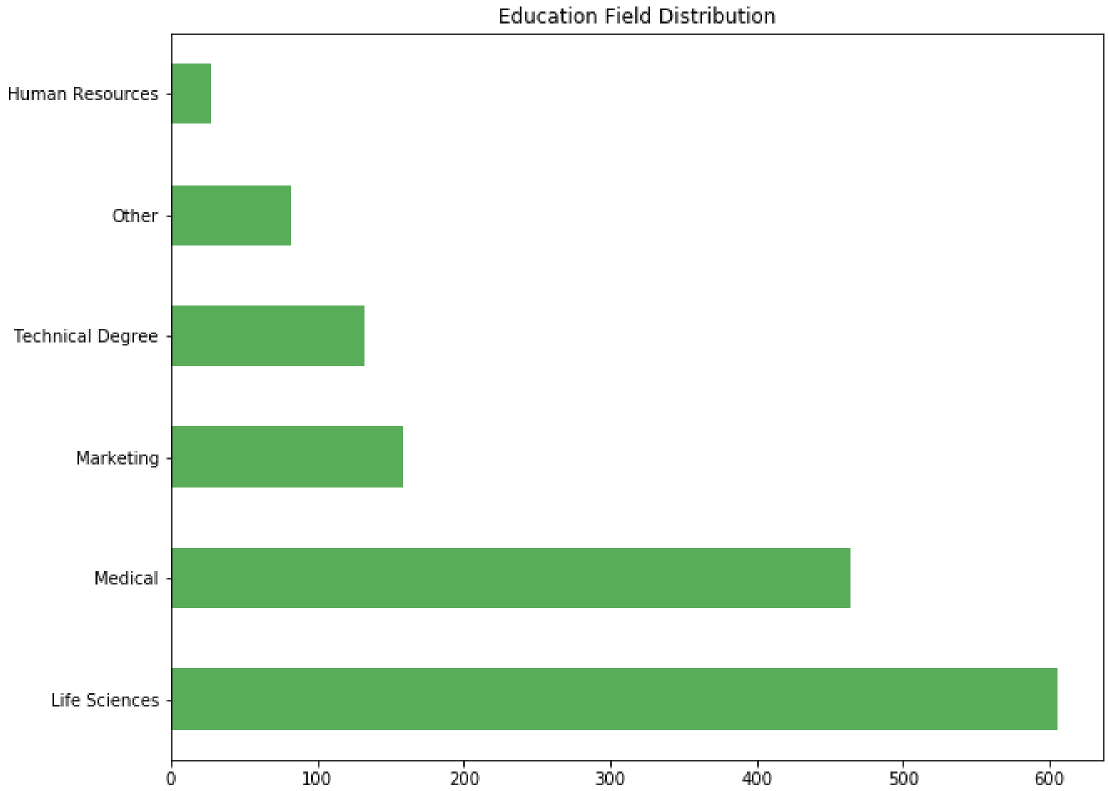
[6]:

```python
# explore data for Attrition by Age
plt.figure(figsize=(14,10))
plt.scatter(dataframe.Attrition,dataframe.Age, alpha=.55)
plt.title("Attrition by Age ")
plt.ylabel("Age")
plt.grid(b=True, which='major',axis='y')
plt.show()
```

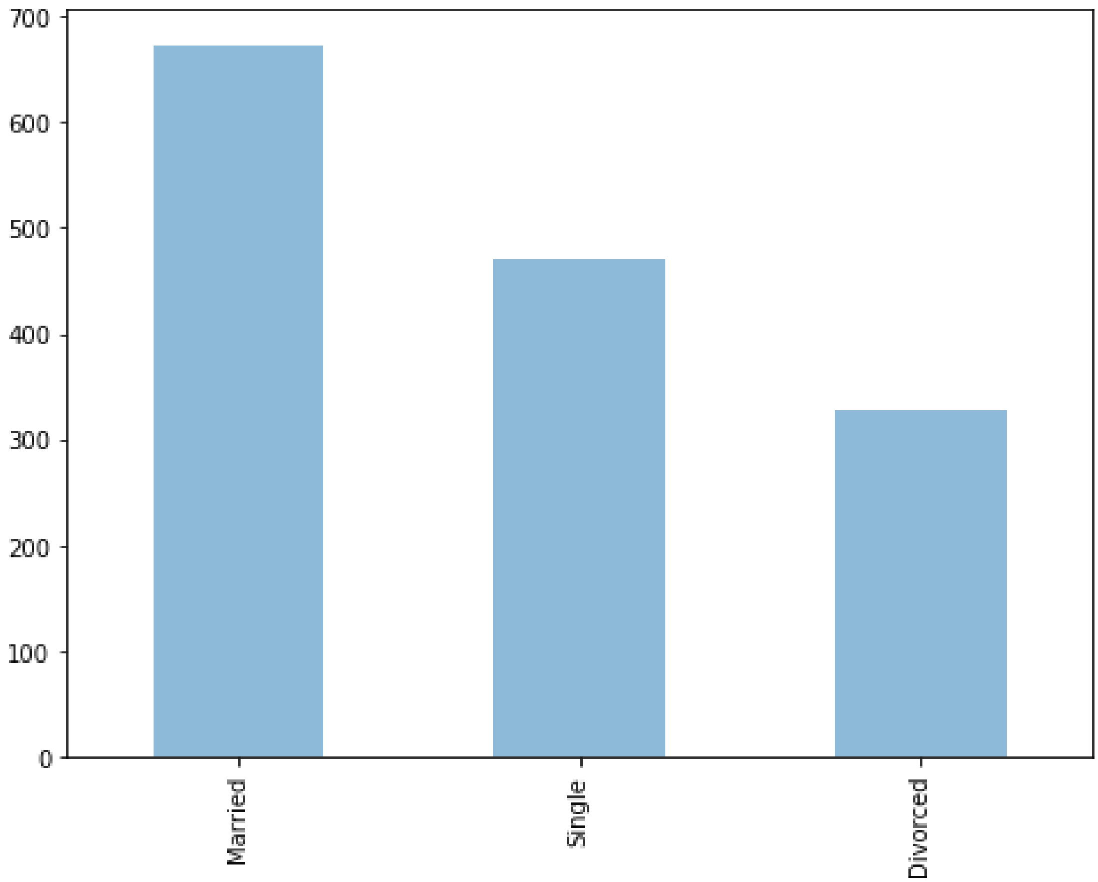
[7]:
```
# explore data for Left employees breakdown
plt.figure(figsize=(8,6))
dataframe.Attrition.value_counts().plot(kind='barh',color='blue',alpha=.65)
plt.title("Attrition breakdown ")
plt.show()
```



[8]:
```
# explore data for Education Field distribution
plt.figure(figsize=(10,8))
dataframe.EducationField.value_counts().plot(kind='barh',color='g',alpha=.65)
plt.title("Education Field Distribution")
plt.show()
```



[9]:
```
# explore data for Marital Status
plt.figure(figsize=(8,6))
dataframe.MaritalStatus.value_counts().plot(kind='bar',alpha=.5)
plt.show()
```



[10]:
```
dataframe.describe()
```

[10]:
```
               Age  DistanceFromHome    Education  EnvironmentSatisfaction  \
count  1470.000000       1470.000000  1470.000000              1470.000000
mean     36.923810          9.192517     2.912925                 2.721769
std       9.135373          8.106864     1.024165                 1.093082
min      18.000000          1.000000     1.000000                 1.000000
25%      30.000000          2.000000     2.000000                 2.000000
50%      36.000000          7.000000     3.000000                 3.000000
75%      43.000000         14.000000     4.000000                 4.000000
max      60.000000         29.000000     5.000000                 4.000000

       JobSatisfaction  MonthlyIncome  NumCompaniesWorked  WorkLifeBalance  \
count      1470.000000    1470.000000         1470.000000      1470.000000
mean          2.728571    6502.931293            2.693197         2.761224
std           1.102846    4707.956783            2.498009         0.706476
min           1.000000    1009.000000            0.000000         1.000000
25%           2.000000    2911.000000            1.000000         2.000000
50%           3.000000    4919.000000            2.000000         3.000000
```

```
75%           4.000000    8379.000000              4.000000            3.000000
max           4.000000   19999.000000              9.000000            4.000000

       YearsAtCompany
count     1470.000000
mean         7.008163
std          6.126525
min          0.000000
25%          3.000000
50%          5.000000
75%          9.000000
max         40.000000
```

[11]:
```
dataframe.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 13 columns):
Age                        1470 non-null int64
Attrition                  1470 non-null object
Department                 1470 non-null object
DistanceFromHome           1470 non-null int64
Education                  1470 non-null int64
EducationField             1470 non-null object
EnvironmentSatisfaction    1470 non-null int64
JobSatisfaction            1470 non-null int64
MaritalStatus              1470 non-null object
MonthlyIncome              1470 non-null int64
NumCompaniesWorked         1470 non-null int64
WorkLifeBalance            1470 non-null int64
YearsAtCompany             1470 non-null int64
dtypes: int64(9), object(4)
memory usage: 149.4+ KB
```

[12]:
```
dataframe.columns
```

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

[13]:
```
dataframe.std()
```

[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
```

[14]:
```python
dataframe['Attrition'].value_counts()
```

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

[15]:
```python
dataframe['Attrition'].dtypes
```

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

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

[17]:
```python
dataframe.head(10)
```

[17]:
```
   Age  Attrition              Department  DistanceFromHome  Education  \
0   41          1                   Sales                 1          2
1   49          0  Research & Development                 8          1
2   37          1  Research & Development                 2          2
3   33          0  Research & Development                 3          4
4   27          0  Research & Development                 2          1
5   32          0  Research & Development                 2          2
6   59          0  Research & Development                 3          3
7   30          0  Research & Development                24          1
8   38          0  Research & Development                23          3
9   36          0  Research & Development                27          3

  EducationField  EnvironmentSatisfaction  JobSatisfaction MaritalStatus  \
0  Life Sciences                        2                4        Single
1  Life Sciences                        3                2       Married
2          Other                        4                3        Single
3  Life Sciences                        4                3       Married
4        Medical                        1                2       Married
5  Life Sciences                        4                4        Single
6        Medical                        3                1       Married
7  Life Sciences                        4                3      Divorced
8  Life Sciences                        4                3        Single
9        Medical                        3                3       Married
```

```
   MonthlyIncome  NumCompaniesWorked  WorkLifeBalance  YearsAtCompany
0           5993                   8                1               6
1           5130                   1                3              10
2           2090                   6                3               0
3           2909                   1                3               8
4           3468                   9                3               2
5           3068                   0                2               7
6           2670                   4                2               1
7           2693                   1                3               1
8           9526                   0                3               9
9           5237                   6                2               7
```

[18]:
```
# building up a logistic regression model
X = dataframe.drop(['Attrition'],axis=1)
X.head()
Y = dataframe['Attrition']
Y.head()
```

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

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

[20]:
```
dataframe['EducationField'].value_counts()
```

[20]:
```
1    606
2    464
3    159
5    132
4     82
6     27
Name: EducationField, dtype: int64
```

[21]:
```
dataframe['Department'].value_counts()
```

[21]:
```
Research & Development    961
Sales                     446
Human Resources            63
```

```
Name: Department, dtype: int64
```

[22]:
```python
dataframe['Department'].replace('Research & Development',1, inplace=True)
dataframe['Department'].replace('Sales',2, inplace=True)
dataframe['Department'].replace('Human Resources', 3, inplace=True)
```

[23]:
```python
dataframe['Department'].value_counts()
```

[23]:
```
1    961
2    446
3     63
Name: Department, dtype: int64
```

[24]:
```python
dataframe['MaritalStatus'].value_counts()
```

[24]:
```
Married     673
Single      470
Divorced    327
Name: MaritalStatus, dtype: int64
```

[25]:
```python
dataframe['MaritalStatus'].replace('Married',1, inplace=True)
dataframe['MaritalStatus'].replace('Single',2, inplace=True)
dataframe['MaritalStatus'].replace('Divorced',3, inplace=True)
```

[26]:
```python
dataframe['MaritalStatus'].value_counts()
```

[26]:
```
1    673
2    470
3    327
Name: MaritalStatus, dtype: int64
```

[27]:
```python
x=dataframe.select_dtypes(include=['int64'])
x.dtypes
```

[27]:
```
Age                        int64
Attrition                  int64
Department                 int64
DistanceFromHome           int64
Education                  int64
EducationField             int64
EnvironmentSatisfaction    int64
JobSatisfaction            int64
MaritalStatus              int64
MonthlyIncome              int64
NumCompaniesWorked         int64
WorkLifeBalance            int64
YearsAtCompany             int64
```

```
dtype: object
```

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

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

[29]:
```
y=dataframe['Attrition']
```

[30]:
```
y.head()
```

[30]:
```
0    1
1    0
2    1
3    0
4    0
Name: Attrition, dtype: int64
```

[31]:
```
y, x = dmatrices('Attrition ~ Age + Department + \
                  DistanceFromHome + Education + EducationField +␣
 ↪YearsAtCompany',
                  dataframe, return_type="dataframe")
print (x.columns)
```

```
Index(['Intercept', 'Age', 'Department', 'DistanceFromHome', 'Education',
       'EducationField', 'YearsAtCompany'],
      dtype='object')
```

[32]:
```
y = np.ravel(y)
```

[33]:
```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression()
model = model.fit(x, y)

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

```
/opt/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433:
FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify a
solver to silence this warning.
  FutureWarning)
```

[33]:
```
0.8408163265306122
```

[34]:
```python
y.mean()
```

[34]:
```
0.16122448979591836
```

[35]:
```python
X_train,X_test,y_train,y_test=sklearn.model_selection.train_test_split(x,y,␣
↪test_size=0.3, random_state=0)
model2=LogisticRegression()
model2.fit(X_train, y_train)
```

```
/opt/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433:
FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify a
solver to silence this warning.
  FutureWarning)
```

[35]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=None, solver='warn',
          tol=0.0001, verbose=0, warm_start=False)
```

[36]:
```python
predicted= model2.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. 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. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

[37]:
```python
probs = model2.predict_proba(X_test)
print (probs)
```

```
[[0.86257761 0.13742239]
 [0.80710189 0.19289811]
 [0.7429987  0.2570013 ]
```

```
[0.83583504 0.16416496]
[0.73307035 0.26692965]
[0.78942615 0.21057385]
[0.85718191 0.14281809]
[0.85697723 0.14302277]
[0.96732187 0.03267813]
[0.93781765 0.06218235]
[0.95112889 0.04887111]
[0.83140356 0.16859644]
[0.86069144 0.13930856]
[0.863881   0.136119  ]
[0.88818146 0.11181854]
[0.88851235 0.11148765]
[0.88418532 0.11581468]
[0.78102191 0.21897809]
[0.79870103 0.20129897]
[0.88654952 0.11345048]
[0.70201258 0.29798742]
[0.94684452 0.05315548]
[0.86687518 0.13312482]
[0.84389943 0.15610057]
[0.60328043 0.39671957]
[0.8112161  0.1887839 ]
[0.91914771 0.08085229]
[0.93333047 0.06666953]
[0.67850927 0.32149073]
[0.87080099 0.12919901]
[0.87277322 0.12722678]
[0.77054173 0.22945827]
[0.86434352 0.13565648]
[0.95829505 0.04170495]
[0.84589968 0.15410032]
[0.86642435 0.13357565]
[0.90489195 0.09510805]
[0.68640634 0.31359366]
[0.90762923 0.09237077]
[0.80686978 0.19313022]
[0.91626105 0.08373895]
[0.82434807 0.17565193]
[0.93702713 0.06297287]
[0.93419719 0.06580281]
[0.89317815 0.10682185]
[0.85163342 0.14836658]
[0.78599372 0.21400628]
[0.84591285 0.15408715]
[0.66035418 0.33964582]
[0.75985595 0.24014405]
[0.92971879 0.07028121]
```

```
[0.79073149 0.20926851]
[0.86251514 0.13748486]
[0.86028777 0.13971223]
[0.87176033 0.12823967]
[0.79087814 0.20912186]
[0.87589802 0.12410198]
[0.84351786 0.15648214]
[0.72814826 0.27185174]
[0.83401865 0.16598135]
[0.90193848 0.09806152]
[0.70822548 0.29177452]
[0.92855494 0.07144506]
[0.84184113 0.15815887]
[0.79759143 0.20240857]
[0.86955841 0.13044159]
[0.91690233 0.08309767]
[0.84801457 0.15198543]
[0.89284306 0.10715694]
[0.63214954 0.36785046]
[0.93929587 0.06070413]
[0.72436084 0.27563916]
[0.85581742 0.14418258]
[0.84210919 0.15789081]
[0.77522163 0.22477837]
[0.71561254 0.28438746]
[0.93625216 0.06374784]
[0.95759882 0.04240118]
[0.79115941 0.20884059]
[0.89387487 0.10612513]
[0.9143774  0.0856226 ]
[0.79373481 0.20626519]
[0.78032498 0.21967502]
[0.79647769 0.20352231]
[0.83618218 0.16381782]
[0.71431018 0.28568982]
[0.97808679 0.02191321]
[0.94675994 0.05324006]
[0.88520539 0.11479461]
[0.79405267 0.20594733]
[0.61481071 0.38518929]
[0.81886235 0.18113765]
[0.74684358 0.25315642]
[0.86722821 0.13277179]
[0.86992409 0.13007591]
[0.81789428 0.18210572]
[0.71822509 0.28177491]
[0.60023923 0.39976077]
[0.83836485 0.16163515]
```

```
[0.88216124 0.11783876]
[0.74418148 0.25581852]
[0.76564261 0.23435739]
[0.98067742 0.01932258]
[0.91939455 0.08060545]
[0.77415323 0.22584677]
[0.92564103 0.07435897]
[0.88199097 0.11800903]
[0.74514347 0.25485653]
[0.90673063 0.09326937]
[0.78928203 0.21071797]
[0.80971647 0.19028353]
[0.93515971 0.06484029]
[0.93924676 0.06075324]
[0.79462059 0.20537941]
[0.81215385 0.18784615]
[0.91649218 0.08350782]
[0.90265873 0.09734127]
[0.84731114 0.15268886]
[0.95376317 0.04623683]
[0.91222675 0.08777325]
[0.86028682 0.13971318]
[0.85822982 0.14177018]
[0.87448572 0.12551428]
[0.75985594 0.24014406]
[0.92296733 0.07703267]
[0.96914997 0.03085003]
[0.94407447 0.05592553]
[0.81720383 0.18279617]
[0.88066242 0.11933758]
[0.77639891 0.22360109]
[0.97128842 0.02871158]
[0.88831439 0.11168561]
[0.78631482 0.21368518]
[0.81840678 0.18159322]
[0.94987331 0.05012669]
[0.95894743 0.04105257]
[0.73447703 0.26552297]
[0.93444274 0.06555726]
[0.73813794 0.26186206]
[0.82247975 0.17752025]
[0.82289185 0.17710815]
[0.89920393 0.10079607]
[0.78516352 0.21483648]
[0.89653967 0.10346033]
[0.91537087 0.08462913]
[0.92820436 0.07179564]
[0.96589553 0.03410447]
```

```
[0.94419804 0.05580196]
[0.93024428 0.06975572]
[0.66112588 0.33887412]
[0.84095505 0.15904495]
[0.82603046 0.17396954]
[0.80610059 0.19389941]
[0.96191568 0.03808432]
[0.93671599 0.06328401]
[0.94770351 0.05229649]
[0.97376472 0.02623528]
[0.79369198 0.20630802]
[0.87741394 0.12258606]
[0.85956848 0.14043152]
[0.95216215 0.04783785]
[0.93160388 0.06839612]
[0.75495757 0.24504243]
[0.74998837 0.25001163]
[0.95590644 0.04409356]
[0.86936376 0.13063624]
[0.81422948 0.18577052]
[0.76650749 0.23349251]
[0.80183602 0.19816398]
[0.92798469 0.07201531]
[0.91054713 0.08945287]
[0.94603047 0.05396953]
[0.93400754 0.06599246]
[0.69063333 0.30936667]
[0.93091068 0.06908932]
[0.74159667 0.25840333]
[0.78516386 0.21483614]
[0.93229165 0.06770835]
[0.80621879 0.19378121]
[0.85290079 0.14709921]
[0.66903659 0.33096341]
[0.9042279  0.0957721 ]
[0.91210155 0.08789845]
[0.87547616 0.12452384]
[0.93020588 0.06979412]
[0.66879074 0.33120926]
[0.89374371 0.10625629]
[0.86196532 0.13803468]
[0.78749466 0.21250534]
[0.53185454 0.46814546]
[0.73337673 0.26662327]
[0.70603668 0.29396332]
[0.85434454 0.14565546]
[0.869108   0.130892  ]
[0.75104191 0.24895809]
```

```
[0.89891506 0.10108494]
[0.79281444 0.20718556]
[0.90787555 0.09212445]
[0.77348776 0.22651224]
[0.88287113 0.11712887]
[0.85302465 0.14697535]
[0.8195964  0.1804036 ]
[0.74239392 0.25760608]
[0.86238441 0.13761559]
[0.77748616 0.22251384]
[0.76912758 0.23087242]
[0.7938589  0.2061411 ]
[0.92209228 0.07790772]
[0.74615104 0.25384896]
[0.87485382 0.12514618]
[0.85477514 0.14522486]
[0.77450251 0.22549749]
[0.87362727 0.12637273]
[0.67359458 0.32640542]
[0.93698936 0.06301064]
[0.82461956 0.17538044]
[0.95188386 0.04811614]
[0.83450941 0.16549059]
[0.81117757 0.18882243]
[0.80629478 0.19370522]
[0.87690301 0.12309699]
[0.6663069  0.3336931 ]
[0.59350144 0.40649856]
[0.98983468 0.01016532]
[0.70381235 0.29618765]
[0.91693005 0.08306995]
[0.92230104 0.07769896]
[0.71009303 0.28990697]
[0.62307399 0.37692601]
[0.76273323 0.23726677]
[0.95379074 0.04620926]
[0.88139107 0.11860893]
[0.85805507 0.14194493]
[0.92153445 0.07846555]
[0.87986341 0.12013659]
[0.80455714 0.19544286]
[0.8045461  0.1954539 ]
[0.91400939 0.08599061]
[0.71996681 0.28003319]
[0.9459133  0.0540867 ]
[0.90887304 0.09112696]
[0.73122211 0.26877789]
[0.98139747 0.01860253]
```

```
[0.85440507 0.14559493]
[0.89904525 0.10095475]
[0.82348836 0.17651164]
[0.83289134 0.16710866]
[0.88059965 0.11940035]
[0.87965985 0.12034015]
[0.87516106 0.12483894]
[0.8154612  0.1845388 ]
[0.88085227 0.11914773]
[0.61440015 0.38559985]
[0.88813952 0.11186048]
[0.89579477 0.10420523]
[0.85493829 0.14506171]
[0.98316036 0.01683964]
[0.7717054  0.2282946 ]
[0.62163203 0.37836797]
[0.82648597 0.17351403]
[0.84082886 0.15917114]
[0.84770539 0.15229461]
[0.84996276 0.15003724]
[0.7568283  0.2431717 ]
[0.86135648 0.13864352]
[0.90742097 0.09257903]
[0.84653325 0.15346675]
[0.81068432 0.18931568]
[0.74291535 0.25708465]
[0.87004234 0.12995766]
[0.83937674 0.16062326]
[0.86204616 0.13795384]
[0.66559201 0.33440799]
[0.90809363 0.09190637]
[0.87063167 0.12936833]
[0.92591545 0.07408455]
[0.84519617 0.15480383]
[0.89988333 0.10011667]
[0.91377645 0.08622355]
[0.79655167 0.20344833]
[0.63617514 0.36382486]
[0.8451662  0.1548338 ]
[0.75229555 0.24770445]
[0.85439954 0.14560046]
[0.99258502 0.00741498]
[0.85979235 0.14020765]
[0.88042046 0.11957954]
[0.82752509 0.17247491]
[0.93110919 0.06889081]
[0.87320755 0.12679245]
[0.88685479 0.11314521]
```

```
[0.83757498 0.16242502]
[0.86470667 0.13529333]
[0.86456218 0.13543782]
[0.89956239 0.10043761]
[0.78624587 0.21375413]
[0.79354081 0.20645919]
[0.88436844 0.11563156]
[0.6517352  0.3482648 ]
[0.94078665 0.05921335]
[0.8998614  0.1001386 ]
[0.72416846 0.27583154]
[0.68855828 0.31144172]
[0.87401796 0.12598204]
[0.86388337 0.13611663]
[0.97515719 0.02484281]
[0.86301941 0.13698059]
[0.54429973 0.45570027]
[0.91312117 0.08687883]
[0.74864711 0.25135289]
[0.86865164 0.13134836]
[0.88768464 0.11231536]
[0.87825968 0.12174032]
[0.85578493 0.14421507]
[0.77096345 0.22903655]
[0.80608869 0.19391131]
[0.85164484 0.14835516]
[0.7748914  0.2251086 ]
[0.70330443 0.29669557]
[0.88920999 0.11079001]
[0.48805675 0.51194325]
[0.92443534 0.07556466]
[0.75730317 0.24269683]
[0.67532223 0.32467777]
[0.91267187 0.08732813]
[0.94004403 0.05995597]
[0.88105134 0.11894866]
[0.88500438 0.11499562]
[0.95620493 0.04379507]
[0.90018491 0.09981509]
[0.94913267 0.05086733]
[0.83164948 0.16835052]
[0.87828332 0.12171668]
[0.81856033 0.18143967]
[0.81510872 0.18489128]
[0.95165724 0.04834276]
[0.86885777 0.13114223]
[0.90404843 0.09595157]
[0.83568878 0.16431122]
```

```
[0.84565343 0.15434657]
[0.79355796 0.20644204]
[0.81574488 0.18425512]
[0.81449702 0.18550298]
[0.83624028 0.16375972]
[0.91389238 0.08610762]
[0.91564837 0.08435163]
[0.68306915 0.31693085]
[0.99086383 0.00913617]
[0.76974325 0.23025675]
[0.79740377 0.20259623]
[0.72822071 0.27177929]
[0.67182682 0.32817318]
[0.79710573 0.20289427]
[0.84931231 0.15068769]
[0.86433739 0.13566261]
[0.85920637 0.14079363]
[0.84420225 0.15579775]
[0.82853445 0.17146555]
[0.92215641 0.07784359]
[0.82960704 0.17039296]
[0.97700212 0.02299788]
[0.90454177 0.09545823]
[0.92773082 0.07226918]
[0.84754954 0.15245046]
[0.76612754 0.23387246]
[0.94894713 0.05105287]
[0.94800941 0.05199059]
[0.75328457 0.24671543]
[0.87742303 0.12257697]
[0.80519574 0.19480426]
[0.93916181 0.06083819]
[0.85990185 0.14009815]
[0.75971578 0.24028422]
[0.90848727 0.09151273]
[0.7529481  0.2470519 ]
[0.94361137 0.05638863]
[0.91817801 0.08182199]
[0.90585801 0.09414199]
[0.77143638 0.22856362]
[0.92392657 0.07607343]
[0.80755229 0.19244771]
[0.9013237  0.0986763 ]
[0.87830849 0.12169151]
[0.8068256  0.1931744 ]
[0.83483933 0.16516067]
[0.53939525 0.46060475]
[0.95106284 0.04893716]
```

```
[0.73235519 0.26764481]
[0.892211   0.107789  ]
[0.80131021 0.19868979]
[0.87926632 0.12073368]
[0.96835844 0.03164156]
[0.81435024 0.18564976]
[0.8595391  0.1404609 ]
[0.59090241 0.40909759]
[0.82620318 0.17379682]
[0.92520542 0.07479458]
[0.81774745 0.18225255]
[0.92599818 0.07400182]
[0.89198781 0.10801219]
[0.70041077 0.29958923]
[0.82018762 0.17981238]
[0.96584774 0.03415226]
[0.87007757 0.12992243]
[0.8985835  0.1014165 ]
[0.89010322 0.10989678]
[0.81133218 0.18866782]
[0.85998392 0.14001608]
[0.83705922 0.16294078]
[0.83833325 0.16166675]
[0.82480592 0.17519408]
[0.94132438 0.05867562]
[0.83011466 0.16988534]
[0.77419827 0.22580173]
[0.69208833 0.30791167]
[0.86186596 0.13813404]
[0.82653322 0.17346678]
[0.84351252 0.15648748]
[0.87151308 0.12848692]
[0.89317815 0.10682185]
[0.82864779 0.17135221]
[0.7290552  0.2709448 ]
[0.9473871  0.0526129 ]
[0.96100837 0.03899163]
[0.9049959  0.0950041 ]
[0.88585723 0.11414277]
[0.84839464 0.15160536]
[0.78874009 0.21125991]
[0.67361889 0.32638111]
[0.93357031 0.06642969]
[0.65079394 0.34920606]
[0.74503232 0.25496768]
[0.9420944  0.0579056 ]
[0.78550077 0.21449923]
[0.90782391 0.09217609]
```

```
 [0.81479395 0.18520605]
 [0.89162714 0.10837286]
 [0.85619491 0.14380509]
 [0.67747664 0.32252336]
 [0.93182493 0.06817507]
 [0.89944427 0.10055573]]
```

[38]:
```python
from sklearn import metrics

print (metrics.accuracy_score(y_test, predicted))
print (metrics.roc_auc_score(y_test, probs[:, 1]))
```

```
0.8435374149659864
0.6500577589526376
```

[39]:
```python
print (metrics.confusion_matrix(y_test, predicted))
print (metrics.classification_report(y_test, predicted))
```

```
[[371   0]
 [ 69   1]]
              precision    recall  f1-score   support

         0.0       0.84      1.00      0.91       371
         1.0       1.00      0.01      0.03        70

   micro avg       0.84      0.84      0.84       441
   macro avg       0.92      0.51      0.47       441
weighted avg       0.87      0.84      0.77       441
```

[40]:
```python
print (X_train)
```

```
      Intercept   Age  Department  DistanceFromHome  Education  \
338         1.0  30.0         2.0               5.0        3.0
363         1.0  33.0         2.0               5.0        3.0
759         1.0  45.0         3.0              24.0        4.0
793         1.0  28.0         1.0              15.0        2.0
581         1.0  30.0         1.0               1.0        3.0
320         1.0  27.0         2.0               2.0        3.0
452         1.0  45.0         2.0               2.0        3.0
195         1.0  37.0         1.0              21.0        3.0
776         1.0  20.0         2.0               9.0        3.0
1295        1.0  41.0         2.0               4.0        1.0
70          1.0  59.0         2.0               1.0        1.0
1135        1.0  46.0         2.0               1.0        4.0
1011        1.0  36.0         2.0               3.0        4.0
10          1.0  35.0         1.0              16.0        3.0
1265        1.0  33.0         1.0               4.0        3.0
```

1270	1.0	34.0	2.0	3.0	2.0
1257	1.0	31.0	2.0	16.0	4.0
271	1.0	47.0	1.0	29.0	4.0
858	1.0	53.0	1.0	7.0	2.0
790	1.0	33.0	1.0	5.0	3.0
1290	1.0	34.0	1.0	9.0	4.0
915	1.0	21.0	1.0	10.0	2.0
64	1.0	36.0	1.0	8.0	3.0
959	1.0	40.0	1.0	2.0	3.0
1274	1.0	31.0	2.0	29.0	4.0
1394	1.0	32.0	1.0	5.0	4.0
1109	1.0	30.0	2.0	29.0	4.0
416	1.0	38.0	1.0	2.0	2.0
1234	1.0	47.0	2.0	2.0	4.0
687	1.0	36.0	1.0	2.0	4.0
…	…	…	…	…	…
1445	1.0	41.0	1.0	28.0	4.0
1201	1.0	23.0	1.0	8.0	1.0
99	1.0	44.0	1.0	23.0	3.0
850	1.0	32.0	2.0	2.0	1.0
448	1.0	40.0	1.0	6.0	3.0
755	1.0	45.0	2.0	11.0	2.0
976	1.0	56.0	1.0	23.0	3.0
115	1.0	37.0	2.0	3.0	3.0
777	1.0	21.0	1.0	10.0	3.0
72	1.0	31.0	1.0	1.0	4.0
845	1.0	40.0	1.0	26.0	2.0
537	1.0	27.0	1.0	10.0	2.0
849	1.0	43.0	2.0	9.0	3.0
174	1.0	45.0	2.0	4.0	2.0
87	1.0	51.0	1.0	9.0	4.0
551	1.0	39.0	3.0	3.0	3.0
705	1.0	39.0	2.0	2.0	5.0
314	1.0	39.0	1.0	10.0	1.0
1420	1.0	41.0	1.0	1.0	3.0
600	1.0	32.0	1.0	4.0	3.0
1094	1.0	40.0	2.0	9.0	2.0
599	1.0	36.0	3.0	13.0	3.0
277	1.0	38.0	2.0	7.0	2.0
1033	1.0	31.0	1.0	1.0	5.0
1383	1.0	36.0	1.0	9.0	4.0
763	1.0	34.0	2.0	10.0	4.0
835	1.0	35.0	3.0	8.0	4.0
1216	1.0	43.0	2.0	2.0	3.0
559	1.0	38.0	1.0	2.0	5.0
684	1.0	40.0	2.0	10.0	4.0

```
      EducationField  YearsAtCompany
```

338	3.0	10.0
363	3.0	1.0
759	2.0	6.0
793	1.0	4.0
581	1.0	2.0
320	1.0	5.0
452	4.0	8.0
195	1.0	8.0
776	3.0	2.0
1295	3.0	22.0
70	1.0	4.0
1135	1.0	26.0
1011	3.0	5.0
10	2.0	5.0
1265	5.0	9.0
1270	1.0	2.0
1257	3.0	1.0
271	1.0	10.0
858	2.0	7.0
790	1.0	3.0
1290	1.0	7.0
915	1.0	2.0
64	5.0	17.0
959	1.0	9.0
1274	3.0	12.0
1394	1.0	1.0
1109	5.0	4.0
416	1.0	1.0
1234	3.0	1.0
687	2.0	11.0
…	…	…
1445	1.0	20.0
1201	2.0	5.0
99	2.0	3.0
850	1.0	1.0
448	1.0	20.0
755	1.0	9.0
976	1.0	19.0
115	1.0	5.0
777	1.0	1.0
72	2.0	1.0
845	2.0	1.0
537	1.0	9.0
849	3.0	4.0
174	1.0	5.0
87	1.0	4.0
551	6.0	8.0
705	1.0	8.0

```
314                2.0             21.0
1420               1.0              5.0
600                1.0             14.0
1094               2.0              8.0
599                6.0              5.0
277                2.0              8.0
1033               1.0             10.0
1383               1.0              5.0
763                1.0              1.0
835                5.0              5.0
1216               2.0             10.0
559                2.0              1.0
684                3.0              1.0

[1029 rows x 7 columns]
```

[41]:

```
#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]]
print(model.predict_proba(kk))
```

```
[[7.14139240e-07 9.99999286e-01]]
```

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: