In [327…
```
# Importing libraries for Data handling and analysis
import numpy as np
import pandas as pd
import seaborn as sns

# Libraries for plotting

import matplotlib.pyplot as plt
%matplotlib inline

# Modelling Algorithms

from sklearn.linear_model import LogisticRegression

#Model building

from patsy import dmatrices
import sklearn
```

In [328…
```
# Importing the Dataset
df = pd.read_csv("IBM Attrition Data.csv")
```

In [329…
```
# Display first five rows of Data
df.head()
```

Out[329]:

	Age	Attrition	Department	DistanceFromHome	Education	EducationField	EnvironmentSatisfactio
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 [330…
```
#Displaying no:of rows and columns
df.shape
```

Out[330]:
```
(1470, 13)
```

In [331…
```
Names = df.columns.values
print(Names)
```

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

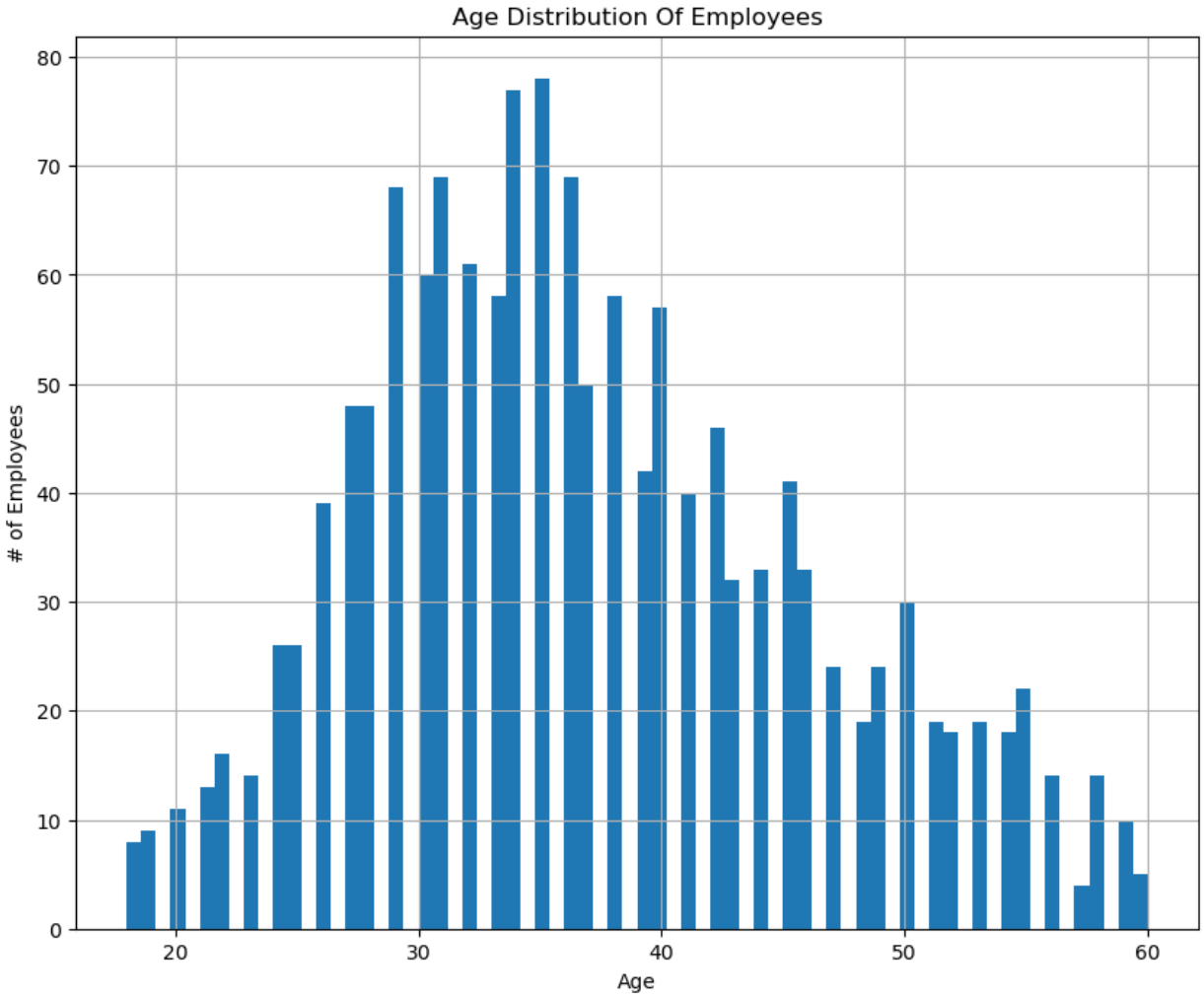
In [332…
```
# Checking for Missing values
df.isnull().sum()
```

Out[332]:
```
Age                       0
Attrition                 0
Department                0
DistanceFromHome          0
Education                 0
EducationField            0
EnvironmentSatisfaction   0
JobSatisfaction           0
MaritalStatus             0
MonthlyIncome             0
NumCompaniesWorked        0
WorkLifeBalance           0
YearsAtCompany            0
dtype: int64
```

In [333…
```
# Displaying the count of 'yes' and 'no' values of the target variable
df['Attrition'].value_counts()
```

Out[333]:
```
No     1233
Yes     237
Name: Attrition, dtype: int64
```

In [334…
```
#Histogram for Age
plt.figure(figsize=(10,8))
df['Age'].hist(bins=70)
plt.title("Age Distribution Of Employees")
plt.xlabel("Age")
plt.ylabel("# of Employees")
plt.show()
```

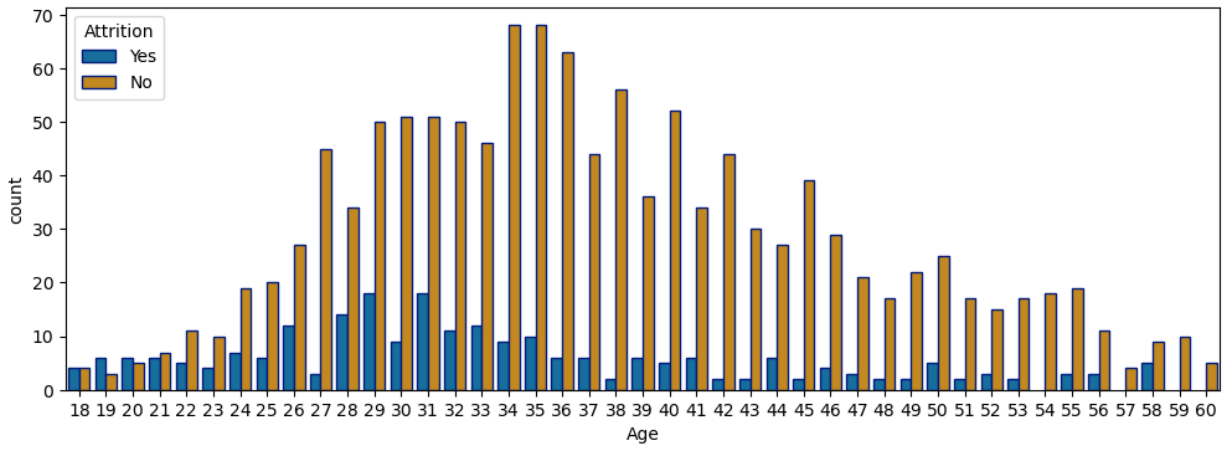


In [335…

```
# Explore Data for Attrition by Age
plt.figure(figsize=(14,10))
plt.scatter(df.Attrition,df.Age, alpha=.55)
plt.title("Attrition by Age ")
plt.ylabel("Age")
plt.grid(visible=True, which='major',axis='y')
plt.show()
```

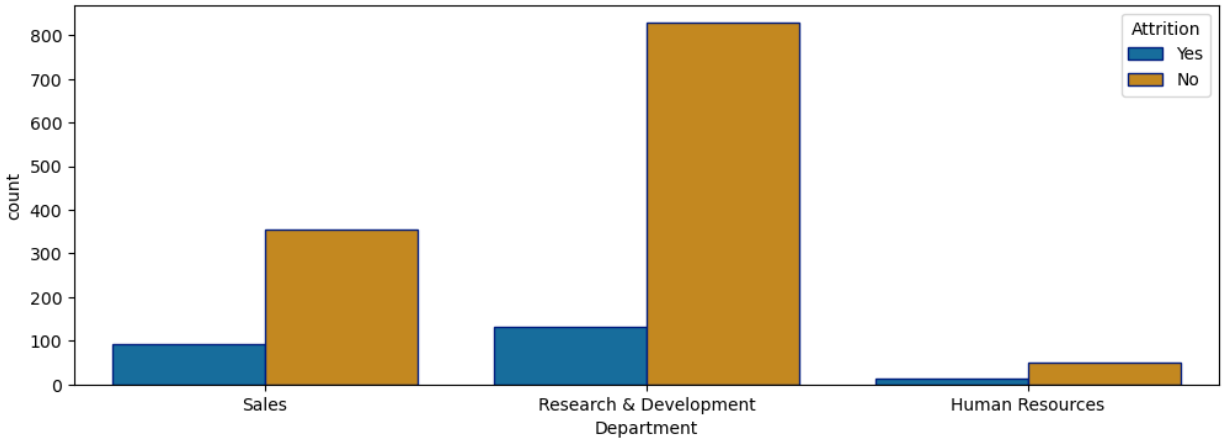
In [336…

```
# Show the number of employees that left and stayed by age
fig_dims = (12, 4)
fig, ax = plt.subplots(figsize=fig_dims)
sns.countplot(x='Age', hue='Attrition', data=df, palette="colorblind", ax=ax, edgecolo
```

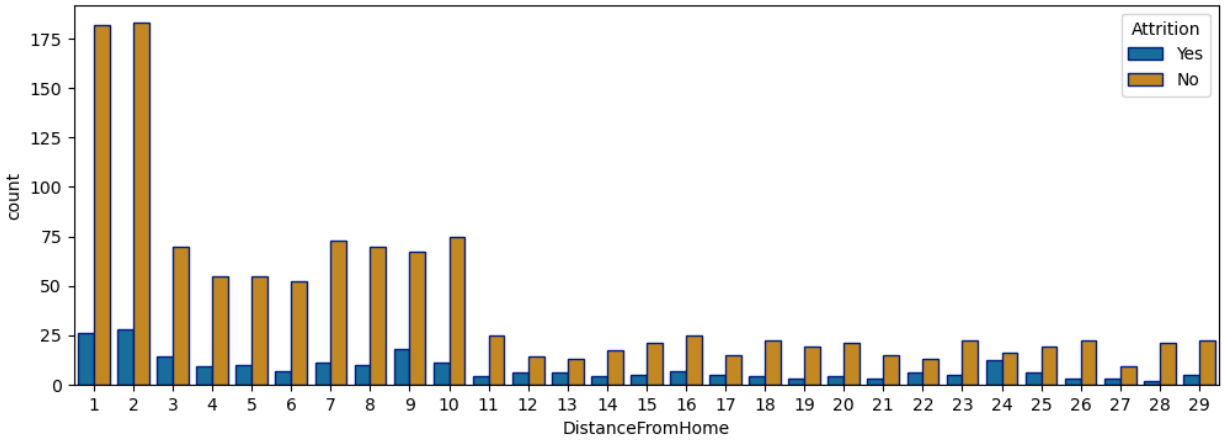


In [337…

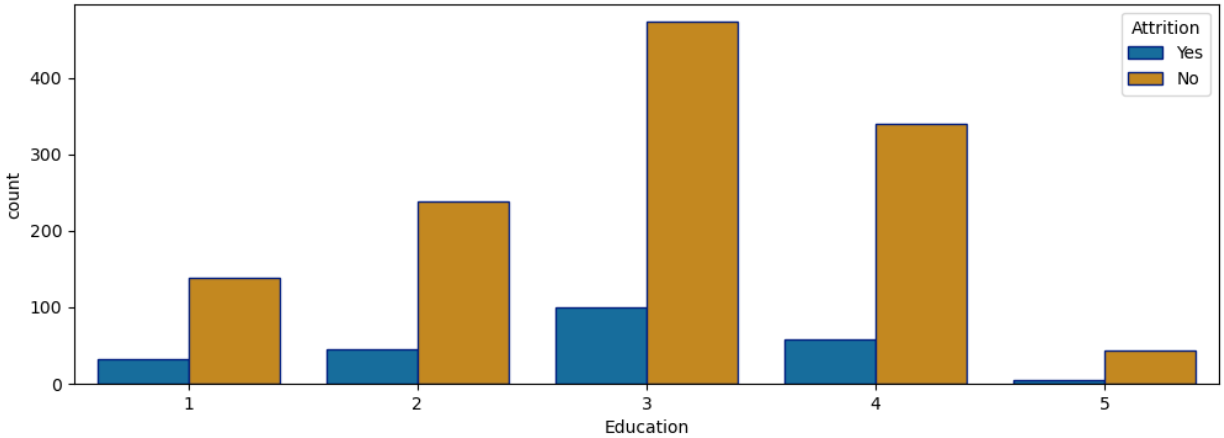
```
# Show the number of employees that left and stayed by Department
fig_dims = (12, 4)
fig, ax = plt.subplots(figsize=fig_dims)
sns.countplot(x='Department', hue='Attrition', data=df, palette="colorblind", ax=ax, e
```



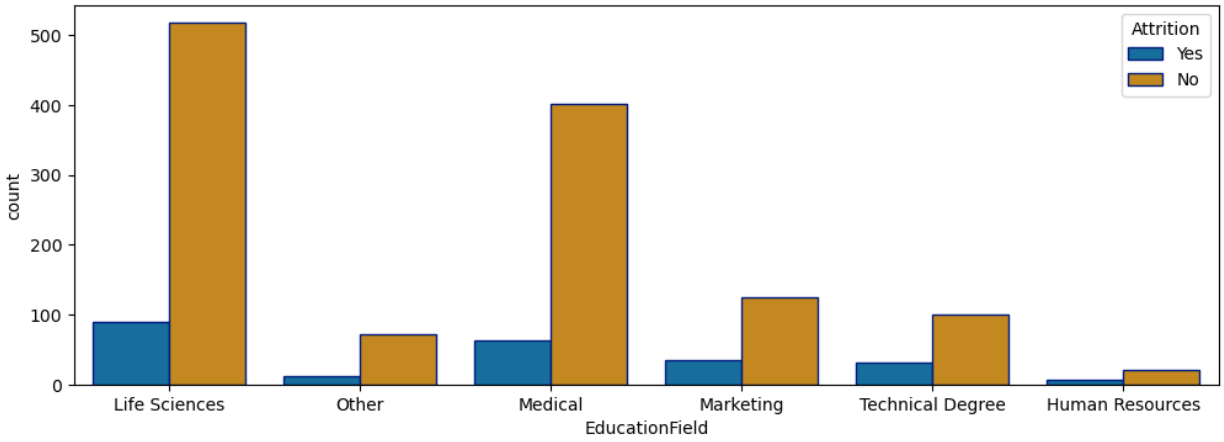
In [338…
```
# Show the number of employees that left and stayed by DistanceFromHome
fig_dims = (12, 4)
fig, ax = plt.subplots(figsize=fig_dims)
sns.countplot(x='DistanceFromHome', hue='Attrition', data=df, palette="colorblind", ax
```



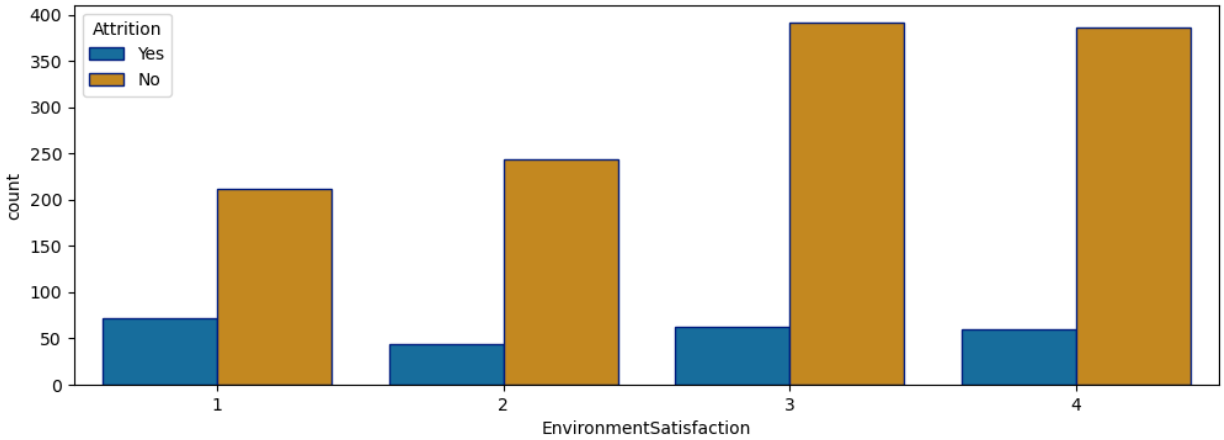
In [339…
```
# Show the number of employees that left and stayed by Education
fig_dims = (12, 4)
fig, ax = plt.subplots(figsize=fig_dims)
sns.countplot(x='Education', hue='Attrition', data=df, palette="colorblind", ax=ax, ed
```



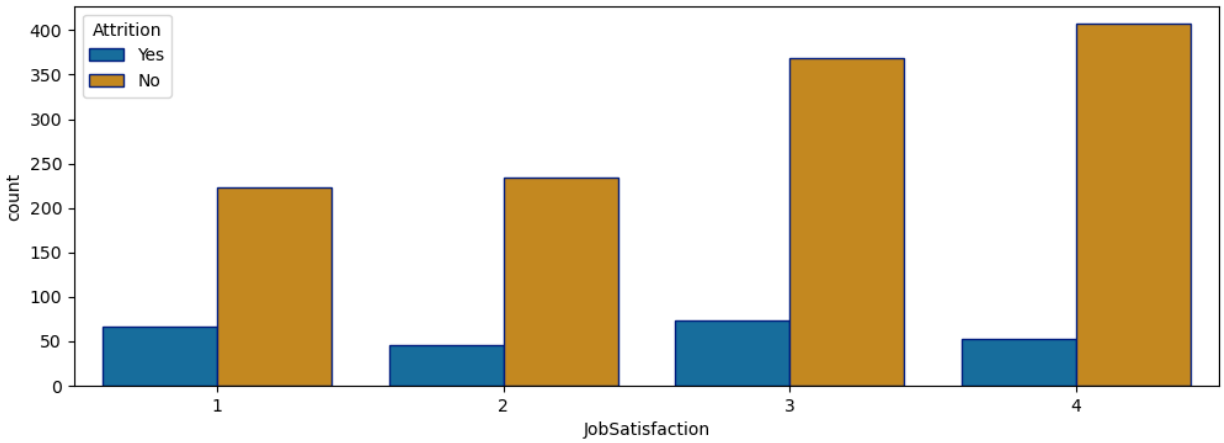
In [340…
```
# Show the number of employees that left and stayed by EducationField
fig_dims = (12, 4)
fig, ax = plt.subplots(figsize=fig_dims)
sns.countplot(x='EducationField', hue='Attrition', data=df, palette="colorblind", ax=a
```



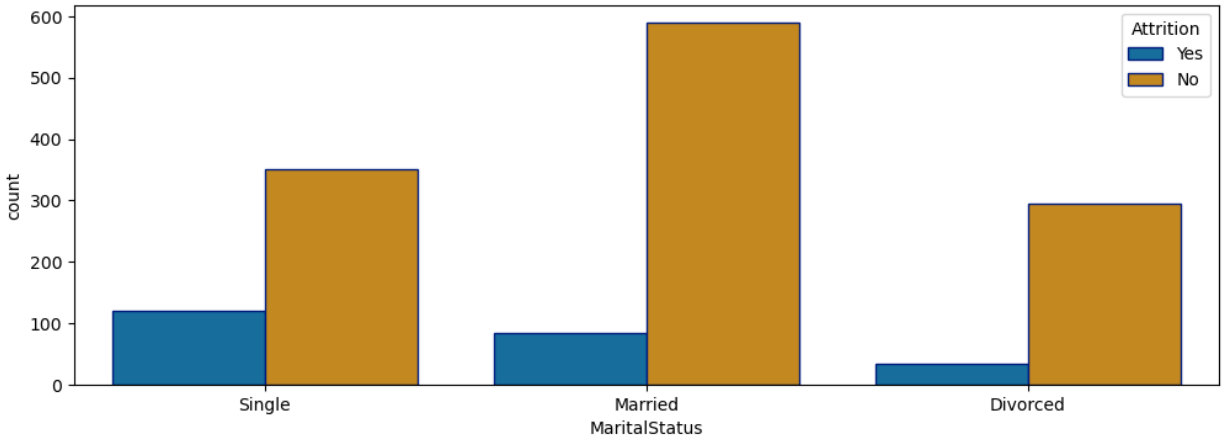
In [341…
```
# Show the number of employees that left and stayed by EnvironmentSatisfaction
fig_dims = (12, 4)
fig, ax = plt.subplots(figsize=fig_dims)
sns.countplot(x='EnvironmentSatisfaction', hue='Attrition', data=df, palette="colorbli
```



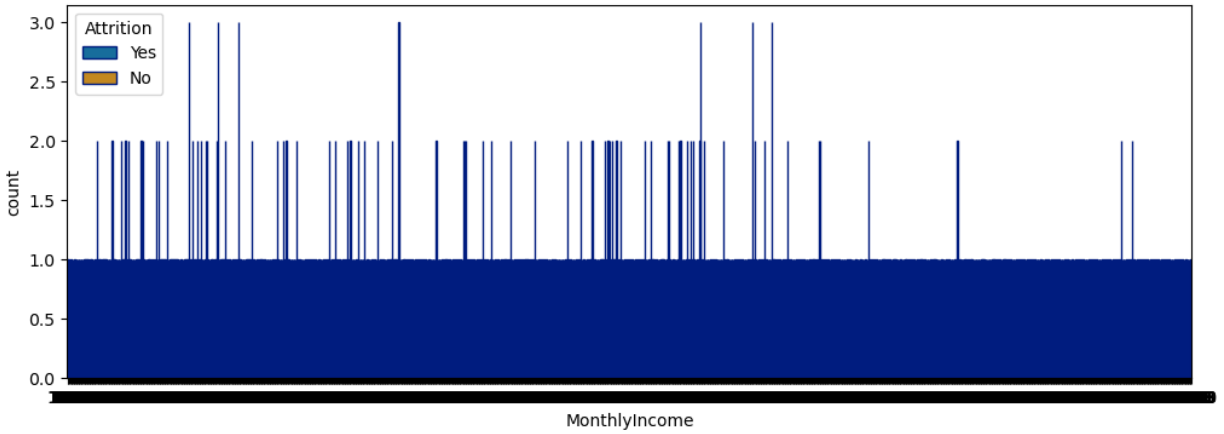
In [342…
```
# Show the number of employees that left and stayed by JobSatisfaction
fig_dims = (12, 4)
fig, ax = plt.subplots(figsize=fig_dims)
sns.countplot(x='JobSatisfaction', hue='Attrition', data=df, palette="colorblind", ax=
```



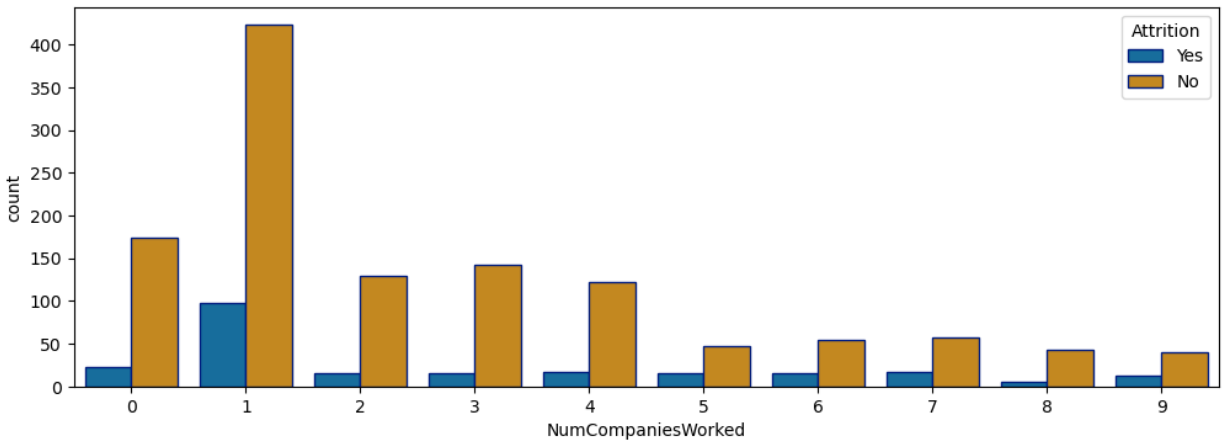
In [343…
```
# Show the number of employees that left and stayed by MaritalStatus
fig_dims = (12, 4)
fig, ax = plt.subplots(figsize=fig_dims)
sns.countplot(x='MaritalStatus', hue='Attrition', data=df, palette="colorblind", ax=ax
```



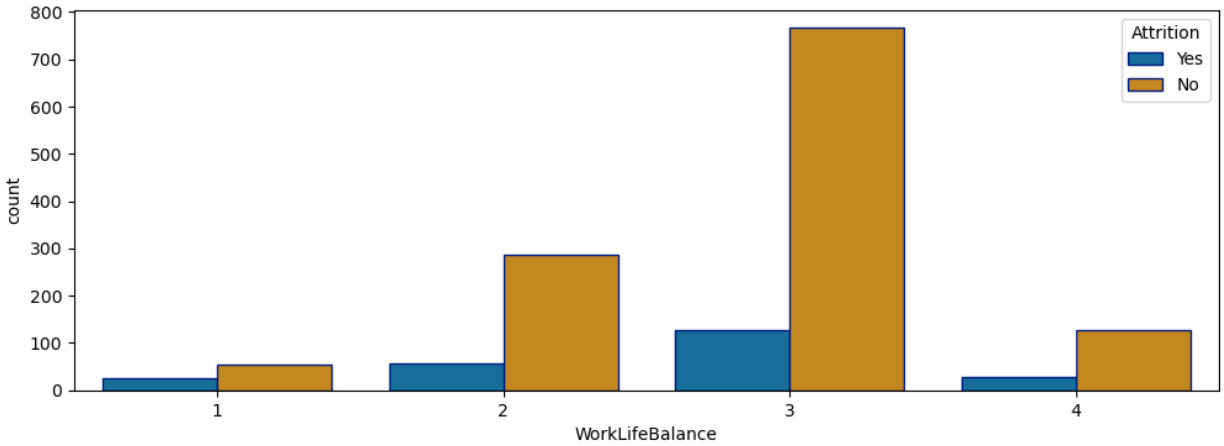
In [344…
```
# Show the number of employees that left and stayed by MonthlyIncome
fig_dims = (12, 4)
fig, ax = plt.subplots(figsize=fig_dims)
sns.countplot(x='MonthlyIncome', hue='Attrition', data=df, palette="colorblind", ax=ax
```



In [345…
```
# Show the number of employees that left and stayed by Number of Companies worked
fig_dims = (12, 4)
fig, ax = plt.subplots(figsize=fig_dims)
sns.countplot(x='NumCompaniesWorked', hue='Attrition', data=df, palette="colorblind",
```

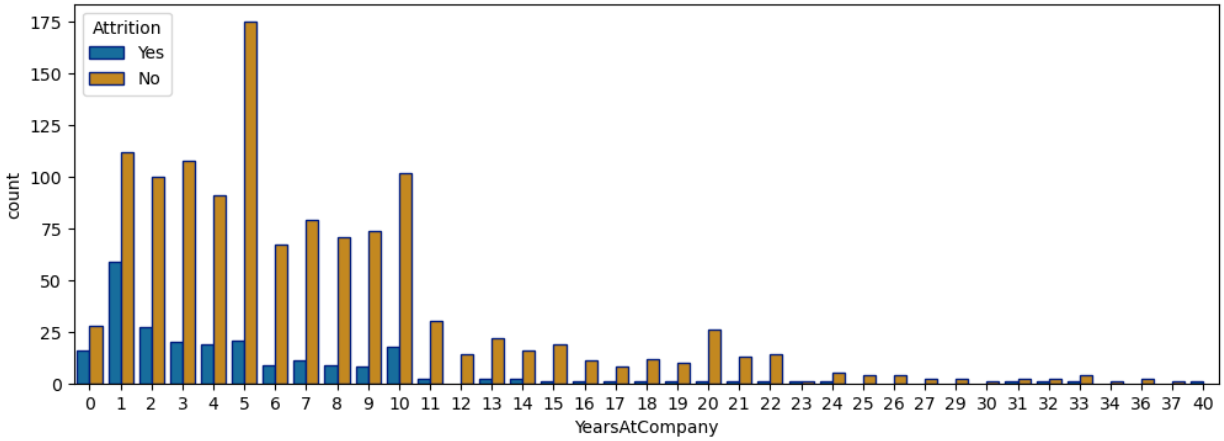


In [346…
```
# Show the number of employees that left and stayed by Work life balance
fig_dims = (12, 4)
fig, ax = plt.subplots(figsize=fig_dims)
sns.countplot(x='WorkLifeBalance', hue='Attrition', data=df, palette="colorblind", ax=
```



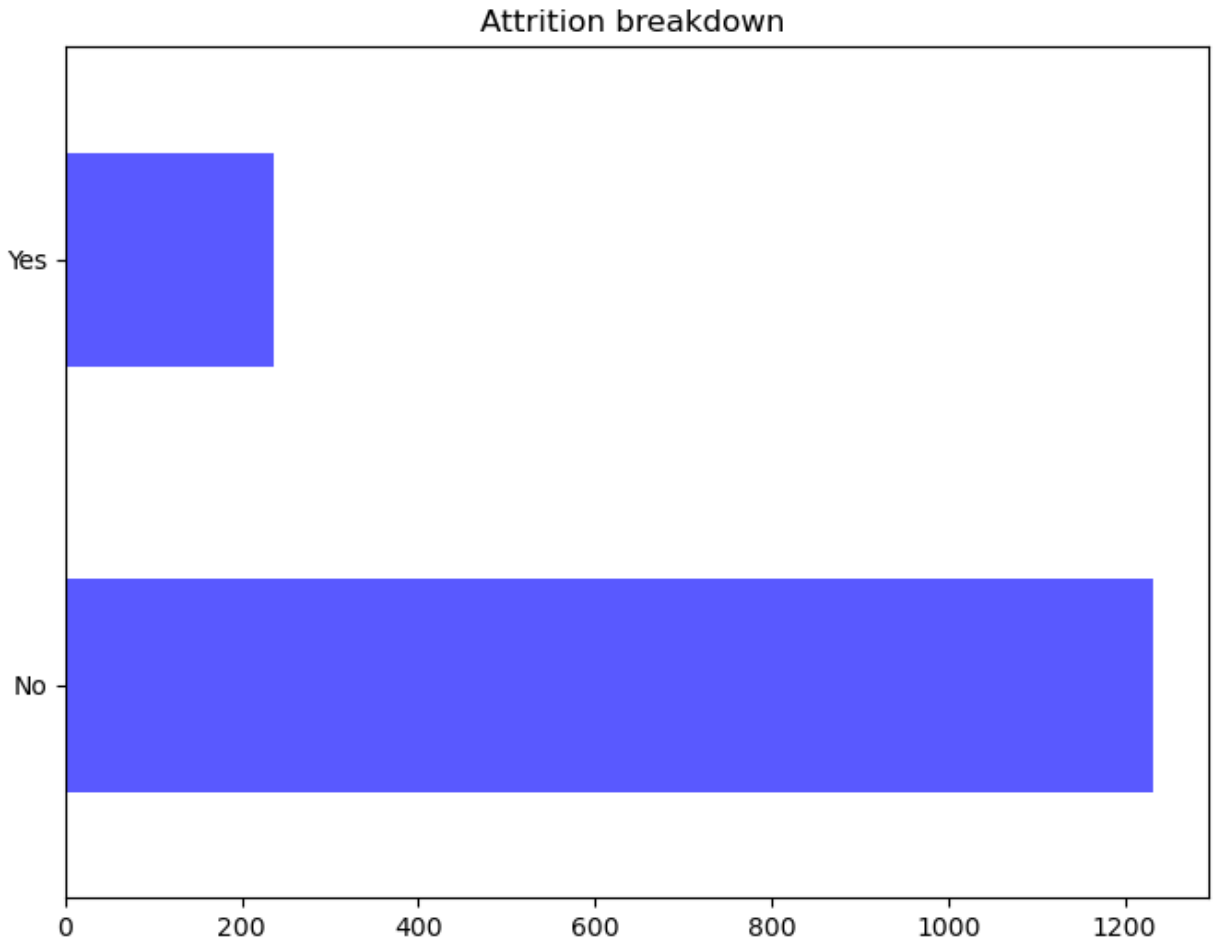
In [347…

```
# Show the number of employees that left and stayed by Years worked at company
fig_dims = (12, 4)
fig, ax = plt.subplots(figsize=fig_dims)
sns.countplot(x='YearsAtCompany', hue='Attrition', data=df, palette="colorblind", ax=a
```



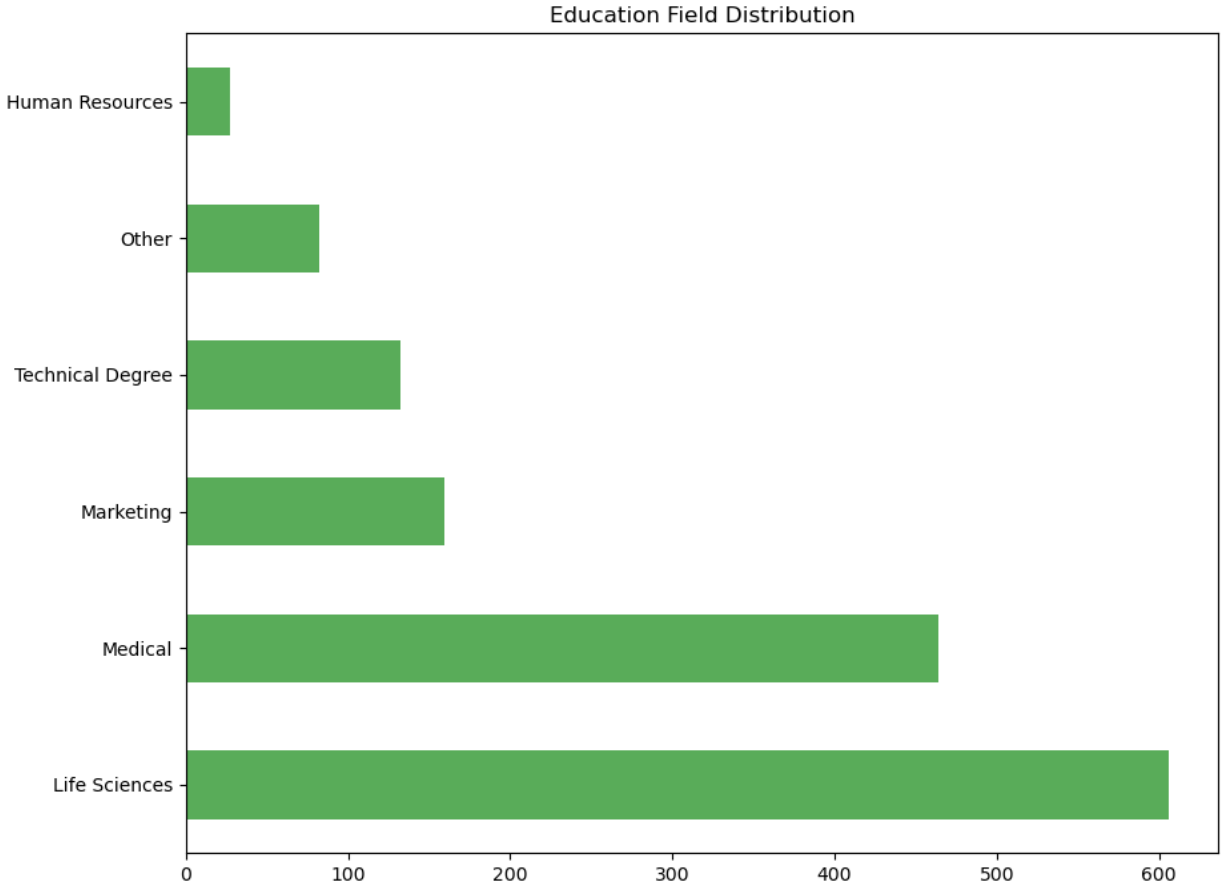
In [348…

```
# Explore Data for left Employees Breakdown
plt.figure(figsize=(8,6))
df.Attrition.value_counts().plot(kind='barh',color='blue',alpha=.65)
plt.title("Attrition breakdown ")
plt.show()
```



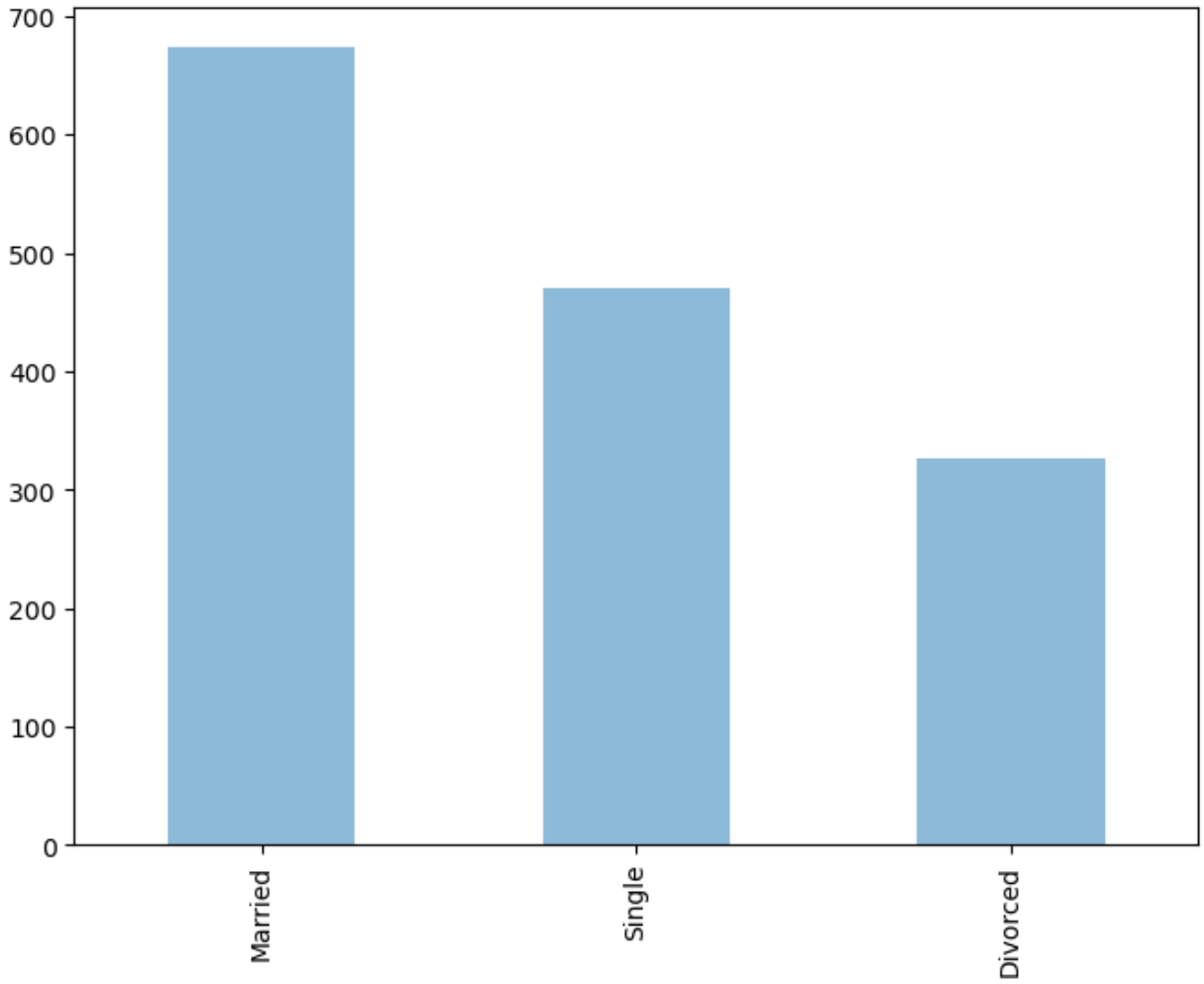
In [349…

```
# Explore Data for Education Field distribution
plt.figure(figsize=(10,8))
df.EducationField.value_counts().plot(kind='barh',color='g',alpha=.65)
plt.title("Education Field Distribution")
plt.show()
```



In [350…

```
# Explore Data for Marital Status
plt.figure(figsize=(8,6))
df.MaritalStatus.value_counts().plot(kind='bar',alpha=.5)
plt.show()
```



In [351…
```
df.describe()
```

Out[351]:

	Age	DistanceFromHome	Education	EnvironmentSatisfaction	JobSatisfaction	Month
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	147
mean	36.923810	9.192517	2.912925	2.721769	2.728571	65(
std	9.135373	8.106864	1.024165	1.093082	1.102846	47(
min	18.000000	1.000000	1.000000	1.000000	1.000000	10(
25%	30.000000	2.000000	2.000000	2.000000	2.000000	29'
50%	36.000000	7.000000	3.000000	3.000000	3.000000	49'
75%	43.000000	14.000000	4.000000	4.000000	4.000000	83'
max	60.000000	29.000000	5.000000	4.000000	4.000000	199

In [352…
```
df.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 [353…
```
df.columns
```

Out[353]:
```
Index(['Age', 'Attrition', 'Department', 'DistanceFromHome', 'Education',
       'EducationField', 'EnvironmentSatisfaction', 'JobSatisfaction',
       'MaritalStatus', 'MonthlyIncome', 'NumCompaniesWorked',
       'WorkLifeBalance', 'YearsAtCompany'],
      dtype='object')
```

In [354…
```
df.std()
```

```
C:\Users\beemr\AppData\Local\Temp\ipykernel_2420\3390915376.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  df.std()
```

Out[354]:
```
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 [355…
```
df['Attrition'].value_counts()
```

Out[355]:
```
No     1233
Yes     237
Name: Attrition, dtype: int64
```

In [356…
```
df['Attrition'].dtypes
```

Out[356]:
```
dtype('O')
```

In [357…
```
df['Attrition'].replace('Yes',1, inplace=True)
df['Attrition'].replace('No',0, inplace=True)
```

In [358…
```
df.head(10)
```

Out[358]:

	Age	Attrition	Department	DistanceFromHome	Education	EducationField	EnvironmentSatisfactio
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	
5	32	0	Research & Development	2	2	Life Sciences	
6	59	0	Research & Development	3	3	Medical	
7	30	0	Research & Development	24	1	Life Sciences	
8	38	0	Research & Development	23	3	Life Sciences	
9	36	0	Research & Development	27	3	Medical	

In [359…
```
# Building Up a Logistic Regression Model
X = df.drop(['Attrition'],axis=1)
X.head()
Y = df['Attrition']
Y.head()
```

Out[359]:
```
0    1
1    0
2    1
3    0
4    0
Name: Attrition, dtype: int64
```

In [360…
```
df['EducationField'].replace('Life Sciences',1, inplace=True)
df['EducationField'].replace('Medical',2, inplace=True)
df['EducationField'].replace('Marketing', 3, inplace=True)
df['EducationField'].replace('Other',4, inplace=True)
df['EducationField'].replace('Technical Degree',5, inplace=True)
df['EducationField'].replace('Human Resources', 6, inplace=True)
```

In [361…
```
df['EducationField'].value_counts()
```

Out[361]:
```
1    606
2    464
3    159
5    132
4     82
6     27
Name: EducationField, dtype: int64
```

In [362…
```
df['Department'].value_counts()
```

Out[362]:
```
Research & Development    961
Sales                     446
Human Resources            63
Name: Department, dtype: int64
```

In [363…
```
df['Department'].replace('Research & Development',1, inplace=True)
df['Department'].replace('Sales',2, inplace=True)
df['Department'].replace('Human Resources', 3, inplace=True)
```

In [364…
```
df['Department'].value_counts()
```

Out[364]:
```
1    961
2    446
3     63
Name: Department, dtype: int64
```

In [365…
```
df['MaritalStatus'].value_counts()
```

Out[365]:
```
Married     673
Single      470
Divorced    327
Name: MaritalStatus, dtype: int64
```

In [366…
```
df['MaritalStatus'].replace('Married',1, inplace=True)
df['MaritalStatus'].replace('Single',2, inplace=True)
df['MaritalStatus'].replace('Divorced',3, inplace=True)
```

In [367…
```
df['MaritalStatus'].value_counts()
```

Out[367]:
```
1    673
2    470
3    327
Name: MaritalStatus, dtype: int64
```

In [368…
```
x=df.select_dtypes(include=['int64'])
x.dtypes
```

Out[368]:
```
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
```

In [369…
```
x.columns
```

Out[369]:
```
Index(['Age', 'Attrition', 'Department', 'DistanceFromHome', 'Education',
       'EducationField', 'EnvironmentSatisfaction', 'JobSatisfaction',
       'MaritalStatus', 'MonthlyIncome', 'NumCompaniesWorked',
       'WorkLifeBalance', 'YearsAtCompany'],
      dtype='object')
```

In [370…
```
y=df['Attrition']
```

In [371…
```
y.head()
```

Out[371]:
```
0    1
1    0
2    1
3    0
4    0
Name: Attrition, dtype: int64
```

In [372…
```
y, x = dmatrices('Attrition ~ Age + Department + \
                  DistanceFromHome + Education + EducationField + YearsAtCompany',
                  df, return_type="dataframe")
print (x.columns)
```

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

In [373…
```
# COrrelation of the columns
df.corr()
```

Out[373]:

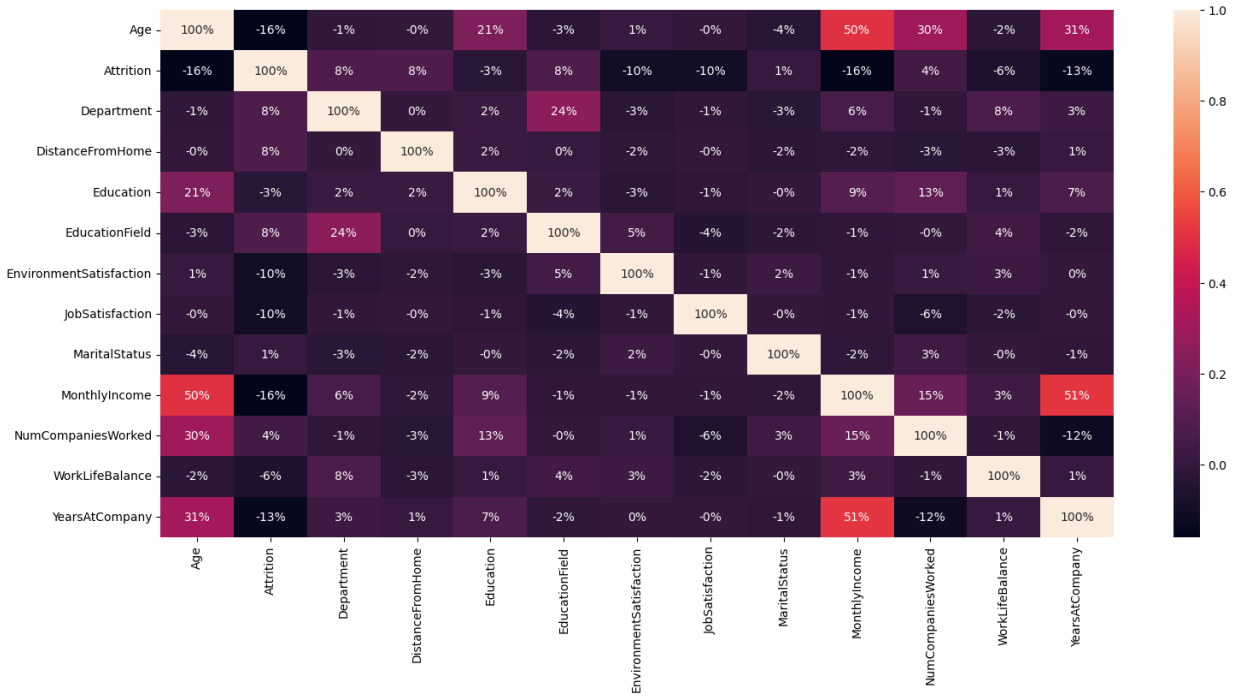
	Age	Attrition	Department	DistanceFromHome	Education	EducationF
Age	1.000000	-0.159205	-0.007652	-0.001686	0.208034	-0.028
Attrition	-0.159205	1.000000	0.077351	0.077924	-0.031373	0.077
Department	-0.007652	0.077351	1.000000	0.002196	0.019636	0.243
DistanceFromHome	-0.001686	0.077924	0.002196	1.000000	0.021042	0.004
Education	0.208034	-0.031373	0.019636	0.021042	1.000000	0.018
EducationField	-0.028312	0.077232	0.243641	0.004815	0.018328	1.000
EnvironmentSatisfaction	0.010146	-0.103369	-0.026110	-0.016075	-0.027128	0.045
JobSatisfaction	-0.004892	-0.103481	-0.006231	-0.003669	-0.011296	-0.044
MaritalStatus	-0.035466	0.011195	-0.030818	-0.021916	-0.000107	-0.023
MonthlyIncome	0.497855	-0.159840	0.056573	-0.017014	0.094961	-0.013
NumCompaniesWorked	0.299635	0.043494	-0.011261	-0.029251	0.126317	-0.002
WorkLifeBalance	-0.021490	-0.063939	0.075507	-0.026556	0.009819	0.041
YearsAtCompany	0.311309	-0.134392	0.029752	0.009508	0.069114	-0.018

In [374…

```
# visualize the correlation
plt.figure(figsize=(18,8))
sns.heatmap(df.corr(), annot=True, fmt='.0%')
```

Out[374]:

```
<AxesSubplot:>
```



In [375…

```
y = np.ravel(y)
```

In [376…

```
from sklearn.linear_model import LogisticRegression
```

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

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

Out[376]:
```
0.8408163265306122
```

In [377…
```
y.mean()
```

Out[377]:
```
0.16122448979591836
```

In [378…
```
X_train,X_test,y_train,y_test=sklearn.model_selection.train_test_split(x,y, test_size=
model2=LogisticRegression()
model2.fit(X_train, y_train)
```

Out[378]:
```
LogisticRegression()
```

In [380…
```
probs = model2.predict_proba(X_test)
print (probs)
```

```
[[0.86179633 0.13820367]
 [0.80754596 0.19245404]
 [0.74123965 0.25876035]
 [0.83441328 0.16558672]
 [0.73499948 0.26500052]
 [0.79097755 0.20902245]
 [0.85615206 0.14384794]
 [0.85699677 0.14300323]
 [0.96699051 0.03300949]
 [0.93685198 0.06314802]
 [0.95099263 0.04900737]
 [0.83101542 0.16898458]
 [0.86296549 0.13703451]
 [0.86581192 0.13418808]
 [0.8875059  0.1124941 ]
 [0.88892618 0.11107382]
 [0.88569715 0.11430285]
 [0.78516592 0.21483408]
 [0.79794491 0.20205509]
 [0.88511291 0.11488709]
 [0.70651623 0.29348377]
 [0.94676682 0.05323318]
 [0.86736257 0.13263743]
 [0.84276459 0.15723541]
 [0.60336899 0.39663101]
 [0.81129201 0.18870799]
 [0.9181372  0.0818628 ]
 [0.93285517 0.06714483]
 [0.68230795 0.31769205]
 [0.87027127 0.12972873]
 [0.8726638  0.1273362 ]
 [0.76968749 0.23031251]
 [0.86435752 0.13564248]
 [0.9575887  0.0424113 ]
 [0.8446148  0.1553852 ]
 [0.86719354 0.13280646]
 [0.90465967 0.09534033]
 [0.68936441 0.31063559]
 [0.90703609 0.09296391]
 [0.80663487 0.19336513]
 [0.91515711 0.08484289]
 [0.82351291 0.17648709]
 [0.93711506 0.06288494]
 [0.93411321 0.06588679]
 [0.89447647 0.10552353]
 [0.85317745 0.14682255]
 [0.78922389 0.21077611]
 [0.84879884 0.15120116]
 [0.66402495 0.33597505]
 [0.76252316 0.23747684]
 [0.928511   0.071489  ]
 [0.78953699 0.21046301]
 [0.86166603 0.13833397]
 [0.85837897 0.14162103]
 [0.87217671 0.12782329]
 [0.78950907 0.21049093]
 [0.87690786 0.12309214]
 [0.84165437 0.15834563]
 [0.72847193 0.27152807]
 [0.83181423 0.16818577]
```

```
[0.90095041 0.09904959]
[0.71077357 0.28922643]
[0.92823012 0.07176988]
[0.84375688 0.15624312]
[0.79544117 0.20455883]
[0.86826144 0.13173856]
[0.91679447 0.08320553]
[0.84763055 0.15236945]
[0.89253705 0.10746295]
[0.62872181 0.37127819]
[0.93875379 0.06124621]
[0.72620352 0.27379648]
[0.85652973 0.14347027]
[0.84226021 0.15773979]
[0.77436416 0.22563584]
[0.71899563 0.28100437]
[0.93587374 0.06412626]
[0.95710059 0.04289941]
[0.79185866 0.20814134]
[0.89370433 0.10629567]
[0.91382029 0.08617971]
[0.79354593 0.20645407]
[0.77934037 0.22065963]
[0.79639027 0.20360973]
[0.83800486 0.16199514]
[0.71395696 0.28604304]
[0.97772708 0.02227292]
[0.94645961 0.05354039]
[0.88617618 0.11382382]
[0.79620158 0.20379842]
[0.6186388  0.3813612 ]
[0.81866466 0.18133534]
[0.74504158 0.25495842]
[0.86779494 0.13220506]
[0.87071136 0.12928864]
[0.81717483 0.18282517]
[0.71840761 0.28159239]
[0.59825956 0.40174044]
[0.83951537 0.16048463]
[0.8835132  0.1164868 ]
[0.74352615 0.25647385]
[0.76631604 0.23368396]
[0.98033031 0.01966969]
[0.91857458 0.08142542]
[0.77432842 0.22567158]
[0.92514804 0.07485196]
[0.88123382 0.11876618]
[0.74587212 0.25412788]
[0.90478357 0.09521643]
[0.78685556 0.21314444]
[0.8114777  0.1885223 ]
[0.93472169 0.06527831]
[0.93836492 0.06163508]
[0.79411747 0.20588253]
[0.81372909 0.18627091]
[0.91610913 0.08389087]
[0.90428335 0.09571665]
[0.84669432 0.15330568]
[0.95384539 0.04615461]
[0.91283686 0.08716314]
```

```
[0.85919591 0.14080409]
[0.85902506 0.14097494]
[0.87519521 0.12480479]
[0.76114683 0.23885317]
[0.92217678 0.07782322]
[0.96859402 0.03140598]
[0.94398205 0.05601795]
[0.81780303 0.18219697]
[0.88058698 0.11941302]
[0.77894275 0.22105725]
[0.97124457 0.02875543]
[0.8880766  0.1119234 ]
[0.78715257 0.21284743]
[0.82001488 0.17998512]
[0.94934532 0.05065468]
[0.95888924 0.04111076]
[0.73559255 0.26440745]
[0.93416982 0.06583018]
[0.7375067  0.2624933 ]
[0.82136765 0.17863235]
[0.82171199 0.17828801]
[0.898967   0.101033  ]
[0.78745754 0.21254246]
[0.8982534  0.1017466 ]
[0.91433807 0.08566193]
[0.92724742 0.07275258]
[0.96594958 0.03405042]
[0.94417351 0.05582649]
[0.93073074 0.06926926]
[0.66320621 0.33679379]
[0.84168659 0.15831341]
[0.82636823 0.17363177]
[0.80616646 0.19383354]
[0.96157626 0.03842374]
[0.93515156 0.06484844]
[0.94778611 0.05221389]
[0.97337817 0.02662183]
[0.7929708  0.2070292 ]
[0.87770188 0.12229812]
[0.86103662 0.13896338]
[0.95185473 0.04814527]
[0.93131494 0.06868506]
[0.75685375 0.24314625]
[0.74997821 0.25002179]
[0.95527857 0.04472143]
[0.86950711 0.13049289]
[0.81376776 0.18623224]
[0.7699062  0.2300938 ]
[0.80077924 0.19922076]
[0.92801113 0.07198887]
[0.90949237 0.09050763]
[0.94557615 0.05442385]
[0.93330455 0.06669545]
[0.6914595  0.3085405 ]
[0.93056868 0.06943132]
[0.74525455 0.25474545]
[0.78594357 0.21405643]
[0.93308592 0.06691408]
[0.80878286 0.19121714]
[0.85096455 0.14903545]
```

```
[0.66956309 0.33043691]
[0.90338376 0.09661624]
[0.91158921 0.08841079]
[0.87300498 0.12699502]
[0.92939047 0.07060953]
[0.66661769 0.33338231]
[0.89098506 0.10901494]
[0.86216714 0.13783286]
[0.78838554 0.21161446]
[0.53099689 0.46900311]
[0.73344347 0.26655653]
[0.71053996 0.28946004]
[0.85530485 0.14469515]
[0.869606   0.130394  ]
[0.75470011 0.24529989]
[0.89823505 0.10176495]
[0.79247854 0.20752146]
[0.90643122 0.09356878]
[0.7765078  0.2234922 ]
[0.88387785 0.11612215]
[0.85404311 0.14595689]
[0.81899383 0.18100617]
[0.74448547 0.25551453]
[0.86259438 0.13740562]
[0.7782263  0.2217737 ]
[0.76928335 0.23071665]
[0.79679238 0.20320762]
[0.92096588 0.07903412]
[0.74573789 0.25426211]
[0.87499218 0.12500782]
[0.85503285 0.14496715]
[0.7729218  0.2270782 ]
[0.87227276 0.12772724]
[0.67395392 0.32604608]
[0.93621654 0.06378346]
[0.82479478 0.17520522]
[0.95146039 0.04853961]
[0.83462456 0.16537544]
[0.81106466 0.18893534]
[0.8082827  0.1917173 ]
[0.87664145 0.12335855]
[0.66787014 0.33212986]
[0.59616875 0.40383125]
[0.98979359 0.01020641]
[0.7063442  0.2936558 ]
[0.91633074 0.08366926]
[0.92133466 0.07866534]
[0.71383914 0.28616086]
[0.62454002 0.37545998]
[0.76440588 0.23559412]
[0.95401642 0.04598358]
[0.8818672  0.1181328 ]
[0.86019894 0.13980106]
[0.92117374 0.07882626]
[0.88019162 0.11980838]
[0.8045145  0.1954855 ]
[0.80585191 0.19414809]
[0.9134548  0.0865452 ]
[0.72101822 0.27898178]
[0.94507993 0.05492007]
```

```
[0.90906895 0.09093105]
[0.7338177  0.2661823 ]
[0.9810918  0.0189082 ]
[0.85490455 0.14509545]
[0.89817059 0.10182941]
[0.82436995 0.17563005]
[0.83387091 0.16612909]
[0.88136631 0.11863369]
[0.8803131  0.1196869 ]
[0.87560623 0.12439377]
[0.81617196 0.18382804]
[0.8802884  0.1197116 ]
[0.61530879 0.38469121]
[0.88826146 0.11173854]
[0.89692801 0.10307199]
[0.85328926 0.14671074]
[0.98297036 0.01702964]
[0.77341256 0.22658744]
[0.62184915 0.37815085]
[0.82739138 0.17260862]
[0.84083343 0.15916657]
[0.84808619 0.15191381]
[0.84937023 0.15062977]
[0.75541027 0.24458973]
[0.86188862 0.13811138]
[0.90684678 0.09315322]
[0.84663282 0.15336718]
[0.81290917 0.18709083]
[0.74053615 0.25946385]
[0.87035273 0.12964727]
[0.83938103 0.16061897]
[0.86267954 0.13732046]
[0.66362941 0.33637059]
[0.90671714 0.09328286]
[0.8705628  0.1294372 ]
[0.92580781 0.07419219]
[0.84386871 0.15613129]
[0.89878143 0.10121857]
[0.91239484 0.08760516]
[0.79867648 0.20132352]
[0.64052612 0.35947388]
[0.84683949 0.15316051]
[0.75503217 0.24496783]
[0.8526455  0.1473545 ]
[0.99251429 0.00748571]
[0.86138207 0.13861793]
[0.88140133 0.11859867]
[0.8275709  0.1724291 ]
[0.93027065 0.06972935]
[0.87307447 0.12692553]
[0.88802391 0.11197609]
[0.83753339 0.16246661]
[0.86481034 0.13518966]
[0.86515542 0.13484458]
[0.89754783 0.10245217]
[0.78409068 0.21590932]
[0.7916512  0.2083488 ]
[0.8838813  0.1161187 ]
[0.65418989 0.34581011]
[0.94059367 0.05940633]
```

```
[0.89913431 0.10086569]
[0.72525643 0.27474357]
[0.69085557 0.30914443]
[0.87591628 0.12408372]
[0.86562237 0.13437763]
[0.9750887  0.0249113 ]
[0.8626512  0.1373488 ]
[0.54036129 0.45963871]
[0.9130226  0.0869774 ]
[0.74609613 0.25390387]
[0.86758148 0.13241852]
[0.88852759 0.11147241]
[0.87911717 0.12088283]
[0.85664968 0.14335032]
[0.77281975 0.22718025]
[0.80713969 0.19286031]
[0.8528551  0.1471449 ]
[0.77595541 0.22404459]
[0.70409829 0.29590171]
[0.88971657 0.11028343]
[0.48963171 0.51036829]
[0.92370638 0.07629362]
[0.75572324 0.24427676]
[0.67788914 0.32211086]
[0.91301002 0.08698998]
[0.94029209 0.05970791]
[0.88009777 0.11990223]
[0.88591154 0.11408846]
[0.95656815 0.04343185]
[0.89909415 0.10090585]
[0.94784639 0.05215361]
[0.83255786 0.16744214]
[0.87822133 0.12177867]
[0.81945812 0.18054188]
[0.81668795 0.18331205]
[0.95036158 0.04963842]
[0.86940137 0.13059863]
[0.90541587 0.09458413]
[0.83661182 0.16338818]
[0.84583661 0.15416339]
[0.79307985 0.20692015]
[0.81808129 0.18191871]
[0.81598424 0.18401576]
[0.8367757  0.1632243 ]
[0.91443027 0.08556973]
[0.91533086 0.08466914]
[0.68503469 0.31496531]
[0.99073754 0.00926246]
[0.76795027 0.23204973]
[0.79871943 0.20128057]
[0.73176835 0.26823165]
[0.67465235 0.32534765]
[0.79712478 0.20287522]
[0.84935622 0.15064378]
[0.86467087 0.13532913]
[0.85844147 0.14155853]
[0.84533258 0.15466742]
[0.83038794 0.16961206]
[0.92192742 0.07807258]
[0.83189723 0.16810277]
```

```
[0.97707544 0.02292456]
[0.90418217 0.09581783]
[0.92691192 0.07308808]
[0.84797365 0.15202635]
[0.76712583 0.23287417]
[0.94895148 0.05104852]
[0.94782078 0.05217922]
[0.75181532 0.24818468]
[0.87880471 0.12119529]
[0.80770781 0.19229219]
[0.93864294 0.06135706]
[0.86008496 0.13991504]
[0.76056741 0.23943259]
[0.90892464 0.09107536]
[0.75402142 0.24597858]
[0.94271835 0.05728165]
[0.91827317 0.08172683]
[0.90545126 0.09454874]
[0.76879475 0.23120525]
[0.92358128 0.07641872]
[0.80705893 0.19294107]
[0.90079524 0.09920476]
[0.87933316 0.12066684]
[0.80585716 0.19414284]
[0.83150076 0.16849924]
[0.53816822 0.46183178]
[0.95031818 0.04968182]
[0.73291935 0.26708065]
[0.89182876 0.10817124]
[0.80079862 0.19920138]
[0.87739799 0.12260201]
[0.96805225 0.03194775]
[0.81741595 0.18258405]
[0.86150081 0.13849919]
[0.59407595 0.40592405]
[0.82625726 0.17374274]
[0.92534964 0.07465036]
[0.81692208 0.18307792]
[0.92586778 0.07413222]
[0.89094582 0.10905418]
[0.70071794 0.29928206]
[0.82181894 0.17818106]
[0.96589931 0.03410069]
[0.8699699  0.1300301 ]
[0.89918541 0.10081459]
[0.88983811 0.11016189]
[0.8143247  0.1856753 ]
[0.85830028 0.14169972]
[0.83878647 0.16121353]
[0.84056579 0.15943421]
[0.82661162 0.17338838]
[0.94075156 0.05924844]
[0.83183532 0.16816468]
[0.77561364 0.22438636]
[0.69399566 0.30600434]
[0.85962131 0.14037869]
[0.82513939 0.17486061]
[0.84107719 0.15892281]
[0.87191012 0.12808988]
[0.89447647 0.10552353]
```

```
[0.82945361 0.17054639]
[0.72856834 0.27143166]
[0.94692818 0.05307182]
[0.96074296 0.03925704]
[0.905002   0.094998  ]
[0.88599983 0.11400017]
[0.8486564  0.1513436 ]
[0.7908699  0.2091301 ]
[0.67303    0.32697   ]
[0.93390201 0.06609799]
[0.65644908 0.34355092]
[0.74382864 0.25617136]
[0.94248401 0.05751599]
[0.78365662 0.21634338]
[0.90655644 0.09344356]
[0.81578725 0.18421275]
[0.89149733 0.10850267]
[0.85791872 0.14208128]
[0.67453877 0.32546123]
[0.93130384 0.06869616]
[0.89999978 0.10000022]]
```

In [381…

```
from sklearn import metrics

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

```
0.8435374149659864
0.6502502887947632
```

In [382…

```
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

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

In [383…

```
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
...         ...   ...         ...               ...        ...
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
...              ...             ...
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]
```

In [388…

```python
importance = np.abs(model.coef_[0])
for i,v in enumerate(importance):
    print('Feature: %0d, Score: %.5f' % (i,v))
```

```
Feature: 0, Score: 0.00060
Feature: 1, Score: 0.04095
Feature: 2, Score: 0.29101
Feature: 3, Score: 0.02636
Feature: 4, Score: 0.01227
Feature: 5, Score: 0.10509
Feature: 6, Score: 0.06505
```

In [236…

```python
from sklearn.linear_model import LogisticRegression
import numpy as np

# Define your input features and target variable
X = [[0, 1], [1, 0], [1, 1], [0, 0]]
y = [0, 1, 1, 0]

# Initialize and fit the logistic regression model
clf = LogisticRegression(random_state=0).fit(X, y)

# Obtain the coefficients and feature importance
coefficients = clf.coef_[0]
importance = np.abs(coefficients)

# Print the results
print("Coefficients:", coefficients)
print("Feature Importance:", importance)
```

```
Coefficients: [ 8.02109869e-01 -4.71500832e-06]
Feature Importance: [8.02109869e-01 4.71500832e-06]
```

In [323…

```
# Add random values to KK according to the parameters mentioned above to check the prc
kk=[[1.0, 23.0, 1.0, 500.0, 3.0, 24.0, 1.0]]
print(model.predict_proba(kk))
```

```
[[6.25571863e-07 9.99999374e-01]]
C:\Users\beemr\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does n
ot have valid feature names, but LogisticRegression was fitted with feature names
  warnings.warn(
```