attrition

Description of columns

* AGE Numerical Value
* ATTRITION Employee leaving the company (0=no, 1=yes)
* BUSINESS TRAVEL (1=No Travel, 2=Travel Frequently, 3=Travel Rarely)
* DAILY RATE Numerical Value - Salary Level
* DEPARTMENT (1=HR, 2=R&D, 3=Sales)
* DISTANCE FROM HOME Numerical Value - THE DISTANCE FROM WORK TO HOME
* EDUCATION Numerical Value. (1 'Below College' 2 'College' 3 'Bachelor' 4 'Master' 5 'Doctor')
* EDUCATION FIELD (1=HR, 2=LIFE SCIENCES, 3=MARKETING, 4=MEDICAL SCIENCES, 5=OTHERS, 8= TECHNICAL)
* EMPLOYEE COUNT Numerical Value
* EMPLOYEE NUMBER Numerical Value - EMPLOYEE ID
* ENVIRONMENT SATISFACTION Numerical Value - SATISFACTION WITH THE ENVIRONMENT (1 'Low' 2 'Medium' 3 'High' 4 'Very High')
* GENDER (1=FEMALE, 2=MALE)
* HOURLY RATE Numerical Value - HOURLY SALARY
* JOB INVOLVEMENT Numerical Value - JOB INVOLVEMENT (1 'Low' 2 'Medium' 3 'High' 4 'Very High')
* JOB LEVEL Numerical Value - LEVEL OF JOB
* JOB ROLE (1=HR REP, 2=HR, 3=LAB TECHNICIAN, 4=MANAGER, 5= MANAGING DIRECTOR, 8= RESEARCH DIRECTOR, 7= RESEARCH SCIENTIST, 8=SALES EXECUTIVE, 9= SALES REPRESENTATIVE)
* JOB SATISFACTION Numerical Value - SATISFACTION WITH THE JOB (1 'Low' 2 'Medium' 3 'High' 4 'Very High')
* MARITAL STATUS (1=DIVORCED, 2=MARRIED, 3=SINGLE)
* MONTHLY INCOME Numerical Value - MONTHLY SALARY
* MONTHLY RATE Numerical Value - MONTHLY RATE
* NUMCOMPANIES WORKED Numerical Value - NO. OF COMPANIES WORKED AT
* OVER 18 (1=YES, 2=NO)
* OVERTIME (1=NO, 2=YES)
* PERCENT SALARY HIKE Numerical Value - PERCENTAGE INCREASE IN SALARY
* PERFORMANCE RATING Numerical Value - PERFORMANCE RATING
* RELATIONS SATISFACTION Numerical Value - RELATIONS SATISFACTION
* STANDARD HOURS Numerical Value - STANDARD HOURS
* STOCK OPTIONS LEVEL Numerical Value - STOCK OPTIONS (Higher the number, the more stock option an employee has)
* TOTAL WORKING YEARS Numerical Value - TOTAL YEARS WORKED
* TRAINING TIMES LAST YEAR Numerical Value - HOURS SPENT TRAINING
* WORK LIFE BALANCE Numerical Value - TIME SPENT BETWEEN WORK AND OUTSIDE
* YEARS AT COMPANY Numerical Value - TOTAL NUMBER OF YEARS AT THE COMPANY
* YEARS IN CURRENT ROLE Numerical Value -YEARS IN CURRENT ROLE
* YEARS SINCE LAST PROMOTION Numerical Value - LAST PROMOTION
* YEARS WITH CURRENT MANAGER Numerical Value - YEARS SPENT WITH CURRENT MANAGER

In [1]:

import pandas as pd   
import numpy as np   
import matplotlib.pyplot as plt   
%matplotlib inline  
from matplotlib.pylab import rcParams  
rcParams['figure.figsize'] = 20, 10  
import seaborn as sns

In [2]:

df = pd.read\_csv("WA\_Fn-UseC\_-HR-Employee-Attrition.csv")  
df.sample(10)

Out[2]:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Age | Attrition | BusinessTravel | DailyRate | Department | DistanceFromHome | Education | EducationField | EmployeeCount | EmployeeNumber | ... | RelationshipSatisfaction | StandardHours | StockOptionLevel | TotalWorkingYears | TrainingTimesLastYear | WorkLifeBalance | YearsAtCompany | YearsInCurrentRole | YearsSinceLastPromotion | YearsWithCurrManager |
| 281 | 42 | No | Travel\_Rarely | 635 | Sales | 1 | 1 | Life Sciences | 1 | 387 | ... | 3 | 80 | 0 | 20 | 3 | 3 | 20 | 16 | 11 | 6 |
| 1067 | 47 | No | Travel\_Rarely | 571 | Sales | 14 | 3 | Medical | 1 | 1503 | ... | 3 | 80 | 1 | 11 | 4 | 2 | 5 | 4 | 1 | 2 |
| 1243 | 45 | No | Travel\_Rarely | 176 | Human Resources | 4 | 3 | Life Sciences | 1 | 1744 | ... | 3 | 80 | 2 | 9 | 2 | 4 | 5 | 0 | 0 | 3 |
| 1387 | 29 | No | Travel\_Rarely | 136 | Research & Development | 1 | 3 | Life Sciences | 1 | 1954 | ... | 1 | 80 | 1 | 6 | 5 | 2 | 5 | 3 | 0 | 2 |
| 1298 | 46 | Yes | Travel\_Rarely | 261 | Research & Development | 21 | 2 | Medical | 1 | 1821 | ... | 4 | 80 | 1 | 13 | 2 | 4 | 9 | 7 | 3 | 7 |
| 436 | 33 | Yes | Travel\_Rarely | 587 | Research & Development | 10 | 1 | Medical | 1 | 584 | ... | 1 | 80 | 3 | 8 | 2 | 3 | 4 | 3 | 1 | 3 |
| 256 | 42 | No | Travel\_Rarely | 269 | Research & Development | 2 | 3 | Medical | 1 | 351 | ... | 3 | 80 | 1 | 10 | 4 | 3 | 9 | 6 | 7 | 8 |
| 1125 | 29 | No | Travel\_Frequently | 995 | Research & Development | 2 | 1 | Life Sciences | 1 | 1590 | ... | 4 | 80 | 1 | 6 | 0 | 4 | 6 | 4 | 1 | 3 |
| 374 | 28 | No | Travel\_Rarely | 304 | Sales | 9 | 4 | Life Sciences | 1 | 498 | ... | 4 | 80 | 0 | 7 | 1 | 3 | 7 | 5 | 0 | 7 |
| 947 | 52 | Yes | Travel\_Rarely | 1030 | Sales | 5 | 3 | Life Sciences | 1 | 1319 | ... | 3 | 80 | 0 | 10 | 2 | 2 | 8 | 7 | 7 | 7 |

10 rows × 35 columns

In [3]:

df.describe()

Out[3]:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Age | DailyRate | DistanceFromHome | Education | EmployeeCount | EmployeeNumber | EnvironmentSatisfaction | HourlyRate | JobInvolvement | JobLevel | ... | RelationshipSatisfaction | StandardHours | StockOptionLevel | TotalWorkingYears | TrainingTimesLastYear | WorkLifeBalance | YearsAtCompany | YearsInCurrentRole | YearsSinceLastPromotion | YearsWithCurrManager |
| count | 1470.000000 | 1470.000000 | 1470.000000 | 1470.000000 | 1470.0 | 1470.000000 | 1470.000000 | 1470.000000 | 1470.000000 | 1470.000000 | ... | 1470.000000 | 1470.0 | 1470.000000 | 1470.000000 | 1470.000000 | 1470.000000 | 1470.000000 | 1470.000000 | 1470.000000 | 1470.000000 |
| mean | 36.923810 | 802.485714 | 9.192517 | 2.912925 | 1.0 | 1024.865306 | 2.721769 | 65.891156 | 2.729932 | 2.063946 | ... | 2.712245 | 80.0 | 0.793878 | 11.279592 | 2.799320 | 2.761224 | 7.008163 | 4.229252 | 2.187755 | 4.123129 |
| std | 9.135373 | 403.509100 | 8.106864 | 1.024165 | 0.0 | 602.024335 | 1.093082 | 20.329428 | 0.711561 | 1.106940 | ... | 1.081209 | 0.0 | 0.852077 | 7.780782 | 1.289271 | 0.706476 | 6.126525 | 3.623137 | 3.222430 | 3.568136 |
| min | 18.000000 | 102.000000 | 1.000000 | 1.000000 | 1.0 | 1.000000 | 1.000000 | 30.000000 | 1.000000 | 1.000000 | ... | 1.000000 | 80.0 | 0.000000 | 0.000000 | 0.000000 | 1.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 25% | 30.000000 | 465.000000 | 2.000000 | 2.000000 | 1.0 | 491.250000 | 2.000000 | 48.000000 | 2.000000 | 1.000000 | ... | 2.000000 | 80.0 | 0.000000 | 6.000000 | 2.000000 | 2.000000 | 3.000000 | 2.000000 | 0.000000 | 2.000000 |
| 50% | 36.000000 | 802.000000 | 7.000000 | 3.000000 | 1.0 | 1020.500000 | 3.000000 | 66.000000 | 3.000000 | 2.000000 | ... | 3.000000 | 80.0 | 1.000000 | 10.000000 | 3.000000 | 3.000000 | 5.000000 | 3.000000 | 1.000000 | 3.000000 |
| 75% | 43.000000 | 1157.000000 | 14.000000 | 4.000000 | 1.0 | 1555.750000 | 4.000000 | 83.750000 | 3.000000 | 3.000000 | ... | 4.000000 | 80.0 | 1.000000 | 15.000000 | 3.000000 | 3.000000 | 9.000000 | 7.000000 | 3.000000 | 7.000000 |
| max | 60.000000 | 1499.000000 | 29.000000 | 5.000000 | 1.0 | 2068.000000 | 4.000000 | 100.000000 | 4.000000 | 5.000000 | ... | 4.000000 | 80.0 | 3.000000 | 40.000000 | 6.000000 | 4.000000 | 40.000000 | 18.000000 | 15.000000 | 17.000000 |

8 rows × 26 columns

In [4]:

list(df.columns)

Out[4]:

['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']

In [5]:

df.info()

<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 1470 entries, 0 to 1469  
Data columns (total 35 columns):  
Age 1470 non-null int64  
Attrition 1470 non-null object  
BusinessTravel 1470 non-null object  
DailyRate 1470 non-null int64  
Department 1470 non-null object  
DistanceFromHome 1470 non-null int64  
Education 1470 non-null int64  
EducationField 1470 non-null object  
EmployeeCount 1470 non-null int64  
EmployeeNumber 1470 non-null int64  
EnvironmentSatisfaction 1470 non-null int64  
Gender 1470 non-null object  
HourlyRate 1470 non-null int64  
JobInvolvement 1470 non-null int64  
JobLevel 1470 non-null int64  
JobRole 1470 non-null object  
JobSatisfaction 1470 non-null int64  
MaritalStatus 1470 non-null object  
MonthlyIncome 1470 non-null int64  
MonthlyRate 1470 non-null int64  
NumCompaniesWorked 1470 non-null int64  
Over18 1470 non-null object  
OverTime 1470 non-null object  
PercentSalaryHike 1470 non-null int64  
PerformanceRating 1470 non-null int64  
RelationshipSatisfaction 1470 non-null int64  
StandardHours 1470 non-null int64  
StockOptionLevel 1470 non-null int64  
TotalWorkingYears 1470 non-null int64  
TrainingTimesLastYear 1470 non-null int64  
WorkLifeBalance 1470 non-null int64  
YearsAtCompany 1470 non-null int64  
YearsInCurrentRole 1470 non-null int64  
YearsSinceLastPromotion 1470 non-null int64  
YearsWithCurrManager 1470 non-null int64  
dtypes: int64(26), object(9)  
memory usage: 402.0+ KB

Columns removed for ID or showed only one value.

* EmployeeNumber (ID)
* EmployeeCount (value = 1)
* StandardHours (Value = 80)
* Over18 (Value = yes)

In [6]:

df = df.drop(['EmployeeNumber','EmployeeCount','StandardHours','Over18'],axis=1)

In [7]:

#change Attrition, BusinessTravel, Gender, and OverTime to numeric  
varList = ['Attrition','BusinessTravel','Gender','OverTime']  
def binary\_map(x):  
 return x.map({'Yes':1,'No':0,'Non-Travel':0,'Travel\_Rarely':1,'Travel\_Frequently':2,'Male':0,'Female':1})  
  
df[varList] = df[varList].apply(binary\_map)

In [8]:

#separate numeric and categorical values  
varNum = ['Age','DailyRate','BusinessTravel','DistanceFromHome','Education','EnvironmentSatisfaction','Gender','HourlyRate','JobInvolvement','JobLevel','JobSatisfaction','MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked','OverTime','PercentSalaryHike', 'PerformanceRating','RelationshipSatisfaction','StockOptionLevel','TotalWorkingYears', 'TrainingTimesLastYear','WorkLifeBalance','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager']  
varCat = ['Department','EducationField','JobRole','MaritalStatus']  
varLabel = ['Attrition']

In [9]:

#convert categories to dummy code  
dfCatDummies = pd.get\_dummies(df[varCat],drop\_first=True)

In [10]:

#combine variables and separate label  
x = df[varNum]  
x = pd.concat([x,dfCatDummies], axis=1)  
y = df[varLabel]

In [11]:

from sklearn.model\_selection import train\_test\_split  
from sklearn.preprocessing import MinMaxScaler  
np.random.seed(0)

In [14]:

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
scaler = MinMaxScaler()  
x\_train[varNum] = scaler.fit\_transform(x\_train[varNum])  
x\_test[varNum] = scaler.fit\_transform(x\_test[varNum])

C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:3: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 This is separate from the ipykernel package so we can avoid doing imports until  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:4: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 after removing the cwd from sys.path.  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s

In [15]:

import statsmodels.api as sm

In [16]:

x\_train\_lm = sm.add\_constant(x\_train)  
lr1 = sm.OLS(y\_train,x\_train\_lm).fit()  
lr1.summary()

C:\Users\nagem\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2542: FutureWarning: Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.  
 return ptp(axis=axis, out=out, \*\*kwargs)

In [93]:

# Checking for the vif2 values of the variables.   
from statsmodels.stats.outliers\_influence import variance\_inflation\_factor  
  
# Creating a dataframe that will contain the names of all the feature variables and their vif1s  
vif1 = pd.DataFrame()  
vif1['Features'] = x\_train.columns  
vif1['vif1'] = [variance\_inflation\_factor(x\_train.values, i) for i in range(x\_train.shape[1])]  
vif1['vif1'] = round(vif1['vif1'], 2)  
vif1 = vif1.sort\_values(by = "vif1", ascending = False)  
vif1

Out[93]:

|  |  |  |
| --- | --- | --- |
|  | Features | vif1 |
| 17 | TotalWorkingYears | 15.29 |
| 24 | EducationField\_Life Sciences | 13.55 |
| 26 | EducationField\_Medical | 11.17 |
| 20 | YearsAtCompany | 11.07 |
| 0 | Age | 10.67 |
| 19 | WorkLifeBalance | 7.38 |
| 8 | JobInvolvement | 6.61 |
| 23 | YearsWithCurrManager | 6.32 |
| 21 | YearsInCurrentRole | 6.23 |
| 13 | PercentSalaryHike | 6.10 |
| 18 | TrainingTimesLastYear | 5.86 |
| 2 | BusinessTravel | 5.53 |
| 9 | JobLevel | 5.08 |
| 4 | Education | 4.97 |
| 25 | EducationField\_Marketing | 4.31 |
| 30 | MaritalStatus\_Single | 4.11 |
| 7 | HourlyRate | 4.08 |
| 1 | DailyRate | 4.05 |
| 28 | EducationField\_Technical Degree | 3.74 |
| 10 | JobSatisfaction | 3.59 |
| 15 | RelationshipSatisfaction | 3.54 |
| 5 | EnvironmentSatisfaction | 3.48 |
| 16 | StockOptionLevel | 3.33 |
| 29 | MaritalStatus\_Married | 3.20 |
| 14 | PerformanceRating | 3.07 |
| 11 | NumCompaniesWorked | 2.72 |
| 27 | EducationField\_Other | 2.58 |
| 22 | YearsSinceLastPromotion | 2.52 |
| 3 | DistanceFromHome | 2.06 |
| 6 | Gender | 1.74 |
| 12 | OverTime | 1.42 |

In [81]:

#separate numeric and categorical values  
varNum = ['Age','DailyRate','BusinessTravel','DistanceFromHome','Education','EnvironmentSatisfaction','Gender','HourlyRate','JobInvolvement','JobLevel','JobSatisfaction','MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked','OverTime','PercentSalaryHike', 'PerformanceRating','RelationshipSatisfaction','StockOptionLevel','TotalWorkingYears', 'TrainingTimesLastYear','WorkLifeBalance','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager']  
varCat = ['EducationField','JobRole','MaritalStatus'] #remove 'Department'  
varLabel = ['Attrition']  
  
#convert categories to dummy code  
dfCatDummies = pd.get\_dummies(df[varCat],drop\_first=True)  
  
#combine variables and separate label  
x = df[varNum]  
x = pd.concat([x,dfCatDummies], axis=1)  
y = df[varLabel]  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
scaler = MinMaxScaler()  
x\_train[varNum] = scaler.fit\_transform(x\_train[varNum])  
x\_test[varNum] = scaler.fit\_transform(x\_test[varNum])

C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:16: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 app.launch\_new\_instance()  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:17: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s

In [89]:

x\_train\_lm = sm.add\_constant(x\_train)  
lr2 = sm.OLS(y\_train,x\_train\_lm).fit()  
#print(lr2.summary())  
  
# Creating a dataframe that will contain the names of all the feature variables and their vif1s  
vif2 = pd.DataFrame()  
vif2['Features'] = x\_train.columns  
vif2['vif2'] = [variance\_inflation\_factor(x\_train.values, i) for i in range(x\_train.shape[1])]  
vif2['vif2'] = round(vif2['vif2'], 2)  
vif2 = vif2.sort\_values(by = "vif2", ascending = False)  
#print(vif2)

C:\Users\nagem\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2542: FutureWarning: Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.  
 return ptp(axis=axis, out=out, \*\*kwargs)

In [92]:

print(vif2)

Features VIF  
17 TotalWorkingYears 15.29  
24 EducationField\_Life Sciences 13.55  
26 EducationField\_Medical 11.17  
20 YearsAtCompany 11.07  
0 Age 10.67  
19 WorkLifeBalance 7.38  
8 JobInvolvement 6.61  
23 YearsWithCurrManager 6.32  
21 YearsInCurrentRole 6.23  
13 PercentSalaryHike 6.10  
18 TrainingTimesLastYear 5.86  
2 BusinessTravel 5.53  
9 JobLevel 5.08  
4 Education 4.97  
25 EducationField\_Marketing 4.31  
30 MaritalStatus\_Single 4.11  
7 HourlyRate 4.08  
1 DailyRate 4.05  
28 EducationField\_Technical Degree 3.74  
10 JobSatisfaction 3.59  
15 RelationshipSatisfaction 3.54  
5 EnvironmentSatisfaction 3.48  
16 StockOptionLevel 3.33  
29 MaritalStatus\_Married 3.20  
14 PerformanceRating 3.07  
11 NumCompaniesWorked 2.72  
27 EducationField\_Other 2.58  
22 YearsSinceLastPromotion 2.52  
3 DistanceFromHome 2.06  
6 Gender 1.74  
12 OverTime 1.42

In [85]:

#separate numeric and categorical values  
varNum = ['Age','DailyRate','BusinessTravel','DistanceFromHome','Education','EnvironmentSatisfaction','Gender','HourlyRate','JobInvolvement','JobLevel','JobSatisfaction','MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked','OverTime','PercentSalaryHike', 'PerformanceRating','RelationshipSatisfaction','StockOptionLevel','TotalWorkingYears', 'TrainingTimesLastYear','WorkLifeBalance','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager']  
varCat = ['EducationField','MaritalStatus'] #remove 'Department','JobRole'  
varLabel = ['Attrition']  
  
#convert categories to dummy code  
dfCatDummies = pd.get\_dummies(df[varCat],drop\_first=True)  
  
#combine variables and separate label  
x = df[varNum]  
x = pd.concat([x,dfCatDummies], axis=1)  
y = df[varLabel]  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
scaler = MinMaxScaler()  
x\_train[varNum] = scaler.fit\_transform(x\_train[varNum])  
x\_test[varNum] = scaler.fit\_transform(x\_test[varNum])  
  
x\_train\_lm = sm.add\_constant(x\_train)  
lr3 = sm.OLS(y\_train,x\_train\_lm).fit()  
print(lr3.summary())  
  
# Creating a dataframe that will contain the names of all the feature variables and their vif3s  
vif3 = pd.DataFrame()  
vif3['Features'] = x\_train.columns  
vif3['vif3'] = [variance\_inflation\_factor(x\_train.values, i) for i in range(x\_train.shape[1])]  
vif3['vif3'] = round(vif3['vif3'], 2)  
vif3 = vif3.sort\_values(by = "vif3", ascending = False)  
print(vif3)

C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:16: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 app.launch\_new\_instance()  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:17: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s

In [86]:

print(lr3.summary())

OLS Regression Results   
==============================================================================  
Dep. Variable: Attrition R-squared: 0.238  
Model: OLS Adj. R-squared: 0.213  
Method: Least Squares F-statistic: 9.431  
Date: Sat, 02 Oct 2021 Prob (F-statistic): 6.05e-40  
Time: 12:41:59 Log-Likelihood: -272.90  
No. Observations: 1029 AIC: 613.8  
Df Residuals: 995 BIC: 781.6  
Df Model: 33   
Covariance Type: nonrobust   
===================================================================================================  
 coef std err t P>|t| [0.025 0.975]  
---------------------------------------------------------------------------------------------------  
const 0.5027 0.108 4.666 0.000 0.291 0.714  
Age -0.2154 0.066 -3.251 0.001 -0.345 -0.085  
DailyRate -0.0394 0.035 -1.119 0.263 -0.109 0.030  
BusinessTravel 0.1519 0.039 3.902 0.000 0.076 0.228  
DistanceFromHome 0.0965 0.035 2.742 0.006 0.027 0.166  
Education 0.0058 0.041 0.143 0.886 -0.074 0.086  
EnvironmentSatisfaction -0.1186 0.028 -4.247 0.000 -0.173 -0.064  
Gender -0.0390 0.021 -1.884 0.060 -0.080 0.002  
HourlyRate -0.0107 0.035 -0.306 0.760 -0.079 0.058  
JobInvolvement -0.1874 0.042 -4.410 0.000 -0.271 -0.104  
JobLevel -0.0639 0.124 -0.516 0.606 -0.307 0.179  
JobSatisfaction -0.0927 0.028 -3.319 0.001 -0.147 -0.038  
MonthlyIncome -0.0611 0.138 -0.444 0.657 -0.331 0.209  
MonthlyRate 0.0103 0.036 0.290 0.772 -0.059 0.080  
NumCompaniesWorked 0.1786 0.042 4.259 0.000 0.096 0.261  
OverTime 0.2285 0.023 9.984 0.000 0.184 0.273  
PercentSalaryHike 0.0107 0.061 0.174 0.862 -0.110 0.131  
PerformanceRating -0.0166 0.044 -0.379 0.705 -0.102 0.069  
RelationshipSatisfaction -0.0399 0.028 -1.421 0.156 -0.095 0.015  
StockOptionLevel -0.0201 0.048 -0.417 0.677 -0.115 0.075  
TotalWorkingYears -0.1056 0.117 -0.905 0.366 -0.334 0.123  
TrainingTimesLastYear -0.0601 0.048 -1.253 0.211 -0.154 0.034  
WorkLifeBalance -0.0608 0.044 -1.390 0.165 -0.147 0.025  
YearsAtCompany 0.2982 0.142 2.104 0.036 0.020 0.576  
YearsInCurrentRole -0.1353 0.082 -1.654 0.098 -0.296 0.025  
YearsSinceLastPromotion 0.1378 0.062 2.236 0.026 0.017 0.259  
YearsWithCurrManager -0.2051 0.079 -2.598 0.010 -0.360 -0.050  
EducationField\_Life Sciences -0.0993 0.081 -1.228 0.220 -0.258 0.059  
EducationField\_Marketing -0.0486 0.085 -0.571 0.568 -0.215 0.118  
EducationField\_Medical -0.1217 0.081 -1.499 0.134 -0.281 0.038  
EducationField\_Other -0.2011 0.091 -2.220 0.027 -0.379 -0.023  
EducationField\_Technical Degree -0.0200 0.087 -0.231 0.817 -0.190 0.150  
MaritalStatus\_Married 0.0224 0.027 0.824 0.410 -0.031 0.076  
MaritalStatus\_Single 0.1115 0.037 3.005 0.003 0.039 0.184  
==============================================================================  
Omnibus: 217.028 Durbin-Watson: 2.015  
Prob(Omnibus): 0.000 Jarque-Bera (JB): 372.874  
Skew: 1.340 Prob(JB): 1.08e-81  
Kurtosis: 4.230 Cond. No. 49.2  
==============================================================================  
  
Warnings:  
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [94]:

#separate numeric and categorical values  
varNum = ['Age','DailyRate','BusinessTravel','DistanceFromHome','Education','EnvironmentSatisfaction','Gender','HourlyRate','JobInvolvement','JobLevel','JobSatisfaction', 'NumCompaniesWorked','OverTime','PercentSalaryHike', 'PerformanceRating','RelationshipSatisfaction','StockOptionLevel','TotalWorkingYears', 'TrainingTimesLastYear','WorkLifeBalance','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager'] #remove 'MonthlyIncome', 'MonthlyRate',  
varCat = ['EducationField','MaritalStatus'] #remove 'Department','JobRole'  
varLabel = ['Attrition']  
  
#convert categories to dummy code  
dfCatDummies = pd.get\_dummies(df[varCat],drop\_first=True)  
  
#combine variables and separate label  
x = df[varNum]  
x = pd.concat([x,dfCatDummies], axis=1)  
y = df[varLabel]  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
scaler = MinMaxScaler()  
x\_train[varNum] = scaler.fit\_transform(x\_train[varNum])  
x\_test[varNum] = scaler.fit\_transform(x\_test[varNum])  
  
x\_train\_lm = sm.add\_constant(x\_train)  
lr4 = sm.OLS(y\_train,x\_train\_lm).fit()  
#print(lr4.summary())  
  
# Creating a dataframe that will contain the names of all the feature variables and their vif4s  
vif4 = pd.DataFrame()  
vif4['Features'] = x\_train.columns  
vif4['vif4'] = [variance\_inflation\_factor(x\_train.values, i) for i in range(x\_train.shape[1])]  
vif4['vif4'] = round(vif4['vif4'], 2)  
vif4 = vif4.sort\_values(by = "vif4", ascending = False)  
#print(vif4)

C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:16: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 app.launch\_new\_instance()  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:17: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2542: FutureWarning: Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.  
 return ptp(axis=axis, out=out, \*\*kwargs)

In [95]:

lr4.summary()

Out[95]:

OLS Regression Results

|  |  |  |  |
| --- | --- | --- | --- |
| Dep. Variable: | Attrition | R-squared: | 0.238 |
| Model: | OLS | Adj. R-squared: | 0.214 |
| Method: | Least Squares | F-statistic: | 10.05 |
| Date: | Sat, 02 Oct 2021 | Prob (F-statistic): | 6.79e-41 |
| Time: | 12:58:29 | Log-Likelihood: | -273.05 |
| No. Observations: | 1029 | AIC: | 610.1 |
| Df Residuals: | 997 | BIC: | 768.1 |
| Df Model: | 31 |  |  |
| Covariance Type: | nonrobust |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | coef | std err | t | P>|t| | [0.025 | 0.975] |
| const | 0.5023 | 0.105 | 4.766 | 0.000 | 0.295 | 0.709 |
| Age | -0.2135 | 0.066 | -3.230 | 0.001 | -0.343 | -0.084 |
| DailyRate | -0.0403 | 0.035 | -1.146 | 0.252 | -0.109 | 0.029 |
| BusinessTravel | 0.1525 | 0.039 | 3.922 | 0.000 | 0.076 | 0.229 |
| DistanceFromHome | 0.0979 | 0.035 | 2.793 | 0.005 | 0.029 | 0.167 |
| Education | 0.0049 | 0.041 | 0.121 | 0.904 | -0.075 | 0.085 |
| EnvironmentSatisfaction | -0.1176 | 0.028 | -4.226 | 0.000 | -0.172 | -0.063 |
| Gender | -0.0384 | 0.021 | -1.858 | 0.064 | -0.079 | 0.002 |
| HourlyRate | -0.0112 | 0.035 | -0.321 | 0.749 | -0.079 | 0.057 |
| JobInvolvement | -0.1874 | 0.042 | -4.418 | 0.000 | -0.271 | -0.104 |
| JobLevel | -0.1110 | 0.059 | -1.884 | 0.060 | -0.227 | 0.005 |
| JobSatisfaction | -0.0927 | 0.028 | -3.323 | 0.001 | -0.147 | -0.038 |
| NumCompaniesWorked | 0.1791 | 0.042 | 4.280 | 0.000 | 0.097 | 0.261 |
| OverTime | 0.2289 | 0.023 | 10.024 | 0.000 | 0.184 | 0.274 |
| PercentSalaryHike | 0.0111 | 0.061 | 0.181 | 0.857 | -0.109 | 0.131 |
| PerformanceRating | -0.0171 | 0.044 | -0.391 | 0.696 | -0.103 | 0.069 |
| RelationshipSatisfaction | -0.0401 | 0.028 | -1.431 | 0.153 | -0.095 | 0.015 |
| StockOptionLevel | -0.0197 | 0.048 | -0.408 | 0.683 | -0.114 | 0.075 |
| TotalWorkingYears | -0.1155 | 0.115 | -1.006 | 0.314 | -0.341 | 0.110 |
| TrainingTimesLastYear | -0.0601 | 0.048 | -1.253 | 0.210 | -0.154 | 0.034 |
| WorkLifeBalance | -0.0604 | 0.044 | -1.382 | 0.167 | -0.146 | 0.025 |
| YearsAtCompany | 0.2962 | 0.142 | 2.092 | 0.037 | 0.018 | 0.574 |
| YearsInCurrentRole | -0.1347 | 0.082 | -1.649 | 0.100 | -0.295 | 0.026 |
| YearsSinceLastPromotion | 0.1374 | 0.062 | 2.234 | 0.026 | 0.017 | 0.258 |
| YearsWithCurrManager | -0.2014 | 0.078 | -2.571 | 0.010 | -0.355 | -0.048 |
| EducationField\_Life Sciences | -0.0986 | 0.081 | -1.221 | 0.222 | -0.257 | 0.060 |
| EducationField\_Marketing | -0.0471 | 0.085 | -0.556 | 0.578 | -0.214 | 0.119 |
| EducationField\_Medical | -0.1213 | 0.081 | -1.496 | 0.135 | -0.280 | 0.038 |
| EducationField\_Other | -0.2008 | 0.090 | -2.221 | 0.027 | -0.378 | -0.023 |
| EducationField\_Technical Degree | -0.0189 | 0.086 | -0.218 | 0.827 | -0.188 | 0.151 |
| MaritalStatus\_Married | 0.0219 | 0.027 | 0.809 | 0.419 | -0.031 | 0.075 |
| MaritalStatus\_Single | 0.1121 | 0.037 | 3.024 | 0.003 | 0.039 | 0.185 |

|  |  |  |  |
| --- | --- | --- | --- |
| Omnibus: | 218.697 | Durbin-Watson: | 2.016 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 377.616 |
| Skew: | 1.346 | Prob(JB): | 1.00e-82 |
| Kurtosis: | 4.251 | Cond. No. | 47.3 |

Warnings:  
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [96]:

vif4

Out[96]:

|  |  |  |
| --- | --- | --- |
|  | Features | vif4 |
| 17 | TotalWorkingYears | 15.29 |
| 24 | EducationField\_Life Sciences | 13.55 |
| 26 | EducationField\_Medical | 11.17 |
| 20 | YearsAtCompany | 11.07 |
| 0 | Age | 10.67 |
| 19 | WorkLifeBalance | 7.38 |
| 8 | JobInvolvement | 6.61 |
| 23 | YearsWithCurrManager | 6.32 |
| 21 | YearsInCurrentRole | 6.23 |
| 13 | PercentSalaryHike | 6.10 |
| 18 | TrainingTimesLastYear | 5.86 |
| 2 | BusinessTravel | 5.53 |
| 9 | JobLevel | 5.08 |
| 4 | Education | 4.97 |
| 25 | EducationField\_Marketing | 4.31 |
| 30 | MaritalStatus\_Single | 4.11 |
| 7 | HourlyRate | 4.08 |
| 1 | DailyRate | 4.05 |
| 28 | EducationField\_Technical Degree | 3.74 |
| 10 | JobSatisfaction | 3.59 |
| 15 | RelationshipSatisfaction | 3.54 |
| 5 | EnvironmentSatisfaction | 3.48 |
| 16 | StockOptionLevel | 3.33 |
| 29 | MaritalStatus\_Married | 3.20 |
| 14 | PerformanceRating | 3.07 |
| 11 | NumCompaniesWorked | 2.72 |
| 27 | EducationField\_Other | 2.58 |
| 22 | YearsSinceLastPromotion | 2.52 |
| 3 | DistanceFromHome | 2.06 |
| 6 | Gender | 1.74 |
| 12 | OverTime | 1.42 |

In [97]:

#separate numeric and categorical values  
varNum = ['Age','DailyRate','BusinessTravel','DistanceFromHome','EnvironmentSatisfaction','Gender','HourlyRate','JobInvolvement','JobLevel','JobSatisfaction', 'NumCompaniesWorked','OverTime','PercentSalaryHike', 'PerformanceRating','RelationshipSatisfaction','StockOptionLevel','TotalWorkingYears', 'TrainingTimesLastYear','WorkLifeBalance','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager'] #remove 'MonthlyIncome', 'MonthlyRate',,'Education'  
varCat = ['EducationField','MaritalStatus'] #remove 'Department','JobRole'  
varLabel = ['Attrition']  
  
#convert categories to dummy code  
dfCatDummies = pd.get\_dummies(df[varCat],drop\_first=True)  
  
#combine variables and separate label  
x = df[varNum]  
x = pd.concat([x,dfCatDummies], axis=1)  
y = df[varLabel]  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
scaler = MinMaxScaler()  
x\_train[varNum] = scaler.fit\_transform(x\_train[varNum])  
x\_test[varNum] = scaler.fit\_transform(x\_test[varNum])  
  
x\_train\_lm = sm.add\_constant(x\_train)  
lr5 = sm.OLS(y\_train,x\_train\_lm).fit()  
#print(lr4.summary())  
  
# Creating a dataframe that will contain the names of all the feature variables and their vif5s  
vif5 = pd.DataFrame()  
vif5['Features'] = x\_train.columns  
vif5['vif5'] = [variance\_inflation\_factor(x\_train.values, i) for i in range(x\_train.shape[1])]  
vif5['vif5'] = round(vif5['vif5'], 2)  
vif5 = vif5.sort\_values(by = "vif5", ascending = False)  
#print(vif5)

C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:16: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 app.launch\_new\_instance()  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:17: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2542: FutureWarning: Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.  
 return ptp(axis=axis, out=out, \*\*kwargs)

In [99]:

lr5.summary()

Out[99]:

OLS Regression Results

|  |  |  |  |
| --- | --- | --- | --- |
| Dep. Variable: | Attrition | R-squared: | 0.238 |
| Model: | OLS | Adj. R-squared: | 0.215 |
| Method: | Least Squares | F-statistic: | 10.39 |
| Date: | Sat, 02 Oct 2021 | Prob (F-statistic): | 2.09e-41 |
| Time: | 13:01:04 | Log-Likelihood: | -273.06 |
| No. Observations: | 1029 | AIC: | 608.1 |
| Df Residuals: | 998 | BIC: | 761.1 |
| Df Model: | 30 |  |  |
| Covariance Type: | nonrobust |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | coef | std err | t | P>|t| | [0.025 | 0.975] |
| const | 0.5039 | 0.104 | 4.828 | 0.000 | 0.299 | 0.709 |
| Age | -0.2124 | 0.065 | -3.245 | 0.001 | -0.341 | -0.084 |
| DailyRate | -0.0404 | 0.035 | -1.149 | 0.251 | -0.109 | 0.029 |
| BusinessTravel | 0.1525 | 0.039 | 3.925 | 0.000 | 0.076 | 0.229 |
| DistanceFromHome | 0.0980 | 0.035 | 2.797 | 0.005 | 0.029 | 0.167 |
| EnvironmentSatisfaction | -0.1177 | 0.028 | -4.235 | 0.000 | -0.172 | -0.063 |
| Gender | -0.0383 | 0.021 | -1.857 | 0.064 | -0.079 | 0.002 |
| HourlyRate | -0.0111 | 0.035 | -0.320 | 0.749 | -0.079 | 0.057 |
| JobInvolvement | -0.1875 | 0.042 | -4.421 | 0.000 | -0.271 | -0.104 |
| JobLevel | -0.1111 | 0.059 | -1.886 | 0.060 | -0.227 | 0.005 |
| JobSatisfaction | -0.0927 | 0.028 | -3.324 | 0.001 | -0.147 | -0.038 |
| NumCompaniesWorked | 0.1795 | 0.042 | 4.301 | 0.000 | 0.098 | 0.261 |
| OverTime | 0.2288 | 0.023 | 10.030 | 0.000 | 0.184 | 0.274 |
| PercentSalaryHike | 0.0111 | 0.061 | 0.182 | 0.856 | -0.109 | 0.131 |
| PerformanceRating | -0.0172 | 0.044 | -0.393 | 0.694 | -0.103 | 0.068 |
| RelationshipSatisfaction | -0.0401 | 0.028 | -1.432 | 0.152 | -0.095 | 0.015 |
| StockOptionLevel | -0.0196 | 0.048 | -0.407 | 0.684 | -0.114 | 0.075 |
| TotalWorkingYears | -0.1153 | 0.115 | -1.006 | 0.315 | -0.340 | 0.110 |
| TrainingTimesLastYear | -0.0600 | 0.048 | -1.253 | 0.211 | -0.154 | 0.034 |
| WorkLifeBalance | -0.0603 | 0.044 | -1.380 | 0.168 | -0.146 | 0.025 |
| YearsAtCompany | 0.2958 | 0.141 | 2.091 | 0.037 | 0.018 | 0.573 |
| YearsInCurrentRole | -0.1348 | 0.082 | -1.650 | 0.099 | -0.295 | 0.025 |
| YearsSinceLastPromotion | 0.1375 | 0.061 | 2.236 | 0.026 | 0.017 | 0.258 |
| YearsWithCurrManager | -0.2010 | 0.078 | -2.569 | 0.010 | -0.354 | -0.047 |
| EducationField\_Life Sciences | -0.0986 | 0.081 | -1.222 | 0.222 | -0.257 | 0.060 |
| EducationField\_Marketing | -0.0469 | 0.085 | -0.554 | 0.580 | -0.213 | 0.119 |
| EducationField\_Medical | -0.1214 | 0.081 | -1.499 | 0.134 | -0.280 | 0.038 |
| EducationField\_Other | -0.2006 | 0.090 | -2.220 | 0.027 | -0.378 | -0.023 |
| EducationField\_Technical Degree | -0.0189 | 0.086 | -0.219 | 0.827 | -0.188 | 0.151 |
| MaritalStatus\_Married | 0.0220 | 0.027 | 0.810 | 0.418 | -0.031 | 0.075 |
| MaritalStatus\_Single | 0.1122 | 0.037 | 3.031 | 0.003 | 0.040 | 0.185 |

|  |  |  |  |
| --- | --- | --- | --- |
| Omnibus: | 218.736 | Durbin-Watson: | 2.015 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 377.721 |
| Skew: | 1.346 | Prob(JB): | 9.53e-83 |
| Kurtosis: | 4.251 | Cond. No. | 46.3 |

Warnings:  
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [100]:

#separate numeric and categorical values  
varNum = ['Age','DailyRate','BusinessTravel','DistanceFromHome','EnvironmentSatisfaction','Gender','HourlyRate','JobInvolvement','JobLevel','JobSatisfaction', 'NumCompaniesWorked','OverTime','PercentSalaryHike', 'PerformanceRating','RelationshipSatisfaction','StockOptionLevel','TotalWorkingYears', 'TrainingTimesLastYear','WorkLifeBalance','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager'] #remove 'MonthlyIncome', 'MonthlyRate',,'Education'  
varCat = ['MaritalStatus'] #remove 'Department','JobRole''EducationField',  
varLabel = ['Attrition']  
  
#convert categories to dummy code  
dfCatDummies = pd.get\_dummies(df[varCat],drop\_first=True)  
  
#combine variables and separate label  
x = df[varNum]  
x = pd.concat([x,dfCatDummies], axis=1)  
y = df[varLabel]  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
scaler = MinMaxScaler()  
x\_train[varNum] = scaler.fit\_transform(x\_train[varNum])  
x\_test[varNum] = scaler.fit\_transform(x\_test[varNum])  
  
x\_train\_lm = sm.add\_constant(x\_train)  
lr6 = sm.OLS(y\_train,x\_train\_lm).fit()  
#print(lr4.summary())  
  
# Creating a dataframe that will contain the names of all the feature variables and their vif6s  
vif6 = pd.DataFrame()  
vif6['Features'] = x\_train.columns  
vif6['vif6'] = [variance\_inflation\_factor(x\_train.values, i) for i in range(x\_train.shape[1])]  
vif6['vif6'] = round(vif6['vif6'], 2)  
vif6 = vif6.sort\_values(by = "vif6", ascending = False)  
#print(vif6)

C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:16: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 app.launch\_new\_instance()  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:17: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2542: FutureWarning: Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.  
 return ptp(axis=axis, out=out, \*\*kwargs)

In [101]:

lr6.summary()

Out[101]:

OLS Regression Results

|  |  |  |  |
| --- | --- | --- | --- |
| Dep. Variable: | Attrition | R-squared: | 0.225 |
| Model: | OLS | Adj. R-squared: | 0.206 |
| Method: | Least Squares | F-statistic: | 11.66 |
| Date: | Sat, 02 Oct 2021 | Prob (F-statistic): | 9.94e-41 |
| Time: | 13:03:30 | Log-Likelihood: | -281.66 |
| No. Observations: | 1029 | AIC: | 615.3 |
| Df Residuals: | 1003 | BIC: | 743.7 |
| Df Model: | 25 |  |  |
| Covariance Type: | nonrobust |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | coef | std err | t | P>|t| | [0.025 | 0.975] |
| const | 0.4071 | 0.073 | 5.564 | 0.000 | 0.264 | 0.551 |
| Age | -0.2035 | 0.066 | -3.101 | 0.002 | -0.332 | -0.075 |
| DailyRate | -0.0404 | 0.035 | -1.149 | 0.251 | -0.109 | 0.029 |
| BusinessTravel | 0.1495 | 0.039 | 3.832 | 0.000 | 0.073 | 0.226 |
| DistanceFromHome | 0.1019 | 0.035 | 2.903 | 0.004 | 0.033 | 0.171 |
| EnvironmentSatisfaction | -0.1181 | 0.028 | -4.238 | 0.000 | -0.173 | -0.063 |
| Gender | -0.0377 | 0.021 | -1.817 | 0.069 | -0.078 | 0.003 |
| HourlyRate | -0.0061 | 0.035 | -0.175 | 0.861 | -0.075 | 0.062 |
| JobInvolvement | -0.1915 | 0.043 | -4.492 | 0.000 | -0.275 | -0.108 |
| JobLevel | -0.1066 | 0.058 | -1.822 | 0.069 | -0.221 | 0.008 |
| JobSatisfaction | -0.0961 | 0.028 | -3.440 | 0.001 | -0.151 | -0.041 |
| NumCompaniesWorked | 0.1849 | 0.042 | 4.419 | 0.000 | 0.103 | 0.267 |
| OverTime | 0.2233 | 0.023 | 9.753 | 0.000 | 0.178 | 0.268 |
| PercentSalaryHike | -0.0036 | 0.061 | -0.059 | 0.953 | -0.124 | 0.117 |
| PerformanceRating | -0.0147 | 0.044 | -0.336 | 0.737 | -0.101 | 0.071 |
| RelationshipSatisfaction | -0.0412 | 0.028 | -1.470 | 0.142 | -0.096 | 0.014 |
| StockOptionLevel | -0.0062 | 0.048 | -0.129 | 0.897 | -0.101 | 0.088 |
| TotalWorkingYears | -0.1308 | 0.115 | -1.137 | 0.256 | -0.356 | 0.095 |
| TrainingTimesLastYear | -0.0697 | 0.048 | -1.451 | 0.147 | -0.164 | 0.025 |
| WorkLifeBalance | -0.0639 | 0.044 | -1.457 | 0.145 | -0.150 | 0.022 |
| YearsAtCompany | 0.3171 | 0.142 | 2.235 | 0.026 | 0.039 | 0.595 |
| YearsInCurrentRole | -0.1269 | 0.082 | -1.549 | 0.122 | -0.288 | 0.034 |
| YearsSinceLastPromotion | 0.1276 | 0.062 | 2.069 | 0.039 | 0.007 | 0.249 |
| YearsWithCurrManager | -0.2079 | 0.079 | -2.645 | 0.008 | -0.362 | -0.054 |
| MaritalStatus\_Married | 0.0285 | 0.027 | 1.047 | 0.295 | -0.025 | 0.082 |
| MaritalStatus\_Single | 0.1228 | 0.037 | 3.315 | 0.001 | 0.050 | 0.195 |

|  |  |  |  |
| --- | --- | --- | --- |
| Omnibus: | 224.442 | Durbin-Watson: | 2.024 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 393.184 |
| Skew: | 1.369 | Prob(JB): | 4.18e-86 |
| Kurtosis: | 4.292 | Cond. No. | 36.1 |

Warnings:  
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [102]:

#separate numeric and categorical values  
varNum = ['Age','DailyRate','BusinessTravel','DistanceFromHome','EnvironmentSatisfaction','Gender','HourlyRate','JobInvolvement','JobLevel','JobSatisfaction', 'NumCompaniesWorked','OverTime','PercentSalaryHike', 'PerformanceRating','RelationshipSatisfaction','TotalWorkingYears', 'TrainingTimesLastYear','WorkLifeBalance','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager'] #remove 'MonthlyIncome', 'MonthlyRate','Education','StockOptionLevel'  
varCat = ['MaritalStatus'] #remove 'Department','JobRole''EducationField',  
varLabel = ['Attrition']  
  
#convert categories to dummy code  
dfCatDummies = pd.get\_dummies(df[varCat],drop\_first=True)  
  
#combine variables and separate label  
x = df[varNum]  
x = pd.concat([x,dfCatDummies], axis=1)  
y = df[varLabel]  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
scaler = MinMaxScaler()  
x\_train[varNum] = scaler.fit\_transform(x\_train[varNum])  
x\_test[varNum] = scaler.fit\_transform(x\_test[varNum])  
  
x\_train\_lm = sm.add\_constant(x\_train)  
lr7 = sm.OLS(y\_train,x\_train\_lm).fit()  
#print(lr4.summary())  
  
# Creating a dataframe that will contain the names of all the feature variables and their vif7s  
vif7 = pd.DataFrame()  
vif7['Features'] = x\_train.columns  
vif7['vif7'] = [variance\_inflation\_factor(x\_train.values, i) for i in range(x\_train.shape[1])]  
vif7['vif7'] = round(vif7['vif7'], 2)  
vif7 = vif7.sort\_values(by = "vif7", ascending = False)  
#print(vif7)

C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:16: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 app.launch\_new\_instance()  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:17: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2542: FutureWarning: Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.  
 return ptp(axis=axis, out=out, \*\*kwargs)

In [104]:

lr7.summary()

Out[104]:

OLS Regression Results

|  |  |  |  |
| --- | --- | --- | --- |
| Dep. Variable: | Attrition | R-squared: | 0.225 |
| Model: | OLS | Adj. R-squared: | 0.207 |
| Method: | Least Squares | F-statistic: | 12.16 |
| Date: | Sat, 02 Oct 2021 | Prob (F-statistic): | 2.83e-41 |
| Time: | 13:05:45 | Log-Likelihood: | -281.67 |
| No. Observations: | 1029 | AIC: | 613.3 |
| Df Residuals: | 1004 | BIC: | 736.7 |
| Df Model: | 24 |  |  |
| Covariance Type: | nonrobust |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | coef | std err | t | P>|t| | [0.025 | 0.975] |
| const | 0.4043 | 0.070 | 5.788 | 0.000 | 0.267 | 0.541 |
| Age | -0.2034 | 0.066 | -3.102 | 0.002 | -0.332 | -0.075 |
| DailyRate | -0.0403 | 0.035 | -1.148 | 0.251 | -0.109 | 0.029 |
| BusinessTravel | 0.1495 | 0.039 | 3.834 | 0.000 | 0.073 | 0.226 |
| DistanceFromHome | 0.1017 | 0.035 | 2.901 | 0.004 | 0.033 | 0.171 |
| EnvironmentSatisfaction | -0.1181 | 0.028 | -4.239 | 0.000 | -0.173 | -0.063 |
| Gender | -0.0378 | 0.021 | -1.821 | 0.069 | -0.078 | 0.003 |
| HourlyRate | -0.0063 | 0.035 | -0.180 | 0.857 | -0.075 | 0.062 |
| JobInvolvement | -0.1915 | 0.043 | -4.494 | 0.000 | -0.275 | -0.108 |
| JobLevel | -0.1065 | 0.058 | -1.822 | 0.069 | -0.221 | 0.008 |
| JobSatisfaction | -0.0962 | 0.028 | -3.452 | 0.001 | -0.151 | -0.042 |
| NumCompaniesWorked | 0.1850 | 0.042 | 4.424 | 0.000 | 0.103 | 0.267 |
| OverTime | 0.2233 | 0.023 | 9.759 | 0.000 | 0.178 | 0.268 |
| PercentSalaryHike | -0.0038 | 0.061 | -0.062 | 0.950 | -0.124 | 0.117 |
| PerformanceRating | -0.0146 | 0.044 | -0.334 | 0.738 | -0.101 | 0.071 |
| RelationshipSatisfaction | -0.0412 | 0.028 | -1.471 | 0.142 | -0.096 | 0.014 |
| TotalWorkingYears | -0.1306 | 0.115 | -1.136 | 0.256 | -0.356 | 0.095 |
| TrainingTimesLastYear | -0.0698 | 0.048 | -1.454 | 0.146 | -0.164 | 0.024 |
| WorkLifeBalance | -0.0640 | 0.044 | -1.461 | 0.144 | -0.150 | 0.022 |
| YearsAtCompany | 0.3177 | 0.142 | 2.242 | 0.025 | 0.040 | 0.596 |
| YearsInCurrentRole | -0.1273 | 0.082 | -1.555 | 0.120 | -0.288 | 0.033 |
| YearsSinceLastPromotion | 0.1275 | 0.062 | 2.069 | 0.039 | 0.007 | 0.248 |
| YearsWithCurrManager | -0.2081 | 0.079 | -2.650 | 0.008 | -0.362 | -0.054 |
| MaritalStatus\_Married | 0.0294 | 0.026 | 1.127 | 0.260 | -0.022 | 0.081 |
| MaritalStatus\_Single | 0.1259 | 0.028 | 4.449 | 0.000 | 0.070 | 0.181 |

|  |  |  |  |
| --- | --- | --- | --- |
| Omnibus: | 224.303 | Durbin-Watson: | 2.024 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 392.756 |
| Skew: | 1.369 | Prob(JB): | 5.18e-86 |
| Kurtosis: | 4.290 | Cond. No. | 35.9 |

Warnings:  
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [105]:

#separate numeric and categorical values  
varNum = ['Age','DailyRate','BusinessTravel','DistanceFromHome','EnvironmentSatisfaction','Gender','HourlyRate','JobInvolvement','JobLevel','JobSatisfaction', 'NumCompaniesWorked','OverTime', 'PerformanceRating','RelationshipSatisfaction','TotalWorkingYears', 'TrainingTimesLastYear','WorkLifeBalance','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager'] #remove 'MonthlyIncome', 'MonthlyRate','Education','StockOptionLevel','PercentSalaryHike'  
varCat = ['MaritalStatus'] #remove 'Department','JobRole''EducationField',  
varLabel = ['Attrition']  
  
#convert categories to dummy code  
dfCatDummies = pd.get\_dummies(df[varCat],drop\_first=True)  
  
#combine variables and separate label  
x = df[varNum]  
x = pd.concat([x,dfCatDummies], axis=1)  
y = df[varLabel]  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
scaler = MinMaxScaler()  
x\_train[varNum] = scaler.fit\_transform(x\_train[varNum])  
x\_test[varNum] = scaler.fit\_transform(x\_test[varNum])  
  
x\_train\_lm = sm.add\_constant(x\_train)  
lr8 = sm.OLS(y\_train,x\_train\_lm).fit()  
#print(lr4.summary())  
  
# Creating a dataframe that will contain the names of all the feature variables and their vif8s  
vif8 = pd.DataFrame()  
vif8['Features'] = x\_train.columns  
vif8['vif8'] = [variance\_inflation\_factor(x\_train.values, i) for i in range(x\_train.shape[1])]  
vif8['vif8'] = round(vif8['vif8'], 2)  
vif8 = vif8.sort\_values(by = "vif8", ascending = False)  
#print(vif8)

C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:16: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 app.launch\_new\_instance()  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:17: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2542: FutureWarning: Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.  
 return ptp(axis=axis, out=out, \*\*kwargs)

In [106]:

lr8.summary()

Out[106]:

OLS Regression Results

|  |  |  |  |
| --- | --- | --- | --- |
| Dep. Variable: | Attrition | R-squared: | 0.225 |
| Model: | OLS | Adj. R-squared: | 0.207 |
| Method: | Least Squares | F-statistic: | 12.70 |
| Date: | Sat, 02 Oct 2021 | Prob (F-statistic): | 7.84e-42 |
| Time: | 13:15:17 | Log-Likelihood: | -281.67 |
| No. Observations: | 1029 | AIC: | 611.3 |
| Df Residuals: | 1005 | BIC: | 729.8 |
| Df Model: | 23 |  |  |
| Covariance Type: | nonrobust |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | coef | std err | t | P>|t| | [0.025 | 0.975] |
| const | 0.4035 | 0.069 | 5.881 | 0.000 | 0.269 | 0.538 |
| Age | -0.2037 | 0.065 | -3.113 | 0.002 | -0.332 | -0.075 |
| DailyRate | -0.0403 | 0.035 | -1.150 | 0.250 | -0.109 | 0.028 |
| BusinessTravel | 0.1496 | 0.039 | 3.843 | 0.000 | 0.073 | 0.226 |
| DistanceFromHome | 0.1016 | 0.035 | 2.902 | 0.004 | 0.033 | 0.170 |
| EnvironmentSatisfaction | -0.1181 | 0.028 | -4.241 | 0.000 | -0.173 | -0.063 |
| Gender | -0.0377 | 0.021 | -1.821 | 0.069 | -0.078 | 0.003 |
| HourlyRate | -0.0062 | 0.035 | -0.178 | 0.859 | -0.075 | 0.062 |
| JobInvolvement | -0.1916 | 0.043 | -4.501 | 0.000 | -0.275 | -0.108 |
| JobLevel | -0.1066 | 0.058 | -1.824 | 0.068 | -0.221 | 0.008 |
| JobSatisfaction | -0.0963 | 0.028 | -3.455 | 0.001 | -0.151 | -0.042 |
| NumCompaniesWorked | 0.1851 | 0.042 | 4.429 | 0.000 | 0.103 | 0.267 |
| OverTime | 0.2234 | 0.023 | 9.770 | 0.000 | 0.178 | 0.268 |
| PerformanceRating | -0.0168 | 0.028 | -0.609 | 0.543 | -0.071 | 0.037 |
| RelationshipSatisfaction | -0.0413 | 0.028 | -1.472 | 0.141 | -0.096 | 0.014 |
| TotalWorkingYears | -0.1304 | 0.115 | -1.135 | 0.256 | -0.356 | 0.095 |
| TrainingTimesLastYear | -0.0698 | 0.048 | -1.456 | 0.146 | -0.164 | 0.024 |
| WorkLifeBalance | -0.0639 | 0.044 | -1.460 | 0.145 | -0.150 | 0.022 |
| YearsAtCompany | 0.3180 | 0.142 | 2.246 | 0.025 | 0.040 | 0.596 |
| YearsInCurrentRole | -0.1274 | 0.082 | -1.558 | 0.120 | -0.288 | 0.033 |
| YearsSinceLastPromotion | 0.1276 | 0.062 | 2.071 | 0.039 | 0.007 | 0.248 |
| YearsWithCurrManager | -0.2080 | 0.078 | -2.651 | 0.008 | -0.362 | -0.054 |
| MaritalStatus\_Married | 0.0294 | 0.026 | 1.125 | 0.261 | -0.022 | 0.081 |
| MaritalStatus\_Single | 0.1258 | 0.028 | 4.455 | 0.000 | 0.070 | 0.181 |

|  |  |  |  |
| --- | --- | --- | --- |
| Omnibus: | 224.321 | Durbin-Watson: | 2.024 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 392.792 |
| Skew: | 1.369 | Prob(JB): | 5.09e-86 |
| Kurtosis: | 4.290 | Cond. No. | 35.5 |

Warnings:  
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [109]:

#separate numeric and categorical values  
varNum = ['Age','BusinessTravel','DistanceFromHome','EnvironmentSatisfaction','Gender','HourlyRate','JobInvolvement','JobLevel','JobSatisfaction', 'NumCompaniesWorked','OverTime', 'PerformanceRating','RelationshipSatisfaction','TotalWorkingYears', 'TrainingTimesLastYear','WorkLifeBalance','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager'] #remove 'MonthlyIncome', 'MonthlyRate','Education','StockOptionLevel','PercentSalaryHike','DailyRate'  
varCat = ['MaritalStatus'] #remove 'Department','JobRole''EducationField',  
varLabel = ['Attrition']  
  
#convert categories to dummy code  
dfCatDummies = pd.get\_dummies(df[varCat],drop\_first=True)  
  
#combine variables and separate label  
x = df[varNum]  
x = pd.concat([x,dfCatDummies], axis=1)  
y = df[varLabel]  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
scaler = MinMaxScaler()  
x\_train[varNum] = scaler.fit\_transform(x\_train[varNum])  
x\_test[varNum] = scaler.fit\_transform(x\_test[varNum])  
  
x\_train\_lm = sm.add\_constant(x\_train)  
lr9 = sm.OLS(y\_train,x\_train\_lm).fit()  
#print(lr4.summary())  
  
# Creating a dataframe that will contain the names of all the feature variables and their vif9s  
vif9 = pd.DataFrame()  
vif9['Features'] = x\_train.columns  
vif9['vif9'] = [variance\_inflation\_factor(x\_train.values, i) for i in range(x\_train.shape[1])]  
vif9['vif9'] = round(vif9['vif9'], 2)  
vif9 = vif9.sort\_values(by = "vif9", ascending = False)  
#print(vif9)

C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:16: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 app.launch\_new\_instance()  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:17: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2542: FutureWarning: Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.  
 return ptp(axis=axis, out=out, \*\*kwargs)

In [110]:

lr9.summary()

Out[110]:

OLS Regression Results

|  |  |  |  |
| --- | --- | --- | --- |
| Dep. Variable: | Attrition | R-squared: | 0.224 |
| Model: | OLS | Adj. R-squared: | 0.207 |
| Method: | Least Squares | F-statistic: | 13.21 |
| Date: | Sat, 02 Oct 2021 | Prob (F-statistic): | 3.93e-42 |
| Time: | 13:21:02 | Log-Likelihood: | -282.35 |
| No. Observations: | 1029 | AIC: | 610.7 |
| Df Residuals: | 1006 | BIC: | 724.2 |
| Df Model: | 22 |  |  |
| Covariance Type: | nonrobust |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | coef | std err | t | P>|t| | [0.025 | 0.975] |
| const | 0.3842 | 0.067 | 5.774 | 0.000 | 0.254 | 0.515 |
| Age | -0.2021 | 0.065 | -3.089 | 0.002 | -0.330 | -0.074 |
| BusinessTravel | 0.1497 | 0.039 | 3.846 | 0.000 | 0.073 | 0.226 |
| DistanceFromHome | 0.1024 | 0.035 | 2.923 | 0.004 | 0.034 | 0.171 |
| EnvironmentSatisfaction | -0.1194 | 0.028 | -4.292 | 0.000 | -0.174 | -0.065 |
| Gender | -0.0389 | 0.021 | -1.879 | 0.061 | -0.079 | 0.002 |
| HourlyRate | -0.0054 | 0.035 | -0.155 | 0.877 | -0.074 | 0.063 |
| JobInvolvement | -0.1941 | 0.043 | -4.566 | 0.000 | -0.278 | -0.111 |
| JobLevel | -0.1072 | 0.058 | -1.834 | 0.067 | -0.222 | 0.007 |
| JobSatisfaction | -0.0978 | 0.028 | -3.515 | 0.000 | -0.152 | -0.043 |
| NumCompaniesWorked | 0.1836 | 0.042 | 4.395 | 0.000 | 0.102 | 0.266 |
| OverTime | 0.2229 | 0.023 | 9.748 | 0.000 | 0.178 | 0.268 |
| PerformanceRating | -0.0172 | 0.028 | -0.624 | 0.533 | -0.071 | 0.037 |
| RelationshipSatisfaction | -0.0407 | 0.028 | -1.453 | 0.147 | -0.096 | 0.014 |
| TotalWorkingYears | -0.1342 | 0.115 | -1.169 | 0.243 | -0.360 | 0.091 |
| TrainingTimesLastYear | -0.0707 | 0.048 | -1.474 | 0.141 | -0.165 | 0.023 |
| WorkLifeBalance | -0.0609 | 0.044 | -1.394 | 0.164 | -0.147 | 0.025 |
| YearsAtCompany | 0.3278 | 0.141 | 2.319 | 0.021 | 0.050 | 0.605 |
| YearsInCurrentRole | -0.1328 | 0.082 | -1.626 | 0.104 | -0.293 | 0.027 |
| YearsSinceLastPromotion | 0.1305 | 0.062 | 2.120 | 0.034 | 0.010 | 0.251 |
| YearsWithCurrManager | -0.2089 | 0.078 | -2.661 | 0.008 | -0.363 | -0.055 |
| MaritalStatus\_Married | 0.0299 | 0.026 | 1.146 | 0.252 | -0.021 | 0.081 |
| MaritalStatus\_Single | 0.1278 | 0.028 | 4.536 | 0.000 | 0.073 | 0.183 |

|  |  |  |  |
| --- | --- | --- | --- |
| Omnibus: | 225.318 | Durbin-Watson: | 2.026 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 395.554 |
| Skew: | 1.373 | Prob(JB): | 1.28e-86 |
| Kurtosis: | 4.297 | Cond. No. | 34.6 |

Warnings:  
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [112]:

vif9

Out[112]:

|  |  |  |
| --- | --- | --- |
|  | Features | vif9 |
| 13 | TotalWorkingYears | 15.11 |
| 16 | YearsAtCompany | 10.97 |
| 0 | Age | 9.88 |
| 15 | WorkLifeBalance | 6.73 |
| 19 | YearsWithCurrManager | 6.25 |
| 17 | YearsInCurrentRole | 6.16 |
| 6 | JobInvolvement | 6.04 |
| 14 | TrainingTimesLastYear | 5.30 |
| 1 | BusinessTravel | 5.24 |
| 7 | JobLevel | 4.95 |
| 5 | HourlyRate | 3.85 |
| 12 | RelationshipSatisfaction | 3.47 |
| 3 | EnvironmentSatisfaction | 3.33 |
| 8 | JobSatisfaction | 3.30 |
| 20 | MaritalStatus\_Married | 2.89 |
| 9 | NumCompaniesWorked | 2.67 |
| 18 | YearsSinceLastPromotion | 2.50 |
| 21 | MaritalStatus\_Single | 2.31 |
| 2 | DistanceFromHome | 1.99 |
| 4 | Gender | 1.72 |
| 10 | OverTime | 1.40 |
| 11 | PerformanceRating | 1.20 |

In [113]:

#separate numeric and categorical values  
varNum = ['Age','BusinessTravel','DistanceFromHome','EnvironmentSatisfaction','Gender','HourlyRate','JobInvolvement','JobLevel','JobSatisfaction', 'NumCompaniesWorked','OverTime', 'PerformanceRating','RelationshipSatisfaction','TotalWorkingYears', 'TrainingTimesLastYear','WorkLifeBalance','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager'] #remove 'MonthlyIncome', 'MonthlyRate','Education','StockOptionLevel','PercentSalaryHike','DailyRate'  
varCat = ['MaritalStatus'] #remove 'Department','JobRole''EducationField',  
varLabel = ['Attrition']  
  
#convert categories to dummy code  
dfCatDummies = pd.get\_dummies(df[varCat],drop\_first=True)  
  
#combine variables and separate label  
x = df[varNum]  
x = pd.concat([x,dfCatDummies], axis=1)  
y = df[varLabel]  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
scaler = MinMaxScaler()  
x\_train[varNum] = scaler.fit\_transform(x\_train[varNum])  
x\_test[varNum] = scaler.fit\_transform(x\_test[varNum])  
  
x\_train\_lm = sm.add\_constant(x\_train)  
lr10 = sm.OLS(y\_train,x\_train\_lm).fit()  
#print(lr4.summary())  
  
# Creating a dataframe that will contain the names of all the feature variables and their vif10s  
vif10 = pd.DataFrame()  
vif10['Features'] = x\_train.columns  
vif10['vif10'] = [variance\_inflation\_factor(x\_train.values, i) for i in range(x\_train.shape[1])]  
vif10['vif10'] = round(vif10['vif10'], 2)  
vif10 = vif10.sort\_values(by = "vif10", ascending = False)  
#print(vif10)

C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:16: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 app.launch\_new\_instance()  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:17: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2542: FutureWarning: Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.  
 return ptp(axis=axis, out=out, \*\*kwargs)

In [114]:

lr10.summary()

Out[114]:

OLS Regression Results

|  |  |  |  |
| --- | --- | --- | --- |
| Dep. Variable: | Attrition | R-squared: | 0.224 |
| Model: | OLS | Adj. R-squared: | 0.207 |
| Method: | Least Squares | F-statistic: | 13.21 |
| Date: | Sat, 02 Oct 2021 | Prob (F-statistic): | 3.93e-42 |
| Time: | 13:23:40 | Log-Likelihood: | -282.35 |
| No. Observations: | 1029 | AIC: | 610.7 |
| Df Residuals: | 1006 | BIC: | 724.2 |
| Df Model: | 22 |  |  |
| Covariance Type: | nonrobust |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | coef | std err | t | P>|t| | [0.025 | 0.975] |
| const | 0.3842 | 0.067 | 5.774 | 0.000 | 0.254 | 0.515 |
| Age | -0.2021 | 0.065 | -3.089 | 0.002 | -0.330 | -0.074 |
| BusinessTravel | 0.1497 | 0.039 | 3.846 | 0.000 | 0.073 | 0.226 |
| DistanceFromHome | 0.1024 | 0.035 | 2.923 | 0.004 | 0.034 | 0.171 |
| EnvironmentSatisfaction | -0.1194 | 0.028 | -4.292 | 0.000 | -0.174 | -0.065 |
| Gender | -0.0389 | 0.021 | -1.879 | 0.061 | -0.079 | 0.002 |
| HourlyRate | -0.0054 | 0.035 | -0.155 | 0.877 | -0.074 | 0.063 |
| JobInvolvement | -0.1941 | 0.043 | -4.566 | 0.000 | -0.278 | -0.111 |
| JobLevel | -0.1072 | 0.058 | -1.834 | 0.067 | -0.222 | 0.007 |
| JobSatisfaction | -0.0978 | 0.028 | -3.515 | 0.000 | -0.152 | -0.043 |
| NumCompaniesWorked | 0.1836 | 0.042 | 4.395 | 0.000 | 0.102 | 0.266 |
| OverTime | 0.2229 | 0.023 | 9.748 | 0.000 | 0.178 | 0.268 |
| PerformanceRating | -0.0172 | 0.028 | -0.624 | 0.533 | -0.071 | 0.037 |
| RelationshipSatisfaction | -0.0407 | 0.028 | -1.453 | 0.147 | -0.096 | 0.014 |
| TotalWorkingYears | -0.1342 | 0.115 | -1.169 | 0.243 | -0.360 | 0.091 |
| TrainingTimesLastYear | -0.0707 | 0.048 | -1.474 | 0.141 | -0.165 | 0.023 |
| WorkLifeBalance | -0.0609 | 0.044 | -1.394 | 0.164 | -0.147 | 0.025 |
| YearsAtCompany | 0.3278 | 0.141 | 2.319 | 0.021 | 0.050 | 0.605 |
| YearsInCurrentRole | -0.1328 | 0.082 | -1.626 | 0.104 | -0.293 | 0.027 |
| YearsSinceLastPromotion | 0.1305 | 0.062 | 2.120 | 0.034 | 0.010 | 0.251 |
| YearsWithCurrManager | -0.2089 | 0.078 | -2.661 | 0.008 | -0.363 | -0.055 |
| MaritalStatus\_Married | 0.0299 | 0.026 | 1.146 | 0.252 | -0.021 | 0.081 |
| MaritalStatus\_Single | 0.1278 | 0.028 | 4.536 | 0.000 | 0.073 | 0.183 |

|  |  |  |  |
| --- | --- | --- | --- |
| Omnibus: | 225.318 | Durbin-Watson: | 2.026 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 395.554 |
| Skew: | 1.373 | Prob(JB): | 1.28e-86 |
| Kurtosis: | 4.297 | Cond. No. | 34.6 |

Warnings:  
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [115]:

vif10

Out[115]:

|  |  |  |
| --- | --- | --- |
|  | Features | vif10 |
| 13 | TotalWorkingYears | 15.11 |
| 16 | YearsAtCompany | 10.97 |
| 0 | Age | 9.88 |
| 15 | WorkLifeBalance | 6.73 |
| 19 | YearsWithCurrManager | 6.25 |
| 17 | YearsInCurrentRole | 6.16 |
| 6 | JobInvolvement | 6.04 |
| 14 | TrainingTimesLastYear | 5.30 |
| 1 | BusinessTravel | 5.24 |
| 7 | JobLevel | 4.95 |
| 5 | HourlyRate | 3.85 |
| 12 | RelationshipSatisfaction | 3.47 |
| 3 | EnvironmentSatisfaction | 3.33 |
| 8 | JobSatisfaction | 3.30 |
| 20 | MaritalStatus\_Married | 2.89 |
| 9 | NumCompaniesWorked | 2.67 |
| 18 | YearsSinceLastPromotion | 2.50 |
| 21 | MaritalStatus\_Single | 2.31 |
| 2 | DistanceFromHome | 1.99 |
| 4 | Gender | 1.72 |
| 10 | OverTime | 1.40 |
| 11 | PerformanceRating | 1.20 |

In [116]:

#separate numeric and categorical values  
varNum = ['Age','BusinessTravel','DistanceFromHome','EnvironmentSatisfaction','Gender','HourlyRate','JobInvolvement','JobLevel','JobSatisfaction', 'NumCompaniesWorked','OverTime', 'PerformanceRating','RelationshipSatisfaction', 'TrainingTimesLastYear','WorkLifeBalance','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager'] #remove 'MonthlyIncome', 'MonthlyRate','Education','StockOptionLevel','PercentSalaryHike','DailyRate','TotalWorkingYears'  
varCat = ['MaritalStatus'] #remove 'Department','JobRole''EducationField',  
varLabel = ['Attrition']  
  
#convert categories to dummy code  
dfCatDummies = pd.get\_dummies(df[varCat],drop\_first=True)  
  
#combine variables and separate label  
x = df[varNum]  
x = pd.concat([x,dfCatDummies], axis=1)  
y = df[varLabel]  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
scaler = MinMaxScaler()  
x\_train[varNum] = scaler.fit\_transform(x\_train[varNum])  
x\_test[varNum] = scaler.fit\_transform(x\_test[varNum])  
  
x\_train\_lm = sm.add\_constant(x\_train)  
lr11 = sm.OLS(y\_train,x\_train\_lm).fit()  
#print(lr4.summary())  
  
# Creating a dataframe that will contain the names of all the feature variables and their vif11s  
vif11 = pd.DataFrame()  
vif11['Features'] = x\_train.columns  
vif11['vif11'] = [variance\_inflation\_factor(x\_train.values, i) for i in range(x\_train.shape[1])]  
vif11['vif11'] = round(vif11['vif11'], 2)  
vif11 = vif11.sort\_values(by = "vif11", ascending = False)  
#print(vif11)

C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:16: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 app.launch\_new\_instance()  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:17: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2542: FutureWarning: Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.  
 return ptp(axis=axis, out=out, \*\*kwargs)

In [117]:

lr11.summary()

Out[117]:

OLS Regression Results

|  |  |  |  |
| --- | --- | --- | --- |
| Dep. Variable: | Attrition | R-squared: | 0.223 |
| Model: | OLS | Adj. R-squared: | 0.207 |
| Method: | Least Squares | F-statistic: | 13.77 |
| Date: | Sat, 02 Oct 2021 | Prob (F-statistic): | 1.97e-42 |
| Time: | 13:27:34 | Log-Likelihood: | -283.04 |
| No. Observations: | 1029 | AIC: | 610.1 |
| Df Residuals: | 1007 | BIC: | 718.7 |
| Df Model: | 21 |  |  |
| Covariance Type: | nonrobust |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | coef | std err | t | P>|t| | [0.025 | 0.975] |
| const | 0.3898 | 0.066 | 5.872 | 0.000 | 0.260 | 0.520 |
| Age | -0.2409 | 0.056 | -4.273 | 0.000 | -0.352 | -0.130 |
| BusinessTravel | 0.1481 | 0.039 | 3.806 | 0.000 | 0.072 | 0.224 |
| DistanceFromHome | 0.1008 | 0.035 | 2.879 | 0.004 | 0.032 | 0.169 |
| EnvironmentSatisfaction | -0.1191 | 0.028 | -4.282 | 0.000 | -0.174 | -0.065 |
| Gender | -0.0382 | 0.021 | -1.849 | 0.065 | -0.079 | 0.002 |
| HourlyRate | -0.0072 | 0.035 | -0.206 | 0.836 | -0.075 | 0.061 |
| JobInvolvement | -0.1953 | 0.043 | -4.594 | 0.000 | -0.279 | -0.112 |
| JobLevel | -0.1464 | 0.048 | -3.062 | 0.002 | -0.240 | -0.053 |
| JobSatisfaction | -0.0966 | 0.028 | -3.474 | 0.001 | -0.151 | -0.042 |
| NumCompaniesWorked | 0.1708 | 0.040 | 4.236 | 0.000 | 0.092 | 0.250 |
| OverTime | 0.2222 | 0.023 | 9.719 | 0.000 | 0.177 | 0.267 |
| PerformanceRating | -0.0190 | 0.027 | -0.692 | 0.489 | -0.073 | 0.035 |
| RelationshipSatisfaction | -0.0408 | 0.028 | -1.456 | 0.146 | -0.096 | 0.014 |
| TrainingTimesLastYear | -0.0703 | 0.048 | -1.465 | 0.143 | -0.164 | 0.024 |
| WorkLifeBalance | -0.0592 | 0.044 | -1.355 | 0.176 | -0.145 | 0.027 |
| YearsAtCompany | 0.2741 | 0.134 | 2.050 | 0.041 | 0.012 | 0.536 |
| YearsInCurrentRole | -0.1317 | 0.082 | -1.612 | 0.107 | -0.292 | 0.029 |
| YearsSinceLastPromotion | 0.1285 | 0.062 | 2.088 | 0.037 | 0.008 | 0.249 |
| YearsWithCurrManager | -0.2107 | 0.078 | -2.684 | 0.007 | -0.365 | -0.057 |
| MaritalStatus\_Married | 0.0289 | 0.026 | 1.109 | 0.268 | -0.022 | 0.080 |
| MaritalStatus\_Single | 0.1268 | 0.028 | 4.502 | 0.000 | 0.072 | 0.182 |

|  |  |  |  |
| --- | --- | --- | --- |
| Omnibus: | 226.406 | Durbin-Watson: | 2.030 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 398.891 |
| Skew: | 1.376 | Prob(JB): | 2.41e-87 |
| Kurtosis: | 4.314 | Cond. No. | 31.4 |

Warnings:  
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [118]:

vif11

Out[118]:

|  |  |  |
| --- | --- | --- |
|  | Features | vif11 |
| 15 | YearsAtCompany | 9.82 |
| 0 | Age | 7.34 |
| 14 | WorkLifeBalance | 6.70 |
| 18 | YearsWithCurrManager | 6.25 |
| 16 | YearsInCurrentRole | 6.16 |
| 6 | JobInvolvement | 6.04 |
| 13 | TrainingTimesLastYear | 5.29 |
| 1 | BusinessTravel | 5.23 |
| 5 | HourlyRate | 3.84 |
| 12 | RelationshipSatisfaction | 3.47 |
| 3 | EnvironmentSatisfaction | 3.33 |
| 7 | JobLevel | 3.30 |
| 8 | JobSatisfaction | 3.28 |
| 19 | MaritalStatus\_Married | 2.89 |
| 17 | YearsSinceLastPromotion | 2.50 |
| 9 | NumCompaniesWorked | 2.49 |
| 20 | MaritalStatus\_Single | 2.31 |
| 2 | DistanceFromHome | 1.99 |
| 4 | Gender | 1.72 |
| 10 | OverTime | 1.40 |
| 11 | PerformanceRating | 1.20 |

In [119]:

#separate numeric and categorical values  
varNum = ['Age','BusinessTravel','DistanceFromHome','EnvironmentSatisfaction','Gender','JobInvolvement','JobLevel','JobSatisfaction', 'NumCompaniesWorked','OverTime', 'PerformanceRating','RelationshipSatisfaction','TotalWorkingYears', 'TrainingTimesLastYear','WorkLifeBalance','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager'] #remove 'MonthlyIncome', 'MonthlyRate','Education','StockOptionLevel','PercentSalaryHike','DailyRate','HourlyRate'  
varCat = ['MaritalStatus'] #remove 'Department','JobRole''EducationField',  
varLabel = ['Attrition']  
  
#convert categories to dummy code  
dfCatDummies = pd.get\_dummies(df[varCat],drop\_first=True)  
  
#combine variables and separate label  
x = df[varNum]  
x = pd.concat([x,dfCatDummies], axis=1)  
y = df[varLabel]  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
scaler = MinMaxScaler()  
x\_train[varNum] = scaler.fit\_transform(x\_train[varNum])  
x\_test[varNum] = scaler.fit\_transform(x\_test[varNum])  
  
x\_train\_lm = sm.add\_constant(x\_train)  
lr12 = sm.OLS(y\_train,x\_train\_lm).fit()  
#print(lr4.summary())  
  
# Creating a dataframe that will contain the names of all the feature variables and their vif12s  
vif12 = pd.DataFrame()  
vif12['Features'] = x\_train.columns  
vif12['vif12'] = [variance\_inflation\_factor(x\_train.values, i) for i in range(x\_train.shape[1])]  
vif12['vif12'] = round(vif12['vif12'], 2)  
vif12 = vif12.sort\_values(by = "vif12", ascending = False)  
#print(vif12)

C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:16: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 app.launch\_new\_instance()  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:17: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2542: FutureWarning: Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.  
 return ptp(axis=axis, out=out, \*\*kwargs)

In [120]:

lr12.summary()

Out[120]:

OLS Regression Results

|  |  |  |  |
| --- | --- | --- | --- |
| Dep. Variable: | Attrition | R-squared: | 0.224 |
| Model: | OLS | Adj. R-squared: | 0.208 |
| Method: | Least Squares | F-statistic: | 13.85 |
| Date: | Sat, 02 Oct 2021 | Prob (F-statistic): | 1.05e-42 |
| Time: | 13:29:08 | Log-Likelihood: | -282.36 |
| No. Observations: | 1029 | AIC: | 608.7 |
| Df Residuals: | 1007 | BIC: | 717.3 |
| Df Model: | 21 |  |  |
| Covariance Type: | nonrobust |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | coef | std err | t | P>|t| | [0.025 | 0.975] |
| const | 0.3816 | 0.064 | 5.926 | 0.000 | 0.255 | 0.508 |
| Age | -0.2024 | 0.065 | -3.096 | 0.002 | -0.331 | -0.074 |
| BusinessTravel | 0.1496 | 0.039 | 3.846 | 0.000 | 0.073 | 0.226 |
| DistanceFromHome | 0.1024 | 0.035 | 2.924 | 0.004 | 0.034 | 0.171 |
| EnvironmentSatisfaction | -0.1191 | 0.028 | -4.292 | 0.000 | -0.174 | -0.065 |
| Gender | -0.0389 | 0.021 | -1.883 | 0.060 | -0.080 | 0.002 |
| JobInvolvement | -0.1944 | 0.042 | -4.581 | 0.000 | -0.278 | -0.111 |
| JobLevel | -0.1066 | 0.058 | -1.829 | 0.068 | -0.221 | 0.008 |
| JobSatisfaction | -0.0976 | 0.028 | -3.513 | 0.000 | -0.152 | -0.043 |
| NumCompaniesWorked | 0.1838 | 0.042 | 4.403 | 0.000 | 0.102 | 0.266 |
| OverTime | 0.2229 | 0.023 | 9.757 | 0.000 | 0.178 | 0.268 |
| PerformanceRating | -0.0172 | 0.027 | -0.626 | 0.532 | -0.071 | 0.037 |
| RelationshipSatisfaction | -0.0407 | 0.028 | -1.452 | 0.147 | -0.096 | 0.014 |
| TotalWorkingYears | -0.1350 | 0.115 | -1.178 | 0.239 | -0.360 | 0.090 |
| TrainingTimesLastYear | -0.0709 | 0.048 | -1.480 | 0.139 | -0.165 | 0.023 |
| WorkLifeBalance | -0.0612 | 0.044 | -1.401 | 0.162 | -0.147 | 0.025 |
| YearsAtCompany | 0.3278 | 0.141 | 2.320 | 0.021 | 0.051 | 0.605 |
| YearsInCurrentRole | -0.1327 | 0.082 | -1.626 | 0.104 | -0.293 | 0.027 |
| YearsSinceLastPromotion | 0.1305 | 0.062 | 2.121 | 0.034 | 0.010 | 0.251 |
| YearsWithCurrManager | -0.2083 | 0.078 | -2.658 | 0.008 | -0.362 | -0.055 |
| MaritalStatus\_Married | 0.0299 | 0.026 | 1.146 | 0.252 | -0.021 | 0.081 |
| MaritalStatus\_Single | 0.1279 | 0.028 | 4.543 | 0.000 | 0.073 | 0.183 |

|  |  |  |  |
| --- | --- | --- | --- |
| Omnibus: | 225.327 | Durbin-Watson: | 2.026 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 395.563 |
| Skew: | 1.373 | Prob(JB): | 1.27e-86 |
| Kurtosis: | 4.297 | Cond. No. | 33.7 |

Warnings:  
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [121]:

#separate numeric and categorical values  
varNum = ['Age','BusinessTravel','DistanceFromHome','EnvironmentSatisfaction','Gender','JobInvolvement','JobLevel','JobSatisfaction', 'NumCompaniesWorked','OverTime','RelationshipSatisfaction','TotalWorkingYears', 'TrainingTimesLastYear','WorkLifeBalance','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager'] #remove 'MonthlyIncome', 'MonthlyRate','Education','StockOptionLevel','PercentSalaryHike','DailyRate', 'PerformanceRating'  
varCat = ['MaritalStatus'] #remove 'Department','JobRole''EducationField',  
varLabel = ['Attrition']  
  
#convert categories to dummy code  
dfCatDummies = pd.get\_dummies(df[varCat],drop\_first=True)  
  
#combine variables and separate label  
x = df[varNum]  
x = pd.concat([x,dfCatDummies], axis=1)  
y = df[varLabel]  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
scaler = MinMaxScaler()  
x\_train[varNum] = scaler.fit\_transform(x\_train[varNum])  
x\_test[varNum] = scaler.fit\_transform(x\_test[varNum])  
  
x\_train\_lm = sm.add\_constant(x\_train)  
lr13 = sm.OLS(y\_train,x\_train\_lm).fit()  
#print(lr4.summary())  
  
# Creating a dataframe that will contain the names of all the feature variables and their vif13s  
vif13 = pd.DataFrame()  
vif13['Features'] = x\_train.columns  
vif13['vif13'] = [variance\_inflation\_factor(x\_train.values, i) for i in range(x\_train.shape[1])]  
vif13['vif13'] = round(vif13['vif13'], 2)  
vif13 = vif13.sort\_values(by = "vif13", ascending = False)  
#print(vif13)

C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:16: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 app.launch\_new\_instance()  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\ipykernel\_launcher.py:17: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
C:\Users\nagem\Anaconda3\lib\site-packages\pandas\core\indexing.py:543: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy  
 self.obj[item] = s  
C:\Users\nagem\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2542: FutureWarning: Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.  
 return ptp(axis=axis, out=out, \*\*kwargs)

In [122]:

lr13.summary()

Out[122]:

OLS Regression Results

|  |  |  |  |
| --- | --- | --- | --- |
| Dep. Variable: | Attrition | R-squared: | 0.224 |
| Model: | OLS | Adj. R-squared: | 0.208 |
| Method: | Least Squares | F-statistic: | 13.83 |
| Date: | Sat, 02 Oct 2021 | Prob (F-statistic): | 1.25e-42 |
| Time: | 13:31:24 | Log-Likelihood: | -282.54 |
| No. Observations: | 1029 | AIC: | 609.1 |
| Df Residuals: | 1007 | BIC: | 717.7 |
| Df Model: | 21 |  |  |
| Covariance Type: | nonrobust |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | coef | std err | t | P>|t| | [0.025 | 0.975] |
| const | 0.3806 | 0.066 | 5.743 | 0.000 | 0.251 | 0.511 |
| Age | -0.2019 | 0.065 | -3.087 | 0.002 | -0.330 | -0.074 |
| BusinessTravel | 0.1503 | 0.039 | 3.862 | 0.000 | 0.074 | 0.227 |
| DistanceFromHome | 0.1025 | 0.035 | 2.926 | 0.004 | 0.034 | 0.171 |
| EnvironmentSatisfaction | -0.1193 | 0.028 | -4.290 | 0.000 | -0.174 | -0.065 |
| Gender | -0.0393 | 0.021 | -1.901 | 0.058 | -0.080 | 0.001 |
| HourlyRate | -0.0056 | 0.035 | -0.162 | 0.871 | -0.074 | 0.063 |
| JobInvolvement | -0.1933 | 0.042 | -4.551 | 0.000 | -0.277 | -0.110 |
| JobLevel | -0.1054 | 0.058 | -1.806 | 0.071 | -0.220 | 0.009 |
| JobSatisfaction | -0.0979 | 0.028 | -3.519 | 0.000 | -0.152 | -0.043 |
| NumCompaniesWorked | 0.1844 | 0.042 | 4.419 | 0.000 | 0.103 | 0.266 |
| OverTime | 0.2230 | 0.023 | 9.757 | 0.000 | 0.178 | 0.268 |
| RelationshipSatisfaction | -0.0400 | 0.028 | -1.428 | 0.154 | -0.095 | 0.015 |
| TotalWorkingYears | -0.1384 | 0.115 | -1.207 | 0.228 | -0.363 | 0.086 |
| TrainingTimesLastYear | -0.0699 | 0.048 | -1.458 | 0.145 | -0.164 | 0.024 |
| WorkLifeBalance | -0.0612 | 0.044 | -1.402 | 0.161 | -0.147 | 0.024 |
| YearsAtCompany | 0.3336 | 0.141 | 2.366 | 0.018 | 0.057 | 0.610 |
| YearsInCurrentRole | -0.1354 | 0.082 | -1.661 | 0.097 | -0.295 | 0.025 |
| YearsSinceLastPromotion | 0.1304 | 0.062 | 2.120 | 0.034 | 0.010 | 0.251 |
| YearsWithCurrManager | -0.2103 | 0.078 | -2.681 | 0.007 | -0.364 | -0.056 |
| MaritalStatus\_Married | 0.0297 | 0.026 | 1.139 | 0.255 | -0.021 | 0.081 |
| MaritalStatus\_Single | 0.1282 | 0.028 | 4.550 | 0.000 | 0.073 | 0.183 |

|  |  |  |  |
| --- | --- | --- | --- |
| Omnibus: | 225.764 | Durbin-Watson: | 2.026 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 396.730 |
| Skew: | 1.375 | Prob(JB): | 7.10e-87 |
| Kurtosis: | 4.299 | Cond. No. | 34.4 |

Warnings:  
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [ ]:

In [ ]:

In [ ]:

#separate numeric and categorical values  
varNum = ['Age','BusinessTravel','DistanceFromHome','EnvironmentSatisfaction','Gender','HourlyRate','JobInvolvement','JobLevel','JobSatisfaction', 'NumCompaniesWorked','OverTime', 'PerformanceRating','RelationshipSatisfaction','TotalWorkingYears', 'TrainingTimesLastYear','WorkLifeBalance','YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion','YearsWithCurrManager'] #remove 'MonthlyIncome', 'MonthlyRate','Education','StockOptionLevel','PercentSalaryHike','DailyRate'  
varCat = ['MaritalStatus'] #remove 'Department','JobRole''EducationField',  
varLabel = ['Attrition']  
  
#convert categories to dummy code  
dfCatDummies = pd.get\_dummies(df[varCat],drop\_first=True)  
  
#combine variables and separate label  
x = df[varNum]  
x = pd.concat([x,dfCatDummies], axis=1)  
y = df[varLabel]  
  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
scaler = MinMaxScaler()  
x\_train[varNum] = scaler.fit\_transform(x\_train[varNum])  
x\_test[varNum] = scaler.fit\_transform(x\_test[varNum])  
  
x\_train\_lm = sm.add\_constant(x\_train)  
lrzzz = sm.OLS(y\_train,x\_train\_lm).fit()  
#print(lr4.summary())  
  
# Creating a dataframe that will contain the names of all the feature variables and their vifzzzs  
vifzzz = pd.DataFrame()  
vifzzz['Features'] = x\_train.columns  
vifzzz['vifzzz'] = [variance\_inflation\_factor(x\_train.values, i) for i in range(x\_train.shape[1])]  
vifzzz['vifzzz'] = round(vifzzz['vifzzz'], 2)  
vifzzz = vifzzz.sort\_values(by = "vifzzz", ascending = False)  
#print(vifzzz)

In [108]:

x\_train.head()

Out[108]:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Age | DailyRate | BusinessTravel | DistanceFromHome | EnvironmentSatisfaction | Gender | HourlyRate | JobInvolvement | JobLevel | JobSatisfaction | ... | RelationshipSatisfaction | TotalWorkingYears | TrainingTimesLastYear | WorkLifeBalance | YearsAtCompany | YearsInCurrentRole | YearsSinceLastPromotion | YearsWithCurrManager | MaritalStatus\_Married | MaritalStatus\_Single |
| 1141 | 0.285714 | 0.099713 | 0.5 | 0.214286 | 0.333333 | 0.0 | 0.257143 | 0.333333 | 0.00 | 0.333333 | ... | 0.333333 | 0.150 | 0.500000 | 0.333333 | 0.150 | 0.222222 | 0.066667 | 0.058824 | 1 | 0 |
| 1330 | 0.595238 | 0.517217 | 0.5 | 0.178571 | 0.000000 | 1.0 | 0.728571 | 0.333333 | 1.00 | 0.666667 | ... | 1.000000 | 0.525 | 0.333333 | 0.666667 | 0.400 | 0.666667 | 0.400000 | 0.823529 | 1 | 0 |
| 1205 | 0.333333 | 0.829986 | 0.5 | 0.035714 | 1.000000 | 0.0 | 0.928571 | 0.666667 | 0.00 | 0.333333 | ... | 0.000000 | 0.025 | 0.333333 | 0.666667 | 0.025 | 0.000000 | 0.000000 | 0.000000 | 0 | 1 |
| 679 | 0.309524 | 0.779053 | 0.0 | 0.678571 | 1.000000 | 1.0 | 0.214286 | 0.666667 | 0.25 | 0.666667 | ... | 1.000000 | 0.225 | 0.333333 | 0.333333 | 0.225 | 0.444444 | 0.000000 | 0.000000 | 1 | 0 |
| 1366 | 0.500000 | 0.824247 | 0.0 | 0.714286 | 0.000000 | 1.0 | 0.028571 | 0.000000 | 0.25 | 0.666667 | ... | 0.666667 | 0.250 | 0.166667 | 0.666667 | 0.075 | 0.111111 | 0.066667 | 0.117647 | 1 | 0 |

5 rows × 23 columns

In [ ]:

In [80]:

x\_train\_logit = sm.add\_constant(x\_train)  
log1 = sm.Logit(y\_train,x\_train\_logit).fit()  
log1.summary()

Warning: Maximum number of iterations has been exceeded.  
 Current function value: 0.283017  
 Iterations: 35

Using Naive Bays

In [18]:

from sklearn.preprocessing import StandardScaler

In [19]:

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
sc = StandardScaler()  
x\_train = sc.fit\_transform(x\_train)  
x\_test = sc.fit\_transform(x\_test)

In [20]:

from sklearn.naive\_bayes import GaussianNB

In [21]:

classifier = GaussianNB()  
classifier.fit(x\_train,y\_train)

C:\Users\nagem\Anaconda3\lib\site-packages\sklearn\naive\_bayes.py:206: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n\_samples, ), for example using ravel().  
 y = column\_or\_1d(y, warn=True)

In [25]:

y\_pred = classifier.predict(x\_test)

In [29]:

from sklearn.metrics import confusion\_matrix, accuracy\_score, ConfusionMatrixDisplay

In [46]:

cm = confusion\_matrix(y\_test, y\_pred,labels=classifier.classes\_)  
ac = accuracy\_score(y\_test,y\_pred)  
#print(cm)  
print('Accuracy: ',ac)  
rcParams['figure.figsize'] = 5, 5  
sns.heatmap(cm,square=True,annot=True,fmt='d',cbar=False,xticklabels={'0','1'},yticklabels={'0','1'})  
plt.title('Confusion Matrix from Naive Bayes Prediction')  
plt.xlabel('predicted label')  
plt.ylabel('actual label')  
plt.show()

Accuracy: 0.6122448979591837

In [42]:

tn, fp, fn, tp = confusion\_matrix(y\_test,y\_pred).ravel()  
print(tn, fp, fn, tp)

213 150 21 57

In [37]:

print(ConfusionMatrixDisplay(cm,display\_labels={'0','1'}))  
#plt.show()

<sklearn.metrics.\_plot.confusion\_matrix.ConfusionMatrixDisplay object at 0x000001A4013E0E88>

Using decision tree

In [47]:

from sklearn import tree

In [48]:

dtree = tree.DecisionTreeClassifier()  
dtree = dtree.fit(x,y)  
tree.plot\_tree(dtree)

Out[48]:

[Text(122.77339538409703, 265.005, 'X[14] <= 0.5\ngini = 0.27\nsamples = 1470\nvalue = [1233, 237]'),  
 Text(45.44191013669619, 251.41500000000002, 'X[19] <= 2.5\ngini = 0.187\nsamples = 1054\nvalue = [944, 110]'),  
 Text(15.470157874470543, 237.82500000000002, 'X[38] <= 0.5\ngini = 0.442\nsamples = 88\nvalue = [59, 29]'),  
 Text(8.594532152483636, 224.235, 'X[7] <= 58.5\ngini = 0.495\nsamples = 60\nvalue = [33, 27]'),  
 Text(3.437812860993454, 210.645, 'X[5] <= 2.5\ngini = 0.435\nsamples = 25\nvalue = [8, 17]'),  
 Text(1.718906430496727, 197.055, 'gini = 0.0\nsamples = 11\nvalue = [0, 11]'),  
 Text(5.156719291490181, 197.055, 'X[11] <= 2156.5\ngini = 0.49\nsamples = 14\nvalue = [8, 6]'),  
 Text(3.437812860993454, 183.46500000000003, 'X[0] <= 32.0\ngini = 0.245\nsamples = 7\nvalue = [1, 6]'),  
 Text(1.718906430496727, 169.875, 'gini = 0.0\nsamples = 6\nvalue = [0, 6]'),  
 Text(5.156719291490181, 169.875, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(6.875625721986908, 183.46500000000003, 'gini = 0.0\nsamples = 7\nvalue = [7, 0]'),  
 Text(13.751251443973816, 210.645, 'X[4] <= 3.5\ngini = 0.408\nsamples = 35\nvalue = [25, 10]'),  
 Text(12.032345013477089, 197.055, 'X[20] <= 1.0\ngini = 0.342\nsamples = 32\nvalue = [25, 7]'),  
 Text(10.313438582980362, 183.46500000000003, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(13.751251443973816, 183.46500000000003, 'X[10] <= 1.5\ngini = 0.278\nsamples = 30\nvalue = [25, 5]'),  
 Text(10.313438582980362, 169.875, 'X[7] <= 80.0\ngini = 0.444\nsamples = 3\nvalue = [1, 2]'),  
 Text(8.594532152483636, 156.28500000000003, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(12.032345013477089, 156.28500000000003, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(17.18906430496727, 169.875, 'X[21] <= 2.5\ngini = 0.198\nsamples = 27\nvalue = [24, 3]'),  
 Text(15.470157874470543, 156.28500000000003, 'X[3] <= 8.0\ngini = 0.49\nsamples = 7\nvalue = [4, 3]'),  
 Text(13.751251443973816, 142.69500000000002, 'gini = 0.0\nsamples = 4\nvalue = [4, 0]'),  
 Text(17.18906430496727, 142.69500000000002, 'gini = 0.0\nsamples = 3\nvalue = [0, 3]'),  
 Text(18.907970735463998, 156.28500000000003, 'gini = 0.0\nsamples = 20\nvalue = [20, 0]'),  
 Text(15.470157874470543, 197.055, 'gini = 0.0\nsamples = 3\nvalue = [0, 3]'),  
 Text(22.34578359645745, 224.235, 'X[31] <= 0.5\ngini = 0.133\nsamples = 28\nvalue = [26, 2]'),  
 Text(20.626877165960725, 210.645, 'X[1] <= 227.0\ngini = 0.071\nsamples = 27\nvalue = [26, 1]'),  
 Text(18.907970735463998, 197.055, 'X[0] <= 30.5\ngini = 0.5\nsamples = 2\nvalue = [1, 1]'),  
 Text(17.18906430496727, 183.46500000000003, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(20.626877165960725, 183.46500000000003, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(22.34578359645745, 197.055, 'gini = 0.0\nsamples = 25\nvalue = [25, 0]'),  
 Text(24.064690026954178, 210.645, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(75.41366239892183, 237.82500000000002, 'X[21] <= 1.5\ngini = 0.154\nsamples = 966\nvalue = [885, 81]'),  
 Text(31.79976896418945, 224.235, 'X[25] <= 0.5\ngini = 0.37\nsamples = 53\nvalue = [40, 13]'),  
 Text(27.50250288794763, 210.645, 'X[0] <= 44.0\ngini = 0.408\nsamples = 7\nvalue = [2, 5]'),  
 Text(25.783596457450905, 197.055, 'gini = 0.0\nsamples = 5\nvalue = [0, 5]'),  
 Text(29.22140931844436, 197.055, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),  
 Text(36.097035040431265, 210.645, 'X[24] <= 6.0\ngini = 0.287\nsamples = 46\nvalue = [38, 8]'),  
 Text(32.65922217943781, 197.055, 'X[10] <= 1.5\ngini = 0.18\nsamples = 40\nvalue = [36, 4]'),  
 Text(30.940315748941085, 183.46500000000003, 'X[1] <= 438.0\ngini = 0.444\nsamples = 12\nvalue = [8, 4]'),  
 Text(27.50250288794763, 169.875, 'X[0] <= 44.5\ngini = 0.375\nsamples = 4\nvalue = [1, 3]'),  
 Text(25.783596457450905, 156.28500000000003, 'gini = 0.0\nsamples = 3\nvalue = [0, 3]'),  
 Text(29.22140931844436, 156.28500000000003, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(34.37812860993454, 169.875, 'X[1] <= 1303.5\ngini = 0.219\nsamples = 8\nvalue = [7, 1]'),  
 Text(32.65922217943781, 156.28500000000003, 'gini = 0.0\nsamples = 7\nvalue = [7, 0]'),  
 Text(36.097035040431265, 156.28500000000003, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(34.37812860993454, 183.46500000000003, 'gini = 0.0\nsamples = 28\nvalue = [28, 0]'),  
 Text(39.53484790142472, 197.055, 'X[10] <= 2.5\ngini = 0.444\nsamples = 6\nvalue = [2, 4]'),  
 Text(37.815941470927996, 183.46500000000003, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),  
 Text(41.25375433192145, 183.46500000000003, 'gini = 0.0\nsamples = 4\nvalue = [0, 4]'),  
 Text(119.02755583365422, 224.235, 'X[19] <= 39.0\ngini = 0.138\nsamples = 913\nvalue = [845, 68]'),  
 Text(117.30864940315749, 210.645, 'X[13] <= 4.5\ngini = 0.136\nsamples = 912\nvalue = [845, 67]'),  
 Text(98.9311224489796, 197.055, 'X[24] <= 14.5\ngini = 0.107\nsamples = 702\nvalue = [662, 40]'),  
 Text(84.73671544089333, 183.46500000000003, 'X[40] <= 0.5\ngini = 0.103\nsamples = 697\nvalue = [659, 38]'),  
 Text(61.504620716211015, 169.875, 'X[0] <= 43.5\ngini = 0.094\nsamples = 668\nvalue = [635, 33]'),  
 Text(39.53484790142472, 156.28500000000003, 'X[1] <= 110.0\ngini = 0.067\nsamples = 518\nvalue = [500, 18]'),  
 Text(32.76665383134386, 142.69500000000002, 'X[30] <= 0.5\ngini = 0.444\nsamples = 3\nvalue = [2, 1]'),  
 Text(31.047747400847133, 129.10500000000002, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),  
 Text(34.48556026184058, 129.10500000000002, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(46.303041971505586, 142.69500000000002, 'X[3] <= 28.5\ngini = 0.064\nsamples = 515\nvalue = [498, 17]'),  
 Text(37.923373122834036, 129.10500000000002, 'X[11] <= 7437.0\ngini = 0.058\nsamples = 503\nvalue = [488, 15]'),  
 Text(26.32075471698113, 115.51500000000001, 'X[7] <= 72.5\ngini = 0.037\nsamples = 375\nvalue = [368, 7]'),  
 Text(24.601848286484405, 101.92500000000001, 'gini = 0.0\nsamples = 217\nvalue = [217, 0]'),  
 Text(28.039661147477858, 101.92500000000001, 'X[38] <= 0.5\ngini = 0.085\nsamples = 158\nvalue = [151, 7]'),  
 Text(21.16403542549095, 88.33500000000001, 'X[13] <= 3.5\ngini = 0.048\nsamples = 121\nvalue = [118, 3]'),  
 Text(17.726222564497498, 74.745, 'X[25] <= 1.5\ngini = 0.019\nsamples = 103\nvalue = [102, 1]'),  
 Text(16.00731613400077, 61.155, 'X[5] <= 3.5\ngini = 0.198\nsamples = 9\nvalue = [8, 1]'),  
 Text(14.288409703504042, 47.565000000000026, 'gini = 0.0\nsamples = 8\nvalue = [8, 0]'),  
 Text(17.726222564497498, 47.565000000000026, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(19.445128994994224, 61.155, 'gini = 0.0\nsamples = 94\nvalue = [94, 0]'),  
 Text(24.601848286484405, 74.745, 'X[36] <= 0.5\ngini = 0.198\nsamples = 18\nvalue = [16, 2]'),  
 Text(22.882941855987678, 61.155, 'X[12] <= 6439.0\ngini = 0.111\nsamples = 17\nvalue = [16, 1]'),  
 Text(21.16403542549095, 47.565000000000026, 'X[5] <= 2.5\ngini = 0.5\nsamples = 2\nvalue = [1, 1]'),  
 Text(19.445128994994224, 33.97500000000002, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(22.882941855987678, 33.97500000000002, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(24.601848286484405, 47.565000000000026, 'gini = 0.0\nsamples = 15\nvalue = [15, 0]'),  
 Text(26.32075471698113, 61.155, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(34.915286869464765, 88.33500000000001, 'X[13] <= 0.5\ngini = 0.193\nsamples = 37\nvalue = [33, 4]'),  
 Text(31.47747400847131, 74.745, 'X[1] <= 1061.5\ngini = 0.49\nsamples = 7\nvalue = [4, 3]'),  
 Text(29.758567577974585, 61.155, 'X[10] <= 3.5\ngini = 0.375\nsamples = 4\nvalue = [1, 3]'),  
 Text(28.039661147477858, 47.565000000000026, 'gini = 0.0\nsamples = 3\nvalue = [0, 3]'),  
 Text(31.47747400847131, 47.565000000000026, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(33.19638043896804, 61.155, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),  
 Text(38.35309973045822, 74.745, 'X[4] <= 1.5\ngini = 0.064\nsamples = 30\nvalue = [29, 1]'),  
 Text(36.634193299961495, 61.155, 'X[13] <= 1.5\ngini = 0.375\nsamples = 4\nvalue = [3, 1]'),  
 Text(34.915286869464765, 47.565000000000026, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),  
 Text(38.35309973045822, 47.565000000000026, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(40.07200616095495, 61.155, 'gini = 0.0\nsamples = 26\nvalue = [26, 0]'),  
 Text(49.525991528686944, 115.51500000000001, 'X[7] <= 99.5\ngini = 0.117\nsamples = 128\nvalue = [120, 8]'),  
 Text(47.80708509819022, 101.92500000000001, 'X[1] <= 153.5\ngini = 0.104\nsamples = 127\nvalue = [120, 7]'),  
 Text(43.5098190219484, 88.33500000000001, 'X[15] <= 15.5\ngini = 0.48\nsamples = 5\nvalue = [3, 2]'),  
 Text(41.79091259145167, 74.745, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),  
 Text(45.228725452445126, 74.745, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(52.10435117443204, 88.33500000000001, 'X[22] <= 1.5\ngini = 0.079\nsamples = 122\nvalue = [117, 5]'),  
 Text(48.66653831343858, 74.745, 'X[17] <= 1.5\ngini = 0.444\nsamples = 6\nvalue = [4, 2]'),  
 Text(46.947631882941856, 61.155, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(50.38544474393531, 61.155, 'gini = 0.0\nsamples = 4\nvalue = [4, 0]'),  
 Text(55.54216403542549, 74.745, 'X[15] <= 12.5\ngini = 0.05\nsamples = 116\nvalue = [113, 3]'),  
 Text(53.82325760492876, 61.155, 'X[7] <= 33.5\ngini = 0.18\nsamples = 30\nvalue = [27, 3]'),  
 Text(52.10435117443204, 47.565000000000026, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(55.54216403542549, 47.565000000000026, 'X[0] <= 28.5\ngini = 0.128\nsamples = 29\nvalue = [27, 2]'),  
 Text(53.82325760492876, 33.97500000000002, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(57.26107046592222, 33.97500000000002, 'X[3] <= 2.0\ngini = 0.069\nsamples = 28\nvalue = [27, 1]'),  
 Text(55.54216403542549, 20.38500000000002, 'X[20] <= 3.5\ngini = 0.375\nsamples = 4\nvalue = [3, 1]'),  
 Text(53.82325760492876, 6.795000000000016, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),  
 Text(57.26107046592222, 6.795000000000016, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(58.97997689641895, 20.38500000000002, 'gini = 0.0\nsamples = 24\nvalue = [24, 0]'),  
 Text(57.26107046592222, 61.155, 'gini = 0.0\nsamples = 86\nvalue = [86, 0]'),  
 Text(51.244897959183675, 101.92500000000001, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(54.68271082017713, 129.10500000000002, 'X[12] <= 3151.0\ngini = 0.278\nsamples = 12\nvalue = [10, 2]'),  
 Text(52.9638043896804, 115.51500000000001, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(56.40161725067385, 115.51500000000001, 'X[13] <= 0.5\ngini = 0.165\nsamples = 11\nvalue = [10, 1]'),  
 Text(54.68271082017713, 101.92500000000001, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(58.12052368117058, 101.92500000000001, 'gini = 0.0\nsamples = 10\nvalue = [10, 0]'),  
 Text(83.4743935309973, 156.28500000000003, 'X[7] <= 99.5\ngini = 0.18\nsamples = 150\nvalue = [135, 15]'),  
 Text(81.75548710050057, 142.69500000000002, 'X[10] <= 3.5\ngini = 0.17\nsamples = 149\nvalue = [135, 14]'),  
 Text(68.43396226415095, 129.10500000000002, 'X[11] <= 2658.5\ngini = 0.095\nsamples = 100\nvalue = [95, 5]'),  
 Text(63.27724297266076, 115.51500000000001, 'X[11] <= 2326.0\ngini = 0.375\nsamples = 8\nvalue = [6, 2]'),  
 Text(61.558336542164035, 101.92500000000001, 'gini = 0.0\nsamples = 6\nvalue = [6, 0]'),  
 Text(64.99614940315749, 101.92500000000001, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(73.59068155564113, 115.51500000000001, 'X[3] <= 20.5\ngini = 0.063\nsamples = 92\nvalue = [89, 3]'),  
 Text(68.43396226415095, 101.92500000000001, 'X[38] <= 0.5\ngini = 0.025\nsamples = 78\nvalue = [77, 1]'),  
 Text(66.71505583365422, 88.33500000000001, 'gini = 0.0\nsamples = 69\nvalue = [69, 0]'),  
 Text(70.15286869464767, 88.33500000000001, 'X[24] <= 4.5\ngini = 0.198\nsamples = 9\nvalue = [8, 1]'),  
 Text(68.43396226415095, 74.745, 'gini = 0.0\nsamples = 8\nvalue = [8, 0]'),  
 Text(71.8717751251444, 74.745, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(78.7474008471313, 101.92500000000001, 'X[4] <= 2.5\ngini = 0.245\nsamples = 14\nvalue = [12, 2]'),  
 Text(77.02849441663457, 88.33500000000001, 'X[9] <= 3.0\ngini = 0.444\nsamples = 3\nvalue = [1, 2]'),  
 Text(75.30958798613786, 74.745, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(78.7474008471313, 74.745, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(80.46630727762803, 88.33500000000001, 'gini = 0.0\nsamples = 11\nvalue = [11, 0]'),  
 Text(95.07701193685021, 129.10500000000002, 'X[13] <= 1.5\ngini = 0.3\nsamples = 49\nvalue = [40, 9]'),  
 Text(89.06083943011167, 115.51500000000001, 'X[12] <= 15294.0\ngini = 0.49\nsamples = 14\nvalue = [8, 6]'),  
 Text(85.62302656911821, 101.92500000000001, 'X[15] <= 18.5\ngini = 0.408\nsamples = 7\nvalue = [2, 5]'),  
 Text(83.90412013862148, 88.33500000000001, 'gini = 0.0\nsamples = 5\nvalue = [0, 5]'),  
 Text(87.34193299961494, 88.33500000000001, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),  
 Text(92.49865229110512, 101.92500000000001, 'X[8] <= 1.5\ngini = 0.245\nsamples = 7\nvalue = [6, 1]'),  
 Text(90.77974586060839, 88.33500000000001, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(94.21755872160185, 88.33500000000001, 'gini = 0.0\nsamples = 6\nvalue = [6, 0]'),  
 Text(101.09318444358875, 115.51500000000001, 'X[12] <= 24803.0\ngini = 0.157\nsamples = 35\nvalue = [32, 3]'),  
 Text(99.37427801309202, 101.92500000000001, 'X[39] <= 0.5\ngini = 0.111\nsamples = 34\nvalue = [32, 2]'),  
 Text(97.65537158259531, 88.33500000000001, 'gini = 0.0\nsamples = 27\nvalue = [27, 0]'),  
 Text(101.09318444358875, 88.33500000000001, 'X[7] <= 61.0\ngini = 0.408\nsamples = 7\nvalue = [5, 2]'),  
 Text(99.37427801309202, 74.745, 'X[11] <= 4605.5\ngini = 0.444\nsamples = 3\nvalue = [1, 2]'),  
 Text(97.65537158259531, 61.155, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(101.09318444358875, 61.155, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(102.81209087408548, 74.745, 'gini = 0.0\nsamples = 4\nvalue = [4, 0]'),  
 Text(102.81209087408548, 101.92500000000001, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(85.19329996149403, 142.69500000000002, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(107.96881016557566, 169.875, 'X[1] <= 1412.0\ngini = 0.285\nsamples = 29\nvalue = [24, 5]'),  
 Text(106.24990373507893, 156.28500000000003, 'X[11] <= 2091.0\ngini = 0.198\nsamples = 27\nvalue = [24, 3]'),  
 Text(104.53099730458221, 142.69500000000002, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(107.96881016557566, 142.69500000000002, 'X[1] <= 383.5\ngini = 0.142\nsamples = 26\nvalue = [24, 2]'),  
 Text(106.24990373507893, 129.10500000000002, 'X[4] <= 3.0\ngini = 0.48\nsamples = 5\nvalue = [3, 2]'),  
 Text(104.53099730458221, 115.51500000000001, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(107.96881016557566, 115.51500000000001, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),  
 Text(109.68771659607239, 129.10500000000002, 'gini = 0.0\nsamples = 21\nvalue = [21, 0]'),  
 Text(109.68771659607239, 156.28500000000003, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(113.12552945706584, 183.46500000000003, 'X[20] <= 1.5\ngini = 0.48\nsamples = 5\nvalue = [3, 2]'),  
 Text(111.40662302656912, 169.875, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(114.84443588756257, 169.875, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),  
 Text(135.6861763573354, 197.055, 'X[0] <= 33.5\ngini = 0.224\nsamples = 210\nvalue = [183, 27]'),  
 Text(122.57951482479784, 183.46500000000003, 'X[7] <= 41.5\ngini = 0.416\nsamples = 61\nvalue = [43, 18]'),  
 Text(118.28224874855603, 169.875, 'X[4] <= 3.5\ngini = 0.42\nsamples = 10\nvalue = [3, 7]'),  
 Text(116.5633423180593, 156.28500000000003, 'X[13] <= 8.0\ngini = 0.219\nsamples = 8\nvalue = [1, 7]'),  
 Text(114.84443588756257, 142.69500000000002, 'gini = 0.0\nsamples = 7\nvalue = [0, 7]'),  
 Text(118.28224874855603, 142.69500000000002, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(120.00115517905276, 156.28500000000003, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),  
 Text(126.87678090103967, 169.875, 'X[19] <= 7.5\ngini = 0.338\nsamples = 51\nvalue = [40, 11]'),  
 Text(123.4389680400462, 156.28500000000003, 'X[15] <= 18.5\ngini = 0.491\nsamples = 23\nvalue = [13, 10]'),  
 Text(121.72006160954948, 142.69500000000002, 'X[8] <= 2.5\ngini = 0.499\nsamples = 19\nvalue = [9, 10]'),  
 Text(120.00115517905276, 129.10500000000002, 'gini = 0.0\nsamples = 5\nvalue = [0, 5]'),  
 Text(123.4389680400462, 129.10500000000002, 'X[11] <= 4004.5\ngini = 0.459\nsamples = 14\nvalue = [9, 5]'),  
 Text(121.72006160954948, 115.51500000000001, 'gini = 0.0\nsamples = 7\nvalue = [7, 0]'),  
 Text(125.15787447054294, 115.51500000000001, 'X[23] <= 1.5\ngini = 0.408\nsamples = 7\nvalue = [2, 5]'),  
 Text(123.4389680400462, 101.92500000000001, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),  
 Text(126.87678090103967, 101.92500000000001, 'gini = 0.0\nsamples = 5\nvalue = [0, 5]'),  
 Text(125.15787447054294, 142.69500000000002, 'gini = 0.0\nsamples = 4\nvalue = [4, 0]'),  
 Text(130.31459376203313, 156.28500000000003, 'X[1] <= 401.5\ngini = 0.069\nsamples = 28\nvalue = [27, 1]'),  
 Text(128.5956873315364, 142.69500000000002, 'X[3] <= 17.0\ngini = 0.444\nsamples = 3\nvalue = [2, 1]'),  
 Text(126.87678090103967, 129.10500000000002, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),  
 Text(130.31459376203313, 129.10500000000002, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(132.03350019252983, 142.69500000000002, 'gini = 0.0\nsamples = 25\nvalue = [25, 0]'),  
 Text(148.79283788987294, 183.46500000000003, 'X[5] <= 1.5\ngini = 0.114\nsamples = 149\nvalue = [140, 9]'),  
 Text(140.62803234501348, 169.875, 'X[31] <= 0.5\ngini = 0.269\nsamples = 25\nvalue = [21, 4]'),  
 Text(138.90912591451675, 156.28500000000003, 'X[20] <= 3.5\ngini = 0.219\nsamples = 24\nvalue = [21, 3]'),  
 Text(135.4713130535233, 142.69500000000002, 'X[19] <= 7.5\ngini = 0.1\nsamples = 19\nvalue = [18, 1]'),  
 Text(133.75240662302656, 129.10500000000002, 'X[9] <= 1.5\ngini = 0.5\nsamples = 2\nvalue = [1, 1]'),  
 Text(132.03350019252983, 115.51500000000001, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(135.4713130535233, 115.51500000000001, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(137.19021948402002, 129.10500000000002, 'gini = 0.0\nsamples = 17\nvalue = [17, 0]'),  
 Text(142.3469387755102, 142.69500000000002, 'X[0] <= 39.5\ngini = 0.48\nsamples = 5\nvalue = [3, 2]'),  
 Text(140.62803234501348, 129.10500000000002, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(144.06584520600694, 129.10500000000002, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),  
 Text(142.3469387755102, 156.28500000000003, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(156.95764343473238, 169.875, 'X[26] <= 0.5\ngini = 0.077\nsamples = 124\nvalue = [119, 5]'),  
 Text(155.23873700423565, 156.28500000000003, 'X[15] <= 20.5\ngini = 0.187\nsamples = 48\nvalue = [43, 5]'),  
 Text(150.94147092799383, 142.69500000000002, 'X[13] <= 8.5\ngini = 0.13\nsamples = 43\nvalue = [40, 3]'),  
 Text(147.50365806700037, 129.10500000000002, 'X[7] <= 93.5\ngini = 0.051\nsamples = 38\nvalue = [37, 1]'),  
 Text(145.78475163650364, 115.51500000000001, 'gini = 0.0\nsamples = 34\nvalue = [34, 0]'),  
 Text(149.2225644974971, 115.51500000000001, 'X[7] <= 95.0\ngini = 0.375\nsamples = 4\nvalue = [3, 1]'),  
 Text(147.50365806700037, 101.92500000000001, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(150.94147092799383, 101.92500000000001, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),  
 Text(154.3792837889873, 129.10500000000002, 'X[30] <= 0.5\ngini = 0.48\nsamples = 5\nvalue = [3, 2]'),  
 Text(152.66037735849056, 115.51500000000001, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),  
 Text(156.09819021948402, 115.51500000000001, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(159.53600308047749, 142.69500000000002, 'X[24] <= 0.5\ngini = 0.48\nsamples = 5\nvalue = [3, 2]'),  
 Text(157.81709664998075, 129.10500000000002, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(161.2549095109742, 129.10500000000002, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),  
 Text(158.6765498652291, 156.28500000000003, 'gini = 0.0\nsamples = 76\nvalue = [76, 0]'),  
 Text(120.74646226415095, 210.645, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(200.10488063149788, 251.41500000000002, 'X[11] <= 2475.0\ngini = 0.424\nsamples = 416\nvalue = [289, 127]'),  
 Text(169.84944166345784, 237.82500000000002, 'X[1] <= 931.0\ngini = 0.423\nsamples = 69\nvalue = [21, 48]'),  
 Text(163.8332691567193, 224.235, 'X[30] <= 0.5\ngini = 0.214\nsamples = 41\nvalue = [5, 36]'),  
 Text(162.11436272622257, 210.645, 'gini = 0.0\nsamples = 26\nvalue = [0, 26]'),  
 Text(165.55217558721603, 210.645, 'X[17] <= 3.5\ngini = 0.444\nsamples = 15\nvalue = [5, 10]'),  
 Text(163.8332691567193, 197.055, 'X[4] <= 3.5\ngini = 0.278\nsamples = 12\nvalue = [2, 10]'),  
 Text(162.11436272622257, 183.46500000000003, 'X[0] <= 35.0\ngini = 0.165\nsamples = 11\nvalue = [1, 10]'),  
 Text(160.39545629572584, 169.875, 'gini = 0.0\nsamples = 9\nvalue = [0, 9]'),  
 Text(163.8332691567193, 169.875, 'X[23] <= 2.5\ngini = 0.5\nsamples = 2\nvalue = [1, 1]'),  
 Text(162.11436272622257, 156.28500000000003, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(165.55217558721603, 156.28500000000003, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(165.55217558721603, 183.46500000000003, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(167.27108201771276, 197.055, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),  
 Text(175.86561417019638, 224.235, 'X[23] <= 2.5\ngini = 0.49\nsamples = 28\nvalue = [16, 12]'),  
 Text(174.14670773969965, 210.645, 'X[10] <= 3.5\ngini = 0.444\nsamples = 18\nvalue = [6, 12]'),  
 Text(172.42780130920292, 197.055, 'X[8] <= 3.5\ngini = 0.375\nsamples = 16\nvalue = [4, 12]'),  
 Text(168.98998844820946, 183.46500000000003, 'X[12] <= 17204.5\ngini = 0.26\nsamples = 13\nvalue = [2, 11]'),  
 Text(167.27108201771276, 169.875, 'gini = 0.0\nsamples = 10\nvalue = [0, 10]'),  
 Text(170.7088948787062, 169.875, 'X[1] <= 1347.5\ngini = 0.444\nsamples = 3\nvalue = [2, 1]'),  
 Text(168.98998844820946, 156.28500000000003, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),  
 Text(172.42780130920292, 156.28500000000003, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(175.86561417019638, 183.46500000000003, 'X[3] <= 5.5\ngini = 0.444\nsamples = 3\nvalue = [2, 1]'),  
 Text(174.14670773969965, 169.875, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),  
 Text(177.5845206006931, 169.875, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(175.86561417019638, 197.055, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),  
 Text(177.5845206006931, 210.645, 'gini = 0.0\nsamples = 10\nvalue = [10, 0]'),  
 Text(230.36031959953792, 237.82500000000002, 'X[42] <= 0.5\ngini = 0.352\nsamples = 347\nvalue = [268, 79]'),  
 Text(204.2812860993454, 224.235, 'X[5] <= 2.5\ngini = 0.261\nsamples = 246\nvalue = [208, 38]'),  
 Text(192.62495186753947, 210.645, 'X[13] <= 4.5\ngini = 0.399\nsamples = 80\nvalue = [58, 22]'),  
 Text(188.75741239892184, 197.055, 'X[30] <= 0.5\ngini = 0.305\nsamples = 64\nvalue = [52, 12]'),  
 Text(187.0385059684251, 183.46500000000003, 'X[0] <= 31.5\ngini = 0.402\nsamples = 43\nvalue = [31, 12]'),  
 Text(181.02233346168657, 169.875, 'X[7] <= 72.0\ngini = 0.497\nsamples = 13\nvalue = [6, 7]'),  
 Text(179.30342703118984, 156.28500000000003, 'gini = 0.0\nsamples = 6\nvalue = [0, 6]'),  
 Text(182.74123989218327, 156.28500000000003, 'X[0] <= 30.5\ngini = 0.245\nsamples = 7\nvalue = [6, 1]'),  
 Text(181.02233346168657, 142.69500000000002, 'gini = 0.0\nsamples = 6\nvalue = [6, 0]'),  
 Text(184.46014632268, 142.69500000000002, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(193.05467847516366, 169.875, 'X[11] <= 2798.0\ngini = 0.278\nsamples = 30\nvalue = [25, 5]'),  
 Text(189.6168656141702, 156.28500000000003, 'X[1] <= 765.0\ngini = 0.444\nsamples = 3\nvalue = [1, 2]'),  
 Text(187.89795918367346, 142.69500000000002, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(191.33577204466692, 142.69500000000002, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(196.49249133615712, 156.28500000000003, 'X[12] <= 23214.5\ngini = 0.198\nsamples = 27\nvalue = [24, 3]'),  
 Text(194.77358490566039, 142.69500000000002, 'X[2] <= 0.5\ngini = 0.142\nsamples = 26\nvalue = [24, 2]'),  
 Text(193.05467847516366, 129.10500000000002, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(196.49249133615712, 129.10500000000002, 'X[36] <= 0.5\ngini = 0.077\nsamples = 25\nvalue = [24, 1]'),  
 Text(194.77358490566039, 115.51500000000001, 'gini = 0.0\nsamples = 22\nvalue = [22, 0]'),  
 Text(198.21139776665382, 115.51500000000001, 'X[19] <= 9.0\ngini = 0.444\nsamples = 3\nvalue = [2, 1]'),  
 Text(196.49249133615712, 101.92500000000001, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(199.93030419715055, 101.92500000000001, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),  
 Text(198.21139776665382, 142.69500000000002, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(190.47631882941855, 183.46500000000003, 'gini = 0.0\nsamples = 21\nvalue = [21, 0]'),  
 Text(196.49249133615712, 197.055, 'X[0] <= 35.5\ngini = 0.469\nsamples = 16\nvalue = [6, 10]'),  
 Text(194.77358490566039, 183.46500000000003, 'gini = 0.0\nsamples = 7\nvalue = [0, 7]'),  
 Text(198.21139776665382, 183.46500000000003, 'X[3] <= 10.0\ngini = 0.444\nsamples = 9\nvalue = [6, 3]'),  
 Text(196.49249133615712, 169.875, 'gini = 0.0\nsamples = 6\nvalue = [6, 0]'),  
 Text(199.93030419715055, 169.875, 'gini = 0.0\nsamples = 3\nvalue = [0, 3]'),  
 Text(215.93762033115132, 210.645, 'X[19] <= 2.5\ngini = 0.174\nsamples = 166\nvalue = [150, 16]'),  
 Text(208.95456295725836, 197.055, 'X[15] <= 11.5\ngini = 0.5\nsamples = 6\nvalue = [3, 3]'),  
 Text(207.23565652676166, 183.46500000000003, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(210.6734693877551, 183.46500000000003, 'X[13] <= 0.5\ngini = 0.375\nsamples = 4\nvalue = [3, 1]'),  
 Text(208.95456295725836, 169.875, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(212.39237581825182, 169.875, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),  
 Text(222.92067770504428, 197.055, 'X[11] <= 19853.0\ngini = 0.149\nsamples = 160\nvalue = [147, 13]'),  
 Text(221.20177127454755, 183.46500000000003, 'X[20] <= 5.5\ngini = 0.14\nsamples = 159\nvalue = [147, 12]'),  
 Text(215.83018867924528, 169.875, 'X[20] <= 0.5\ngini = 0.121\nsamples = 155\nvalue = [145, 10]'),  
 Text(210.24374278013093, 156.28500000000003, 'X[7] <= 67.5\ngini = 0.444\nsamples = 9\nvalue = [6, 3]'),  
 Text(208.5248363496342, 142.69500000000002, 'gini = 0.0\nsamples = 3\nvalue = [0, 3]'),  
 Text(211.96264921062763, 142.69500000000002, 'gini = 0.0\nsamples = 6\nvalue = [6, 0]'),  
 Text(221.41663457835963, 156.28500000000003, 'X[40] <= 0.5\ngini = 0.091\nsamples = 146\nvalue = [139, 7]'),  
 Text(215.4004620716211, 142.69500000000002, 'X[7] <= 41.5\ngini = 0.08\nsamples = 144\nvalue = [138, 6]'),  
 Text(208.5248363496342, 129.10500000000002, 'X[34] <= 0.5\ngini = 0.245\nsamples = 28\nvalue = [24, 4]'),  
 Text(205.08702348864074, 115.51500000000001, 'X[3] <= 23.0\ngini = 0.147\nsamples = 25\nvalue = [23, 2]'),  
 Text(203.368117058144, 101.92500000000001, 'X[22] <= 1.5\ngini = 0.08\nsamples = 24\nvalue = [23, 1]'),  
 Text(201.64921062764728, 88.33500000000001, 'X[17] <= 2.5\ngini = 0.5\nsamples = 2\nvalue = [1, 1]'),  
 Text(199.93030419715055, 74.745, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(203.368117058144, 74.745, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(205.08702348864074, 88.33500000000001, 'gini = 0.0\nsamples = 22\nvalue = [22, 0]'),  
 Text(206.80592991913747, 101.92500000000001, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(211.96264921062763, 115.51500000000001, 'X[22] <= 3.0\ngini = 0.444\nsamples = 3\nvalue = [1, 2]'),  
 Text(210.24374278013093, 101.92500000000001, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(213.68155564112436, 101.92500000000001, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(222.276087793608, 129.10500000000002, 'X[7] <= 97.5\ngini = 0.034\nsamples = 116\nvalue = [114, 2]'),  
 Text(218.83827493261455, 115.51500000000001, 'X[11] <= 2770.5\ngini = 0.018\nsamples = 111\nvalue = [110, 1]'),  
 Text(217.11936850211782, 101.92500000000001, 'X[38] <= 0.5\ngini = 0.245\nsamples = 7\nvalue = [6, 1]'),  
 Text(215.4004620716211, 88.33500000000001, 'gini = 0.0\nsamples = 6\nvalue = [6, 0]'),  
 Text(218.83827493261455, 88.33500000000001, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(220.55718136311128, 101.92500000000001, 'gini = 0.0\nsamples = 104\nvalue = [104, 0]'),  
 Text(225.71390065460147, 115.51500000000001, 'X[4] <= 3.5\ngini = 0.32\nsamples = 5\nvalue = [4, 1]'),  
 Text(223.99499422410474, 101.92500000000001, 'gini = 0.0\nsamples = 4\nvalue = [4, 0]'),  
 Text(227.43280708509818, 101.92500000000001, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(227.43280708509818, 142.69500000000002, 'X[1] <= 456.0\ngini = 0.5\nsamples = 2\nvalue = [1, 1]'),  
 Text(225.71390065460147, 129.10500000000002, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(229.1517135155949, 129.10500000000002, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(226.57335386984983, 169.875, 'X[12] <= 9932.5\ngini = 0.5\nsamples = 4\nvalue = [2, 2]'),  
 Text(224.8544474393531, 156.28500000000003, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(228.29226030034656, 156.28500000000003, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),  
 Text(224.639584135541, 183.46500000000003, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(256.43935309973045, 224.235, 'X[27] <= 0.5\ngini = 0.482\nsamples = 101\nvalue = [60, 41]'),  
 Text(242.04351174432037, 210.645, 'X[11] <= 4035.0\ngini = 0.402\nsamples = 61\nvalue = [44, 17]'),  
 Text(234.3084328070851, 197.055, 'X[7] <= 47.5\ngini = 0.472\nsamples = 21\nvalue = [8, 13]'),  
 Text(232.58952637658837, 183.46500000000003, 'gini = 0.0\nsamples = 4\nvalue = [4, 0]'),  
 Text(236.02733923758183, 183.46500000000003, 'X[12] <= 20762.0\ngini = 0.36\nsamples = 17\nvalue = [4, 13]'),  
 Text(234.3084328070851, 169.875, 'X[20] <= 1.5\ngini = 0.231\nsamples = 15\nvalue = [2, 13]'),  
 Text(232.58952637658837, 156.28500000000003, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(236.02733923758183, 156.28500000000003, 'X[7] <= 53.5\ngini = 0.133\nsamples = 14\nvalue = [1, 13]'),  
 Text(234.3084328070851, 142.69500000000002, 'X[10] <= 3.5\ngini = 0.5\nsamples = 2\nvalue = [1, 1]'),  
 Text(232.58952637658837, 129.10500000000002, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(236.02733923758183, 129.10500000000002, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(237.74624566807856, 142.69500000000002, 'gini = 0.0\nsamples = 12\nvalue = [0, 12]'),  
 Text(237.74624566807856, 169.875, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),  
 Text(249.77859068155564, 197.055, 'X[23] <= 14.5\ngini = 0.18\nsamples = 40\nvalue = [36, 4]'),  
 Text(248.0596842510589, 183.46500000000003, 'X[34] <= 0.5\ngini = 0.142\nsamples = 39\nvalue = [36, 3]'),  
 Text(244.62187139006545, 169.875, 'X[4] <= 1.5\ngini = 0.059\nsamples = 33\nvalue = [32, 1]'),  
 Text(242.90296495956872, 156.28500000000003, 'X[17] <= 1.5\ngini = 0.5\nsamples = 2\nvalue = [1, 1]'),  
 Text(241.184058529072, 142.69500000000002, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(244.62187139006545, 142.69500000000002, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(246.34077782056218, 156.28500000000003, 'gini = 0.0\nsamples = 31\nvalue = [31, 0]'),  
 Text(251.49749711205237, 169.875, 'X[5] <= 1.5\ngini = 0.444\nsamples = 6\nvalue = [4, 2]'),  
 Text(249.77859068155564, 156.28500000000003, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(253.2164035425491, 156.28500000000003, 'gini = 0.0\nsamples = 4\nvalue = [4, 0]'),  
 Text(251.49749711205237, 183.46500000000003, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(270.83519445514054, 210.645, 'X[24] <= 1.5\ngini = 0.48\nsamples = 40\nvalue = [16, 24]'),  
 Text(266.10820177127454, 197.055, 'X[10] <= 3.5\ngini = 0.497\nsamples = 28\nvalue = [15, 13]'),  
 Text(261.81093569503275, 183.46500000000003, 'X[12] <= 7594.5\ngini = 0.43\nsamples = 16\nvalue = [5, 11]'),  
 Text(258.3731228340393, 169.875, 'X[23] <= 7.0\ngini = 0.375\nsamples = 4\nvalue = [3, 1]'),  
 Text(256.65421640354253, 156.28500000000003, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]'),  
 Text(260.092029264536, 156.28500000000003, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(265.24874855602616, 169.875, 'X[19] <= 13.0\ngini = 0.278\nsamples = 12\nvalue = [2, 10]'),  
 Text(263.52984212552946, 156.28500000000003, 'gini = 0.0\nsamples = 9\nvalue = [0, 9]'),  
 Text(266.9676549865229, 156.28500000000003, 'X[2] <= 0.5\ngini = 0.444\nsamples = 3\nvalue = [2, 1]'),  
 Text(265.24874855602616, 142.69500000000002, 'gini = 0.0\nsamples = 1\nvalue = [0, 1]'),  
 Text(268.6865614170196, 142.69500000000002, 'gini = 0.0\nsamples = 2\nvalue = [2, 0]'),  
 Text(270.4054678475164, 183.46500000000003, 'X[13] <= 6.0\ngini = 0.278\nsamples = 12\nvalue = [10, 2]'),  
 Text(268.6865614170196, 169.875, 'gini = 0.0\nsamples = 9\nvalue = [9, 0]'),  
 Text(272.1243742780131, 169.875, 'X[22] <= 3.0\ngini = 0.444\nsamples = 3\nvalue = [1, 2]'),  
 Text(270.4054678475164, 156.28500000000003, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]'),  
 Text(273.84328070850984, 156.28500000000003, 'gini = 0.0\nsamples = 2\nvalue = [0, 2]'),  
 Text(275.56218713900654, 197.055, 'X[22] <= 27.0\ngini = 0.153\nsamples = 12\nvalue = [1, 11]'),  
 Text(273.84328070850984, 183.46500000000003, 'gini = 0.0\nsamples = 11\nvalue = [0, 11]'),  
 Text(277.2810935695033, 183.46500000000003, 'gini = 0.0\nsamples = 1\nvalue = [1, 0]')]

# using neural\_network[¶](#using-neural_network)

In [49]:

from sklearn.neural\_network import MLPClassifier

In [59]:

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
#x\_train = StandardScaler(x\_train)  
#x\_test = StandardScaler(x\_test)  
mlp = MLPClassifier(solver='lbfgs', alpha=0.05, hidden\_layer\_sizes=(5,2),random\_state=1)  
mlp.fit(x\_train,y\_train.values.ravel())  
mlp.score(x\_test,y\_test)

Out[59]:

0.8231292517006803

In [60]:

#increase layer size to 10  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
#x\_train = StandardScaler(x\_train)  
#x\_test = StandardScaler(x\_test)  
mlp = MLPClassifier(solver='lbfgs', alpha=0.05, hidden\_layer\_sizes=(10,2),random\_state=1)  
mlp.fit(x\_train,y\_train.values.ravel())  
mlp.score(x\_test,y\_test)

Out[60]:

0.19501133786848074

In [78]:

#increase layer size to 20  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.3, random\_state = 23)  
#x\_train = StandardScaler(x\_train)  
#x\_test = StandardScaler(x\_test)  
mlp = MLPClassifier(solver='adam', alpha=0.05, hidden\_layer\_sizes=(5,2),random\_state=1,max\_iter=400)  
mlp.fit(x\_train,y\_train.values.ravel())  
mlp.score(x\_test,y\_test)

Out[78]:

0.8231292517006803