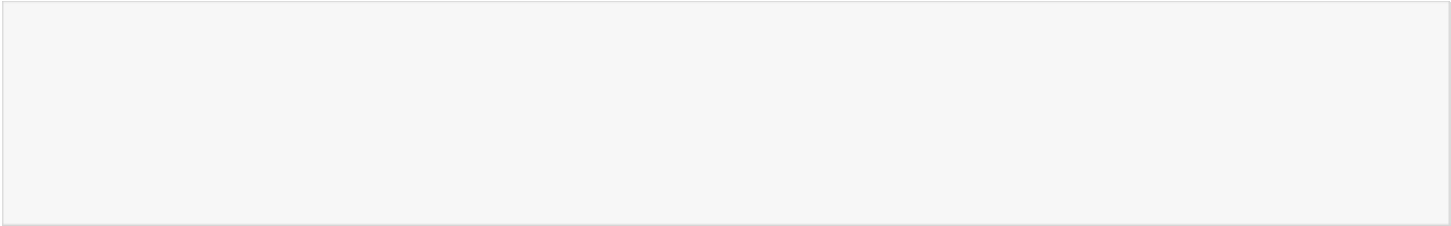
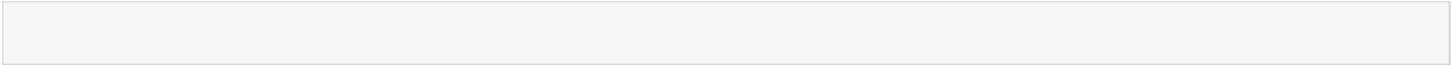
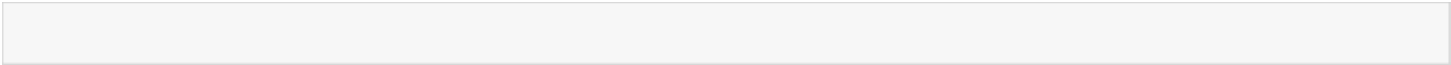
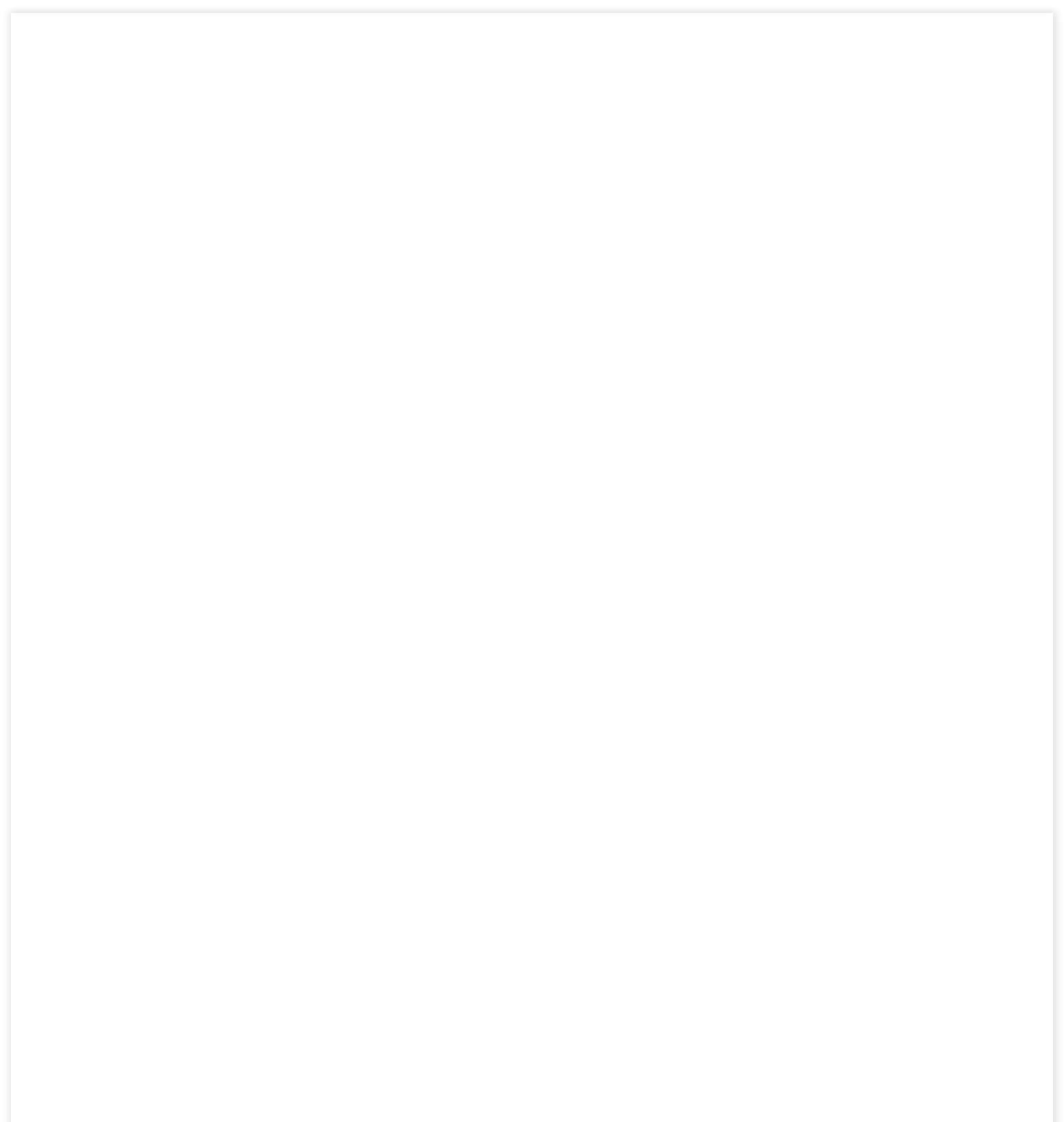
In [5]:



**import pandas as pd import numpy as np**

In [6]:

data = pd.read\_csv("attrition\_data.csv")

|  |  |
| --- | --- |
| In [7]: |  |
| data.isna().sum() |
| Out[7]: |
| Age | 0 |
| Attrition | 0 |
| BusinessTravel | 0 |
| Department | 0 |
| DistanceFromHome | 0 |
| Education | 0 |
| EducationField | 0 |
| EmployeeCount | 0 |
| EmployeeID | 0 |
| Gender | 0 |
| JobLevel | 0 |
| JobRole | 0 |
| MaritalStatus | 0 |
| MonthlyIncome | 0 |
| NumCompaniesWorked | 19 |
| Over18 | 0 |
| PercentSalaryHike | 0 |
| StandardHours | 0 |
| StockOptionLevel | 0 |
| TotalWorkingYears | 9 |
| TrainingTimesLastYear | 0 |
| YearsAtCompany | 0 |
| YearsSinceLastPromotion | 0 |
| YearsWithCurrManager | 0 |
| dtype: int64 |  |

In [8]:

print(data["NumCompaniesWorked"].mean()) print(data["TotalWorkingYears"].mean())

2.6948303347756775

11.279936378095888

In [9]:

new\_NumCampaniesWorked=np.where(data["NumCompaniesWorked"].isnull(),3,data["NumCompaniesW orked"])

data["NumCompaniesWorked"]=new\_NumCampaniesWorked

new\_TotalWorkingYears=np.where(data["TotalWorkingYears"].isnull(),11,data["TotalWorkingYe ars"])

data["TotalWorkingYears"]=new\_TotalWorkingYears

data.isna().sum() Out[9]:

Age 0

Attrition 0

BusinessTravel 0

Department 0

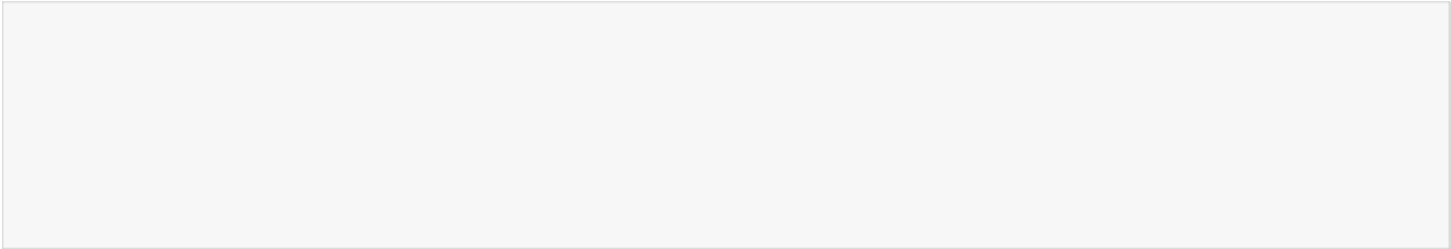
DistanceFromHome 0

Education 0

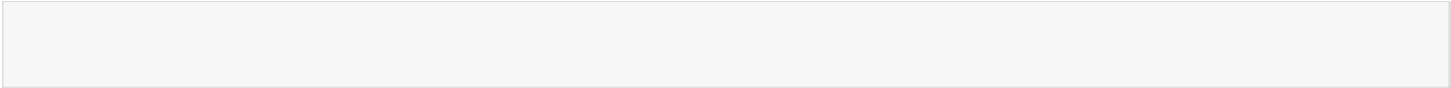
|  |  |
| --- | --- |
| EducationField | 0 |
| EmployeeCount | 0 |
| EmployeeID | 0 |
| Gender | 0 |
| JobLevel | 0 |
| JobRole | 0 |
| MaritalStatus | 0 |
| MonthlyIncome | 0 |
| NumCompaniesWorked | 0 |
| Over18 | 0 |
| PercentSalaryHike | 0 |
| StandardHours | 0 |
| StockOptionLevel | 0 |
| TotalWorkingYears | 0 |
| TrainingTimesLastYear | 0 |
| YearsAtCompany | 0 |
| YearsSinceLastPromotion | 0 |
| YearsWithCurrManager dtype: int64 | 0 |
| In [10]: |  |
| df = pd.DataFrame(data) df.NumCompaniesWorked = | df.NumCompaniesWorked.astype(int) |



In [11]:



**from sklearn.preprocessing import** LabelEncoder lb=LabelEncoder() data["Attrition"]=lb.fit\_transform(data["Attrition"]) data['BusinessTravel']=lb.fit\_transform(data['BusinessTravel']) data['Department']=lb.fit\_transform(data["Department"]) data['EducationField']=lb.fit\_transform(data['EducationField']) data['Gender']=lb.fit\_transform(data['Gender']) data['MaritalStatus']=lb.fit\_transform(data['MaritalStatus']) data['JobRole']=lb.fit\_transform(data['JobRole']) data['Over18']=lb.fit\_transform(data['Over18'])



df.TotalWorkingYears = df.TotalWorkingYears.astype(int)

In [12]:



**from sklearn.ensemble import** RandomForestClassifier

In [13]:



rf\_model=RandomForestClassifier(n\_estimators=1000,max\_features=2,oob\_score=**True**)

In [14]:

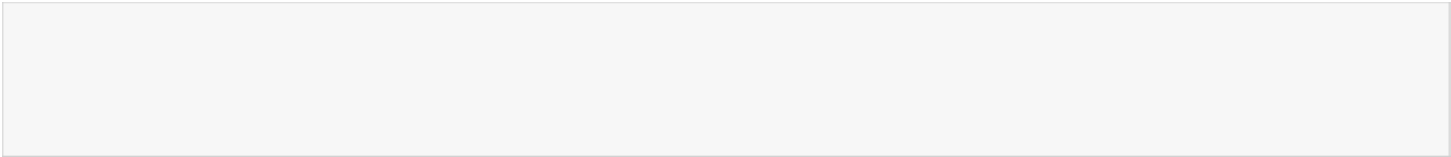


data.columns

Out[14]:

Index(['Age', 'Attrition', 'BusinessTravel', 'Department', 'DistanceFromHome', 'Education', 'EducationField', 'EmployeeCount', 'EmployeeID', 'Gender', 'JobLevel', 'JobRole', 'MaritalStatus', 'MonthlyIncome', 'NumCompaniesWorked', 'Over18', 'PercentSalaryHike', 'StandardHours', 'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear', 'YearsAtCompany', 'YearsSinceLastPromotion', 'YearsWithCurrManager'],

dtype='object') In [15]:



features=['Age', 'BusinessTravel', 'Department', 'DistanceFromHome', 'Education', 'EducationField', 'EmployeeCount', 'Gender', 'JobLevel', 'JobRole', 'MaritalStatus', 'MonthlyIncome', 'NumCompaniesWorked', 'Over18', 'PercentSalaryHike', 'StandardHours', 'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear', 'YearsAtCompany', 'YearsSinceLastPromotion', 'YearsWithCurrManager']

In [16]:





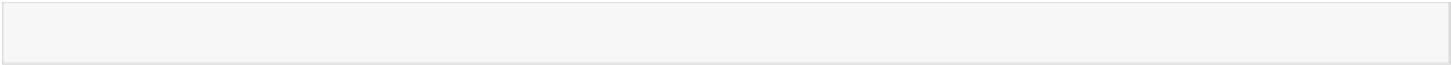
rf\_model.fit(X=data[features],y=data["Attrition"])

Out[16]:

RandomForestClassifier(bootstrap=True, ccp\_alpha=0.0, class\_weight=None,

criterion='gini', max\_depth=None, max\_features=2, max\_leaf\_nodes=None, max\_samples=None, min\_impurity\_decrease=0.0, min\_impurity\_split=None, min\_samples\_leaf=1, min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0, n\_estimators=1000, n\_jobs=None, oob\_score=True, random\_state=None, verbose=0, warm\_start=False)

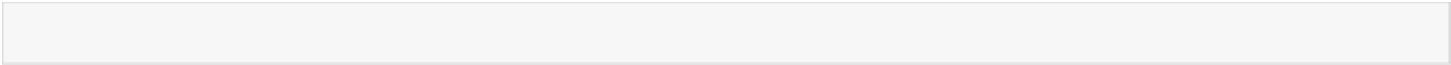
In [17]:



print('OOB Accuracy') print(rf\_model.oob\_score\_);

OOB Accuracy 1.0

In [18]:



**for** feature,imp **in** zip(features,rf\_model.feature\_importances\_): print(feature,imp)

Age 0.09602606705644955

BusinessTravel 0.028267820733520573

Department 0.026665488571615627

DistanceFromHome 0.07033905724278072

Education 0.04069760348498781

EducationField 0.04148643595174499

EmployeeCount 0.0

Gender 0.01750223591997137

JobLevel 0.03782353719191878

JobRole 0.055879838785980185

MaritalStatus 0.03972305068704668

MonthlyIncome 0.0928028078667828

NumCompaniesWorked 0.05515163453342869

Over18 0.0

PercentSalaryHike 0.06573684832795022

StandardHours 0.0

StockOptionLevel 0.034121186895815815

TotalWorkingYears 0.08438728154337624

TrainingTimesLastYear 0.04518781453247599

YearsAtCompany 0.06987386451068431

YearsSinceLastPromotion 0.04368278582917235

YearsWithCurrManager 0.054644640334297286

Age , Monthly income , TotalWorkingYears are most significant

In [19]:



**from sklearn import** tree,preprocessing

In [20]:



predictors=pd.DataFrame([data['MonthlyIncome'], data['Age'],data['TotalWorkingYears']]).T

In [21]:



tree\_model=tree.DecisionTreeClassifier(max\_depth=6)

In [22]:

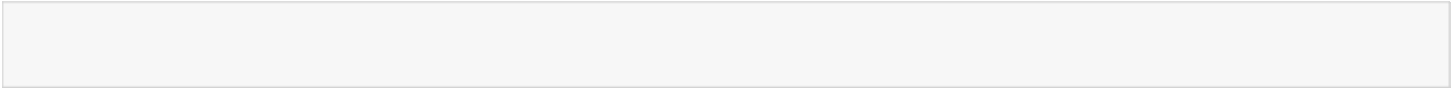
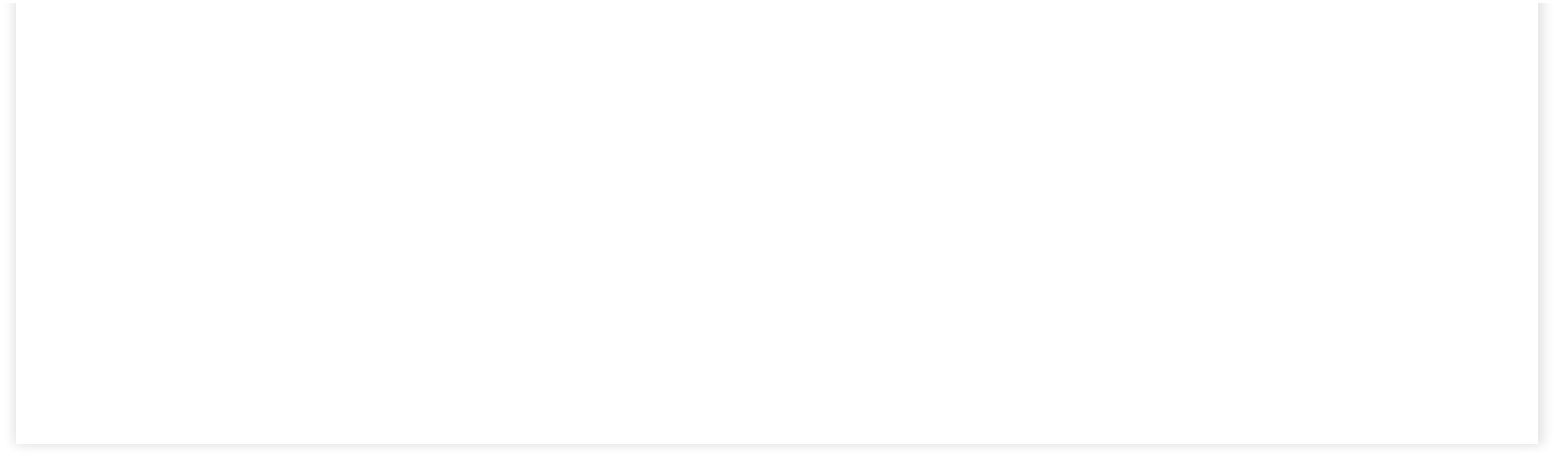


tree\_model.fit(X=predictors, y=data['Attrition'])

Out[22]:

DecisionTreeClassifier(ccp\_alpha=0.0, class\_weight=None, criterion='gini',

max\_depth=6, max\_features=None, max\_leaf\_nodes=None,



min\_impurity\_decrease=0.0, min\_impurity\_split=None, min\_samples\_leaf=1, min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0, presort='deprecated', random\_state=None, splitter='best')

In [23]:

**with** open('attrition\_tree.dot','w')**as** f: f=tree.export\_graphviz(tree\_model,feature\_names=['MonthlyIncome','Age','TotalWorkingY

ears'],out\_file=f);

In [24]:

tree\_model.score(X=predictors,y=data['Attrition']) Out[24]:

0.8650793650793651

**RULES:**

**• If TotalWorkingYears <=1.5 and age<=33.5 and MonthlyIncome<=19425 then survived and if MonthlyIncome>19425.0 then no attrition.**

**• If TotalWorkingYears <=1.5 and age<=33.5 and MonthlyIncome>20325 then attrition**

**• If TotalWorkingYears <=1.5 and age<=23.5 and MonthlyIncome<=32530 then attrition**

**• If TotalWorkingYears <=1.5 and age<=18.5 then attrition**

**• If TotalWorkingYears <=1.5 and MonthlyIncome<=112610 and age>20 then attrition**

**• If TotalWorkingYears <=1.5 and MonthlyIncome<=155870 then attrition.**

**• If TotalWorkingYears <=1.5 and age<=20 MonthlyIncome<=173350 then no attrition.**

**• If TotalWorkingYears <=1.5 and age<=20 and MonthlyIncome>1773350 then attrition is there.**

**• If TotalWorkingYears >=1.5 and age<=33.5 and TotalWorkingYears<=39 then no attrition**

**• If TotalWorkingYears >=1.5 and age<=33.5 and TotalWorkingYears<=39 and MonthlyIncome<=10300 then no attrition**

**• If TotalWorkingYears >=1.5 and age<=33.5 and TotalWorkingYears>5.5 then attrition**

**• If TotalWorkingYears >=1.5 and age<=33.5 and TotalWorkingYears<=5.5 and MonthlyIncome<=137390 then attrition**

**• If age<=33.5 and TotalWorkingYears<=39 and MonthlyIncome<=10300 then no attrition**

