HR Analytics with PGM

Fateme Rahimi

July 22, 2020

1 HR Analytics

Attrition is a problem that impacts all businesses, industry and size of the company. Employee attrition leads to significant costs for a business, including the cost of business disruption, hiring new staff and training new staff. As such, there is great interest in understanding staff attrition.

To this end, the use of probabilistic graphic models to find probability of attrition can help the HR unit's ability to hire motivated and sustainable employees, In addition, it can help to reduce the factors that reduce the attractiveness of work, to create a better work environment for their employees. Here we use dataset that presents an employee survey from IBM, indicating if there is attrition or not.

1.1 Importing libraries and loading data

First, we downloaded the data from the Kaggle site:

https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset

and read it using the Pandas library.

[2]:
```
import pandas as pd
df=pd.read_csv("datasets-HR-Employee-Attrition.csv")
```

This dataset contain 1470 record and 35 feature.

[3]:
```
print(df.shape)
df.head(5)
```

```
(1470, 35)
```

[3]:
```
   Age Attrition     BusinessTravel  DailyRate              Department  \
0   41       Yes      Travel_Rarely       1102                   Sales
1   49        No  Travel_Frequently        279  Research & Development
2   37       Yes      Travel_Rarely       1373  Research & Development
3   33        No  Travel_Frequently       1392  Research & Development
4   27        No      Travel_Rarely        591  Research & Development

   DistanceFromHome  Education EducationField  EmployeeCount  EmployeeNumber  \
0                 1          2  Life Sciences              1               1
1                 8          1  Life Sciences              1               2
```

```
2                 2          2          Other             1               4
3                 3          4  Life Sciences             1               5
4                 2          1        Medical             1               7

   ...  RelationshipSatisfaction StandardHours  StockOptionLevel  \
0  ...                         1            80                 0
1  ...                         4            80                 1
2  ...                         2            80                 0
3  ...                         3            80                 0
4  ...                         4            80                 1

   TotalWorkingYears  TrainingTimesLastYear WorkLifeBalance  YearsAtCompany  \
0                  8                      0               1               6
1                 10                      3               3              10
2                  7                      3               3               0
3                  8                      3               3               8
4                  6                      3               3               2

  YearsInCurrentRole  YearsSinceLastPromotion  YearsWithCurrManager
0                  4                        0                     5
1                  7                        1                     7
2                  0                        0                     0
3                  7                        3                     0
4                  2                        2                     2

[5 rows x 35 columns]
```

1.2 Data cleaning

we don't have any miss value in this dataset:

[15]:
```
df.isnull().sum()
```

[15]:
```
Age                         0
Attrition                   0
BusinessTravel              0
DailyRate                   0
Department                  0
DistanceFromHome            0
Education                   0
EducationField              0
EmployeeCount               0
EmployeeNumber              0
EnvironmentSatisfaction     0
Gender                      0
HourlyRate                  0
JobInvolvement              0
```

```
JobLevel                    0
JobRole                     0
JobSatisfaction             0
MaritalStatus               0
MonthlyIncome               0
MonthlyRate                 0
NumCompaniesWorked          0
Over18                      0
OverTime                    0
PercentSalaryHike           0
PerformanceRating           0
RelationshipSatisfaction    0
StandardHours               0
StockOptionLevel            0
TotalWorkingYears           0
TrainingTimesLastYear       0
WorkLifeBalance             0
YearsAtCompany              0
YearsInCurrentRole          0
YearsSinceLastPromotion     0
YearsWithCurrManager        0
dtype: int64
```

1.3 Encode Catogorical Attributes

We have 9 categorical attribute,we need to decode them.

[5]:
```
m=df.select_dtypes(include=['object']).columns
print("catogorical attribute:",m)
for i in m:
    df[i] = df[i].astype('category')
    df[i] = df[i].cat.codes
```

```
catogorical attribute: Index(['Attrition', 'BusinessTravel', 'Department',
'EducationField', 'Gender',
       'JobRole', 'MaritalStatus', 'Over18', 'OverTime'],
      dtype='object')
```

2 Finding Four Correlated Features

Our data has 35 attributes that are very time consuming to perform calculations using all of these attributes. Given the existing operating system, we choose four features that they have most correlation with attrition. In the following, we will implement the algorithms with these 5 attribute.

[7]:
```
corrmat = df.corr()
corrmat.sort_values("Attrition",inplace=True)
corrmat.head(2)
```

[7]:
```
                        Age  Attrition  BusinessTravel  DailyRate  Department  \
TotalWorkingYears  0.680381  -0.171063        0.034226   0.014515   -0.015762
JobLevel           0.509604  -0.169105        0.019311   0.002966    0.101963

                   DistanceFromHome  Education  EducationField  EmployeeCount  \
TotalWorkingYears          0.004628   0.148280       -0.027848            NaN
JobLevel                   0.005303   0.101589       -0.044933            NaN

                   EmployeeNumber  ...  RelationshipSatisfaction  \
TotalWorkingYears       -0.014365  ...                  0.024054
JobLevel                -0.018519  ...                  0.021642

                   StandardHours  StockOptionLevel  TotalWorkingYears  \
TotalWorkingYears            NaN          0.010136           1.000000
JobLevel                     NaN          0.013984           0.782208

                   TrainingTimesLastYear  WorkLifeBalance  YearsAtCompany  \
TotalWorkingYears              -0.035662         0.001008        0.628133
JobLevel                       -0.018191         0.037818        0.534739

                   YearsInCurrentRole  YearsSinceLastPromotion  \
TotalWorkingYears            0.460365                 0.404858
JobLevel                     0.389447                 0.353885

                   YearsWithCurrManager
TotalWorkingYears              0.459188
JobLevel                       0.375281

[2 rows x 35 columns]
```

[9]:
```
corrmat.tail(5)
```

[9]:
```
                    Age  Attrition  BusinessTravel  DailyRate  Department  \
OverTime       0.028062   0.246118        0.016543   0.009135    0.007481
Attrition     -0.159205   1.000000        0.000074  -0.056652    0.063991
EmployeeCount       NaN        NaN             NaN        NaN         NaN
Over18              NaN        NaN             NaN        NaN         NaN
StandardHours       NaN        NaN             NaN        NaN         NaN

               DistanceFromHome  Education  EducationField  EmployeeCount  \
OverTime               0.025514  -0.020322        0.002259            NaN
Attrition              0.077924  -0.031373        0.026846            NaN
EmployeeCount               NaN        NaN             NaN            NaN
Over18                      NaN        NaN             NaN            NaN
StandardHours               NaN        NaN             NaN            NaN

               EmployeeNumber  ...  RelationshipSatisfaction  StandardHours  \
```

```
OverTime             -0.024037  ...                 0.048493            NaN
Attrition            -0.010577  ...                -0.045872            NaN
EmployeeCount              NaN  ...                      NaN            NaN
Over18                     NaN  ...                      NaN            NaN
StandardHours              NaN  ...                      NaN            NaN

               StockOptionLevel  TotalWorkingYears  TrainingTimesLastYear  \
OverTime              -0.000449           0.012754              -0.079113
Attrition             -0.137145          -0.171063              -0.059478
EmployeeCount               NaN                NaN                    NaN
Over18                      NaN                NaN                    NaN
StandardHours               NaN                NaN                    NaN

               WorkLifeBalance  YearsAtCompany  YearsInCurrentRole  \
OverTime             -0.027092       -0.011687           -0.029758
Attrition            -0.063939       -0.134392           -0.160545
EmployeeCount              NaN             NaN                 NaN
Over18                     NaN             NaN                 NaN
StandardHours              NaN             NaN                 NaN

               YearsSinceLastPromotion  YearsWithCurrManager
OverTime                     -0.012239             -0.041586
Attrition                    -0.033019             -0.156199
EmployeeCount                      NaN                   NaN
Over18                             NaN                   NaN
StandardHours                      NaN                   NaN

[5 rows x 35 columns]
```

[10]:
```
df4=df[["Attrition","TotalWorkingYears","JobLevel","OverTime","MaritalStatus"]]
```

3 Learning Bayesian Networks from Data

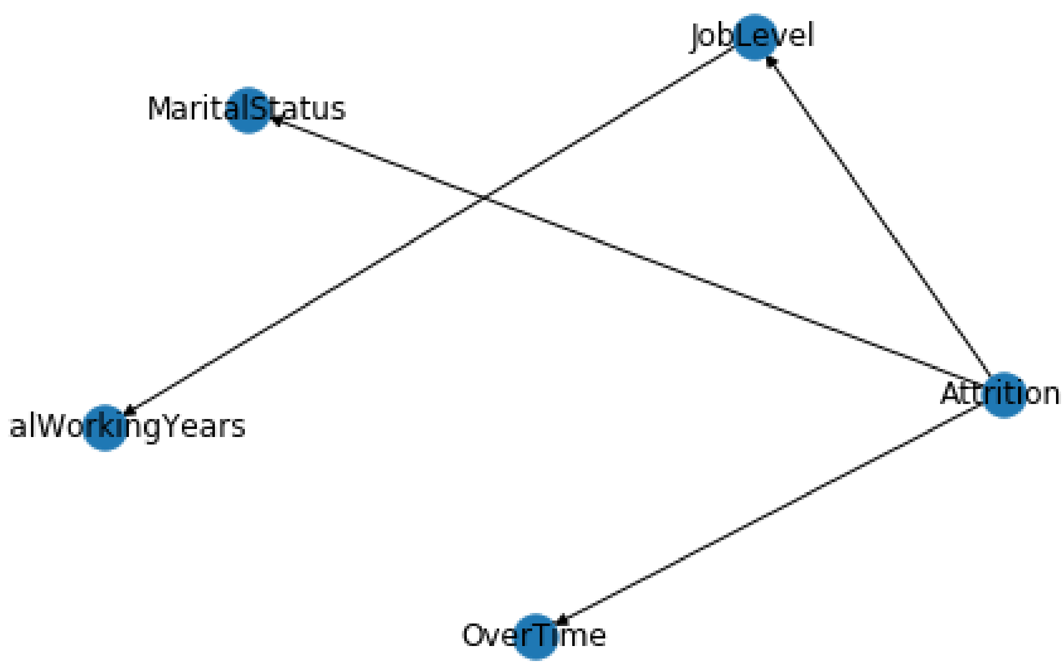
Because we do not have prior knowledge of the independency or dependency of these attributes, it is necessary to use an algorithm to find the best DAG structure for this data.

[14]:
```
from pgmpy.estimators import ExhaustiveSearch
es = ExhaustiveSearch(df4, scoring_method=BicScore(df4))
best_model = es.estimate()
print(best_model.edges())
```

```
[('Attrition', 'JobLevel'), ('Attrition', 'MaritalStatus'), ('Attrition',
'OverTime'), ('JobLevel', 'TotalWorkingYears')]
```

4 Plot bayesian network

[129]:
```
import networkx as nx
import pylab as plt
nx.draw(model, with_labels=True)
plt.show()
```



5 Finding CPDs

In order to run inference algorithms, it is necessary to find CPDs of this structure using our data.

[66]:
```
from pgmpy.models import BayesianModel
from pgmpy.estimators import MaximumLikelihoodEstimator
from pgmpy.factors.discrete import TabularCPD
model = BayesianModel([('Attrition', 'JobLevel'), ('Attrition',␣
↪'MaritalStatus'), ('Attrition', 'OverTime'), ('JobLevel', 'TotalWorkingYears')]
)
model.fit(df4)
model.get_cpds()
```

[66]:
```
[<TabularCPD representing P(Attrition:2) at 0x1e513908>,
 <TabularCPD representing P(JobLevel:5 | Attrition:2) at 0x1e8ed148>,
```

```
<TabularCPD representing P(MaritalStatus:3 | Attrition:2) at 0x1e513e08>,
<TabularCPD representing P(OverTime:2 | Attrition:2) at 0x115b8d08>,
<TabularCPD representing P(TotalWorkingYears:40 | JobLevel:5) at 0x116c2ec8>]
```

[56]:
```
model.get_independencies()
```

[56]:
```
(JobLevel _|_ MaritalStatus | OverTime)
(JobLevel _|_ MaritalStatus | TotalWorkingYears, OverTime)
(TotalWorkingYears _|_ MaritalStatus, OverTime | JobLevel)
(TotalWorkingYears _|_ MaritalStatus | OverTime)
(TotalWorkingYears _|_ OverTime | MaritalStatus, JobLevel)
(TotalWorkingYears _|_ MaritalStatus | JobLevel, OverTime)
(MaritalStatus _|_ TotalWorkingYears | JobLevel)
(MaritalStatus _|_ JobLevel, TotalWorkingYears | OverTime)
(MaritalStatus _|_ TotalWorkingYears | JobLevel, OverTime)
(MaritalStatus _|_ JobLevel | TotalWorkingYears, OverTime)
(OverTime _|_ TotalWorkingYears | JobLevel)
(OverTime _|_ TotalWorkingYears | MaritalStatus, JobLevel)
```

6 Inference with VariableElimination

Here we obtain the probability of Attrition using the VariableElimination distribution algorithm.

[95]:
```
from pgmpy.inference import VariableElimination
hr_infer = VariableElimination(model)
prob_offer = hr_infer.query(variables = ['Attrition'])
```

```
Finding Elimination Order: : 100%|| 4/4 [00:00<00:00, 999.95it/s]
Eliminating: OverTime: 100%|| 4/4 [00:00<00:00, 99.99it/s]
```

[96]:
```
print(prob_offer)
```

```
+--------------+------------------+
| Attrition    |   phi(Attrition) |
+==============+==================+
| Attrition(0) |           0.8388 |
+--------------+------------------+
| Attrition(1) |           0.1612 |
+--------------+------------------+
```

7 Get query with evidence

This may be the most practical. Suppose you have a number of candidates for a job that you know about these 5 attributes, you can add these evidence as the input of the algorithm and calculate the attrition probability for each candidate and if you see a significant difference in this quantity select the candidate you want accordingly.

for example here we set "TotalWorkingYears"=1,"JobLevel"=0, "OverTime"=1 as evidence, you can see in this case probability of attrition or not are equally likely.

[121]:
```
prob_Attrition=hr_infer.query(['Attrition'], evidence={"TotalWorkingYears":
↪1,"JobLevel":0,"OverTime":1})
print(prob_Attrition)
```

```
Finding Elimination Order: : 100%|| 1/1 [00:00<00:00, 52.63it/s]
Eliminating: MaritalStatus: 100%|| 1/1 [00:00<00:00, 249.99it/s]
```

```
+--------------+------------------+
| Attrition    |   phi(Attrition) |
+==============+==================+
| Attrition(0) |           0.5503 |
+--------------+------------------+
| Attrition(1) |           0.4497 |
+--------------+------------------+
```

8 Inference with Belief Propagation

[123]:
```
from pgmpy.inference import BeliefPropagation
belief_propagation = BeliefPropagation(model)
res = belief_propagation.query(variables = ['Attrition'])
```

```
Eliminating: OverTime: 100%|| 3/3 [00:00<00:00, 136.35it/s]
```

[125]:
```
print (res)
```

```
+--------------+------------------+
| Attrition    |   phi(Attrition) |
+==============+==================+
| Attrition(0) |           0.8388 |
+--------------+------------------+
| Attrition(1) |           0.1612 |
+--------------+------------------+
```

9 Limitations

As mentioned earlier, in this model, due to the limitations of the operating system, we used only 5 attributes to build the model, and we did not have prior knowledge about the interdependence of these attributes. If more features are selected and we can add our prior knowledge to the model, we will eventually have better accuracy for the model output.