Project Report
IBM: Employee attrition's analysis

Edoardo Gervasoni s1043824 - edoardo.gervasoni@student.ru.nl
Alberto Monaco s1043826 - alberto.monaco@student.ru.nl
Marco Spanò s1045892 - marco.spano@student.ru.nl

Bayesian Networks course — Radboud Universiteit — December 16, 2019

1 Problem Domain

The phenomenon of **Attrition** refers to a gradual reduction in personnel or work force in a company caused by resignation or retirement.

An unchecked process of attrition can cause numerous unfavourable issues to a company, including overall poor company performance, low employees morale, and even more attrition in worst of case. Remaining employees could also be called to perform tasks they are not completely trained to or unsuited duties for their background, which in turn leaves the staff feeling unappreciated, underpaid and overworked. Such conditions could quickly go out of control and turn into the mass exit of employees, preventing the company from living up to the promises.

Another major problem in employees attrition is its impact on costs for the organization: job postings, hiring processes, paperwork and new hire training are only some of the common expenses of losing personnel.

In this context, the scope of the following project is to investigate some causal relationships among economical and social variables related to the workers, in order to outline features and patterns which affect the attrition practice. All the discoveries and the deductions will be carried out through the construction of a Bayesian Network model.

2 Data and preprocessing

2.1 Data table

For the project drafting we have used the dataset "HR Employee Attrition", which contains information about a wide set of employees. The date are publicly accessible on the Kaggle platform at
https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset.

The dataset is composed of 1470 rows and 35 columns. Each record represents an employee, while each variable represents personal feature (i.e. *Age*, *Gender*...) or working data (such as *Income* or *JobRole*). In the following table we describe briefly the data attributes. In blue are colored the names of the variables considered in the final network.

Name	Type	Values	Description
Age	Continuos	18 - 60	Numerical age of the employee
Attrition	Binary	"Yes", "No"	Target variable indicating the attrition of the employee
BusinessTravel	Ordinal	"Non-Travel", "Travel_Rarely", "Travel_Frequently"	Target variable indicating the attrition of the employee
DailyRate	Continuos	102 - 1499	Daily rate for the employee

Department	Categorical	"Human Resources", "Research & Development", "Sales"	Working sector of the company
DistanceFromHome	Continuos	1 - 29	Distance between the working company and home in miles
Education	Ordinal	1 - 5	Variable indicating the level of education (1=‘Below College’, 2=‘College’, 3=‘Bachelor’, 4=‘Master’, 5=‘Doctor’)
EducationField	Categorical	"Human Resources", "Life Sciences", "Marketing", "Medical", "Other", "Technical Degree"	Education area where the employee performed his/her study
EmployeeCount	Continuos	1	Variable with no meaning, everytime equals to one
EmployeeNumber	Continuos	1 - 2068	ID of the employee
Gender	Binary	"Male", "Female"	Gender of the employee
HourlyRate	Continuos	30 - 100	Hourly rate for the employee
JobInvolvement	Ordinal	1 - 4	Grade of the involvement of the employee in the working environment (1 ‘Low’, 2 ‘Medium’, 3 ‘High’, 4 ‘Very High’)
JobLevel	Ordinal	1 - 5	Working level inside the company (from 1=junior to 5=partner)
JobRole	Categorical	1 - 9	Role of the employee inside the company (in order: 1 "Healthcare Representative", 2 "Human Resources", 3 "Laboratory Technician", 4 "Manager", 5 "Manufacturing Director", 6 "Research Director", 7 "Research Scientist", 8 "Sales Executive", 9 "Sales Representative")
MaritalStatus	Categorical	"Married", "Non-married", "Divorced"	Variable indicating if the employee is in a marriage or not
MonthlyIncome	Continuos	1009 - 19999	Salary of the employee
MonthlyRate	Continuos	2094 - 26999	Monthly rate for the employee
NumCompaniesWorked	Continuos	0 - 9	Number of companies the employee has worked before the actual one
Over18	Categorical	"Y"	Indicates if the employee is more than 18 years old (everytime has value "Y")
OverTime	Binary	"Yes", "No"	Indicates if the employee do works overtime
PercentSalaryHike	Continuos	11 - 25	Percentage indicating how much the salary increased from 2015 to 2016
PerformanceRating	Ordinal	1 - 4	Grade for the current performance of the employee(1 ‘Low’, 2 ‘Good’, 3 ‘Excellent’, 4 ‘Outstanding’)
StandardHours	Continuos	80	Variable with no special meaning, everytime equal to ‘80’
StockOptionLevel	Ordinal	0 - 3	Indicates the right to buy a certain value of stocks of the company (‘0’ Low, ‘3’ High)

TotalSatisfaction	Continuos	1 - 4	Average between ‘EnvironmentSatisfaction’, ‘RelationshipSatisfaction’ and ‘Job-Satisfaction’, grades of the satisfaction of the employee respectively for the working environment, relationship with colleagues and type of job performed
TotalWorkingYears	Continuos	0 - 40	Total amount of years worked during employee’s life
TrainingTimesLastYear	Continuos	0 - 6	Total amount of hours spent the previous year in job training
WorkLifeBalance	Ordinal	1 - 4	Ratio between amount of time spent working and free time (1 ‘Bad’, 2 ‘Good’, 3 ‘Better’, 4 ‘Best’)
YearsAtCompany	Continuos	0 - 40	Total amount of years worked inside the company
YearsInCurrentRole	Continuos	0 - 18	Total amount of years since the employee is working in the same role
YearsSinceLastPromotion	Continuos	0 - 15	Total amount of years passed since the last promotion of the employee
YearsWithCurrManager	Continuos	0 - 17	Total amount of years since the employee is working under the same manager

2.2 Preprocessing

Once loaded the data described above, we have exploited some exploration techniques to analyze variables roles and characteristics. In particular, the visualizations showed that some variables were bad recorded (for example some of them had the same value for all the rows) and others were simply not useful for our purposes, reason why we have not considered them in our project. We decided to discard the following attributes:

Over18, *StandardHours*, *EmployeeCount*, *EmployeeNumber*, *MonthlyRate*, *HourlyRate*, *DailyRate*, *JobInvolvement*, *EducationField*, *TrainingTimesLastYear*, *YearsWithCurrManager*, *Department*, *YearsInCurrentRole*, *JobRole*, *PerformanceRating*.

We have also opted to include a new variable called *TotalSatisfaction*, which is constructed from the mean values of 3 columns of the original dataset: *EnvironmentSatisfaction*, *RelationshipSatisfaction* and *JobSatisfaction*.

The next step consisted in reorganizing the categorical variables present in the dataset, giving an order to the several categories (for *JobLevel*, *WorkLifeBalance*, *TotalSatisfaction*, *Education*, *StockOptionLevel* and *BusinessTravel*) and setting as binary the ones that do not have a proper order (*Gender*, *OverTime* and in particular *MaritalStatus*, where we divided the employees in who is married and who is not).

3 The Network

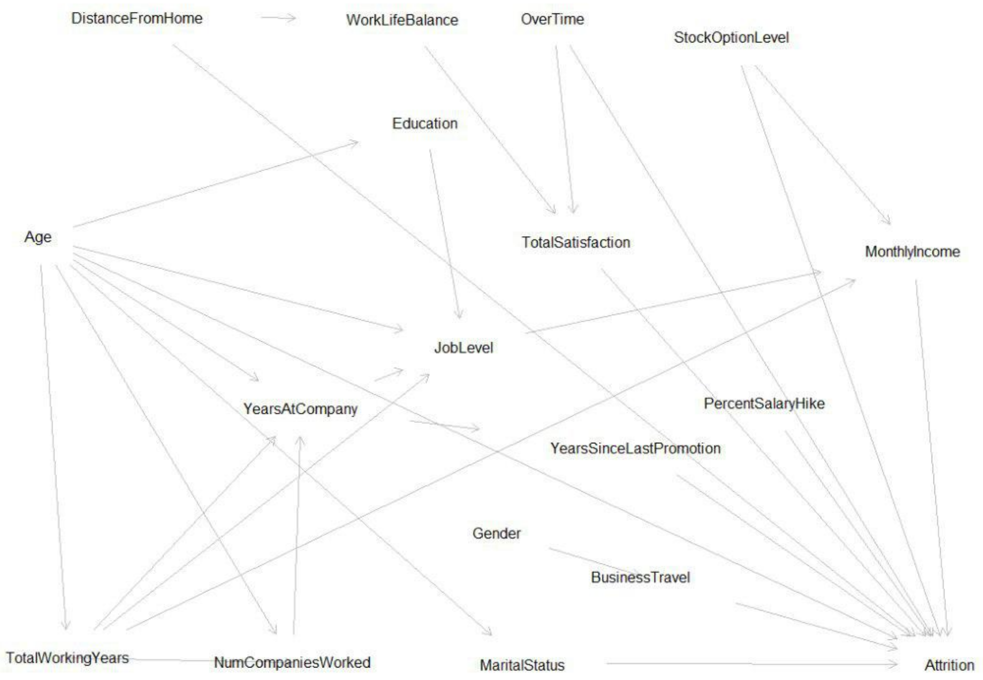


Figure 1: Our final DAG

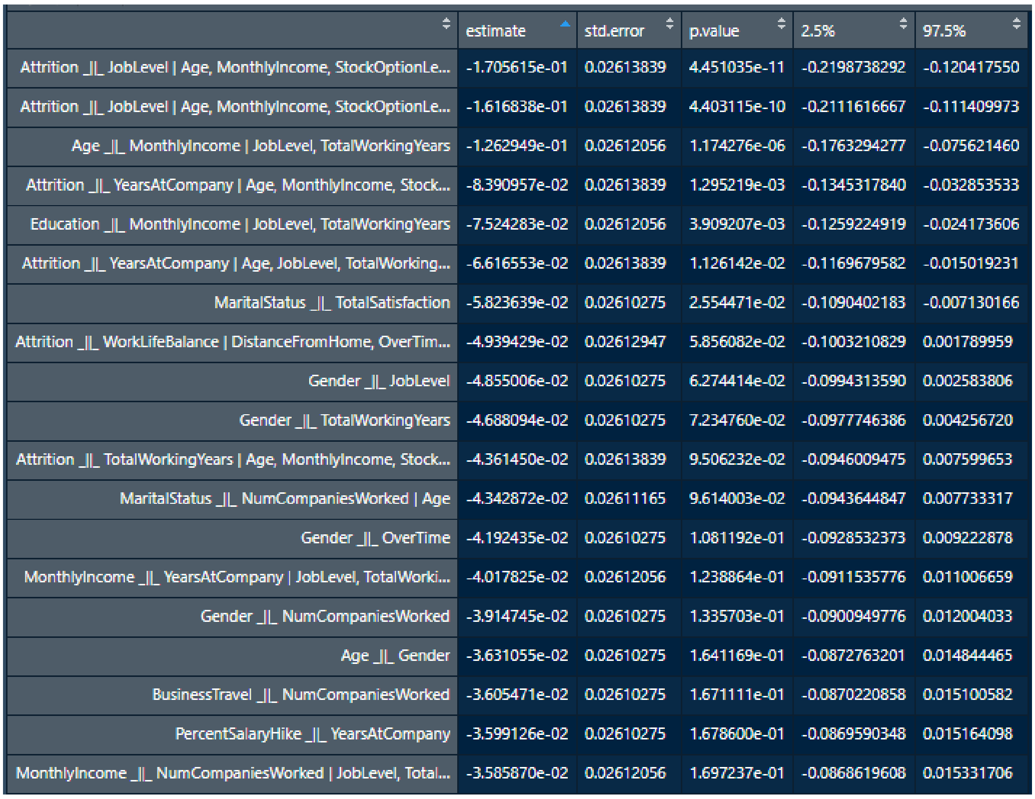
During the network building, we decided to place the nodes that were measured *a priori* on the top-left part of the screen. All the other variables, which we considered to be causal conseguences from the first ones, have been included after them, with the final outcome variable *Attrition* in the right-bottom corner. The main reason of such a disposition of the nodes is to have a better and clearer visualization.
The network was built using the online software Dagitty, and then tested and maniputated through R software, suitable to work with Bayesian networks, thanks to *bnlearn*, *lavaan*, *dagitty* and *bayesianNetworks* libraries.

3.1 Building the network

Given the heterogeneity of variables types (5 categorical ordinal variables, 3 binary and all the other continuous), we have decided to exploit the *polychoric correlation matrix*, in order to fit a hybrid discrete-continuous model.
Thus, we have extracted the polychoric correlation matrix from our dataset by the appropriate R function, and then we have built a first network considering the causal relationships we thought there were among the variables. After having tested the conditional independences within the nodes with the *LocalTest* function applied on the Polychoric matrix, we chose both to add some edges that we hadn't considered and to remove some others. We have repeated these steps up to the final network that is represented in the previous subsection.

3.2 Results

Here we report some results of the *LocalTest* function on our final DAG.



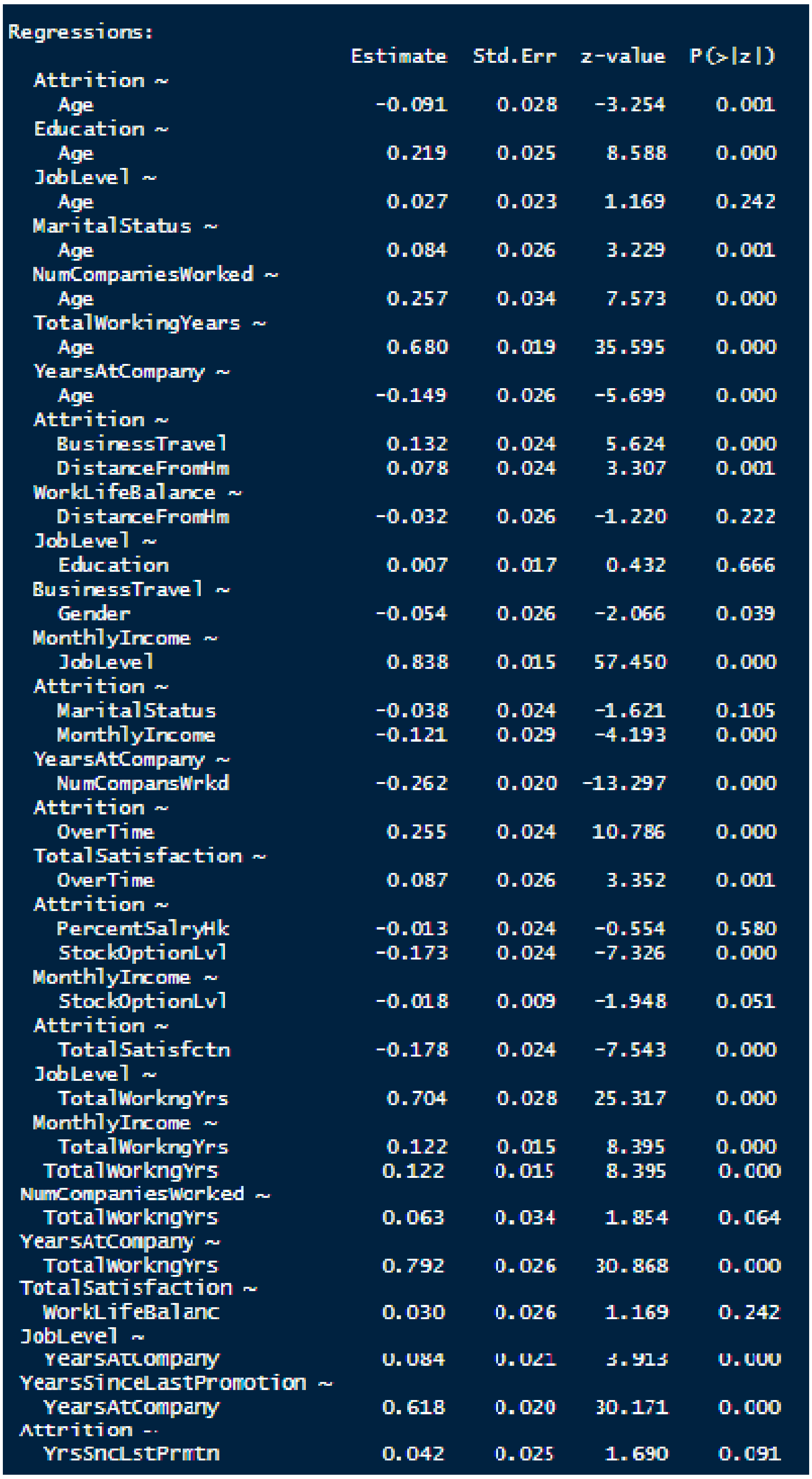
	estimate	std.error	p.value	2.5%	97.5%
Attrition _||_ JobLevel | Age, MonthlyIncome, StockOptionLe...	-1.705615e-01	0.02613839	4.451035e-11	-0.2198738292	-0.120417550
Attrition _||_ JobLevel | Age, MonthlyIncome, StockOptionLe...	-1.616838e-01	0.02613839	4.403115e-10	-0.2111616667	-0.111409973
Age _||_ MonthlyIncome | JobLevel, TotalWorkingYears	-1.262949e-01	0.02612056	1.174276e-06	-0.1763294277	-0.075621460
Attrition _||_ YearsAtCompany | Age, MonthlyIncome, Stock...	-8.390957e-02	0.02613839	1.295219e-03	-0.1345317840	-0.032853533
Education _||_ MonthlyIncome | JobLevel, TotalWorkingYears	-7.524283e-02	0.02612056	3.909207e-03	-0.1259224919	-0.024173606
Attrition _||_ YearsAtCompany | Age, JobLevel, TotalWorking...	-6.616553e-02	0.02613839	1.126142e-02	-0.1169679582	-0.015019231
MaritalStatus _||_ TotalSatisfaction	-5.823639e-02	0.02610275	2.554471e-02	-0.1090402183	-0.007130166
Attrition _||_ WorkLifeBalance | DistanceFromHome, OverTim...	-4.939429e-02	0.02612947	5.856082e-02	-0.1003210829	0.001789959
Gender _||_ JobLevel	-4.855006e-02	0.02610275	6.274414e-02	-0.0994313590	0.002583806
Gender _||_ TotalWorkingYears	-4.688094e-02	0.02610275	7.234760e-02	-0.0977746386	0.004256720
Attrition _||_ TotalWorkingYears | Age, MonthlyIncome, Stock...	-4.361450e-02	0.02613839	9.506232e-02	-0.0946009475	0.007599653
MaritalStatus _||_ NumCompaniesWorked | Age	-4.342872e-02	0.02611165	9.614003e-02	-0.0943644847	0.007733317
Gender _||_ OverTime	-4.192435e-02	0.02610275	1.081192e-01	-0.0928532373	0.009222878
MonthlyIncome _||_ YearsAtCompany | JobLevel, TotalWorki...	-4.017825e-02	0.02612056	1.238864e-01	-0.0911535776	0.011006659
Gender _||_ NumCompaniesWorked	-3.914745e-02	0.02610275	1.335703e-01	-0.0900949776	0.012004033
Age _||_ Gender	-3.631055e-02	0.02610275	1.641169e-01	-0.0872763201	0.014844465
BusinessTravel _||_ NumCompaniesWorked	-3.605471e-02	0.02610275	1.671111e-01	-0.0870220858	0.015100582
PercentSalaryHike _||_ YearsAtCompany	-3.599126e-02	0.02610275	1.678600e-01	-0.0869590348	0.015164098
MonthlyIncome _||_ NumCompaniesWorked | JobLevel, Total...	-3.585870e-02	0.02612056	1.697237e-01	-0.0868619608	0.015331706

Figure 2: Conditional independencies tests

From the table above, in the first two rows we have the conditional independence tests between *Attrition* and *JobLevel*, with coefficients ranging from ~ -0.17 to ~ -0.16 and with pretty low p-values. We have tried to add the direct edge between them in the DAG, but this leads to some discrepancies: for example, looking at the coefficient of $MonthlyIncome \rightarrow Attrition$, the positive correlation becomes negative, which is against our casual interpretation of the relation between these two nodes. Therefore we decided to avoid the edge insertion, leaving the previous DAG structure.

Moreover, for the same reason explained before, we chose not to add other edges considered significative by the same test, such as $MaritialStatus \rightarrow StockOptionLevel$, since their relation cannot be actually interpreted as causal.

Finally, in the next figure we are going to display the outcomes obtained by the fitting performed on the polychoric correlation matrix.



```
Regressions:
                           Estimate  Std.Err  z-value  P(>|z|)
  Attrition ~
    Age                      -0.091    0.028   -3.254    0.001
  Education ~
    Age                       0.219    0.025    8.588    0.000
  JobLevel ~
    Age                       0.027    0.023    1.169    0.242
  MaritalStatus ~
    Age                       0.084    0.026    3.229    0.001
  NumCompaniesWorked ~
    Age                       0.257    0.034    7.573    0.000
  TotalWorkingYears ~
    Age                       0.680    0.019   35.595    0.000
  YearsAtCompany ~
    Age                      -0.149    0.026   -5.699    0.000
  Attrition ~
    BusinessTravel            0.132    0.024    5.624    0.000
    DistanceFromHm            0.078    0.024    3.307    0.001
  WorkLifeBalance ~
    DistanceFromHm           -0.032    0.026   -1.220    0.222
  JobLevel ~
    Education                 0.007    0.017    0.432    0.666
  BusinessTravel ~
    Gender                   -0.054    0.026   -2.066    0.039
  MonthlyIncome ~
    JobLevel                  0.838    0.015   57.450    0.000
  Attrition ~
    MaritalStatus            -0.038    0.024   -1.621    0.105
    MonthlyIncome            -0.121    0.029   -4.193    0.000
  YearsAtCompany ~
    NumCompansWrkd           -0.262    0.020  -13.297    0.000
  Attrition ~
    OverTime                  0.255    0.024   10.786    0.000
  TotalSatisfaction ~
    OverTime                  0.087    0.026    3.352    0.001
  Attrition ~
    PercentSalryHk           -0.013    0.024   -0.554    0.580
    StockOptionLvl           -0.173    0.024   -7.326    0.000
  MonthlyIncome ~
    StockOptionLvl           -0.018    0.009   -1.948    0.051
  Attrition ~
    TotalSatisfctn           -0.178    0.024   -7.543    0.000
  JobLevel ~
    TotalWorkngYrs            0.704    0.028   25.317    0.000
  MonthlyIncome ~
    TotalWorkngYrs            0.122    0.015    8.395    0.000
   TotalWorkngYrs             0.122    0.015    8.395    0.000
 NumCompaniesWorked ~
   TotalWorkngYrs             0.063    0.034    1.854    0.064
 YearsAtCompany ~
   TotalWorkngYrs             0.792    0.026   30.868    0.000
 TotalSatisfaction ~
   WorkLifeBalanc             0.030    0.026    1.169    0.242
 JobLevel ~
   YearsAtCompany             0.084    0.021    3.913    0.000
 YearsSinceLastPromotion ~
   YearsAtCompany             0.618    0.020   30.171    0.000
 Attrition ~
   YrsSncLstPrmtn             0.042    0.025    1.690    0.091
```

Figure 3: Results of the fitting

The results achieved show that most of the edges of our final DAG are significative. In particular we have found several important estimates:

$JobLevel \to MonthlyIncome$; $Age \to TotalWorkingYears$; $TotalWorkngYrs \to YearsAtCompany$; $TotalWorkngYrs \to JobLevel$; $YearsAtCompany \to YearsSinceLastPromotion$;

All of these have an estimate coefficient of at least 0.60 and are positively correlated, as well as logically casual associated.
It has to be underlined that the the relations between $JobLevel \to Education$ and $PercentSalaryHike \to Attrition$ have been kept, despite having bad results according to the data available. The reason lies in the strong causal relation that often these variables have in the real world.
Furthermore, we should consider that the relations expressed in the results are linear, while it could be possible that *Education* and *JobLevel* or *PercentSalaryHike* and *Attrition* are linked in a non-linear way (e.g. quadratic, exponential, etc.).

4 Prediction

Once we have tested the structure of the network through the *LocalTest* function and fitted the parameters, the next step of the project consisted in performing a prediction task on the target variable, *Attrition*, with respect to its parent nodes in the DAG. In particular, the prediction phase has been realized by using 3 different classification models: Classic Logistic Regression, Stepwise Logistic Regression and Decision Tree.

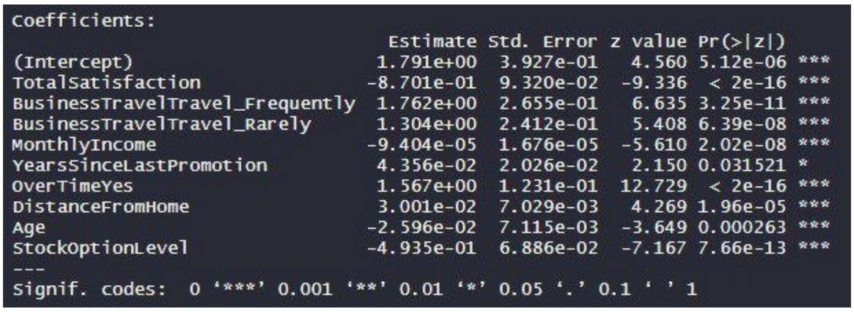
Here we present the results obtained:

	Accuracy	F1-Measure	AUC
Log. Reg.	0.75	0.74	0.75
Step Log. Reg.	0.74	0.73	0.74
Decision Tree	0.75	0.76	0.75

As we can see, our models perform pretty well despite their semplicity, since they can reach good performance measures, with $\sim 75\%$ of Accuracy for all of them.

However, the most important result that emerged here concerns the variables considered relevant by the classifiers. As a matter of fact, there is a great match between significant features according to the classification model and consistent nodes as stated by the polychoric correlation matrix fitting. These outcomes confirm our interpretation about the causal relations within the data.

To have an example of the above statement, the results obtained by Stepwise Linear Regression are reported:



```
Coefficients:
                            Estimate Std. Error z value Pr(>|z|)    
(Intercept)                1.791e+00  3.927e-01   4.560 5.12e-06 ***
TotalSatisfaction         -8.701e-01  9.320e-02  -9.336  < 2e-16 ***
BusinessTravelTravel_Frequently  1.762e+00  2.655e-01   6.635 3.25e-11 ***
BusinessTravelTravel_Rarely      1.304e+00  2.412e-01   5.408 6.39e-08 ***
MonthlyIncome             -9.404e-05  1.676e-05  -5.610 2.02e-08 ***
YearsSinceLastPromotion    4.356e-02  2.026e-02   2.150 0.031521 *  
OverTimeYes                1.567e+00  1.231e-01  12.729  < 2e-16 ***
DistanceFromHome           3.001e-02  7.029e-03   4.269 1.96e-05 ***
Age                       -2.596e-02  7.115e-03  -3.649 0.000263 ***
StockOptionLevel          -4.935e-01  6.886e-02  -7.167 7.66e-13 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Figure 4: Results obtained by the Stepwise Logistic Regression

For instance, *PercentSalaryHike*, which estimate coefficient was -0.013 and *p-value*$= 0.58$ as expressed from Figure 2, has to be considered not very helpful to predict the attrition attitude of employees, and in fact it isn't even included by the regression. Moreover, despite *YearsSinceLastPromotion* has been kept by the model, its associated coefficent is the least significative among the regressors, confirming the coefficient of the edge between it and Attrition (0.042 with a *p-value* $= 0.091$, slightly higher than the typical threshold of 0.05).

5 Discussion

The first goal of this analysis is to define and understand the main causes behind employee's attrition. Through the Bayesian Network represented in Figure 1, we were able to identify the most relevant nodes related to this problem.
In particular, looking at the results obtained from the fitting on the polychoric matrix, we can state that important causes of Attrition are associated with themes concerning the salary, free time and the possibility to buy company's stocks. In fact, the Income of employees (*MonthlyIncome* is negative correlated with *Attrition*), *OverTime* and the *StockOptionLevel* are some of the most important features emerged by our investigation. Other relevant roots of this problem are the employees' age, their distance from home, their overall satisfaction and if their job requires to travel frequently or not. Furthermore, it could be interesting to underline that factors referring to their promotion (*YearsSinceLastPromotion*) or rewards (*PercentSalaryHike*) are not very useful to discover Attrition evidence, according to our data.

As a last step, we also performed a prediction on *Attrition* with its parent nodes as explanatory variables using three different models, Classic Logistic Regression, Stepwise Logistic Regression and Decision Tree; the results obtained in this way confirmed our conclusions about the causal relations between the considered nodes.

6 References

[1] https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset
[2] https://www.highspeedtraining.co.uk/hub/causes-of-employee-turnover/
[3] https://jobzology.com/staff-attrition-vs-staff-turnover-whats-the-difference/
[4] http://dagitty.net/
[5] http://dagitty.net/learn/index.html

Project Report
IBM: Employee attrition's analysis

Edoardo Gervasoni s1043824 - edoardo.gervasoni@student.ru.nl
Alberto Monaco s1043826 - alberto.monaco@student.ru.nl
Marco Spanò s1045892 - marco.spano@student.ru.nl

Bayesian Networks course — Radboud Universiteit — January 30, 2020

Abstract

Over the course of the first assignment we focused on the understanding of the main causes behind employees' attrition. The Bayesian network, constructed to investigate the causal relationships among the available nodes, has been built up thanks to the software *Dagitty*. After several adjustments, it has been tested through several useful methods to prove the conditional independencies returned from the network arrangement on the data. These tests showed some unexpected results (factors referring to the promotions were not very helpful to capture Attrition) and corroborated some insight previously taken into account from our *a priori* knowledge.
In this report we are going to describe the second part of this project. It will consist in applying two different algorithms, able to infer the structure of a graphical model entirely from a set of data, and in comparing their outcomes with the results previously achieved by our own Bayesian network. Throughout the project, it has to be considered the different approach while comparing the models: the Network we have built takes into account patterns and structures that come from the knowledge of the authors, their backgrounds and thoughts about the casual relationships among the nodes; the two algorithms, **PC stable** and **Tabu Search**, conversely, construct DAGs working directly on the data, without any biased judgment.

1 Introduction

Automatic learning for Bayesian Network is a very powerful task that is raising importance nowadays, since it permits to discover simple and easy-to-read structures from data given as input. Our work aimed to learn a Bayesian Network model from a dataset about employees' attrition and to compare it to our hand-crafted network we previously worked on.

1.1 Algorithms

In order to achieve this task, we decided to perform the learning of the Bayesian network using two notable algorithms belonging to two different families, PC-stable and Tabu search. Both are a better and more complex variant of two simpler algorithms from which, as we will discuss later in this section, they inherit the same approach: these are respectively *PC* for PC-stable and *hill climbing* for Tabu search.
We decided to utilize those two algorithms first because they are well-known and easy to use (they are both already implemented in several packages), second for their suitability for our dataset, and last because they approach the same problem from two really different ways, as one is a constraint-based algorithm while the other is a score-based one. In this context, it is important to compare both the results obtained by these algorithms with each other, as well as with the network that we constructed in the Assignment 1.
We refer to PC-stable as the one first described in the paper of Colombo and Maathuis *A modification of the PC algorithm yielding order-independent skeletons* [3], while for Tabu search we can cite the paper *Tabu search—part I* [6] and *Tabu search—part II* [7].

PC-stable is an order-independent constraint-based algorithm that uses conditional independence tests on the dataset to discover a causal structure. This type of algorithm starts with a complete undirected graph and then prunes it by deleting all the edges that do not satisfy the conditional independence test (that could be information based, ad-hoc or statistical), in order to obtain the final skeleton of the graph. Subsequently it uses some specific rules trying to find the direction of the edges.
PC-stable is a better variant of the basic PC-algorithm, that is order-dependent and was found to be unstable especially for high-dimensional data; for example in [4], after being executed 25 times on the same dataset, it was found to have 2000 unstable edges on a total amount of 5000 in at least 50% of the cases tried. PC-stable follows the same passages of the PC but with some modifications applied, obtaining an order-independent algorithm that turns out to be more reliable and partially parallelizable.

The second algorithm is Tabu search, an iterative technique to solve an optimization problem that eventually gives us the Bayesian network as outcome. The first and simplest algorithm of this family is called *hill climbing* and it uses a score function to evaluate the actual network given a dataset (this function is the one used for the optimization problem). Starting from an initial solution (usually an empty graph), it attempts to improve the accuracy going step by step towards a local minimum.
Tabu search[2] is a memory-based strategy that changes hill climbing to end up with a better outcome. The main difference of this algorithm is that, at each step it prohibits to undo and reverse recent moves, since those would take the algorithm away from the local minimum and back to the previous state. To fulfill this requirement, the algorithm, after each step, creates Tabu entries. They consist of a list of moves that will be inserted in the table for a small number of times and that are checked at every step to block any attempt to perform them. The algorithm stops when it can't improve its current solution. Using this approach, Tabu search can outperform hill climbing, exploring more the set of possible solutions at every step (it does not go back to previous states because of the table, so it can only explore new possible paths) finding, eventually, a better one.

1.2 Parameters

In the two algorithms there are few parameters that can be changed, affecting the final outcome. Starting with PC-stable, two are the most important parameters that can be tuned: *alpha* and *type*.
Alpha, the parameter we chose to use in our project, is the significance level of the statistical test for conditional and unconditional independencies, and it is normally set at 0.05 as default[5]. High value of alpha will lead the Bayesian network to obtain more edges (since the variables, according to the test, are not independent), while low values of alpha will create as output a Bayesian network with less edges.
The *type* parameter, instead, rules the variety of statistical test to perform. Different types of tests are present and usually each one is more suited to a specific kind of data.

Looking at Tabu search, two are the most important parameters: the *Score Function* and the *Tabu Length*.
The second one, Tabu length, is the parameter that governs the size of the table. As we have mentioned before, a bigger list for storing Tabu moves will push the algorithm to explore portions of the search space that were not already been visited before, improving the overall final solution. On the other hand, a bigger table will raise significantly the computational time of the algorithm, as for each step more checks of the moves must be performed. A possible default value for this parameter is 10.
On the contrary, it is possible to vary the score function of the current Bayesian Network, by changing the parameter of the same name. Several score functions are available and all of them aim to maximize the likelihood of our model looking at the data.

1.3 Dataset

Throughout our work we are going to use a dataset involving the attrition burden among the employees of a company[1]. The data is composed of 1470 rows and 18 columns, with all the attributes not changed from the first assignment we have worked on.
In each row of the dataset, an employee is described through those 18 variables, that represent personal (as *Age, Gender*, ...) or working characteristics (like *Income* or *JobRole*).

2 Methods

We carried out our project thanks to the R software. In particular, we have used two functions of the package *bnlearn* in order to construct the DAGs: *pc.stable* and *tabu*. The first one allowed us to build a Bayesian Network starting from the data on employee attrition's through the pc algorithm; the second one was used to learn the structure of the DAG with the tabu search approach.

2.1 Preprocessing

The initial part of our project consisted of a preprocessing phase, which is necessary in order to obtain data that can be processed more easily and more efficiently by the algorithms. One common problem in structure learning is the fact that usually the data available contain variables with too many levels, thus implying a high number of degrees of freedom and too many combinations of values to consider. To avoid these complications, we first selected the variables that have been used in the Assignment 1 and then we discretized those that were continuous. After that, we proceeded gradually decreasing the number of classes that the variables could take. At the end of this process, we ended up with most of our variables having 2 levels (for *Age*, *Attrition*, *BusinessTravel*, *DistanceFromHome*, *Gender*, *JobLevel*, *MaritalStatus*, *NumCompaniesWorked*, *OverTime*, *StockOptionLevel*, *TotalWorkingYears*, *WorkLifeBalance*, *YearsAtCompany*, *TotalSatisfaction*); the levels of *Education*, *MonthlyIncome* and *PercentSalaryHike* have been reduced to 3, while those of *YearsSinceLastPromotion* to 5.

2.2 Distance Measures

After the preprocessing step and the execution of the two structure learning algorithms, we compared the results obtained with each other and with the DAG that we manually built in the first Assignment. In order to do this, we decided to use some functions always provided by the *bnlearn* package:

- *all.equal*: checks whether two networks have the same structure.
- *hamming*: is the Hamming Distance between two networks, which computes the number of different edges in their skeletons.
- *shd*: Structural Hamming Distance, which counts how many edges are different between the CPDAGs of the two networks, taking into account their v-structures. Thus, the SHD of two DAGs belonging to the same Markov equivalence class, is guaranteed to be zero.
- *compare*: compares two networks, taking into account also arc directions, with more complete approach. The possible output are: True positive (TP), which tells us the number of arcs that are the same in both the two arguments; False positive (FP) is the number of arcs which are present in the first network (considered as baseline) but missing in the second one or that have different orientations; False negative (FN) refers to edges that are present in the second argument but not in the first one, or have different directions.
- *graphviz.compare*: is a graphical comparison from the *Rgraphviz* package, which takes one network as reference, and plots all other networks such that true positive arcs are in black, false positive arcs are in red, while false negative arcs are in blue, and drawn using a dashed line.

3 Results

In this section we are going to explicitly show the main differences emerged in the graphical model representations.
We will start showing the structure learning results of the algorithms Tabu Search and PC Stable, in comparison with the original Bayesian Network:

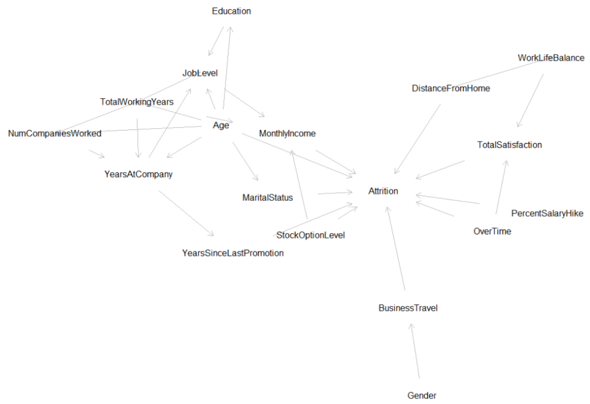


Figure 1: Network of the First Assignment

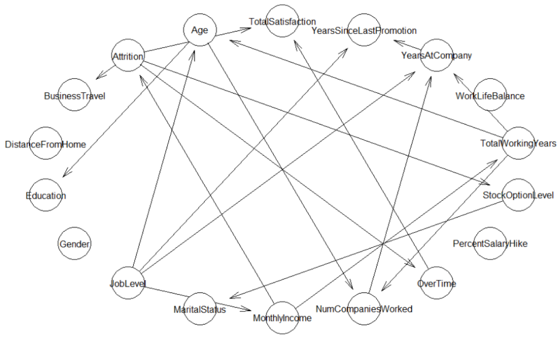


Figure 2: Network obtained with Tabu Search

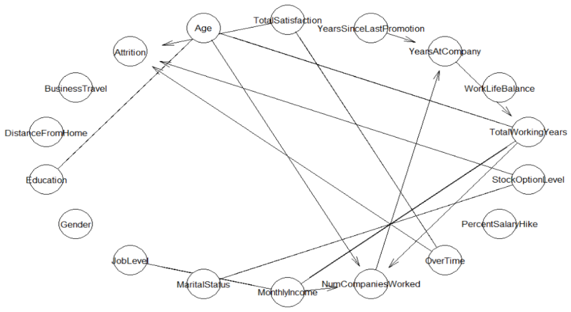


Figure 3: Network obtained with PC.stable

We investigated the differences among the networks through the indicators described in the previous section. First of all, the function *all.equal* showed that the results were all different, as we expected.
Let's now look at the *Hamming Distance* and the *SHD*:

Comparison	Hamming Distance	SHD
PC - Tabu	6	13
PC - Hand-constructed	19	27
Hand-constructed - Tabu	15	32

As we can see, the major differences come from the comparison between the hand-constructed Bayesian Network with respect to the two algorithm implementations.
Now let's consider the results of the function *compare*:

First Argument v Second Argument	TP	FP	FN
PC - Tabu	3	16	12
Hand-constructed - PC	6	9	24
Tabu - Hand-constructed	9	21	10

Finally we show the differences between the DAGs obtained with *graphviz.compare*, where the leftmost Network is considered as the baseline:

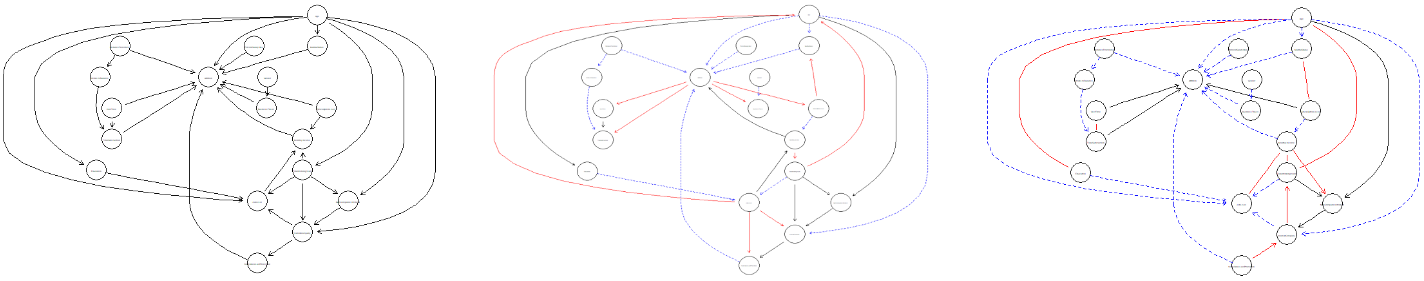


Figure 4: From left to right: our hand-constructed network; its comparison with the result of Tabu Search; its comparison with the result of PC.Stable (TP in black, FP in red and FN in blue)

Once we have achieved the outcomes of the two structure learning models, we decided to vary some parameters in both the algorithms. In the first case we modified the parameter *alpha*, with levels 0.2, 0.5 and 0.75, while in the second one we varied *tabu* (the tabu table size), assigning values of 20, 50 and 100 to the table length. Here we show some of the networks obtained:

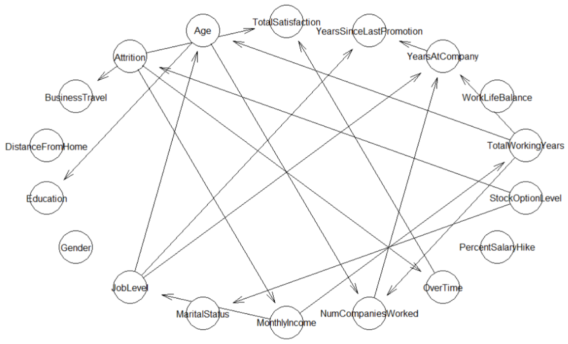


Figure 5: Tabu Search, tabu parameter: 100

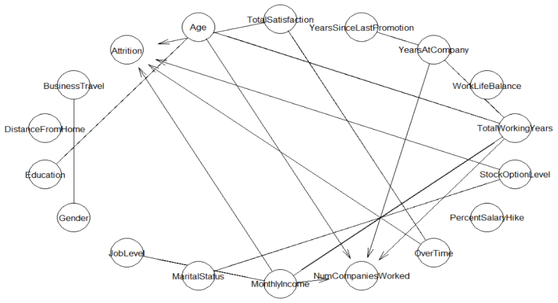


Figure 6: PC Stable, $\alpha = 0.20$

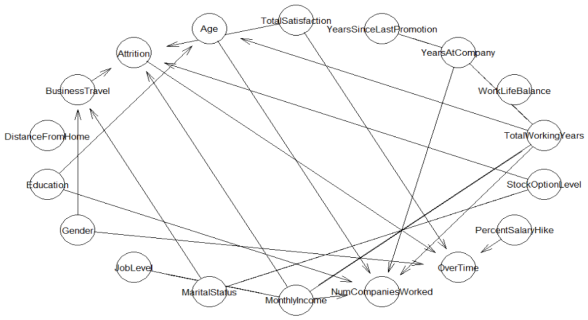


Figure 7: PC Stable, $\alpha = 0.75$

The Tabu table size seems to not really affect the edges arrangement, except for some arrowheads (we also tried to assign lower values, obtaining the same results). On the other hand, *alpha* leads the PC Stable to detect a greater number of connections in the graph. In particular, the number of edges increases at the same pace of the alpha rise.

In the following tables, we can confirm the graphical insights received. For the PC stable:

Comparison with alpha 0.05	Hamming Distance	SHD
$\alpha = 0.2$	1	4
$\alpha = 0.5$	8	12
$\alpha = 0.75$	7	14

For the Tabu Search:

Comparison with table size 10	Hamming Distance	SHD
table size: 20	0	0
table size: 50	0	0
table size: 100	0	0

4 Discussion

The results showed that the two networks constructed using structure learning algorithms are less different with each other than to our manually built DAG, as we can see from the tables of the Hamming Distance and the SHD. We expected this result, given that the two algorithms consider only the data that are provided to them, while the manual construction process of the network takes into account previous knowledge too. For instance, in our DAG we decided to keep the relations $Education \rightarrow JobLevel$ and $PercentSalaryHike \rightarrow Attrition$, despite having bad results according to the data available (high p-values and low estimate coefficients in the polychoric matrix), since we thought there was a strong causal relation between these variables in the real world. The two structure learning algorithms, on the other hand, didn't maintain these edges. This discrepancy of building methods therefore inevitably leads to differences between the constructed networks.

Regarding the two algorithms, Tabu Search is the one that performed better, since it managed to detect more relationships between the variables than the PC Stable, and notably, 9 of these relationships are also present in our DAG of the first Assignment (TP = 9).

Furthermore, we noticed that, in the case of this first technique, the variation of the parameter *tabu* does not lead to significant different results, since only the direction of some edge changes; therefore we considered the default value 10 to be optimal. The statement about PC is totally different, as increasing the value of *alpha* leads to a greater number of connections detected. However, this result is due to the fact that we are just increasing the significance level of the statistical test for the independence between nodes; for instance, selecting $\alpha = 1$, we would consider every node linked to each other. Thus we chose to maintain the typical threshold of 0.05.

So, while the Tabu Search algorithm seems to not be affected by hyperparameter tuning, the PC.Stable has the problem that its outcomes are particularly sensitive to changes of alpha. A possible solution to this issue can be found in [5]: here we define a function $y_i = f(\theta) + \epsilon_i$, with ϵ_i being a Gaussian noise term, while $f(\theta)$ can be seen as a black box objective function, where θ depends on α, the significance level, and T, the type of test. Each time we consider a new observation, a probabilistic model (typically a Gaussian process), is fitted to the data collected so far. In this way we obtain an optimization problem, where we want to maximize f in order to get the optimal value for α and T.

A possible future improvement could be the implementation of such an approach, where the alpha value and the type of test are chosen together as the best parameters according to the data available.

References

[1] IBM HR Analytics Employee Attrition Performance. https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset 1.3

[2] Beretta, S., Castelli, M., Gonçalves, I., Henriques, R., Ramazzotti, D.: Learning the structure of bayesian networks: A quantitative assessment of the effect of different algorithmic schemes. Complexity **2018** (2018) 1.1

[3] Colombo, D., Maathuis, M.H.: A modification of the pc algorithm yielding order-independent skeletons. arXiv preprint arXiv:1211.3295 (2012) 1.1

[4] Colombo, D., Maathuis, M.H.: Order-independent constraint-based causal structure learning. The Journal of Machine Learning Research **15**(1), 3741–3782 (2014) 1.1

[5] Córdoba, I., Garrido-Merchán, E.C., Hernández-Lobato, D., Bielza, C., Larranaga, P.: Bayesian optimization of the pc algorithm for learning gaussian bayesian networks. In: Conference of the Spanish Association for Artificial Intelligence. pp. 44–54. Springer (2018) 1.2, 4

[6] Glover, F.: Tabu search—part i. ORSA Journal on computing **1**(3), 190–206 (1989) 1.1

[7] Glover, F.: Tabu search—part ii. ORSA Journal on computing **2**(1), 4–32 (1990) 1.1