*Automatic model selection, predictive methods, machine learning and statistical learning.*

*AMSE – 17/01/2021*

# IBM HR Analytics Employee attrition

Elisa SEBASTIAN
Eva MANUKYAN
Caroline REBOUILLAT

# Table of contents

# 1 Introduction

Attrition is the decrease or loss of a measurable quantity of people in a given set and a limited period of time.

The attrition rate is an indicator frequently used to measure the level of loss of customers or subscribers to a product, service, or brand. It is used to assess a company's ability to retain its customers.

This indicator is therefore more consumer-oriented, but you may hear about a **company's attrition rate**.
In this case, it refers to the ***loss, natural or otherwise, of employees due, for example, to death, retirement or an excessively high usure***.

Although job renewal in a company is completely normal, many employees leave a company because they are dissatisfied with **working conditions**.

These unwanted departures can harm the company's **image** but also its **productivity**.

The aim of this study is therefore to **predict attrition** of valuable employees *(excluding retirement or death)* in order to **avoid loss of talent** and understand the factors that influence the employee's decision. Also, companies invest a lot of time and resources in employee recruiting and training, therefore, retaining talent can also reduce these indirect costs.

This project could therefore be of particular use to the human resources of companies experiencing abnormally high attrition.

To do so, we use a database from the **IBM company**[1] on **1470 employees** (*of which* ***237 left*** *the company of their own free will)*.

In this project, you will find **some studies** that have been done on this subject (***Section 2 - literature review***), the description of the data base with our pre-processing (***Section 3 – Database description and preprocessing***) and some descriptive statistics to present the variables and see what factors could potentially influence attrition (***Section43 - Descriptive statistics***).

Then, we present models of **automatic variable selection** since we have **25 variables** with dummies (***Section 5 - Automatic model selection***).
This section is composed of a ***Principal Component Analysis***, a ***Forward Stepwise Selection***, a ***logistic regression, Ridge, Lasso*** and ***Elastic Net*** Models.

We then compare these models to other techniques, and, more precisely, machine learning algorithms (***Section 6 - Machine learning methods***). The methods are the ***gradient descent, tree-based methods, neural networks, SVM*** and ***K-means***.

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

Finally, you will find our conclusion and comparison of our models in ***section 7.***

Through this study, we have shown that the most important factors concerning attrition are temporal variables such as *age, number of years worked in the company, total working years, years in the current role* and with the *current manager.*

But also, other variables such as the *monthly income*, the *marital status* and the *training times last year* plays their role.

Finally, we show that the best model for making predictions was neural networks, with **96,94% accuracy** on a test sample, and regarding other metrics (*recall, precision, F1 score, ROC curve...*).

# 2 Literature review

It is fairly intuitive that attrition causes problems at the corporate level.

According to *Prakash and Chowdhury*[2], a consequent **attrition** rate increases the **investment** (*in time and in money*) made for new employees in any company. And, unfortunately, when this attrition is high, then it is impossible for this investment to be translated into profit.

In addition, some companies may not be able to replace jobs that have been lost, and it can be difficult for human resources to **fill the empty space** in the organization.

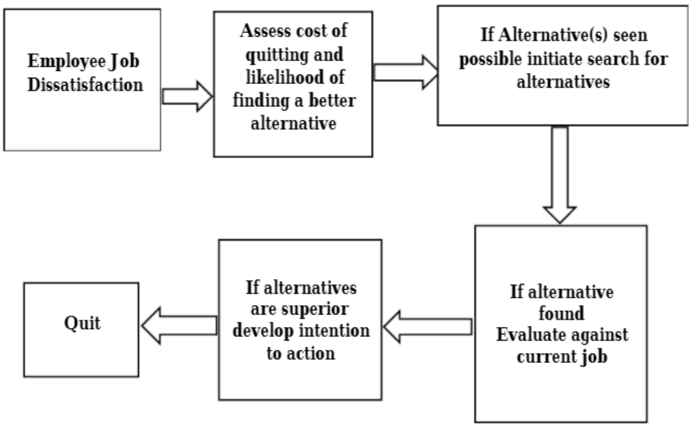
But managers and human resources do have a role to play.

Lot of studies shows the usefulness of **human resource management (HMR)** in order to identify relationships with **productivity**. As an example, *M. Marchington, A. Wilkinson, R.Donnelly and A.Kynighou* [3] discussed in detail about employee engagement, knowledge intensive firms and talent management.

In any case, many results show that this HRM plays a role in the productivity of employees and therefore has positive effects on **capital growth** of companies, such as in the study of *K. Deepak, J.Guthrie, P. Wright* [4].

To do so, human resources need to understand this phenomenon of attrition, which is not necessarily new.

In **1982**, *Bill Mobley* suggested the **Traditional model of attrition**:


Employee Job Dissatisfaction
Assess cost of quitting and likelihood of finding a better alternative
If Alternative(s) seen possible initiate search for alternatives
If alternative found Evaluate against current job
If alternatives are superior develop intention to action
Quit

An employee evaluates his current job and level of satisfaction. If he is dissatisfied, he assesses the cost of quitting and success of searching for an alternative job. Then, the decision to leave the job or not is taken.

[2] Prakash, S. and Chowdhury, R. (2004).“Managing attrition in BPO, In search of Excellence”

[3] . Marchington, M.; Wilkinson, A.; Donnelly, R.; Kynighou, A. Human Resource Management at Work; Kogan Page Publishers: London, UK, 2016.

[4] Deepak, K.D.; Guthrie, J.; Wright, P. Human Resource Management and Labor Productivity: Does Industry Matter? Acad. Manag. J. 2005, 48, 135–145

It remains to be seen what job satisfaction depends on.

As an example, *S. Angelo, Denisi and Ricky W. Griffin*[5] take into consideration the **nature of work, pay and benefits, supervision and co-workers**. And, unlike Mobley's model, they add the fact that sometimes, the **search for alternatives** can lead to increase the **satisfaction** on the present job.

It is therefore important to consider both **internal factors** (the company) and **external factors** (the labour market) that impact the attrition rate.

Of course, many authors highlight different factors. We can think of *D. Alao and A. Adeyemo*[6] who showed that attrition often results from demographics and job-related factors such as the **salary** and the **duration of the employment relationship**.

The purpose of this project is now to consider different factors that could influence workers to leave their jobs.

You will therefore find the following section, the description of the database and its pre-processing, where you will find the **variables** that we think are relevant at this stage of the study.

[5] Angelo S Denisi , Ricky W Griffin: (2009) Human Resources Management , BiztantraPublication, New Delhi 2nd edition

[6] Alao, D.; Adeyemo, A. Analyzing employee attrition using decision tree algorithms. Comput. Inf. Syst. Dev. Inf. Allied Res. J. 2013, 4, 17–28.

# 3 Database description and pre-processing

The database we are going to use is distributed by **IBM analytics** and was created by data **scientist** within this company using **HR data**.

This database contains 35 variables related to **1470 employees**. It informs about the personal characteristics of employees and their working conditions.

***We have selected 25 of them:***

- *Age*
- *Distance from home in miles*
- *Hourly rate in $*
- *Salary hike in %*
- *Monthly income in $*
- *Working years*
- *Number of companies where the employee has worked.*
- *Number of years at IBM*
- *Years since last promotion*
- *Years in current role*
- *Years with the current manager*
- *Training times last year in weeks*
- *Gender*
- *Marital Status*
- *Work overtime or not.*
- *Business travels*
- *Job role*
- *Education Field*
- *Education*
- *Environment satisfaction*
- *Relationship satisfaction*
- *Job satisfaction*
- *Job involvement*
- *Work life balance*

***This dataset also contains the variable "Attrition" which is the target feature in this study, encoded as:***

- 0 if the employee did not leave the company *(1233 employees*)
- 1 if the employee left the company, except death and retirement (*237 employees*)

There are no missing values in this database.

We recoded **qualitative variables** in **dummies** to transform the n values of a class into n binary variables. *For example, starting from the "BusinessTravel" variable we created 3 new variables:*

- Non_travel: 1 if yes, 0 if no
- Travel_Frequently: 1 if yes, 0 if no
- Travel_Rarely: 1 if yes, 0 if no

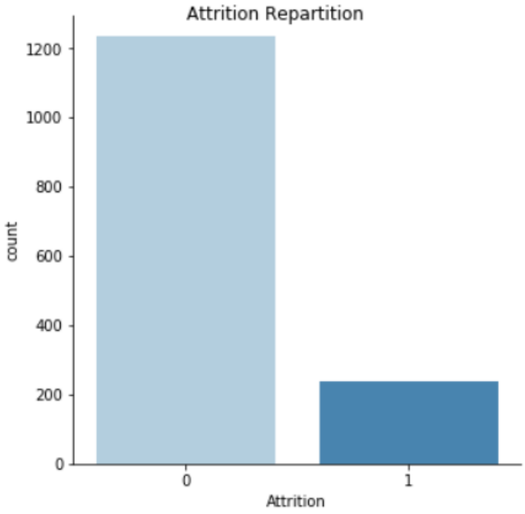
Then, we **standardize** the dataset: the quantitative variables have been centred to the mean and component wise scale to unit variance. Indeed, different orders of magnitude can lead to lower performance in the models we applied.

In machine learning and econometrics, a model can make good predictions about the data it has trained with but can be very poorly generalized to other data. This is called **overfitting**.

In order to give a realistic indicator of the **performance** of our models, we use **80%** of our database to train the models (***train set***) and the remaining **20%** to test them (***test set***).

For the ***section 6*** about machine learning methods, we also chose to **rebalance** our database.

*Originally, here is the breakdown between employees affected by attrition and those not affected:*



We will rebalance the data using **SMOTE** technic ***(Synthetic Minority Oversampling Technique).***

In fact, we **oversampled the minority class** in the training set by duplicating examples in the attrition class to have better predictions.

*How does it work? This algorithm:*

- Select examples that are close in the feature space (typically **5 nearest neighbors**)
- Draw a line between the examples in the feature space and drawing a new sample at a point along that line.

The best way to obtain a fairly "realistic" database is, over the SMOTE method, which is synthetic, to **undersample the majority class** *(employees who are not affected by attrition*) by simply deleting lines from the database **randomly**.

To conclude, the data we will use in ***Section 6*** will therefore be composed of **62 explanatory variables** and the **attrition** variable, with **1976 observations** in the ***train set*** (*with half of them concerning attrition*) and **294** in the ***test set.***

Concerning ***section 5 - automatic model selection***, we will show you the data pre-processing throughout the analysis as we must select variables and possibly add variables to take into account nonlinearities and relationships between variables.

# 4 Descriptive statistics

## 4.1 Means

Regarding the quantitative variables, the employees in this database have on average:

- **37 years old**
- A distance of **9.19 miles from work** to their home.
- **An hourly rate of $65.89**.
- A monthly **salary of $6502.93.**
- A **salary hike of 15.2%.**
- **Worked about 11 years** in their lives.
- Worked in **2 to 3 companies before** returning to IBM.
- Stayed about **7 years in the IBM company** and **4 years without changing role** within the company.
- Their **last promotion was 2 years ago**.
- Stayed with the **same manager for 4 years.**
- A **training time last year of 2.7 weeks**.

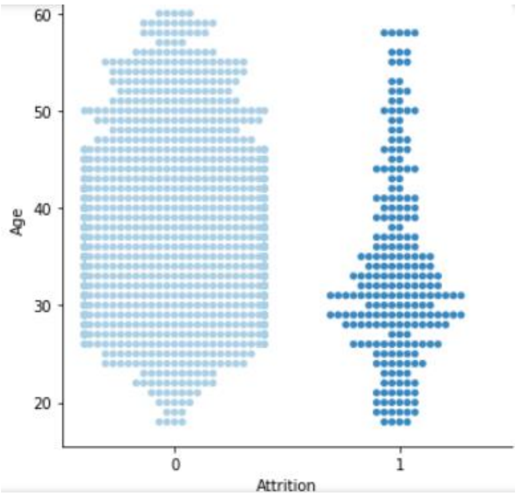
*Regarding the qualitative variables, most of the employees in this database are:*

- **Men**
- **Married**
- Who **do not work overtime**
- Who **travel rarely**
- Who are **Sales Executive**
- With an education field of **life sciences**
- Who have a **bachelor**
- With a **high environment and relationship satisfaction, a high job involvement, a very high job satisfaction** and a **better work life balance**

## 4.2 Plots

### *4.2.1 Age*

*How is attrition dependent on the age?*



The **attrition** is maximum between **28 and 32** years-old employees and ***fall with an increasing age.***

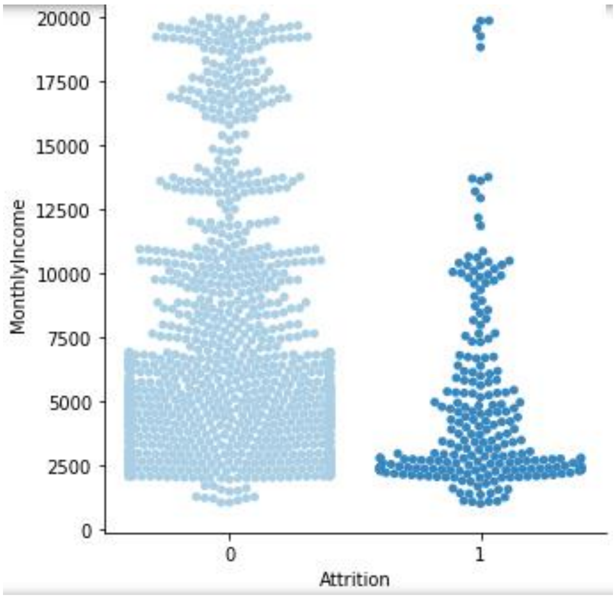
This result could be interpreting as older people looking for **stability** in their work.

Furthermore, people with an **age below 20** have less chances to leave the company, certainly because this is their first job.

In any case, from every age there is attrition or not.

### *4.2.2 Monthly income*

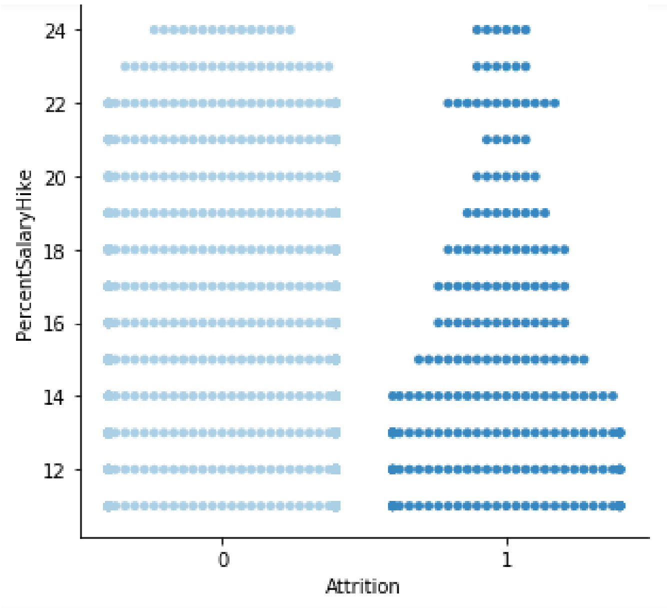
*Is income a factor towards employee attrition?*



The **attrition** is larger for employee with a relatively low monthly income **(< $3000)** and, for higher monthly income, the attrition decreases.

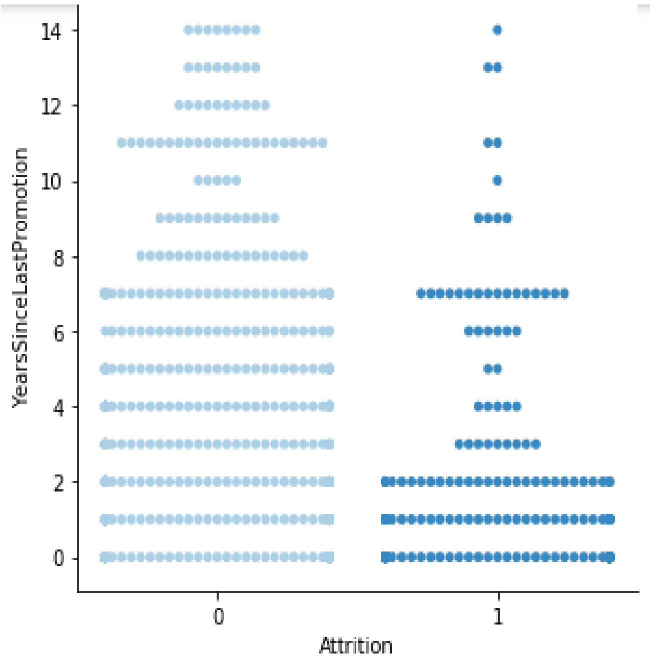
It is possible that those employees tend to move to a different job for a better standard of living.

### *4.2.3 Hike percentages*



Employees who receive a **higher hike** percentage have more chances to stay in the company, certainly because this hike can **motivate** them.

### *4.2.4 Years since last promotion*



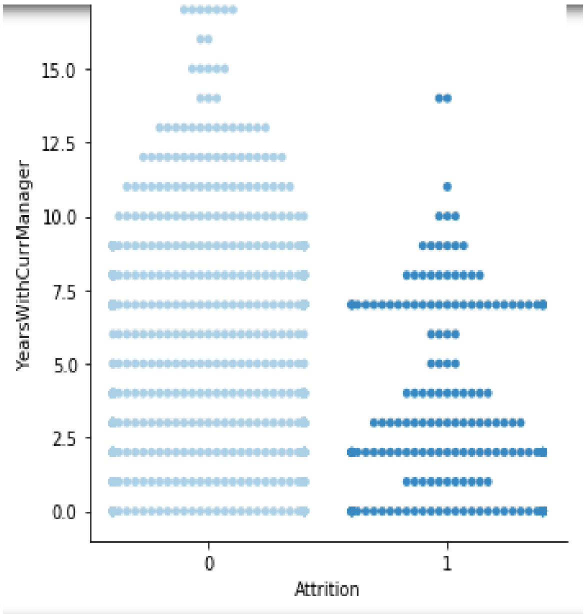
Counter-intuitively, in the **attrition** class, there are more employees that have a recent promotion (*between 0 and 3 years ago*).

But this may be due to other factors: these employees may be paid less at the base or may have received a bonus precisely because they wanted to quit their job.

Afterwards, attrition decreases but reappears for those who received a promotion 7 years before.

This is more intuitive, as employees may want more recognition.

### *4.2.5 Spending years with current manager*

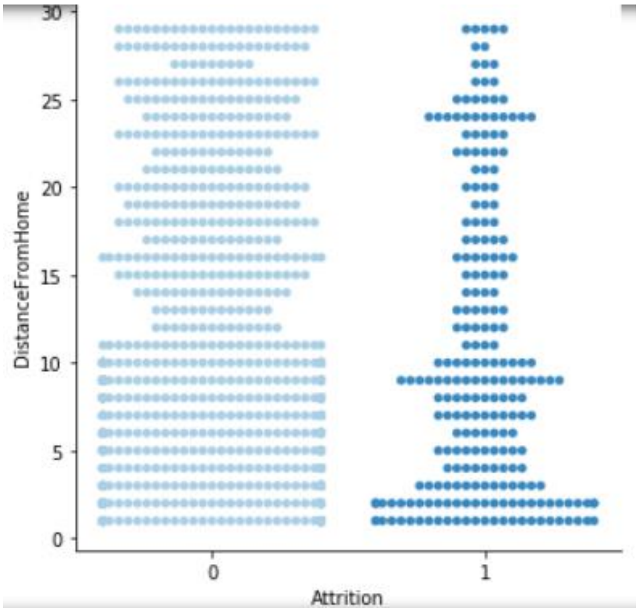


For employees who are **starting** to work with their manager, there is a relatively high **attrition**, probably because of the adaptation. Then, the attrition decreases the first year.

The **second year**, employees also tend to **leave** their job, but, after (***up to 7 years with their manager***) attrition seems to **decrease**. But at this point attrition reappears. This could be explained by the fact that people need to progress in their career.

Finally, **after 7 years** spent at the side of their manager, the employees seem to be satisfied with their job.

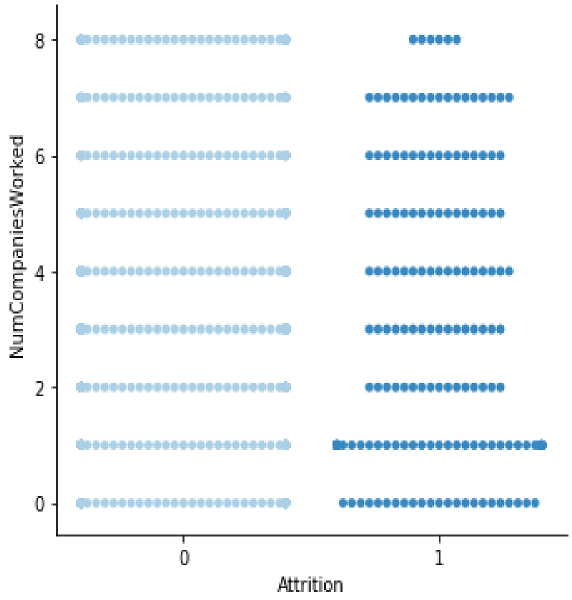
### *4.2.6 Distance from home*



In the **attrition** class, employees who leave **near** their place of work are more present.

Again, this seems counter-intuitive, but surely there are external factors that influence this result.

### *4.2.7 Number of compagnies worked.*

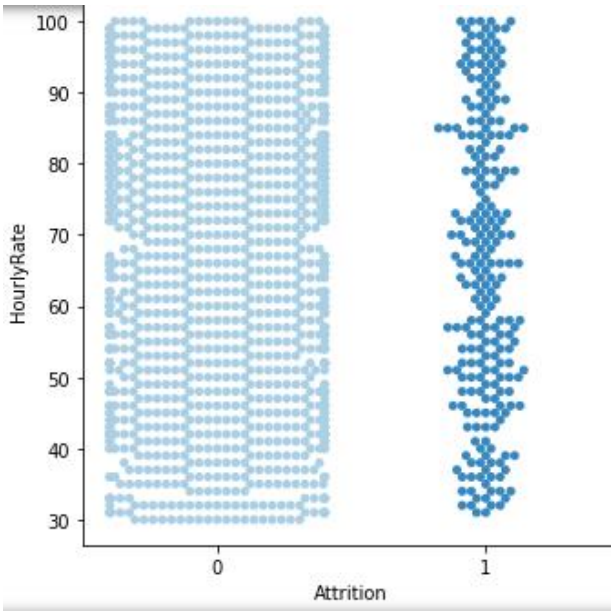


The relationship between the number of companies the employee has worked for and whether he or she leaves the company does not seem to be especially strong.

There is a **slight decrease in attrition** from **2 companies** where the individual has worked.

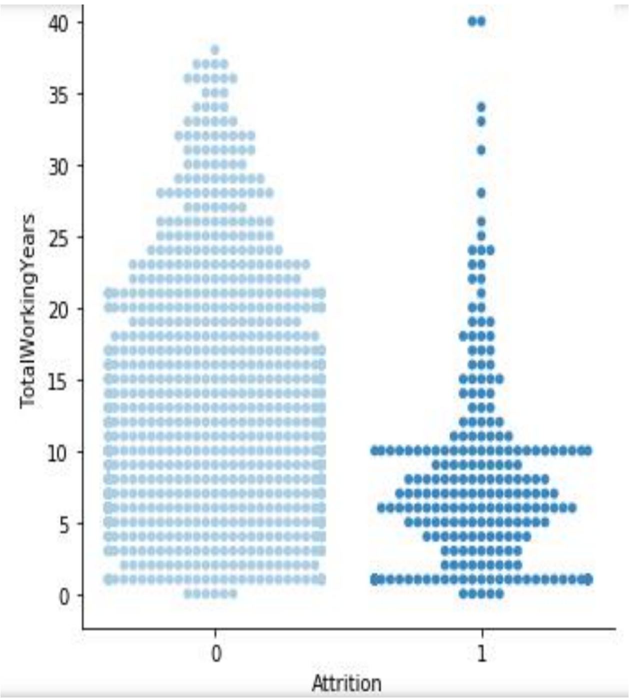
Attrition is nevertheless exceptionally **low** for individuals who worked in **8 different companies**, but this variable could be related to age and therefore stability.

### *4.2.8 Hourly rate*



The hourly rate seems to be unrelated to leave the job or not regarding this swarm plot.

### *4.2.9 Total working years*

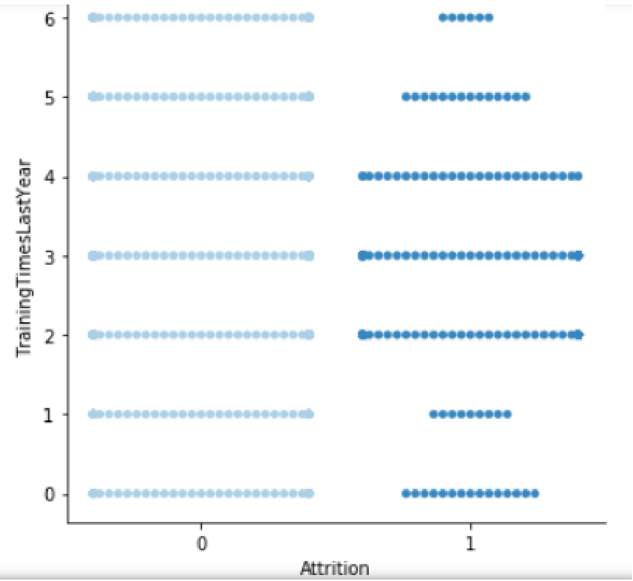


Although we see that the distribution of employees is greater for fewer years of work, we can see that this phenomenon is even more present in the attrition category.

Indeed, people who have worked only **one year** in their life are more prone to **attrition** (*surely because they are in the experimentation phase*) and we see that this is also the case for people who have worked **5 and 10 years** (*this could be people who wish to evolve professionally*).

In any case it is likely that people who have worked for **more than 12 years** in their lives are **less likely to experience attrition**.

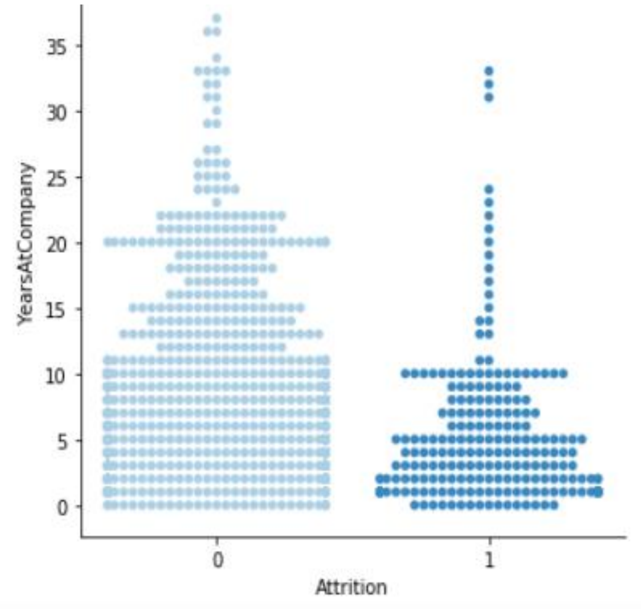
### *4.2.10 Training times last year*



In this graph, we see that employees not affected by attrition have the same distribution regarding the number of weeks of training within the company.

Nevertheless, when we look at those who left their jobs, we notice that those who received the **least training** (*0 or 1 week*) or, on the contrary, those who received a lot of training (*5 or 6 weeks*) seem to be more **satisfied** with their jobs than those who received *2 to 4 weeks of training*.

### *4.2.11 Years at company*



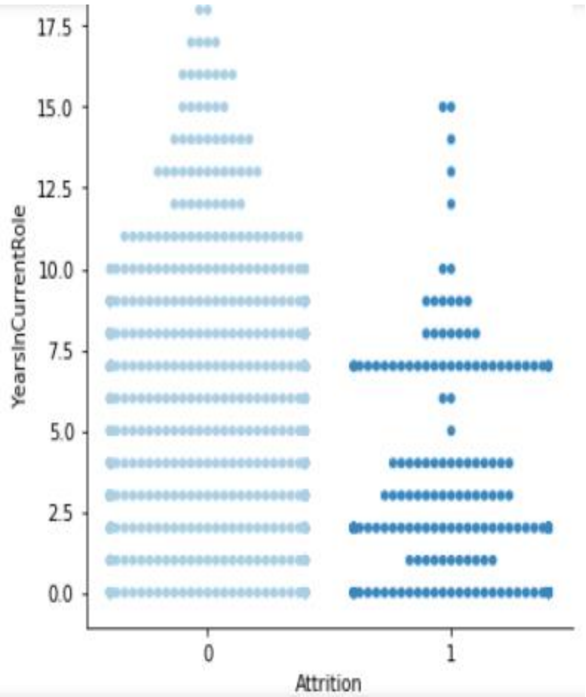
Intuitively, people who leave their jobs due to **attrition** are most often those who have been with the company for **less than 10 years**.

In fact, if an employee feels badly or aspires to leave their company, they usually do not wait more than 10 years to do so.

If we go into more detail, we see that people who are in their **first year** of work or **between 6 and 9 years of work in the company** are slightly **less concerned** relative to the group of 0 to 10 years of work.

Indeed, making the decision to leave the company in the first year can be more complicated, not to mention the **experimentation effect**. And, finally, the 10th year can **raise questions** about one's career.

### *4.2.12 Years in current role*



This swarm plot looks like graph **4.2.5** (*years spending with the same manager*) since we can distinguish 3 "floors".

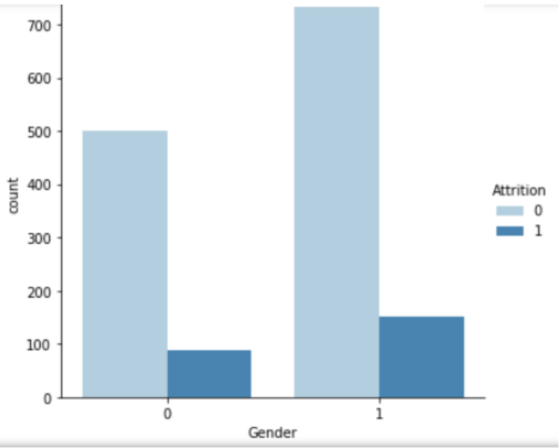
It seems normal that the number of years with the same role is linked to the number of years spent with the same manager.

The **first year** spent in a specific role is subject to attrition. Then, the phenomenon is less present but reappears in the **second year** and the **7th year**.

In any case, we see that people who **leave their job** have often been in the same job for **less than 1 year, 2 years and 7 years**.

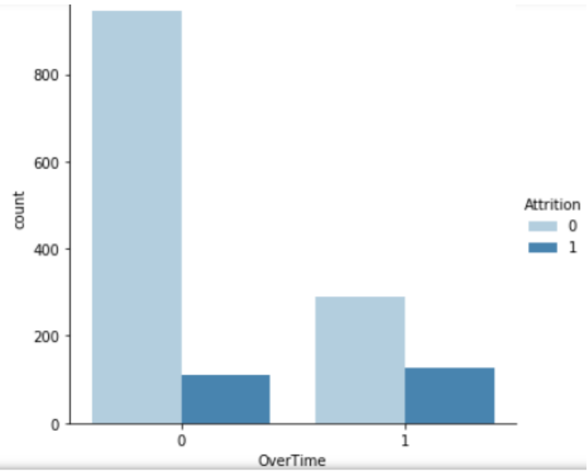
Also, after the 7 years, the employees do not seem to see at all to want to leave their job.

### *4.2.13 Gender*



There are more **men** (*class 1*) concerning by **attrition**, but they are also more **present** in the database.

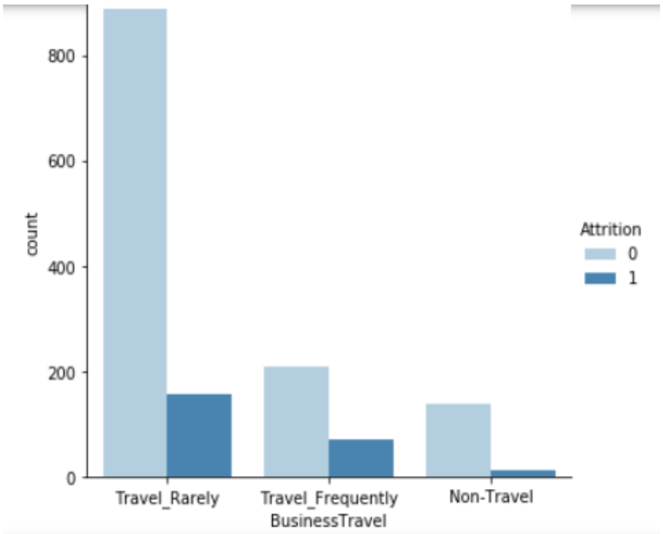
### *4.2.14 Overtime*



In the **attrition** class, there are more employees that do **overtime**, even if they are less in the database.

Therefore, working overtime could lead to attrition, probably related to **tiredness** or **stress**.

### *4.2.15 Travel*

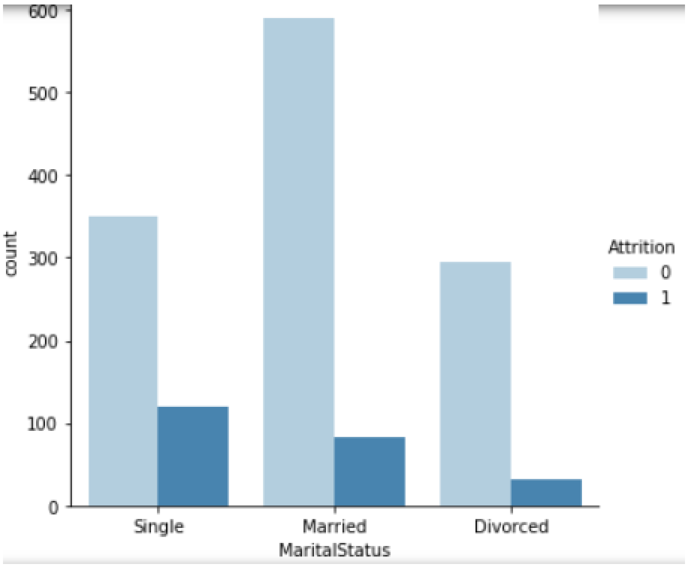


We can see that most of employees that leave their job due to attrition **travel rarely** (*but they are also more present in the database*).

If we look at the ratio *attrition vs no attrition*, it is bigger for people who **travel frequently**.

It is therefore difficult to show clearly the relationship between attrition and travel in the workplace. This will be discussed in the following sections.

### *4.2.16 Marital status*

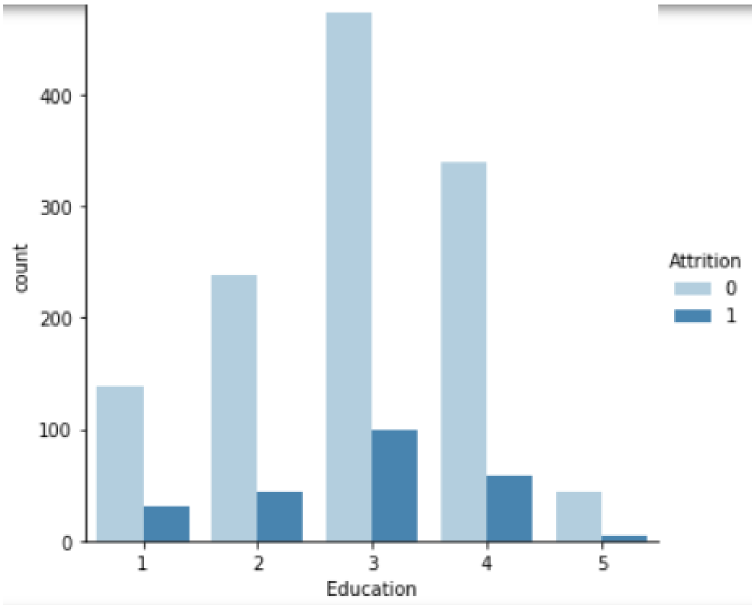


**Divorced** employees are **less concern by attrition** than single or married people.

As before, if we look at the ratio **attrition vs no attrition**, the higher is for **single people**.

This could be because they could have **less psychological support**, or that they are more able to **change jobs** or move **geographically**.

### *4.2.17 Education*



**Education**

1 'Below College'

2 'College'

3 'Bachelor'

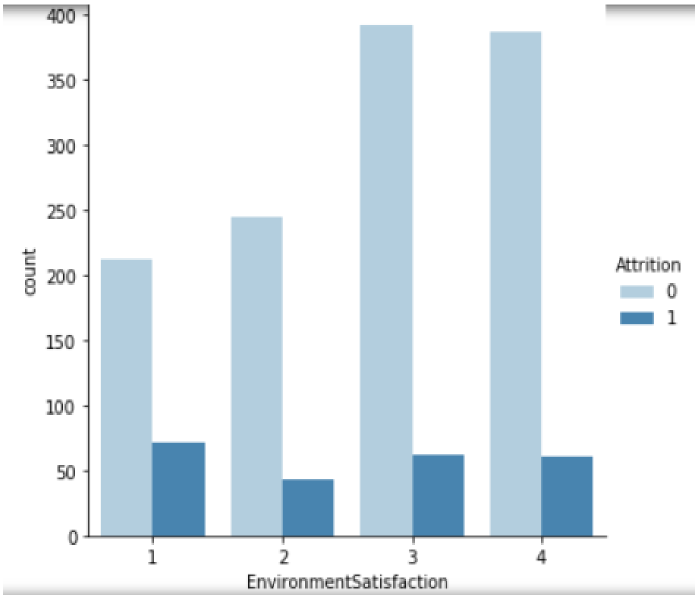
4 'Master'

5 'Doctor'

Most of employees who leave their job have a **bachelor's degree.**

Nevertheless, regarding the ratio attrition vs no attrition it is almost the same for all level of education.

### *4.2.18 Environment satisfaction*

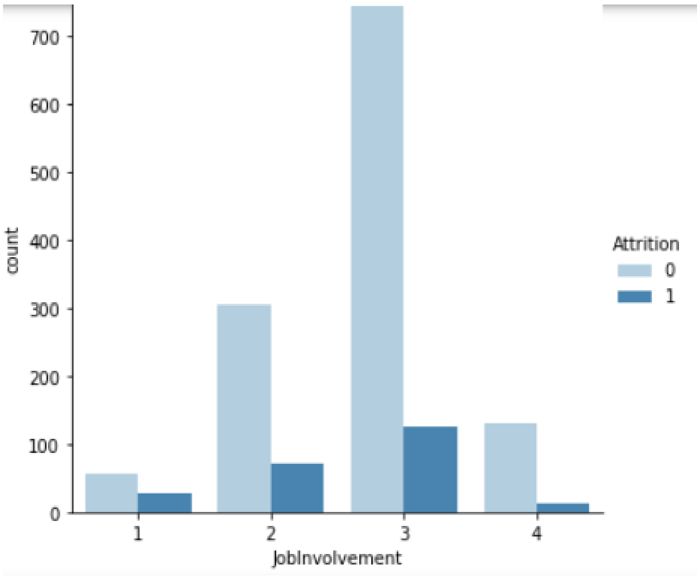


**EnvironmentSatisfaction**
1 'Low'
2 'Medium'
3 'High'
4 'Very High'

Employee that has a **low satisfaction** of their environment are more concern by **attrition**, which is not surprising.

The environment is therefore important to keep talent in the company.

### *4.2.19 Job involvement*

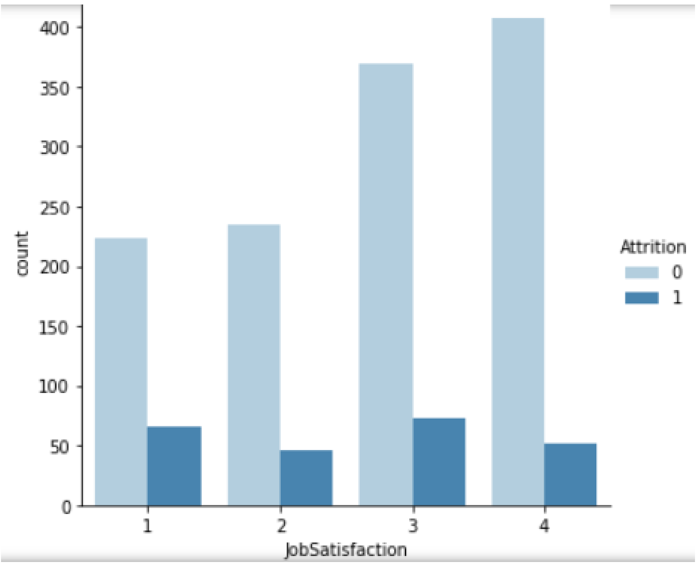


**JobInvolvement**
1 'Low'
2 'Medium'
3 'High'
4 'Very High'

Employees with a **high job involvement** are more present in the attrition class but also in the entire database.

If we look at people with a **low job involvement**, this is the category which have the most **attrition** regarding the **repartition** attrition vs no attrition.

### *4.2.20 Job satisfaction*

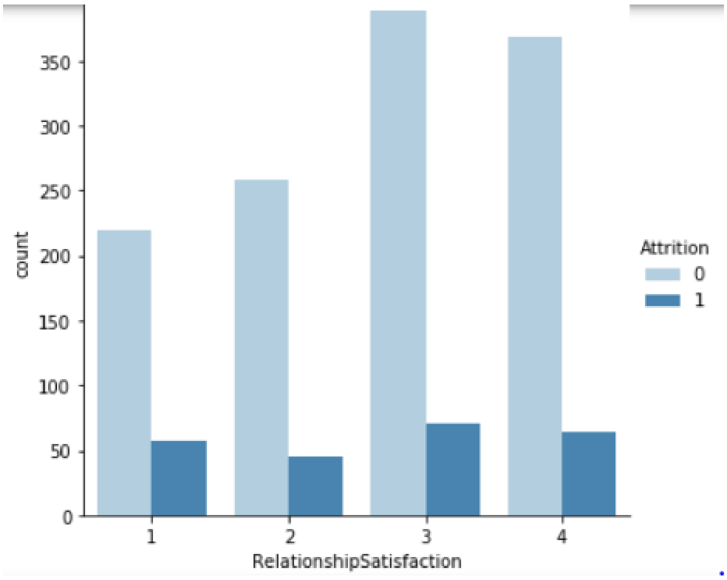


**JobSatisfaction**
1 'Low'
2 'Medium'
3 'High'
4 'Very High'

Unsurprisingly, employees who are the most likely to suffer from **attrition** are those with **low job satisfaction** in terms of attrition vs. no attrition.

Nevertheless, if we look at the employees who are affected by attrition the majority class are those who have **high job satisfaction** (*but they are also second in the database, and therefore well represented*).

### *4.2.21 Relationship satisfaction*

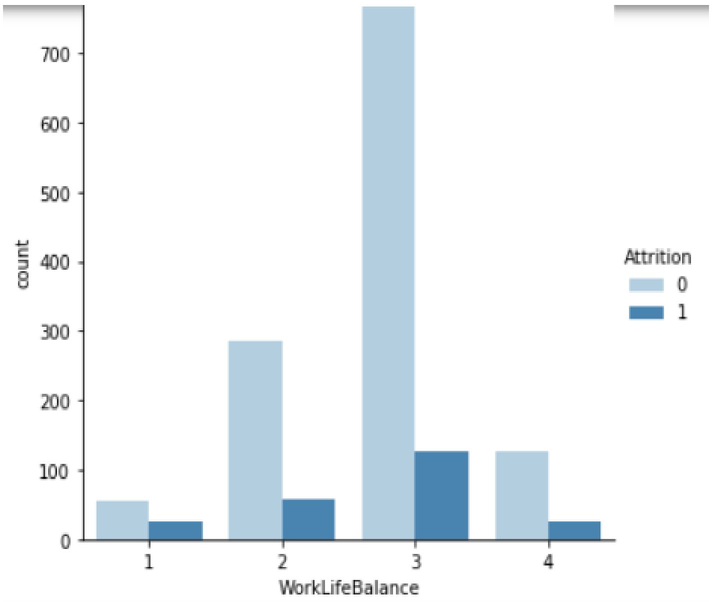


**RelationshipSatisfaction**
1 'Low'
2 'Medium'
3 'High'
4 'Very High'

The interpretation is as the job satisfaction in ***section* 4.20.**

Indeed, perhaps being satisfied with one's job means being satisfied with one's relationships at work (or vice versa).

### *4.2.22 Work life balance*

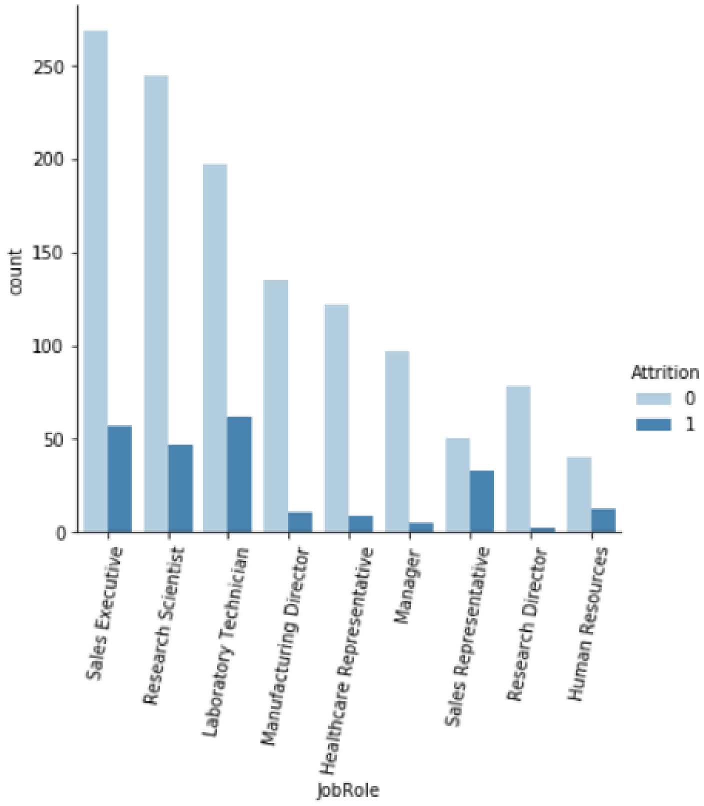


**WorkLifeBalance**
1 'Bad'
2 'Good'
3 'Better'
4 'Best'

Again, the employees who are in the majority to **quit their jobs** are those in category 3 (here, those who said their **work life balance was better)**, **but**, if we look at the **attrition vs. no attrition ratio**, it is higher **for category 1**, i.e. for people who consider that they have a **bad work life balance**.

Of course, a poor work life balance can be due to or result in **stress** or **depression**, which can impact or come from **work**.

### *4.2.23 Job role*

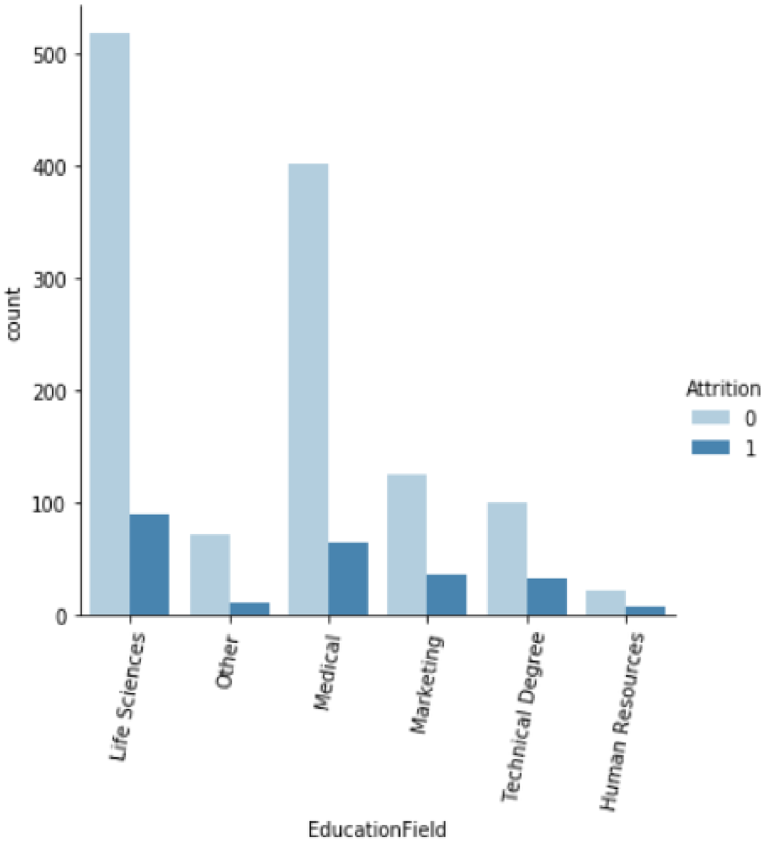


Attrition seems to concern any job role, but **laboratory technician** is the majority.

If we take our indicator: *people affected by attrition / people not affected by attrition*, then it is higher for **sales representatives.**

Finally, again using this indicator, the people who are less likely to leave their jobs are **manufacturing directors, healthcare representatives, managers,** and **research directors**.

### ***4.2.24 Education field***



This variable is necessarily somewhat related to ***Section 4.2.23 Job Role****.*

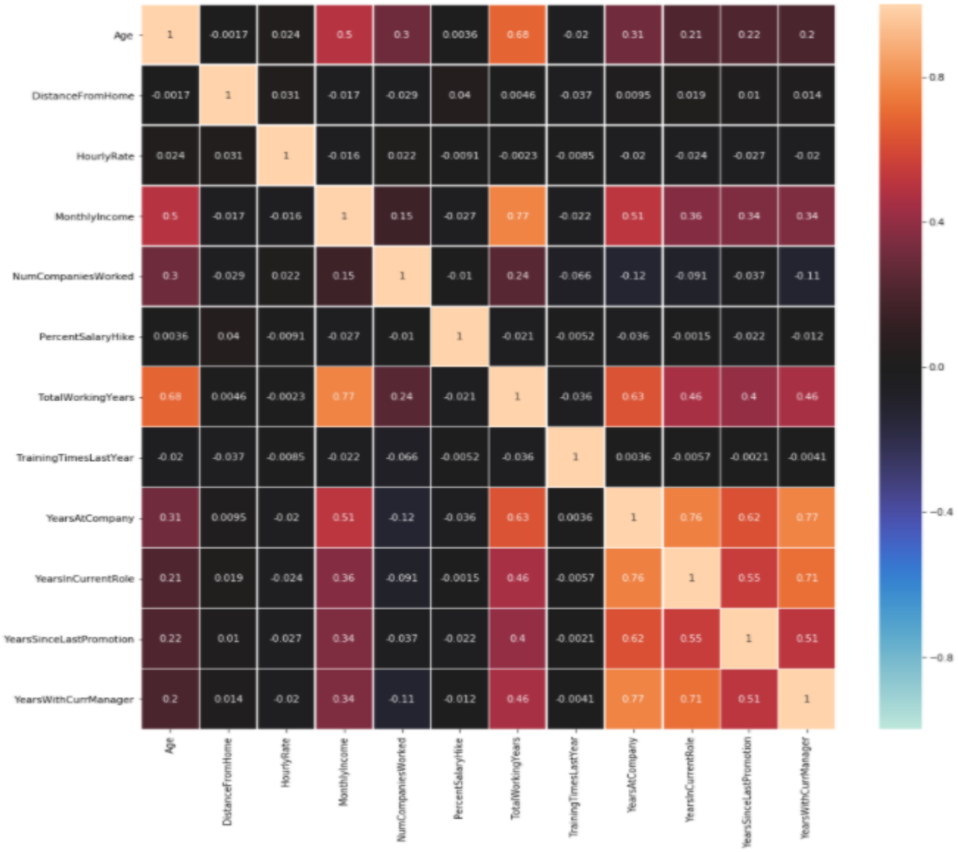
We can say here that there are more employees coming from the **education field "life science"** who are affected by attrition.

However, if we try to remove this distributional effect (*because they are the most present in the database*), we realize that it is **human resources, technical degrees** and **marketing** education field which are the most affected by **attrition** (*looking at the ratio attrition vs. no attrition*).

## 4.3 Correlations

Let us look at the correlation table for quantitative variables.

The table below is based on Pearson's Chi-Square where 0 means no association and 1 is a full association.

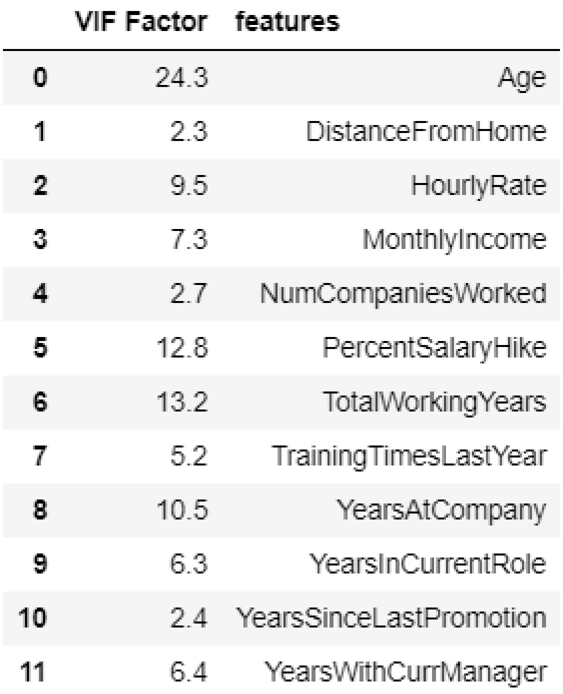


| | Age | DistanceFromHome | HourlyRate | MonthlyIncome | NumCompaniesWorked | PercentSalaryHike | TotalWorkingYears | TrainingTimesLastYear | YearsAtCompany | YearsInCurrentRole | YearsSinceLastPromotion | YearsWithCurrManager |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Age | 1 | -0.0017 | 0.024 | 0.5 | 0.3 | 0.0036 | 0.68 | -0.02 | 0.31 | 0.21 | 0.22 | 0.2 |
| DistanceFromHome | -0.0017 | 1 | 0.031 | -0.017 | -0.029 | 0.04 | 0.0046 | -0.037 | 0.0095 | 0.019 | 0.01 | 0.014 |
| HourlyRate | 0.024 | 0.031 | 1 | -0.016 | 0.022 | -0.0091 | -0.0023 | -0.0085 | -0.02 | -0.024 | -0.027 | -0.02 |
| MonthlyIncome | 0.5 | -0.017 | -0.016 | 1 | 0.15 | -0.027 | 0.77 | -0.022 | 0.51 | 0.36 | 0.34 | 0.34 |
| NumCompaniesWorked | 0.3 | -0.029 | 0.022 | 0.15 | 1 | -0.01 | 0.24 | -0.066 | -0.12 | -0.091 | -0.037 | -0.11 |
| PercentSalaryHike | 0.0036 | 0.04 | -0.0091 | -0.027 | -0.01 | 1 | -0.021 | -0.0052 | -0.036 | -0.0015 | -0.022 | -0.012 |
| TotalWorkingYears | 0.68 | 0.0046 | -0.0023 | 0.77 | 0.24 | -0.021 | 1 | -0.036 | 0.63 | 0.46 | 0.4 | 0.46 |
| TrainingTimesLastYear | -0.02 | -0.037 | -0.0085 | -0.022 | -0.066 | -0.0052 | -0.036 | 1 | 0.0036 | -0.0057 | -0.0021 | -0.0041 |
| YearsAtCompany | 0.31 | 0.0095 | -0.02 | 0.51 | -0.12 | -0.036 | 0.63 | 0.0036 | 1 | 0.76 | 0.62 | 0.77 |
| YearsInCurrentRole | 0.21 | 0.019 | -0.024 | 0.36 | -0.091 | -0.0015 | 0.46 | -0.0057 | 0.76 | 1 | 0.55 | 0.71 |
| YearsSinceLastPromotion | 0.22 | 0.01 | -0.027 | 0.34 | -0.037 | -0.022 | 0.4 | -0.0021 | 0.62 | 0.55 | 1 | 0.51 |
| YearsWithCurrManager | 0.2 | 0.014 | -0.02 | 0.34 | -0.11 | -0.012 | 0.46 | -0.0041 | 0.77 | 0.71 | 0.51 | 1 |

***<u>Let us see which are the correlations >0.5 in absolute value:</u>***

- We can mostly see a **square of highly correlated variables in the bottom left corner**: *YearsAtCompany, YearsInCurrentRole, YearsSinceLastPromotion and YearsWithCurrentManager* which seems an evidence.
  For example, spending many years with the same manager seems to be logical if employees spend many years in the company or in their current role and vice versa.
- *TotalWorkingYears* is strongly correlated with other variables such as *Age, MonthlyIncome and YearsAtCompany (for the same reasons then above).*
  Indeed, if the employee has a high monthly income he will potentially want to stay in the company and vice versa: if the employee has many years of seniority it is normal that his salary is more substantial.
  Also, it seems clear that more an employee has spent years in a company, more likely older he is.

***In order to predict multicollinearity, it is more relevant to do a VIF test (Variance inflation factor):***



|  | VIF Factor | features |
|---|---|---|
| **0** | 24.3 | Age |
| **1** | 2.3 | DistanceFromHome |
| **2** | 9.5 | HourlyRate |
| **3** | 7.3 | MonthlyIncome |
| **4** | 2.7 | NumCompaniesWorked |
| **5** | 12.8 | PercentSalaryHike |
| **6** | 13.2 | TotalWorkingYears |
| **7** | 5.2 | TrainingTimesLastYear |
| **8** | 10.5 | YearsAtCompany |
| **9** | 6.3 | YearsInCurrentRole |
| **10** | 2.4 | YearsSinceLastPromotion |
| **11** | 6.4 | YearsWithCurrManager |

The **numerical value of the VIF** is the **percentage the variance** (*i.e the standard error squared*) **is inflated for each coefficien**t.

*For example, the variance of the coefficient DistanceFromHome is 103% (203%-100%) bigger than what we would have expected if there was no multicollinearity (if there was no correlations with other predictors).*

**A rule of thumb for the variance inflation factor is:**

- 1: not correlated.
- Between 1 and 5: moderately correlated.
- Greater than 5: highly correlated.

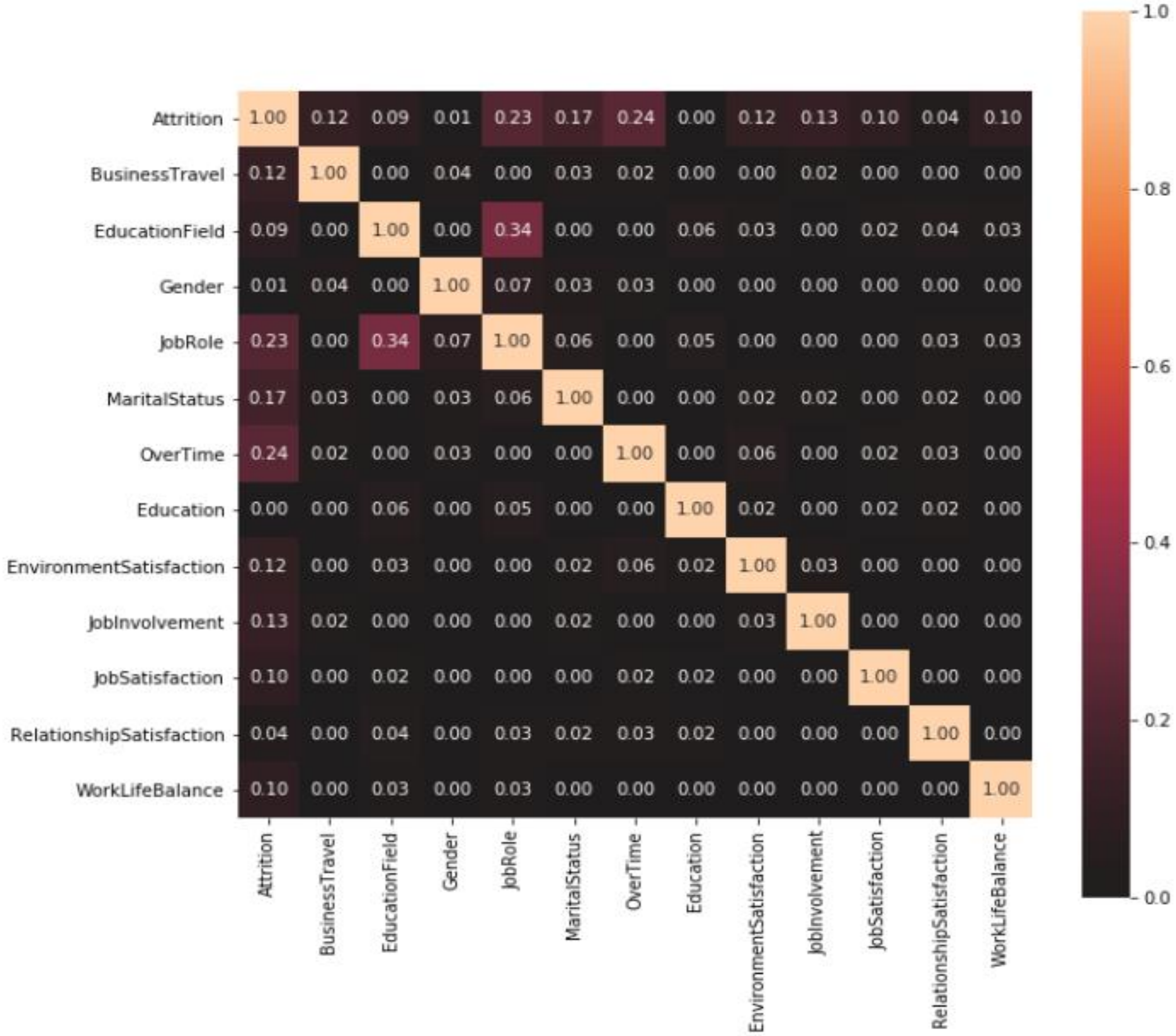
Exactly how large a VIF must be before it causes issues is a subject of debate. But what is known is that more the VIF increases, the less reliable your regression results are going to be (**but in general, a VIF above 10 indicates high correlation and is cause for concern**).

We can see that ***Age, PercentSalaryHike*** *and* ***TotalWorkingYears*** are variables in this case.

Subsequently, we **standardize** the variables and we no longer found a VIF greater than 10 (***all VIF <5***).
In addition, you will find in ***section 5*** models that allow us to **select the variables**.

Regarding the qualitative variables, let us look at the ***Cramer's V*** method: it is based on a **nominal variation of Pearson's Chi-Square Test** (*as before, 0 means no association and 1 is full association*).

***<u>Note</u>****: Cramer's V is symmetrical: it is insensitive to swapping variable x1 and x2.*



There are no high associations here (all are <|0.35|).

We also check the associations with **Teil's U method**: it is an ***asymmetric measure of association between categorical features***: *<u>given the value of x1, how many possible states does x2 have, and how often do they occur?</u>*

Just like Cramer's V, the output value is on the range of [0,1], with the same interpretations as before — but unlike Cramer's V, it is **asymmetric**, meaning that knowing x1 means we know x2, but not vice-versa.

We obtain that all associations were **below |0.15|**, meaning that qualitative variables seem to be unrelated.

## 4.4 Conclusion

***To conclude this section of descriptive statistics, we learned that:***

- Attrition is for employees between **28 and 32 years old.**
- Attrition is larger for employee with a monthly **income < $3000**.
- Employees who receive a **higher hike** percentage have more chances to stay in the company.
- In the **attrition** class, employees who live **near** their place of work are more present.
- Attrition concerning variables related to years such as **years with the same manager, years in the current role, years since the last promotion, years in the company** are higher in the lowest part but work in "floors".
- There is a **slight decrease in attrition** from **2 companies** where the individual has worked.
- We notice that those who received the **least training** (*0 or 1 week*) or, on the contrary, those who received a lot of training (*5 or 6 weeks*) seem to be more **satisfied** with their jobs than those who received *2 to 4 weeks of training*.
- Working **overtime** could lead to attrition.
- If we look at the ratio **attrition vs no attrition**, the higher is for **single people** who **travel frequently**, with a **low relationship satisfaction, job satisfaction, job involvement, environment satisfaction and a bad work life balance**. **Human resources, technical degrees** and **marketing** education field are the most affected by attrition, and, more specifically it is higher for **sales representatives.**
- There are big correlations concerning ***YearsAtCompany, YearsInCurrentRole, YearsSinceLastPromotion, YearsWithCurrentManager, TotalWorkingYears, Age*** *and* ***MonthlyIncome.***
- There are no associations between qualitative variables.

# 5 Econometric models

In econometric analysis, when one estimates an **OLS regression**, one must be careful about the **number of features**.

Actually, if the number of variables (p) is equal, higher or very near to the number of observations (n), then an OLS regression will not work.

Assuming that the true relationship between the response and the predictors is **approximately linear**, the least squares estimates will have **low bias**.

However, if **n is not far from p**, the **variability** in the least squares fit will be **large** and will cause **poor accuracy**.

Also, if **p > n**, estimates' variances are **infinite** so the method cannot be used.

It means that, if one increases the number of features such p ≈ n, one could have less errors on the training sample, but it does not mean that the model provides better forecasts: *this is the problem of* ***over-fitting***.

In our case, the aim of our framework is to provide some **classification**, so we will not use OLS regression.

However, we will consider these issues and deal with overfitting.

We start our analysis with a table of data that contains **52 features**: *14 quantitative and 38 binaries*, with the reference dummies dropped.

We are going to select the **most important quantitative ones** with ***Principal Component Analysis (PCA),*** then, implement a ***Forward Stepwise Selection*** and select the best model according to four ***Generalized Information Criteria (AIC, BIC, Cp and Adjusted R-squared)***. Then, we will create new features to consider **non-linearities** and use selection models to keep the most important ones.

## 5.1 Initial variables selection

### *5.1.1 Principal Components Analysis (PCA)*

The **Principal Component Analysis (PCA)** is a mathematical technique to rewrite a **complex system of correlations** thanks to **linear combinations** of the variables.

The aim is to **reduce the information** (*p variables into m << p factors*).

In most cases, these linear combinations are **interpretable**.

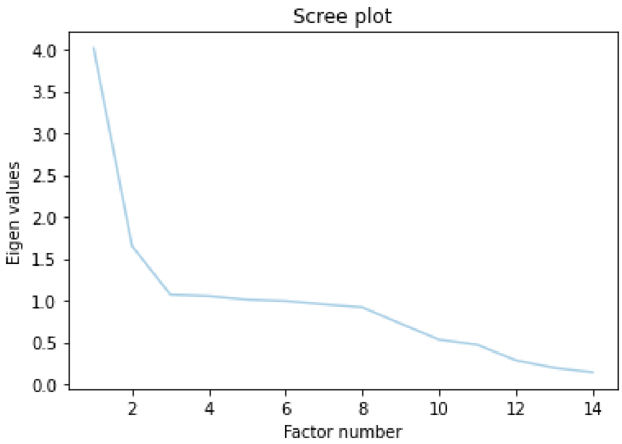
This analysis chooses some **weights** such that *the first linear combination maximizes the variance of the Principal Components (PC)*, such that *the linear combination has the maximum variability in the direction of the correlation*, such that *X's are uncorrelated*.

PCs are **special linear combinations of p variables X1, X2, ..., Xp**. They depend on the **variance-covariance matrix** (*or correlation matrix p*) and on **eigenvalues**.

If one variable as a bigger scale than the others, then naturally, the first component will be too much weighted. **To solve it, one applies the PCA on the correlation matrix** because it is less sensitive to the variance of the variables.

Applicated to our data, we will compute PCs from **quantitative variables**, *14 variables*, based on the **covariance matrix** since we have already **rescaled** the data.

First, once the principal components are calculated, we can check whether the PCA works well on the data. To do so, we can display a **scree plot** that displays the **eigenvalues** depending on the factor number.

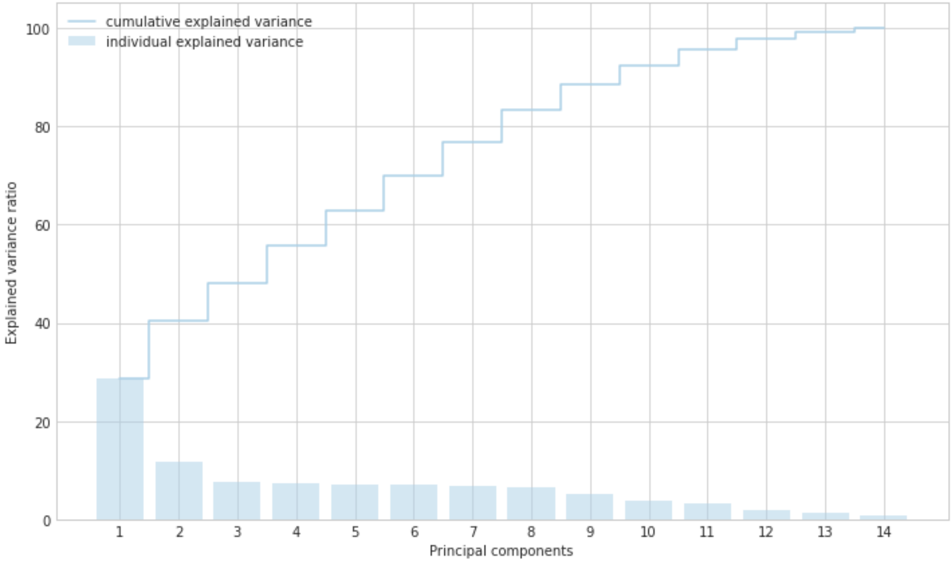


We can observe that the curve is **decreasing** very rapidly from 1 to 8 factor numbers.

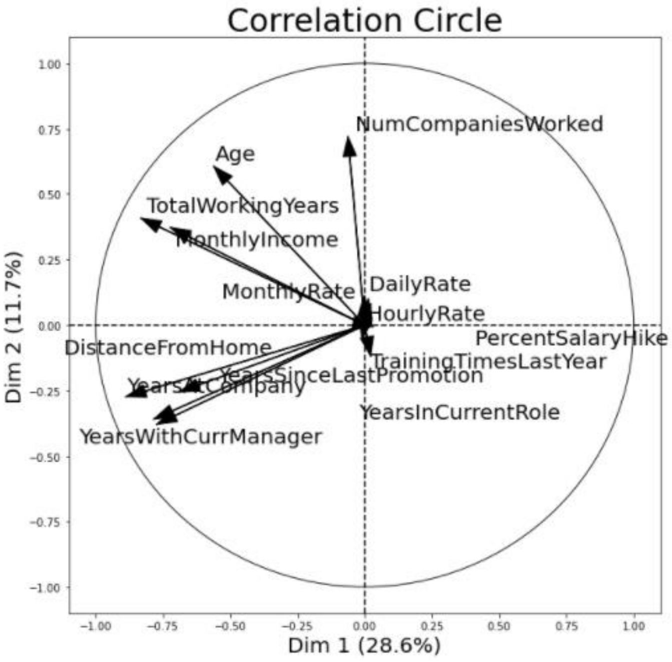
This shows that *the first principal components catch most of the variation.*

Hence, the PCA is working well with our data.

Moreover, having a look at the **<u>cumulative sum of explained variances of each PC</u>** is a first accurate information: we find out that with the **9 first principal components**, we catch up about **88.45% of the information contained by the 14 variables**. We can visualize it on the <u>following plot</u>.



This graph shows that we can use the 9 first principal components to bring to the model almost all the information brought by our 14 initial variables: it can **reduce the number of variables in the model by only using these 9 PCs**.



We can also check *<u>how each principal component is correlated to the variables</u>*, in the aim to **isolate groups** of variables and better understand the information contained in each principal component.

The graph on the left-and-side is an example, it shows the **correlation between the variable** as well as the **first, and second principal components.**

We can see that some variables are very **poorly correlated** to the **first two PCs (***Hourly Rate, Daily Rate, Training Times Last Year, etc.*).

Moreover, two groups are a kind of obvious: variables that are **highly correlated with PC1** (*Years with Current Manager, Years At Company, Monthly Income, Total Working Years, etc.)* and those that are highly correlated with **PC2** (*Age and Number of Companies Worked*).

Finally, we decide to replace **all our quantitative variables by the nine first components**, to reduce the number of features but to keep the maximum of information.

We are now dealing with a table of **47 variables**.

For now, we want to select the best model. To do so, we can use the ***Best Subset Selection (BSS)***, the ***Forward Stepwise Selection (FSS)*** and the ***Backward Stepwise Selection (BackSS)***. *However, as BSS is too slow and BackSS is almost the same as FSS, we have chosen to use FSS.*

### *5.1.2 Forward Stepwise Selection (FSS)*

As said, previously***, BSS, FSS*** and ***BackSS*** are three algorithms that can operate **variables selection**.

Concerning **BSS**, the aim is to *make all possible models and select the best one depending on criteria*. There are 3 steps:

- **1) Build the simplest model**: no predictors (*predict the sample mean for each observation*).
- 2) **For each value of k** (the number of predictors), fit all the **possible models** (*each model is a different combination of predictors for a given value of k*) and pick the best one.
- 3) Select the **best among each best model** for each value of k by using ***Generalized Information Criteria (GIC)*** for example.

However, this technique can be very **slow**.

In our case, we have **47 features**, that implies that we will build **$2^{47}$ models** = **140 737 488 355 328 models...** *Hence, we cannot apply this method.*

With the **FSS**, we do not create and evaluate all the models. It is less heavy and quicker, but it can fail.

*For example, if the best model with one variable is M1 = X1 and the best model with two variables is M2 = X2 + X3, it failed because if X1 is selected by the first model, it keeps on being in other models.*

The main point of this algorithm is: ***for k = 0,1, ..., N-1,*** consider all **N-k models** that **augment the predictors in Mk** with one **additional** predictor and choose the best among these N-k models.

The best one is the one that has the **smallest RSS or highest R-squared**.

Then, we select a **single best model** among these ones (with **GIC** for example).

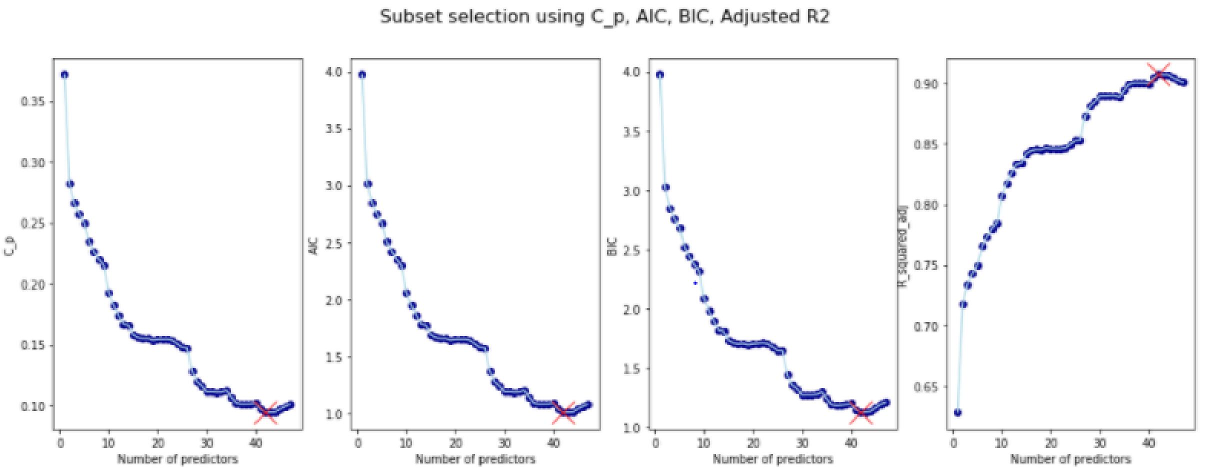
Finally, the idea of the **BackSS** is to start with the **full model** and at each iteration and remove the least statistically significant variables.

We do not apply it to our database because it has the same advantages as **FSS** (*it is quite fast and accurate*) and the same disadvantages (*it can fail when for example, the best model for k = 2 does not contain the variable used in the best model for k = 1*).

After having run the **FSS with logistic models** (*trained on a train set determined by the previously created dataframe and treated with the SMOTE function to get balanced data*), we find out that, the model with **42 variables is the best**, according to the **R-squared and the Residuals Sum of Squares (RSS)**.

However, we cannot just trust the RSS and the R-squared because more variables we have in the model, the higher R-squared and smaller the RSS are. We need to find other criteria that **penalizes for the number of variables**: we have chosen ***AIC, BIC, Cp and Adjusted R-squared***.

It seems that the best model for the 4 criteria contains **42 variables**, as shown below:



Finally, we modify the data frame previously created by **dropping five variables** that have been excluded by the **FSS**. With the new one, we create other train set and test set by using **SMOTE**, that we will use for the following models.

## 5.2 Estimating econometrics models and additional variables selection.

We will see that it is important to take **non-linearities** into account, as soon as we cannot be sur that the relation between the **target variable** and each **quantitative feature** is linear.

Moreover, each model's **hyperparameter** that we are going to estimate (*excluding the basic logistic regression because it does not have hyperparameter*) is determined by an optimisation process.

Actually, for a given model (*for example LASSO*), we evaluate its accuracy by a **10-fold cross-validation** for several value of its hyperparameter(s) and we pick up the best one to estimate our model applied on our data.

### *5.2.1 Taking into account non-linearities.*

**Non-linearities** are an issue which is complicated to deal with.

We must wonder if there is a perfect linear relation between these quantitative variables and the target variable. Indeed, it is very rare, so we have to check it and, if it is not the case, take into account non-linearities.

We have to consider 6 questions:

- How do we decide what order of polynomial to try to fit?
- Do we need to include *cross-coupling terms* for multivariate regression?
- Is there an easy way to automate the process?
- How to ensure we do not overfit to the data?
- Is our machine learning model robust against measurement noise?

A solution is **to look at the plots** of each feature with the target variable and determine whether there is a *linear relationship*. But as soon as we have a lot of variables, and we found this solution a little bit unprecise, we aimed to work differently.

To cleverly add variables that catch non-linearities without overfitting the data, we can use the **Lasso Regression** and **Elastic Net**.

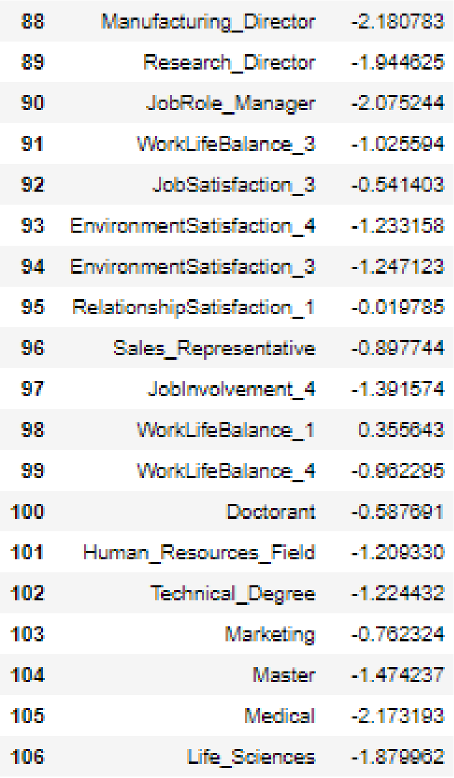
Hence, we have created new features by creating variables by elevating all our quantitative variables at the power 2 and 3 and cross all of them (our quantitative variables are PC1, PC2, PC3, PC5, PC7 and PC9, selected previously by FSS).

Next, we will add these new variables to the dataframe created in the 5.1.2 section and we will apply a **logistic regression** with the **l1 penalty term** *(a Lasso method on a logistic regression)* to select variables that are the most useful. Hence, we will do the same with Elastic Net and compare both approaches and select the best ones.

By this way, we could have got a model that considered a lot of non-linearities without overfitting, thanks to the penalization.

However, we are going to build a **logistic regression** and a **Ridge logistic regression** with all the variables we would have created as well, to compare their results to the other models' ones.

### *5.2.2 Basic Logistic Regression*



| | | |
|---|---|---|
| 88 | Manufacturing_Director | -2.180783 |
| 89 | Research_Director | -1.944625 |
| 90 | JobRole_Manager | -2.075244 |
| 91 | WorkLifeBalance_3 | -1.025594 |
| 92 | JobSatisfaction_3 | -0.541403 |
| 93 | EnvironmentSatisfaction_4 | -1.233158 |
| 94 | EnvironmentSatisfaction_3 | -1.247123 |
| 95 | RelationshipSatisfaction_1 | -0.019785 |
| 96 | Sales_Representative | -0.897744 |
| 97 | JobInvolvement_4 | -1.391574 |
| 98 | WorkLifeBalance_1 | 0.355643 |
| 99 | WorkLifeBalance_4 | -0.962295 |
| 100 | Doctorant | -0.587691 |
| 101 | Human_Resources_Field | -1.209330 |
| 102 | Technical_Degree | -1.224432 |
| 103 | Marketing | -0.762324 |
| 104 | Master | -1.474237 |
| 105 | Medical | -2.173193 |
| 106 | Life_Sciences | -1.879962 |

Logistic regression is one of the most popular classification algorithms. It attributes a coefficient to each feature depending on how it increases the probability of **y = 1**.

On the right, we can observe a part of our the most important coefficients (*we did not display the 119 coefficients, you can find them in the notebook*).

The odds of leaving the company when the employee is a manufacturing director is exp(-2.18) = 0.11, compared to an employee working in human resources.

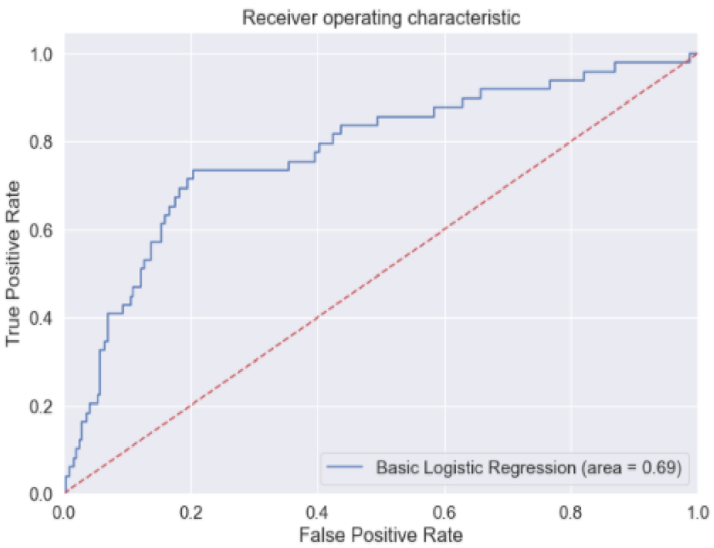
The out-sample accuracy of this model is about **81.63%,** that is a good score.

The confusion matrix tells us that the model has predicted 215 non-attritions while it was really non-attrition but also only 25 attritions when it was really attrition. Actually, it does not accurately classify labels 1, as shown in the following table.

- The **precision** is the ratio of correctly predicted positive observations to all observations classified as positive: **TruePositive/TruePositive+FalsePositive**.
- The **recall** also called sensitivity is the ratio of correctly predicted positive observations to all positive observations: **TruePositive/TruePositive+FalseNegative.**
- The **F1 score** is the weighted average of precision and recall **2*(Recall*Precision)/(Recall+Precision).**

| | Precision | Recall | F1 score |
|---|---|---|---|
| **No - 0** | 0.90 | 0.88 | 0.89 |
| **Yes - 1** | 0.45 | 0.51 | 0.48 |
| | | | |
| **Weighted avg** | 0.83 | 0.82 | 0.82 |

- **Precision**: ratio of attrition correctly predicted to all observations classified as attrition is about **45%**, the same ratio is better for no attrition (**45%**).
- **Recall:** it is the same phenomenon for the recall. Ratio of attrition correctly predicted to all attrition observations is about **51%**, while the same ratio is better for no attrition (**88%**).
- So we see with the **F1 score**, that the model classifies better the non-attrition class rather than the attrition class. On average, the F1 score is **82%**.



According to the **ROC curve**, the area between the basic logistic regression prediction's curve (in blue) and the random classification (*in red, where the true positive rate equals the false positive rate*) is equal to **0.69**. Thus, we can say that the classification produced by our model is far from being random.

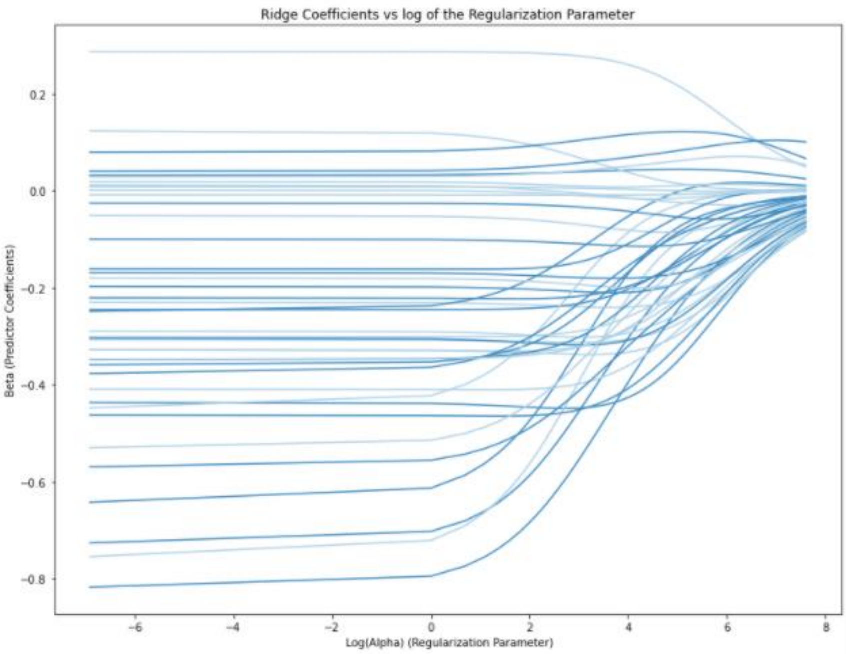
### *5.2.3 Ridge model with Logistic Regression*

Ridge regression method is one of the so-called "***shrinkage methods***", which is usually applied to a regression model when there is instability resulting from **collinearity** of predictors.

When the predictors are collinear or almost collinear, the matrix $X' * X$ becomes singular (*rarely the case*) or almost singular, then the inverse would respond sensitively to errors, which results in instability of prediction with such a model.

Ridge regression, however, make a **trade-off between bias and variance** in prediction. By introducing a relatively small bias, you may expect a large reduction in the variance, and thus in the mean-squared error.

More precisely, the **Ridge Regression** is expressed as the **logistic regression** in our case with the addition of a penalty term called ***L2 regularization***.



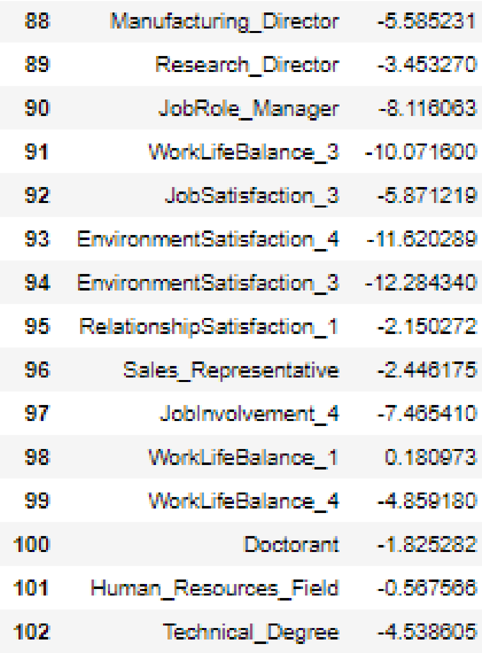
The L2 term is equal to the **square of the magnitude** of the coefficients.

The **hyperparameter alpha** determines the strength of the shrinkage towards zero of the coefficients: the larger lambda is, the more the coefficients are shrinked towards zero (*see on the right an example of the evolution of the coefficient depending on the log of the hyperparameter alpha, with the dataframe with only 42 variables*).

This is by this way that we reduce the **variance** and grow the **bias**.

Of course, every coefficient will be non-zero, so it is not a selection model, we keep the same number of variables and we just **shrink** their effect.

It strikes that the Ridge's coefficients are higher than the logistic regression's one.



| | | |
|---|---|---|
| 88 | Manufacturing_Director | -5.585231 |
| 89 | Research_Director | -3.453270 |
| 90 | JobRole_Manager | -8.116063 |
| 91 | WorkLifeBalance_3 | -10.071600 |
| 92 | JobSatisfaction_3 | -5.871219 |
| 93 | EnvironmentSatisfaction_4 | -11.620289 |
| 94 | EnvironmentSatisfaction_3 | -12.284340 |
| 95 | RelationshipSatisfaction_1 | -2.150272 |
| 96 | Sales_Representative | -2.446175 |
| 97 | JobInvolvement_4 | -7.465410 |
| 98 | WorkLifeBalance_1 | 0.180973 |
| 99 | WorkLifeBalance_4 | -4.859180 |
| 100 | Doctorant | -1.825282 |
| 101 | Human_Resources_Field | -0.567566 |
| 102 | Technical_Degree | -4.538605 |

To take the same example as previously, the odds of leaving the company when the employee is a manufacturing director is exp(-5.59) = **0.003,** compared to an employee working in human resources.

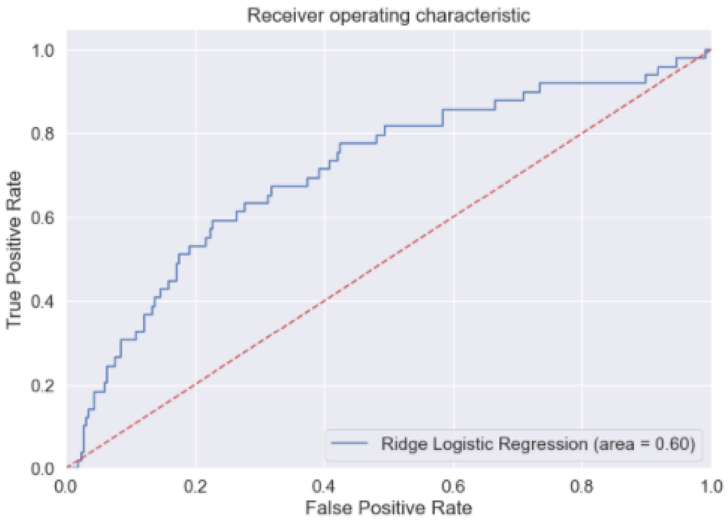
The out-sample accuracy of the Ridge model is **78.91%,** the same as the basic logistic model's one.

The confusion matrix tells us that the model has predicted **216** non-attritions while it was really non-attrition but also only 16 attritions when it was really attrition.

Actually, it does not accurately classify labels 1, as shown in the following table.

| | **Precision** | **Recall** | **F1 score** |
|---|---|---|---|
| **No - 0** | 0.87 | 0.88 | 0.87 |
| **Yes - 1** | 0.36 | 0.33 | 0.34 |
| | | | |
| **Weighted avg** | 0.78 | 0.79 | 0.79 |

- **Precision**: ratio of attrition correctly predicted to all observations classified as attrition is about **36%**, the same ratio is better for no attrition (**87%**).
- **Recall:** it is the same phenomenon for the recall. Ratio of attrition correctly predicted to all attrition observations is about **33%**, while the same ratio is better for no attrition (**88%**).
- So we see with the **F1 score**, that the model classifies better the non-attrition class rather than the attrition class. On average, the F1 score is **79%**.



According to the ROC curve, the area between the Ridge with logistic regression prediction's curve (in blue) and the random classification (in red) is equal to **0.60**. Thus, we can say that the classification produced by our model is far from being random.

### *5.2.4 Lasso model with Logistic Regression*

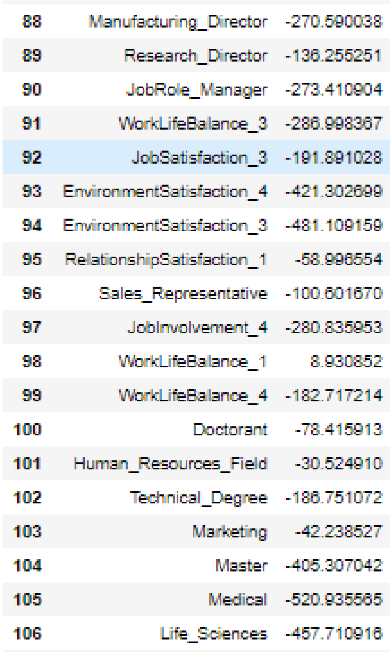
The **Least Absolute Shrinkage and Selection Operator (LASSO)** regression is quite similar to the Ridge regression, but it does not have the same penalty term: it is called the ***L1 penalty term***.

Similar to Ridge regression, an alpha value of zero spits out the basic logistic equation, however given a suitable alpha value Lasso regression.

It can converge coefficients to **zero** so it is a selection model.

However, it performs badly when variables are **colinear**, that is the case for our quantitative variables because they are **principal components**.

Lasso logistic regression selects 114 variables.



| | | |
|---|---|---|
| **88** | Manufacturing_Director | -270.590038 |
| **89** | Research_Director | -136.255251 |
| **90** | JobRole_Manager | -273.410904 |
| **91** | WorkLifeBalance_3 | -286.998367 |
| **92** | JobSatisfaction_3 | -191.891028 |
| **93** | EnvironmentSatisfaction_4 | -421.302699 |
| **94** | EnvironmentSatisfaction_3 | -481.109159 |
| **95** | RelationshipSatisfaction_1 | -58.996554 |
| **96** | Sales_Representative | -100.601670 |
| **97** | JobInvolvement_4 | -280.835953 |
| **98** | WorkLifeBalance_1 | 8.930852 |
| **99** | WorkLifeBalance_4 | -182.717214 |
| **100** | Doctorant | -78.415913 |
| **101** | Human_Resources_Field | -30.524910 |
| **102** | Technical_Degree | -186.751072 |
| **103** | Marketing | -42.238527 |
| **104** | Master | -405.307042 |
| **105** | Medical | -520.935565 |
| **106** | Life_Sciences | -457.710916 |

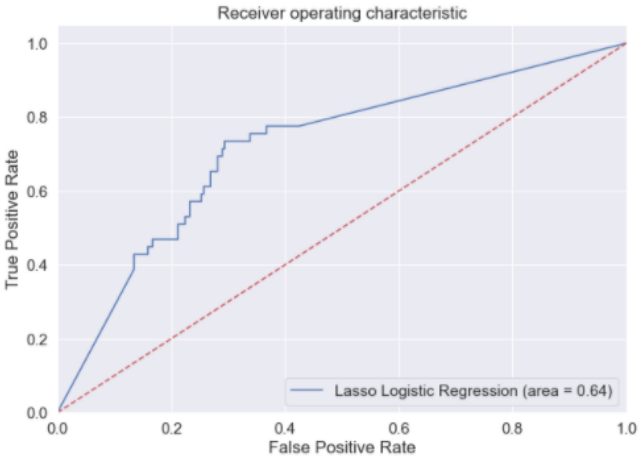
The Lasso's coefficients are very large.

The odds of leave the company when the employee is a manufacturing director is exp(-270.6) = 0.00000…, compared to an employee working in human resources.

We see that Lasso has severely penalized the coefficients.

Moreover, this model is interesting because it is quite accurate (**78.57%** of out-sample accuracy) and uses less information than the basic logistic regression.

Looking at the confusion matrix, this model predicts **210 non-attrition** while they are really non-attritions and only **21 attritions** while they are really attrition.



| | Precision | Recall | F1 score |
|---|---|---|---|
| **No - 0** | 0.88 | 0.86 | 0.87 |
| **Yes - 1** | 0.38 | 0.43 | 0.40 |
| | | | |
| **Weighted avg** | 0.80 | 0.79 | 0.79 |

- **Precision**: ratio of attrition correctly predicted to all observations classified as attrition is about **38%**, the same ratio is better for no attrition (**88%**).
- **Recall:** it is the same phenomenon for the recall. Ratio of attrition correctly predicted to all attrition observations is about **43%**, while the same ratio is better for no attrition (**86%**).
- So we see with the **F1 score**, that the model classifies better the non-attrition class rather than the attrition class. On average, the F1 score is **79%**.

According to the ROC curve, the area between the LASSO logistic regression prediction's curve (in blue) and the random classification (in red) is equal to **0.64**.

Thus, we can say that the classification produced by our model is far from being random. However, it is less than the two first models, while this model is said to be more accurate on the test set.

### 5.2.5 Elastic Net model with Logistic Regression

**Elastic Net** allows us to set and choose an **alpha value**, and to tune the alpha parameter where alpha = 0 corresponds to Ridge and alpha = 1 to Lasso.

- If **alpha = 0**, the penalty function reduces to the **L1 (Ridge) term**.
- If **alpha = 1**, the penalty function reduces to the **L2 (Lasso) term**.

Therefore, we can choose an alpha value between 0 and 1 to optimize the Elastic Net.

Actually, this will shrink some coefficients and set some to 0 for sparse selection.

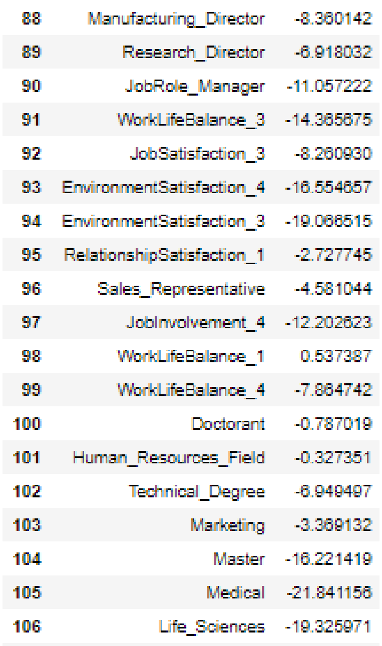
The Elastic Net can be a naive Elastic Net if there is a double shrinkage effect.

There are two steps in the process of Elastic Net:

- The "Lasso part" (L1 term) select variables.
- The "Ridge part" (L2 term) improves the prediction by stabilizing the way that highly correlated variables are used.

This method is a perfect ***trade-off between the Lasso and the Ridge regression***: we merge the efficiency of Lasso in selecting accurate variables and the stabilizing nature of Ridge regression.

Elastic Net solves the problem of Lasso: no more issues with groups of highly correlated variable.



| | | |
|---|---|---|
| 88 | Manufacturing_Director | -8.360142 |
| 89 | Research_Director | -6.918032 |
| 90 | JobRole_Manager | -11.057222 |
| 91 | WorkLifeBalance_3 | -14.365675 |
| 92 | JobSatisfaction_3 | -8.260930 |
| 93 | EnvironmentSatisfaction_4 | -16.554657 |
| 94 | EnvironmentSatisfaction_3 | -19.066515 |
| 95 | RelationshipSatisfaction_1 | -2.727745 |
| 96 | Sales_Representative | -4.581044 |
| 97 | JobInvolvement_4 | -12.202623 |
| 98 | WorkLifeBalance_1 | 0.537387 |
| 99 | WorkLifeBalance_4 | -7.864742 |
| 100 | Doctorant | -0.787019 |
| 101 | Human_Resources_Field | -0.327351 |
| 102 | Technical_Degree | -6.949497 |
| 103 | Marketing | -3.369132 |
| 104 | Master | -16.221419 |
| 105 | Medical | -21.841156 |
| 106 | Life_Sciences | -19.325971 |

The coefficients are less high than Lasso's ones.

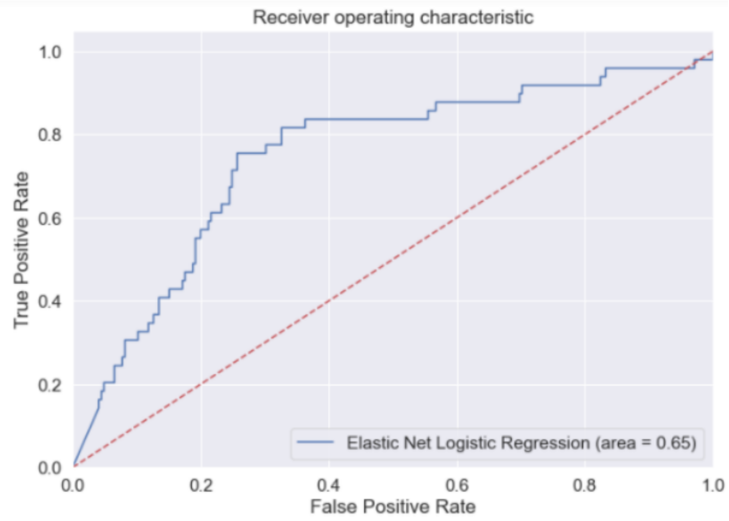
To take the same example as the beginning, the odds of leaving the company when the employee is a manufacturing director is exp(-8.36) = **0.00023**, compared to an employee working in human resources.

The out-sample accuracy is **75.85%** and the model has selected **94 variables**.

Its confusion matrix shows us that this model predicts **200** non-attrition while they are really non-attritions and only **23** attritions while they are really attrition.

| | Precision | Recall | F1 score |
|---|---|---|---|
| **No - 0** | 0.88 | 0.82 | 0.85 |
| **Yes - 1** | 0.34 | 0.47 | 0.39 |
| | | | |
| **Weighted avg** | 0.79 | 0.76 | 0.77 |

- **Precision**: ratio of attrition correctly predicted to all observations classified as attrition is about **34%**, the same ratio is better for no attrition (**88%**).
- **Recall:** it is the same phenomenon for the recall. Ratio of attrition correctly predicted to all attrition observations is about **47%**, while the same ratio is better for no attrition (**82%**).
- So we see with the **F1 score**, that the model classifies better the non-attrition class rather than the attrition class. On average, the F1 score is **77%**.



This model is less accurate than the others, and this is also shown by the ROC curve: the classification produced by our model is far from being random, and the area is equal to **0.65.** However, it is the worst score.

## 5.3 Models comparison

To sum-up the advantages and disadvantages of all the methods we can say that **PCA is** a well-established mathematical technique for reducing the **dimensionality** of data, while keeping as much **variation** as possible.

It allows us to **remove correlated features** (*improves algorithm performance and reduce overfitting*) and improves **visualization** of the data in high dimension.

But Principal Components are not as readable and interpretable as original features, the data must be standardized, and we need to select the number of PC with care in order to do not loose information.

**BSS** makes all possible model, so we are sure to pick the **best choice**. But it is extremely long to run/execute the algorithm due to the advantage above.

**FSS and BackSS** are much **quicker** than BSS but can **fail** because of the variables that are stuck in each model.

**Ridge Model** improves the prediction by stabilizing the way that highly correlated variables are used, but do not select variables (*only a shrinkage effect*).

**Lasso** is efficient in selecting accurate variables but performs poorly when variables are **colinear**.

**Elastic Net** is the ***trade-off between Ridge and Lasso*** (*selection + no more issue of highly correlated variable*). One disadvantage is the **computational cost.**

**Comparing our 5 models' results, we can observe that:**

- The basic **logistic regression** has the best **out-sample accuracy** and the larger area under its **ROC** curve. This is the model that predicts the best both categories. It could mean that we did not need to select variables. The model is accurate and does not overfit the data.
- The **Ridge and Lasso regressions** have almost the same out-sample accuracy (*Ridge is a little bit better*) and Lasso model has selected **113 variables over 119**.
- The **Elastic Net logistic regression** is the **less accurate** model out of sample, but it selected less variables: **94**.

# 6 Machine learning methods

## 6.1 Logistic regression from scratch – gradient descent

Gradient descent is used to find the value of parameters that **minimizes a cost function J**. In other words, the gradient is an **optimization algorithm** that allows to find the **minimum** of any **convex function** by progressively **converging** towards it.

It is thanks to this algorithm that the machine learns, *i.e. finds the best model.*

In our case, we apply this method in an equivalent way to do a **logistic regression** because we want to predict if there is attrition or not.

We use the **vectorization method** because it's more efficient in terms of running time.

To make a long story short, here is how the steps of this algorithm work:

First, we start from a **random starting point** and then measure the **value of the slope** at this point by calculating the **derivative** of the function.

Then, we progress a certain **'alpha'** distance in the direction of the descending slope (*this distance is called the Learning Rate*).

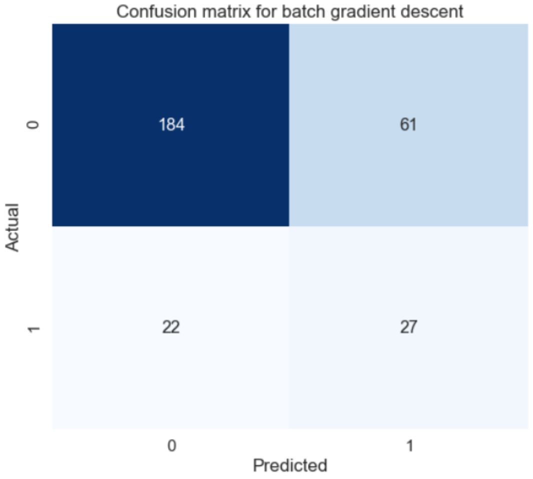
Finally, we repeat these **two steps** in a loop (*the Gradient Descent algorithm is thus an iterative algorithm*).

But in our case, we use the **batch Gradient Descent** where all our training set is taken into account to take a single step (*reevaluate the training set at each step towards the global minimum*).

More in details, here is how we implemented the algorithm:

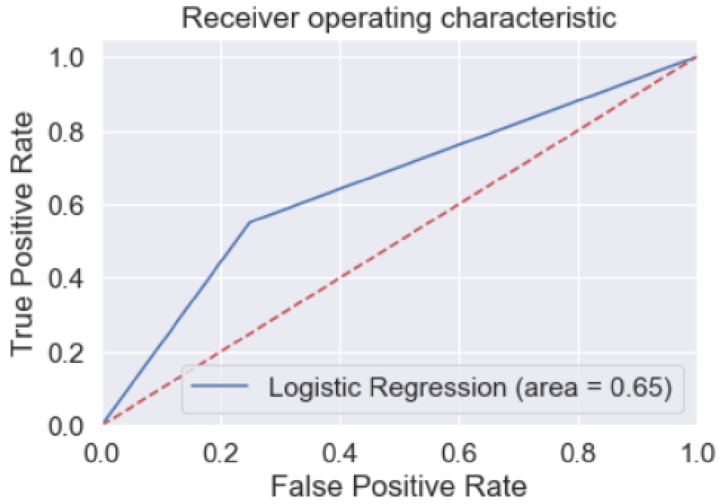
- A prediction function as
  $$\boldsymbol{h = 1/(1 + exp(-matrice\_multiplication\ (x, theta)))}$$
  where x are our explanatory variables and theta the parameters associated, to predict attrition.
- A cost function as
  $\boldsymbol{J -= matrice\_multiplication(y.transpose(), log(h)) +. matrice\_multiplication((1 - y).transpose(), log(1 - h))}$ which is the cost function used for logistic regression and were y is our target variable (attrition).
- The gradient descent returns
  $\boldsymbol{matrice\_multiplication(x.transpose(), prediction\_function(theta, x) - y).}$
  We **initialize** our parameters **theta to 0** (starting).
  Then, in a loop, with a **learning rate of 0.1** and **2000 iterations maximum**, we update them $\boldsymbol{theta = theta + alpha * (x.transpose()\ @\ (y - prediction\_function(theta, x)))}$. Therefore, we have a new value of the cost function. We compute the difference between the old and the new one in absolute value. If this **difference** is **inferior to 0.0001,** we stop our algorithm.

We find an accuracy of **72.95%.**



| | Precision | Recall | F1 score |
|---|---|---|---|
| **No - 0** | 0.89 | 0.75 | 0.82 |
| **Yes - 1** | 0.31 | 0.55 | 0.39 |
| | | | |
| **Weighted avg** | 0.80 | 0.72 | 0.75 |

- **<u>Precision</u>**: ratio of attrition correctly predicted to all observations classified as attrition is about **31%**, the same ratio is better for no attrition (**89%**).
- **<u>Recall:</u>** it is the same phenomenon for the recall. Ratio of attrition correctly predicted to all attrition observations is about **55%**, while the same ratio is better for no attrition (**75%**).
- So we see with the **<u>F1 score</u>**, that the model classifies better the non-attrition class rather than the attrition class. On average, the F1 score is **75%**.



This plot (***True positive rate vs false positive rate)*** shows the ability of our classifier.

The more the area is close to **1** (*and therefore far from the red curve where the true positive rate equals the false positive rate*), the better the classifier, here its 0.65.

You can refer to the ***section*** **5.2** to see that making a **selection of variables** before doing a logistic regression gives a **better accuracy**. In any case, the purpose of this section is to show the **link between machine learning and econometrics**.

## 6.2 Tree-based methods for classification

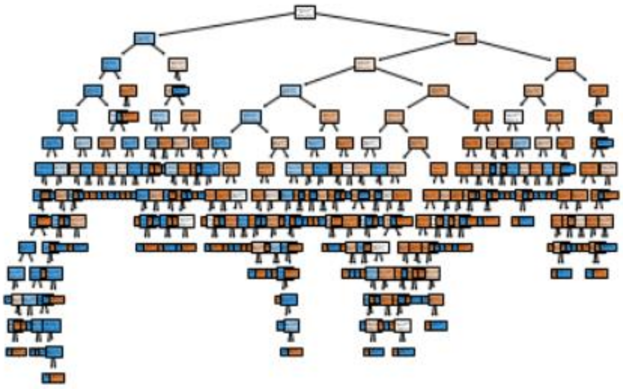
The tree-based methods consist in assigning to each region the most occurring class of observations.

### *6.2.1 Simple classification tree*

The algorithm takes all individuals and looks for the **variable** that **best separates attrition and no attrition**.

To make its choice the algorithm will test each variable and uses a metric (*usually that calculates information gain*) to make its choice.

We started by making a **simple decision tree** which is unpruned with **an accuracy on the test set of 73.47%** *(and the gini metrics that we explain later).*



This unpruned tree is unexplainable and not so easy to understand. We therefore optimize the decision tree performance by pruning.

**The classification rate** (in a given region) is the *proportion of training observations that do not belong to the most common class.* We will here use the **Gini index** and **entropy**, that are both measuring the **impurity** of a node:

- **Gini index**: *measure of total variance across the 2 classes.*
  Small values indicate that a node contains a predominant class.
  The computation is done as follows:

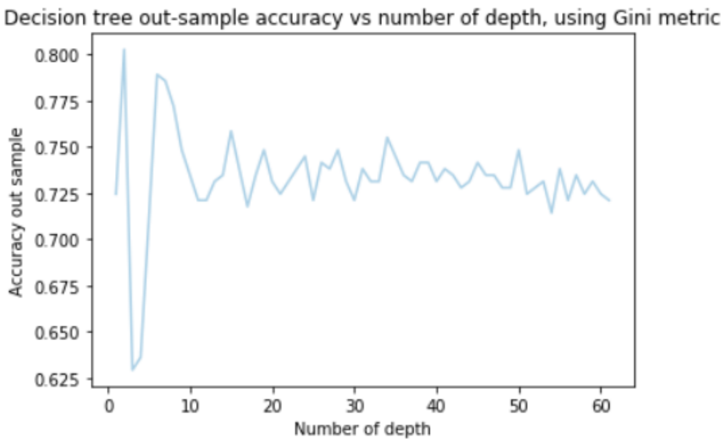
$$\boldsymbol{Gini = 1 - \sum_{i=1}^{n} p^2(Ci)}$$

  *Where p(Ci) is the probability of class ci in a node (n=2).*

- **Cross-entropy**: this is also a measure of purity (a small value indicates that the node is pure).
  The computation is done as follows:

$$\boldsymbol{Entropy = \sum_{i=1}^{n} -p(Ci = log2(p(Ci))}$$

To select the number of variables for the algorithm, we first use **Gini**'s metric and test **from 1 to 62** (*number of variables*) **depths.**

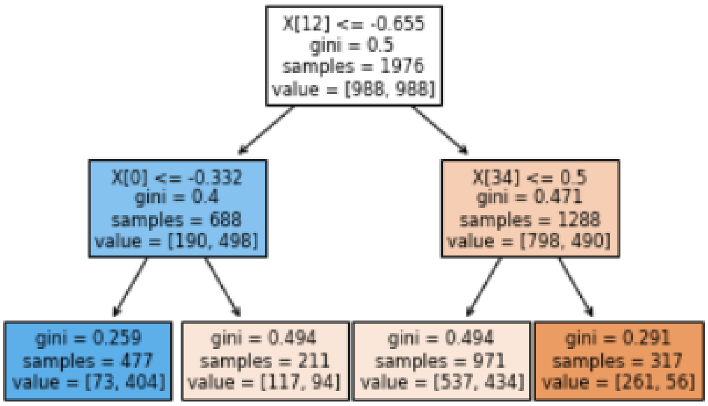


The max **accuracy in the test set** that we can have is about **80.27%** with **2 depths** (*below or above the accuracy decreases*).

A higher value of maximum depth causes **overfitting**, and a lower value causes **underfitting**.

We thus obtain both a higher accuracy and a more interpretable tree.

***The binary tree structure has 7 nodes and has the following structure (in the training set):***



***X[12], training times last year rescaled variable,*** X**[0]*, the employee's age*** rescaled variable and ***X[34]: the marketing education field*** are the 3 variables used to make the split (we can see the 3 split nodes).

Since this is rescaled variables, the decision, for example when training times last year is <=0.655 is not so interpretable.

However, we can look at the first **gini score, 0.5** : the node is not pure (*different from 0*) so we know that the samples contained within the node belong to different classes.

The notation "**samples**", for example equal to 1976 in the first node, correspond to the size of our **database** (in terms of observations).

Then, the **value=[988,988]** tells us how many samples at the first node fall into each category (no attrition vs attrition). The prediction is then made for a given node by taking the class that occurs the most.

***In relation to these definitions, we can therefore interpret a "path" of this tree:***

We start at the root node which asks if the **training times last year is <= -0.655**.

When this assumption is **false** it goes to the internal node on the **right** *where the gini score is 0.471 and the total number of samples is 1288.*

This node will ask if the individual is in the marketing field or not (*<=0.5 → 0 → not in the marketing field*).

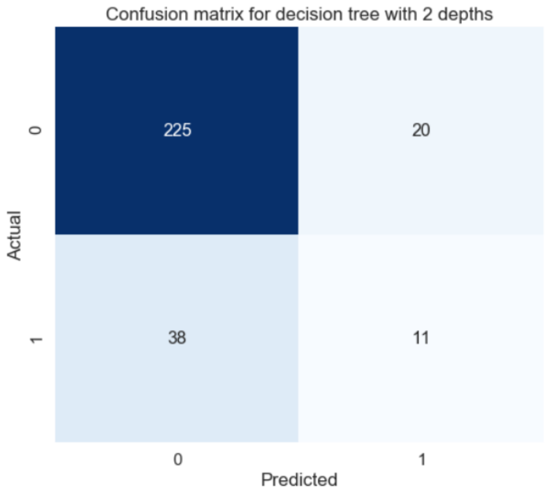
In this case, it moves to the left and the decision tree will predict that those individuals are not concerned by attrition.

We wanted to use **entropy** to see the differences with the Gini criteria.

We obtain the same maximum accuracy on the test set with **2 depths max** of **80.27%.**

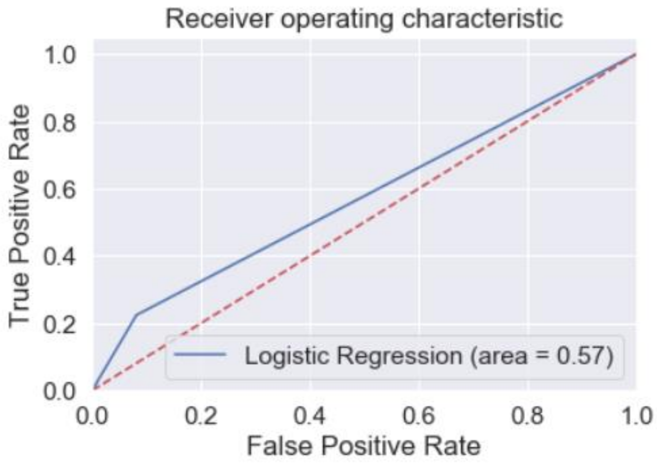
And, with those 2 depths max using entropy we obtain the same tree classifier as with the Gini criteria.

*Here are the confusion matrix, some metrics and the ROC curve:*



| | Precision | Recall | F1 score |
|---|---|---|---|
| **No - 0** | 0.86 | 0.92 | 0.89 |
| **Yes - 1** | 0.35 | 0.22 | 0.27 |
| | | | |
| **Weighted avg** | 0.77 | 0.8 | 0.78 |

- **Precision**: ratio of attrition correctly predicted to all observations classified as attrition is about **35%**, the same ratio is better for no attrition (**86%**).
- **Recall:** it is the same phenomenon for the recall. Ratio of attrition correctly predicted to all attrition observations is about **22%**, while the same ratio is better for no attrition (**92%**).
- So we see with the **F1 score**, that the model classifies better the non-attrition class rather than the attrition class. On average, the F1 score is **78%**.



Here, the area is not so good (**only 0.57**) *regarding the true positive rate vs the false positive rate.*

To conclude on this section, the best classification tree in terms of accuracy out-sample is the one obtained with 2 depths.

The advantages of simple classification tree are that this method is **easy** and **interpretable**. Nevertheless, this not the best in terms of prediction **accuracy** and **unstable trees**.

We will therefore look at 2 other tree-based methods in the following sections.

### *6.2.2 Bagging*

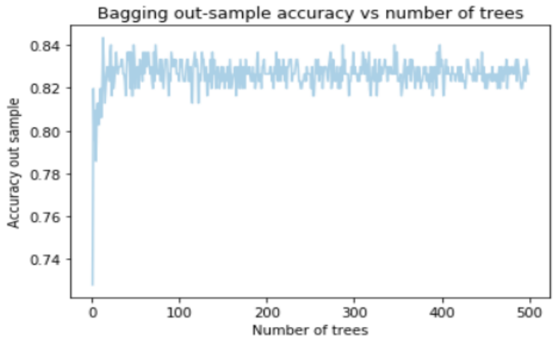
Bagging is used to **reduce the variance** of machine learning methods.

The idea is to use **several bootstrap samples** (*chosen randomly with replacement*) in the training set, then each subset data is used to train their **decision trees**, and finally to **average their prediction** (or in our case, take the majority**).** *Hence its name bagging → boostrap aggregating.*

Therefore, this method helps **overcome the problem of overfitting** because it reduces the variance.

It is important to choose the **number of trees** that will maximize, in our case, the accuracy out sample (test set).

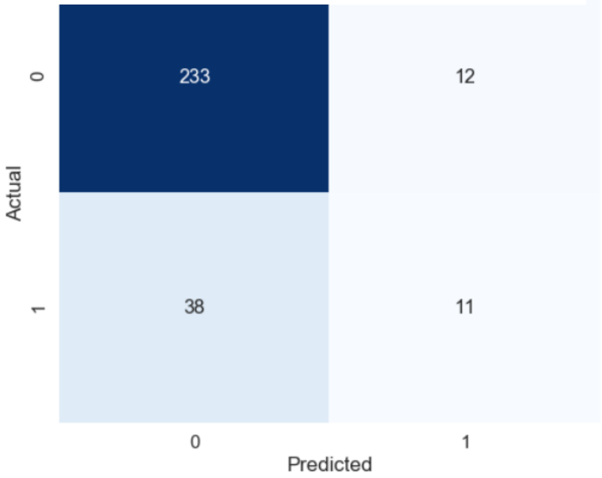
To do so, we do **500 bagging models** (and therefore, number of trees going from 1 to 500), using the entropy criteria.



In any case, we can see that using more than 30 trees is not improving the accuracy at all.

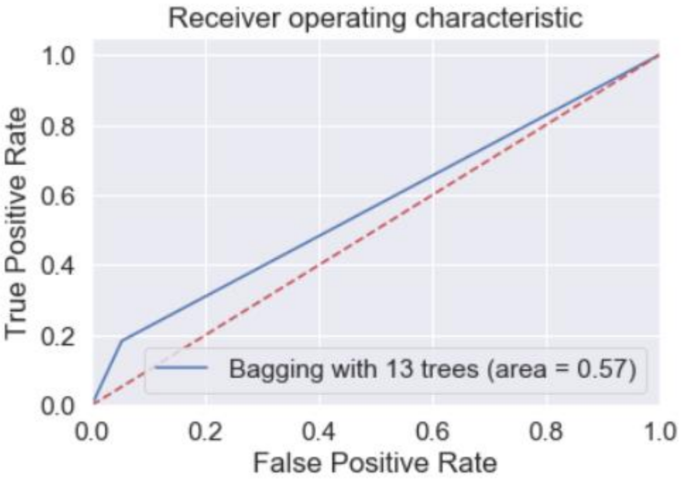
We found a **maximum accuracy** in the test set of **84.35%** for **13 trees**.

*Here there are the confusion matrix, some metrics and the ROC curve:*



| | Precision | Recall | F1 score |
|---|---|---|---|
| **No - 0** | 0.86 | 0.95 | 0.90 |
| **Yes - 1** | 0.48 | 0.22 | 0.31 |
| | | | |
| **Weighted avg** | 0.80 | 0.83 | 0.80 |

- **Precision**: ratio of attrition correctly predicted to all observations classified as attrition is about **48%**, the same ratio is better for no attrition (**86%**).
- **Recall:** it is the same phenomenon for the recall. Ratio of attrition correctly predicted to all attrition observations is about **22%**, while the same ratio is better for no attrition (**95%**).
- So we see with the **F1 score**, that the model classifies better the non-attrition class rather than the attrition class. On average, the F1 score is **80%**.



This is the same result as the logistic regression in the section before (area of 0.57).

For 13 number of trees using **10 folds** to evaluate the model, we find, with no surprise a higher accuracy **92%**.

### *6.2.3 Random Forest*

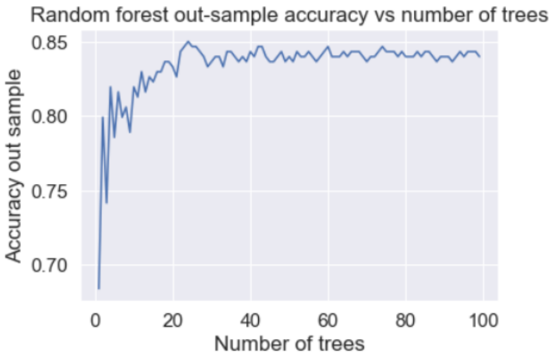
Unlike the bagging method that is using all features to grow trees, random forest **decorrelated trees**.

It is the same method as bagging, but this algorithm chooses a **random sample of predictors at each split of the tree**. As before, the tree is grown to the largest and prediction is given based on the aggregation of predictions from n number of trees.

If we take as before 13 trees, we obtain an **accuracy in the test set** of **83.33%** which is lower than for bagging.

Since the trees are supposed to be decorrelated, we could have expected a better accuracy on the test set.

However, we should not lose sight of the fact that these algorithms give a different accuracy each time they are implemented, and therefore the accuracy can vary by about 3%.

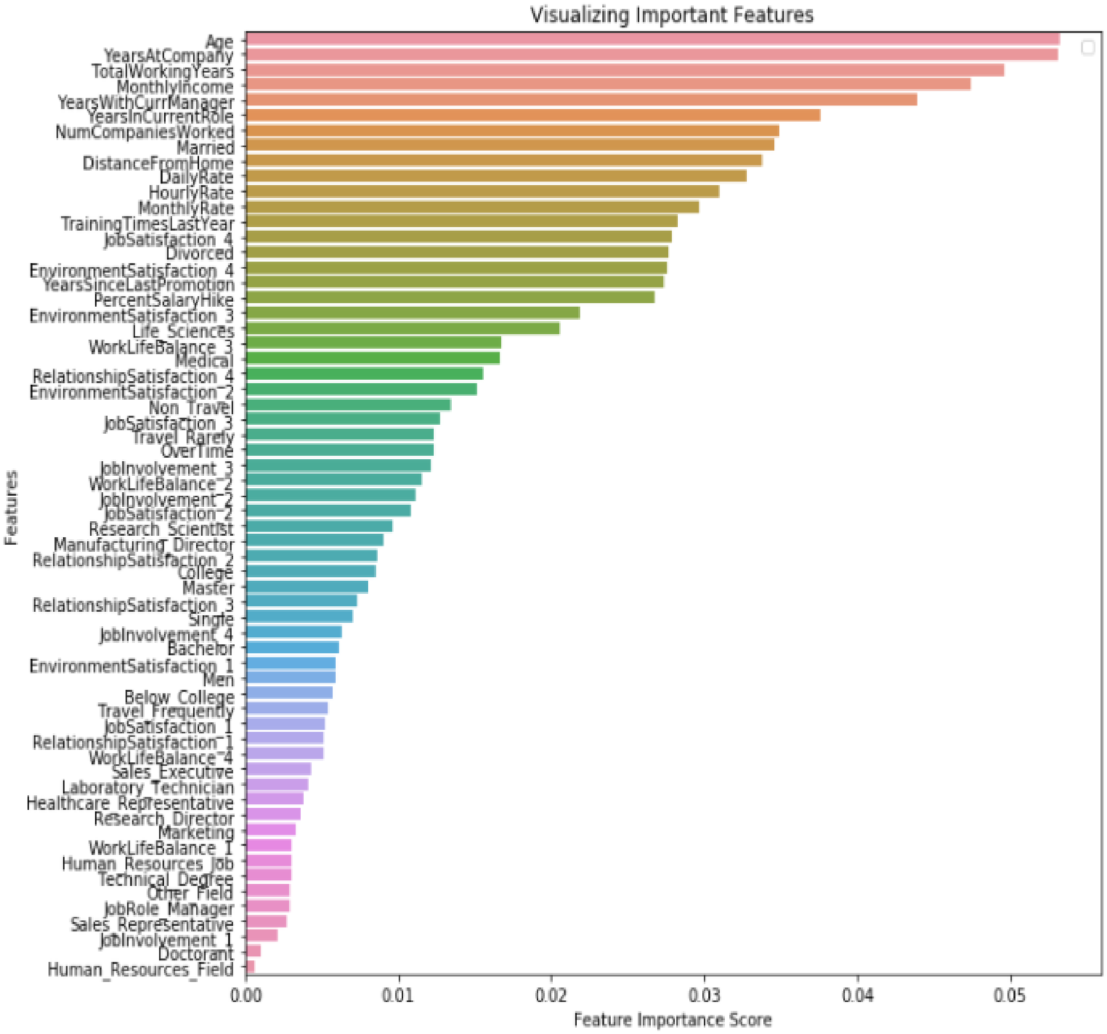


Also, if we try to maximize the accuracy out sample, we obtain an **accuracy of 85.03%** with **24 trees** which is greater than bagging.

With this method, we can show the **<u>relative contribution of each feature in the prediction</u>** by computing the relevance score of each feature in the training phase.

More in detail, we use the criteria of **gini importance** also known as the **total decrease in node impurity:** "*how much the model fit or accuracy decreases when you drop a variable*".

The larger the decrease, the more significant the variable is.



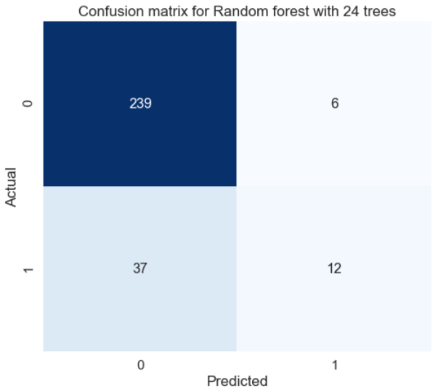
***We see that the 10 most We see significant variables are:***

- Age
- Number of years at the company
- Total working years
- Monthly income
- Years with the current manager
- Years in the current role
- Number of companies worked in
- Married
- Hourly rate
- Training times last year

After this step, we generate the model on selected features (***we drop the 26 least significant variables***) with 24 **trees**, and we obtain **an accuracy in the test set** of **85.37%.**

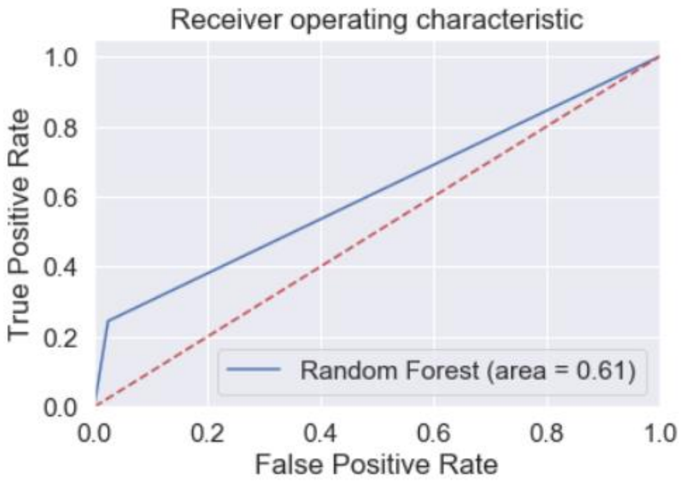
The accuracy increased compared to the random forest with all variables.

This is because it **removes misleading data and noise**, resulting in an increased accuracy (a lesser number of features also reduces the training time.)



| | **Precision** | **Recall** | **F1 score** |
|---|---|---|---|
| **No - 0** | 0.87 | 0.98 | 0.92 |
| **Yes - 1** | 0.67 | 0.24 | 0.36 |
| | | | |
| **Weighted avg** | 0.83 | 0.85 | 0.82 |

- **Precision**: ratio of attrition correctly predicted to all observations classified as attrition is about **67%**, the same ratio is better for no attrition (**87%**).
- **Recall:** it is the same phenomenon for the recall. Ratio of attrition correctly predicted to all attrition observations is about **24%**, while the same ratio is better for no attrition (**98%**).
- So we see with the **F1 score**, that the model classifies better the non-attrition class rather than the attrition class. On average, the F1 score is **82%**.



*The area is better than for logistic regression and bagging.*

To conclude, the advantages of using random forest is that it handles higher dimensionality data well and can maintains accuracy for missing data.

### 6.3 Support Vector Machine (SVM)

***Support Vector Machine*** comes is a **supervised** machine learning algorithm that solve problems in regression, anomaly detection or classification in our case.

They are known for their solid theoretical guarantees, their great **flexibility**, and their **ease** of use even without much knowledge of data mining.

SVMs are based on the idea of finding a **hyperplane** that best **divides a dataset** into **two classes**. The more features we consider the easier it is to identify and distinguish both.

There are many possible ways of drawing a line that separates the two classes, however, in SVM, it is determined **by the margins and the support vectors:**

- **Maximization of the margin:** optimal hyperplane maximizes the **distance between the separation boundary** and the **points of each class** (*which are contained in the support vector*) that are **closest** to it. Maximizing this distance provides some **reinforcement** so that future data points can be classified with more confidence.

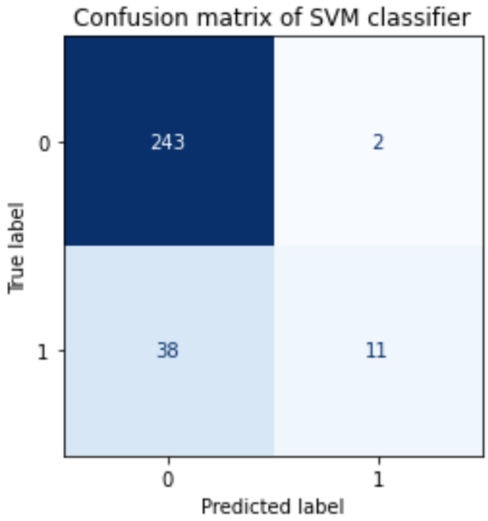
For our SVM we chose **Radial Basis Function kernel** (*RBF kernel*) which is a real valued function whose value depends only on the **distance between the input and some fixed point**, either the **origin**, or some **other** ***fixed point C***, called a **center**. The distance is usually **Euclidean distance**.

We find an **accuracy** of **86, 39%** in the test set.

Apart from looking at the accuracy of the model, it is also interesting to look at other parameters that allow us to see the **quality** of our model.

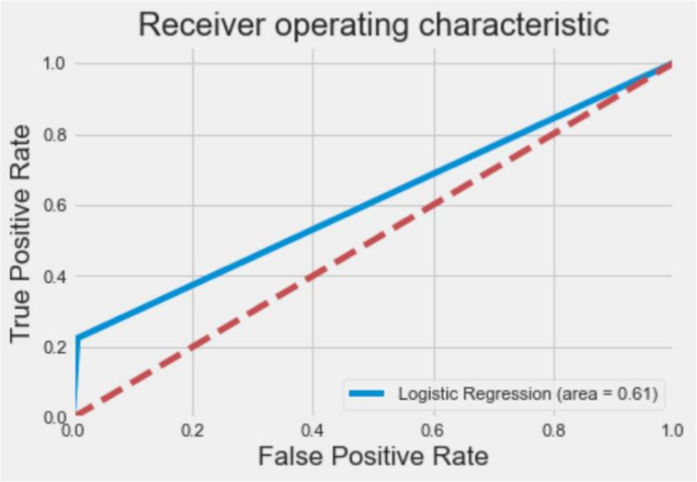
For this, we propose the confusion matrix below, from which several metrics can be derived, such as:

- **Recall** also called **sensitivity** is the ability of a model is designed to find all the **relevant cases** within a dataset. It corresponds to the *ratio of correctly predicted positive observations to all observations in actual class*.
- **Precision** is the ability of a classifier **not to label positive to the negatives**. It corresponds to the *ratio of correctly predicted positive observations to the total predicted positive observations.*
- **F1 score** is the weighted average of **Precision** and **Recall**. Therefore, F1- score takes **both false positives** and **false negatives** into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution (as in our case).



| | Precision | Recall | F1 score |
|---|---|---|---|
| **No - 0** | 0.86 | 0.99 | 0.92 |
| **Yes - 1** | 0.85 | 0.22 | 0.35 |
| | | | |
| **Weighted avg** | 0.86 | 0.86 | 0.83 |

- **Precision**: ratio of attrition correctly predicted to all observations classified as attrition is about **85%**, the same ratio is better for no attrition (**86%**).
- **Recall:** it is the same phenomenon for the recall. Ratio of attrition correctly predicted to all attrition observations is about **22%**, while the same ratio is better for no attrition (**99%**).
- So we see with the **F1 score**, that the model classifies better the non-attrition class rather than the attrition class. On average, the F1 score is **83%**.



This plot (***True positive rate vs false positive rate)*** shows the ability of our classifier.

The more the area is close to **1** (*and therefore far from the red curve where the true positive rate equals the false positive rate*), the better the classifier. Here the area under curve is 0.61.

## 6.4 Neural networks

The basic idea behind a neural network is to **simulate lots of densely interconnected brain cells** inside a computer so you can get it to **learn things, recognize patterns, and make decisions** in a ***humanlike way.*** The amazing thing about a neural network is that we don't have to program it to learn explicitly: *it learns all by itself, just like a brain.*

A neural network is constructed by nesting many **neurons**, in such a way that the output of one neuron is the input of another.

The formal neuron is designed as an **automaton** with a **transfer function** that transforms its inputs into outputs according to precise rules.

For example, a neuron sums its **inputs**, compares the resulting sum to a **threshold** value, and responds by emitting a signal if this sum is greater than or equal to this threshold.

*Efficient operation of Neural Network requires the selection of several parameters such as :*

- **Transfer function.** Taking into account the fact that we want to predict attrition and as we want a probability for output we will use a **sigmoid activation function** on the output layer.

- **Gradient batch size** which is used to train our neural network.
  The **batch size** is a hyperparameter that defines the **number of samples** to work through before updating the internal model parameters. It is like a ***for-loop iterating*** over one or more samples and making predictions. At the **end** of the batch, the **predictions** are compared to the **expected output variables** and an **error** is calculated.

- **The number of epochs:** a hyperparameter that defines the number times that the learning algorithm will work through the entire training dataset.

- **Droup Out rate:** During training, some **number of layer outputs are randomly ignored or "dropped out."** This has the effect of making the layer look-like and be treated-like a layer with a different number of nodes and connectivity to the prior layer.

- Because we are working on a categorization problem with only 2 category we will **calculate our loss with binary cross entropy**.

- **Optimizer ADAM.** After creating our model, we compile it. It's an efficiency step since it **transforms** the simple sequence of layers that we defined into a **highly efficient series of matrix**. This optimization algorithms requires the tuning of **learning rate**. It can be used instead of the classical stochastic gradient descent procedure to update network weights iterative based in training data.

- **kernel_initializer**. It allows to determine the **statistical distribution or function** to use for **initializing the weights**. In case of statistical distribution, the library will generate numbers from that statistical distribution.
  We chose ***truncated normal distribution*** which selects random numbers from a normal distribution whose mean is close to 0 and values are close to 0.
  It is called *truncated* because we are *cutting off the tails from a normal distribution*.

For the selection of optimal parameters for the Neural Network, we wanted to use a very powerful optimization algorithm which should have allowed us to select optimal parameters for our Neural Network. This algorithm is called *Grid Search Algorithm.*

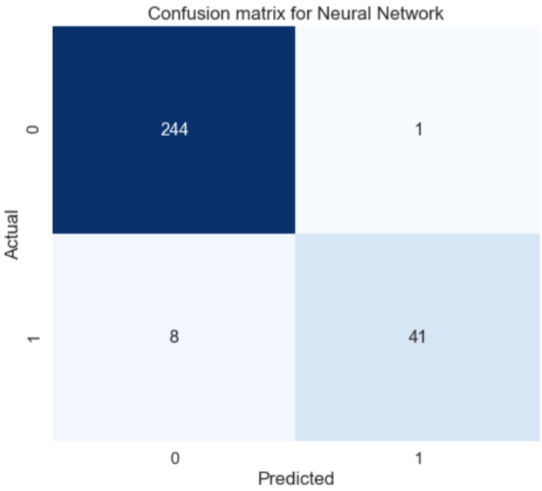
Unfortunately, we couldn't get it work, so we selected parameters by **'trial-and-error' method.** We ended up with: **dropout = 0.1, epochs = 100 and a batch_size = 30**

We will evaluate our model as in the previous sections. However, we have also a 30 Fold Cross validation (*the training set will be cut in 30 folds, randomly, and the neural network will be train on each, using the other fold for validation*)

We obtain a maximum **accuracy of 100%** but let's not forget that the important thing is to test the model on the test set to avoid overfitting.

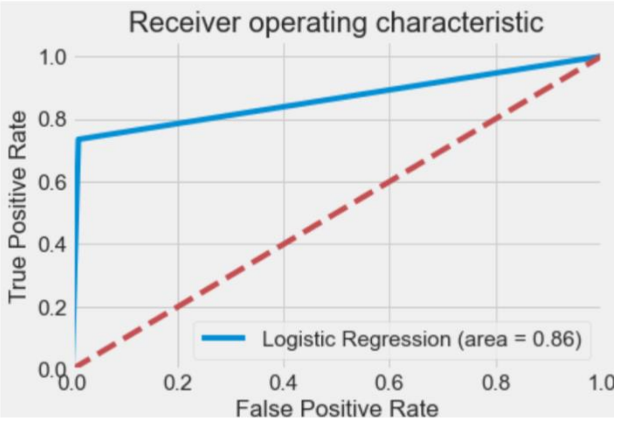
The **accuracy of the test set** is about **96,94%.**

As for previous technics, we will get deeper and analyse the confusion matrix.



| | Precision | Recall | F1 score |
|---|---|---|---|
| **No - 0** | 0.97 | 1.00 | 0.98 |
| **Yes - 1** | 0.98 | 0.84 | 0.90 |
| | | | |
| **Weighted avg** | 0.97 | 0.97 | 0.97 |

- **Precision**: ratio of attrition correctly predicted to all observations classified as attrition is about **98%**, the same ratio is almost the same for no attrition (**97%**).
- **Recall:** Ratio of attrition correctly predicted to all attrition observations is about **84%**, while the same ratio is better for no attrition (**100%**).
- We see with the **F1 score**, that the model classifies the non-attrition class as well as the attrition class. On average, the F1 score is **97%**.



The more the area is close to **1** (*and therefore far from the red curve where the true positive rate equals the false positive rate*), the better the classifier. Here the area under curve is 0.86 (the best of our models).

## 6.5 K-means

The k-means is a ***clustering algorithm***, in other words it allows to perform **unsupervised** analyses *(unlabeled multidimensional dataset)*, to identify a pattern within the data and to **group individuals** with similar characteristics.

This algorithm searches for a **pre-determined number of clusters**.

Here is a simple conception of what a clustering looks like:

- ***Step 0 – initialization :*** **k individuals** are randomly drawn. These k individuals correspond to the initial centers of the **k classes.**
- ***Step 1 :*** We calculate the **distance** between the **individuals** and **each center**. *Several metrics exist to define the proximity between 2 individuals. The "classical" method is based on the Euclidean distance.*
- ***Step 2:*** Each individual is assigned to the **nearest center.**
- ***Step 3:*** We calculate the centers of gravity of the groups that become the **new centers**.
- ***Iterative loop:*** *steps 1, 2 and 3 are repeated as long as the individuals are reassigned to new groups after an iteration.*

The advantage is that it is **simple, robust and easy** to understand but the disadvantage is that before getting started, we need to **determine the number of clusters** we want to obtain, *which is not necessarily intuitive.*

A **large K-number** can lead to an **overly fragmented partitioning** of the data. This will prevent the discovery of interesting patterns in the data.

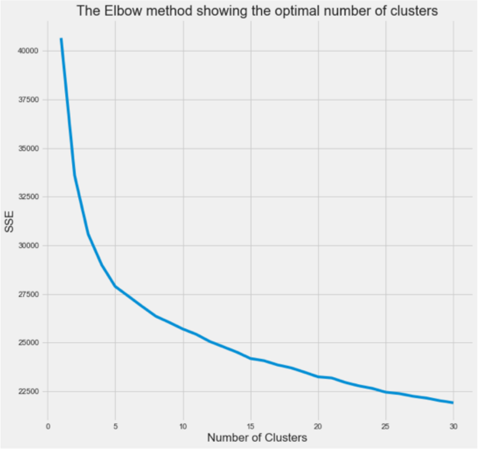
On the other hand, a number of clusters that is **too small** will potentially lead to clusters that are **too generalized** and contain a lot of data. *In this case, there will be no "fine" patterns to discover.*

In order to select the optimal number of clusters, we have focused on 2 technics:

- *Elbow method*
- *Silhouette coefficient*

### *6.5.1 Elbow method*

The quality of the cluster assignments is determined by computing the **sum of the squared error (SSE)** after the centroids **converge**, or match the previous iteration's assignment. The SSE is defined as the *sum of the squared Euclidean distances of each point to its closest centroid*. Since this is a measure of error, the objective of *k*-means is to try to **minimize** this value.



Plotting SSE as a function of the number of clusters, notice that ***SSE continues to decrease as k increases***. As more centroids are added, the distance from each point to its closest centroid will decrease.

We can observe a spot where the **SSE curve starts to bend known** as the **elbow point**. The x-value of this point is thought to be a reasonable trade-off between error and number of clusters. In this example, **the elbow is located at x=5.**

However, we decided also to compute the number of clusters using a python package *kneed*. Using this package, we also ended up with 5 clusters.

### *6.5.2 Silhouette coefficient*

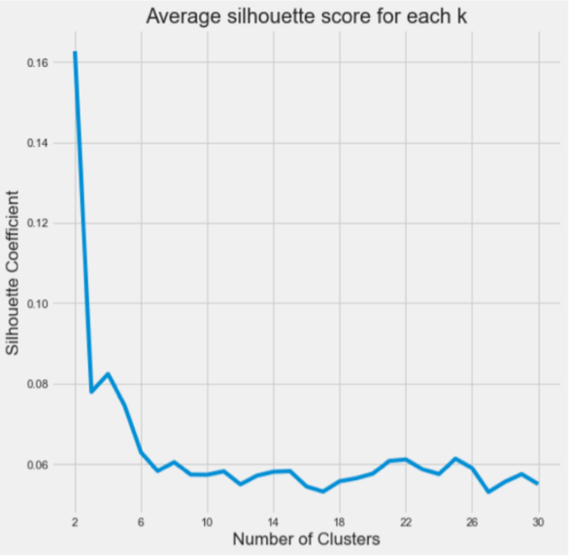
**The silhouette coefficient is a *measure of cluster cohesion and separation*. It** quantifies how well a data point fits into its assigned cluster based on two factors:

- *How close the data point is to other points in the cluster.*
- *How far away the data point is from points in other clusters.*

Silhouette coefficient values range **between -1 and 1**. *Larger numbers indicate that samples are closer to their clusters than they are to other clusters.*

In the *scikit-learn implementation* of the silhouette coefficient, the average silhouette coefficient of all the samples is summarized into **one score**.

So, this time, instead of computing SSE, we compute the silhouette coefficient.

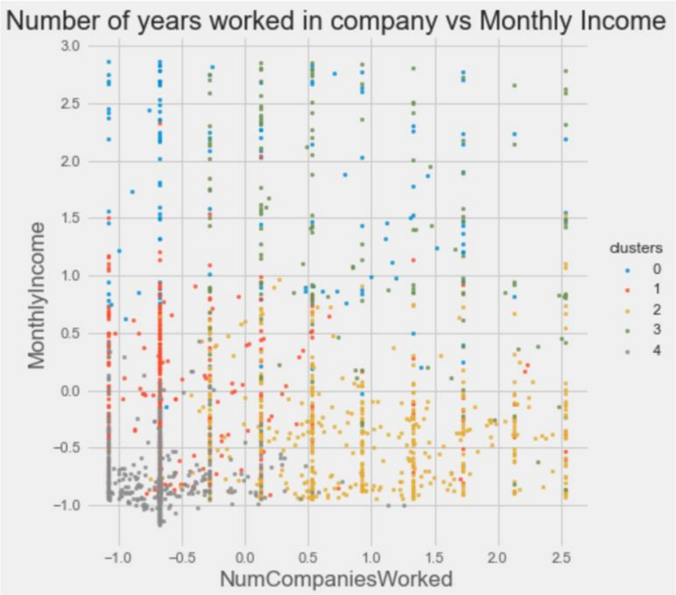


Plotting the **average silhouette scores for each k** shows that the best choice for k is **4-5**.

*Normally, we have to choose the number of clusters which maximize the silhouette coefficient, but according to our plot this coefficient is maximal for 2 clusters, but* as was said above, too small number of clusters leads to very generalized patterns.

After having define the number of clusters, we implement K-means based on a **distance matrix**.

Displaying k-means with a **large number of variables** is very **complicated** even impossible (*with dummy variables we have 62 features in total*), so we have chosen to display **our 5 clusters** in function of **two variables**: ***years worked at the company*** and ***monthly income***. Note that those 2 variables have been **<u>rescaled</u>** in the data preprocessing section.



Looking at figure we can observe that there is a clear cluster for employees who have a **low monthly income** and **do not work in the company for a long time** (**<u>cluster 4</u>**), *other things being equal*.

Another cluster can be detected (**<u>cluster 2</u>**). This cluster is probably composed of employees who have a **monthly income between low and medium** and who **work in the company long enough**, *other features being ignored*.

Moreover, we also see a strong concentration of observations (**<u>cluster 1</u>**) which can be described as IBM's employees who **work not for so long** in the company, but who have a **monthly income greater** than employees from the **cluster 4**, *other characteristics being ignored*.

Unfortunately, we can't get any conclusion for clusters 0 and 3. These groups of employees seem to be more or less similar.

In terms of evaluating accuracy. We should remember that k-means is not a classification tool, thus analyzing accuracy is not a good idea. We can do this, but this is not what k-means is for.

# 7 Conclusion

## 7.1 Comparison and results

**Metrics of used models:**

| Models | Description | Accuracy | Precision | Recall | F1 score | Area (ROC curve) |
|---|---|---|---|---|---|---|
| **Logistic** | Non linearities, crossed variables, PCA | 81,63% | 83% | 82% | 82% | 0,69 |
| **Ridge** | Non linearities, crossed variables, PCA | 78,91% | 78% | 79% | 79% | 0,60 |
| **Lasso** | Non linearities, crossed variables, PCA | 78,57% | 80% | 79% | 79% | 0,64 |
| **Elastic Net** | Non linearities, crossed variables, PCA | 75,85% | 79% | 76% | 77% | 0,65 |
| **Gradient descent** | Simple logistic regression | 72,95% | 80% | 72% | 75% | 0,65 |
| **Tree** | 2 depths max | 80,27% | 77% | 80% | 78% | 0,57 |
| **Bagging** | 13 trees | 84,35% | 80% | 83% | 80% | 0,57 |
| **Random forest** | 24 trees with dropping the 26 least significant variables | 85,37% | 83% | 85% | 82% | 0,61 |
| **SVM** |  | 86,39% | 86% | 86% | 83% | 0,61 |
| **Neural network** |  | 96,94% | 97% | 97% | 97% | 0,86 |

| |
|---|
| 1rst best |
| 2nd best |
| 3rd best |

There is often a **trade-off** between the **interpretability** of **econometric models** and the **performance of machine learning models**.

Indeed, as we can see in this table, first **3 best models** (*or at least those with the most performing metrics*) are the **neural network**, the **SVM** and the **random forest**.

Nevertheless, the first **logistic regression** competes with random forest in terms of ***precision*** and ***F1 score.***

The **choice of variables** is therefore a real issue when it concerns implementing an **econometric model**.

Also, when we are interested in the **area below the ROC curve**, we see that best models are once again the **neural network**, the **regression logistic** that we spoke about just before but also the one implemented by **gradient descent** which considered only basic variables.

In general, with the descriptive statistics, the importance of the variables in the random forest and the principal components we could see that **temporal variables** *such as age, number of years worked in the company, total working years, years in the current role* and with the *current manager* are important regarding attrition.

But also, other variables such as the *monthly income*, the *marital status* and the *training times last year* plays their role.

## 7.2 Limits

We only used **PCA** (i.e., *on quantitative variables*) and not **MCA** (i.e., *on qualitative variables*).

Machine learning models such as **bagging** or **random** forest can have **different accuracies** every time we compute them. *It may therefore be difficult to find exactly the same results as we do by applying these algorithms to the same database.*

Most of used models do not allow much **interpretation**, since in the econometric models we have modified the variables (*rescale, PCA, cross...*) and the machine learning models are rather used for **prediction**.

The database is quite small in terms of **observations**, and most algorithms better predict those **who are not affected by attrition** rather than those who are, which is rather unfortunate in this study.

# 7 Bibliography

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

2 Prakash, S. and Chowdhury, R. (2004).“Managing attrition in BPO, In search of Excellence”

3 Marchington, M.; Wilkinson, A.; Donnelly, R.; Kynighou, A. Human Resource Management at Work; Kogan Page Publishers: London, UK, 2016.

4 Deepak, K.D.; Guthrie, J.; Wright, P. Human Resource Management and Labor Productivity: Does Industry Matter? Acad. Manag. J. 2005, 48, 135–145

5 Angelo S Denisi , Ricky W Griffin: (2009) Human Resources Management , BiztantraPublication, New Delhi 2nd edition

6 Alao, D.; Adeyemo, A. Analyzing employee attrition using decision tree algorithms. Comput. Inf. Syst. Dev. Inf. Allied Res. J. 2013, 4, 17–28.