University of South-Eastern Norway
Faculty of Business School
Institute of industry economics,
Master's Thesis
Study program: Innovation and Technology Management
Spring 2022

Robin Færaas

Importance of data analytics and innovation

A proof of concept how to tackle data analytics for business growth

Foreword

Denne masteroppgaven er det siste kapittelet i min 2-årige masterutdannelse i Innovasjon og teknologiledelse, i tilegg en bachelor i data ved universitett i Sør-øest Norge (USN). I løpet av masterstudiet har jeg vært igjennom metoder og verktøy for å gjennomføre innovasjon, digitale transformasjoner og etablere nye forretningsmodeller.

Med en spesialisering i entrepenørskap og et utveksling opphold og ønsket derfor å gå i dybden på digitalisering av forretningsaktiviteter som gjøre business hverdagen enklere.

De siste to årene har jeg blitt ekstra oppmerksom på hvordan bedrifter benytter seg av digitalisert data, både egen, men også kundenes. Dette har inspiret meg til å gå nærmenre på hvordan denne type data kan utvikle konkuranse fortrinn.

jeg ønsker å avslutte masterstudiet ved å koble sammen alle læringskomponentene i en avsluttende mastergradsoppgave. Prosessen har vært utrolig lærerik og jeg er veldig motivert til å ta med meg denne kunnskapen inn i arbeidslivet.

Universitet i Sør-Øst Norge

Kongsberg, Mai 2022

Forfatters navn her

Summary

The thesis concentrates on how data analytics can help innovation and business growth in the emerging times of industrial revolution 4.0. As a business grows, multiple internal and external factors can affect it.

The purpose was to determine the most influential factors for a digital transformation. The name says digital. Is the technology the most important or the organization?

To deal with this information, a home lab was created, with different datasets from open sources typically generated in a company. Then using data science framework techniques to gather new insight about existing data is not apparent by just looking at the data since the information is now digitized as rows and columns stored in a database.

The result tells us that in the first stage of a transformation, automation and digitization of a report are the most important and closely related to IT. The organization's culture and skills are much more significant in the other steps.

The recommendation provides a holistic view and a better understanding of the benefits of analytic data. During business growth, nothing lasts forever. In that case, businesses need to be acceptable to change, triggering a business idea about data analytics in business

Keywords – Data Analytic, Digital Transformation, Storytelling, Business growth, skills and roles.

Content

Figures

Tables

1 Introduction

1.1 Problem formulation, Data as a resource

In a fast-changing world with more information at our fingertips than ever before, it can easily be overwhelming to process it all. Today we are storing zettabytes of data, and it will continue to grow.(David Reinsel, John Gantz, John Rydning , 2018). Due to the digitalization that many companies do.

Today the adoption and effective use of ICT (Information Communication Technology) is not only a form of business innovation but also an important to stay competitive. Most of the tools used, such as Excel, Databases, ERP, CRM, Teams, or Python, can now be applied by everyone, not only a specialist.

Even though many organizations are working on the digitalization process this decade, it has created two types of organizations.

- **Data Driven:** A business that is using data to enhance decision making and support business intelligence. Together with data in the core of their business models and strategies.
- **Non-Data Driven:** Are companies that have not transitioned yet for different reasons or older organizations. Typically these companies use data as information.

To go from **Non-Data Driven** to **Data Driven** organization need a form for digital transformation. The HBR addressed this issue by looking at what gives the company value in the digital era and rethinking how you work. “Without more fundamental business transformation, digitalization on its own is a road to nowhere"(Paul Leinwand P. and Mahadeva M. Matt Mani , 2021).

Making people work in new ways in the same old organizational model can be changed into an idea to break up old structures, so new ideas and experiences in outcome-oriented teams collaborate across the organization

1.2 Motivation for the thesis

I want to do this kind of research because at the beginning of the fourth industrial revolution, also referred to as digital transformation. People talked about IoT, machine learning, data analytics, etc. They are thinking about how significant a digital transformation is. Data is trivial for new technology native companies, but for older businesses, this is not so trivial. The report will be a proof of concept of a small-scale business with the tools and techniques for data analytics, complemented by theory about data, digital transformation, and business growth.

Many of the data sources come in different forms and shapes, and over time they lose value(Inmon, 2015). Using data analytics techniques on quantitative data some essential hidden insight can be detected that usually not taught of. In addition there is valuable information lying outside the organization, typically stored in public records and APIs.

1.3 SMEs

Innovation helps productivity, long-term growth, and staying competitive. The study will mainly focus on SMEs, especially since SMEs can find this a challenging task due to their limited resources. The inadequate resource makes it challenging for SMEs to be as innovative as a larger company. (Srdjan Dragutinovic , 2021)

Giving SMEs a deeper insight into how to use data analytics to see what works and define why and how it worked. The demand for data analytics will continue to grow, so if SMEs want a competitive advantage, it is just to get started. It can be a daunting task if they have not done it before. However, better decision-making benefits such as better customer attraction, increased employee productivity, and systematic efficiency improvements are the way to go. Looking at data already available to shape the foundation (Srdjan Dragutinovic , 2021).

1.4 Research questions

That lead to these research question for the thesis

1. Is Data management and Innovation management more connected than before?

2. How vital is Business culture to digital transformation?
3. What kind of skills is needed to succeed in digital transformation? What are things people need to learn when they work with digital transformation?
4. Does IPR(Intellectual Property Rights) management have business value?

The idea behind the first research question is see if better data management, can contribute to better innovation management. Is it true that data analysis can give insight that were not thought of. This is relevant for making good decisions on innovation that bring a company forward.

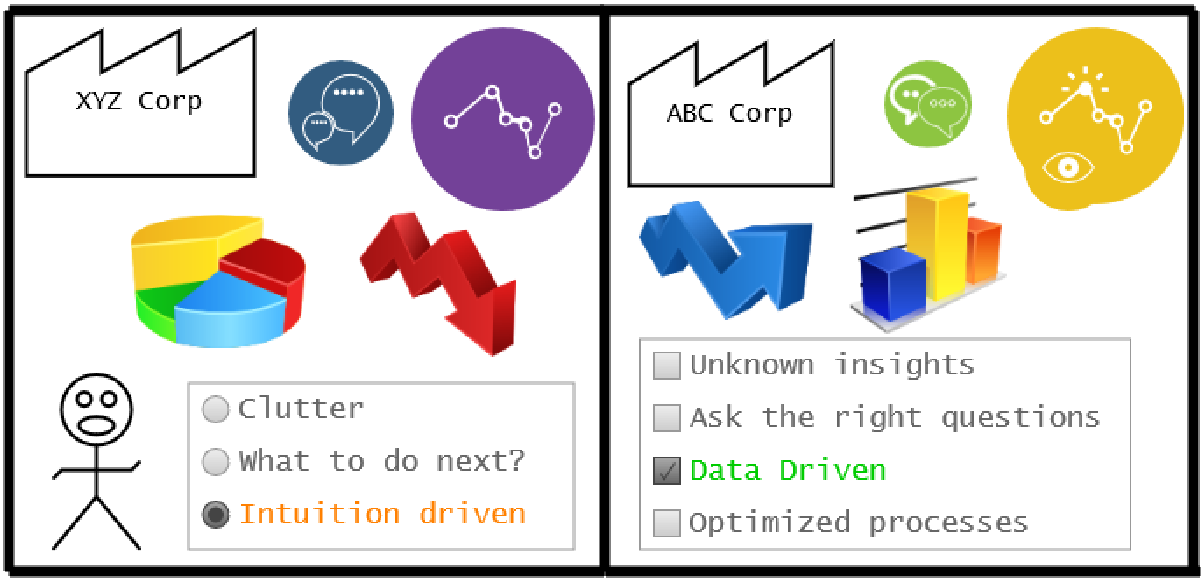
Next is the business culture and skills. Will a digital transformation only concentrate on technology or do people need to change their mindset and learn new workflows. When employees and managers are changing their daily routines some resistance will happen. Talking about this problem is essential for having a healthy business. A healthy business will perform better, so finding the most significant skills for business growth is the main objective.

Finally, IPR. This is typical external data that tells us about companies / a country's innovation degree. which has got a considerable increase in the recant years about this data. Knowing what your competitor are doing puts you in a position to react to that. I will try to collect and extract useful information from these data.

Since this is a proof of concept, it will form into a possible business idea that can help SMEs know what to look for if they have not already started. To perform the proper analysis, they need to know which data might be more valuable and the appropriate knowledge for the team. This will not include how to set up infrastructure in a "safe" way, regarding backup of files, network configuration. Or any cloud configuration.

In this process, a gap is identified during the 4.0 industrial revolution. Figure 1.1 describes XYZ Corp. A typical SMEs created before the revolution is not aware of data as a resource, only as information. If they do not change their direction, it is possible to be outperformed by ABC Corp as a newly started business or a business that has already changed its direction. With the new ways of working.

Figur 1.1: Old style and new style



2 Literature review

2.1 Business perspective, innovation and trends

Consistency is essential for winning a championship, building a business, or losing weight. To close the gap and look for the next big thing, being good at innovation and making an excellent decision requires improvement over time. In James clear's book Atomic habit, he addressed this problem by looking at atomic habits. The slight change is not even noticeable at the moment, but over time getting 1 percent better every day at the end of the year results in a 37% increase. Being 1 percent worse, it will decline to almost zero. It points out not because the desire for change is bad, but having the wrong system for change(Clear, 2018).

Undoubtedly, digital technologies offer new opportunities for Small and medium-sized enterprises (SMEs). But it seems more difficult for SMEs to adopt than larger enterprises. "The new opportunities, called data analytics, a series of techniques used to effectively analyze large amounts of data from structured and unstructured sources, can become a crucial driver of enterprise competitiveness. Strengthen position in both local and global markets, through product or service innovation and improved production processes"(Bianchini og Michalkova, 2019). With more significant information about customers, competitors, and suppliers. Companies can make more strategic decisions based on data. Due to the Internet of things (IoT), data generation and collection have significantly increased. That means SMEs and entrepreneurs may have difficulties accessing and analyzing relevant data. In addition, they do not necessarily have the specialists needed or limited digital skills by managers and employees. That makes it harder to adopt.

Data analytics can help identify a pattern, relationships, and interactions. But raw data need to be cleaned, standardized, and organized before statical analysis can start. Enterprise information software such as ERP, CRM, and SCM have been implemented to complement data growth.

Not Having trust in the data coming from Big data sources is unlikely to bring change in the organization. With a lack of managerial awareness and skills, it can be hard to change traditional business practices, which have worked well in the past. If the understanding of a digitalization project is tied to the cost of implementing data infrastructure, then the benefits and the need for change lose their value. Going from an "intuition drivenbusiness to a data-drivenbusiness requires strong commitment and a good understanding of data(Saldanha, 2019).

An example of this could be an employee working with excel, being pretty good at using the tools, but after a while, with new tools and structures, excel is not so relevant anymore to use. Instead, the data is stored in a database, making it more convenient and reliable to use SQL to query. But this employee is not committed to the new workflow, so they copy the data from the database and paste it into excel to work with it.

1. The employee may create data redundancy, which can influence the important value.
2. the employee uses more time and is not productive.
3. The employee can run into tool limitations, such as too much data to handle.

With more available tools, employees can be trained to perform data analytics on business problems. Having that "in-house"can generate some advantages. Naturally, the employees better understand the data sources, such as production process, customer interaction, or supply chain. Comparing that to outsourcing, you could get an expert on the field but no relations to company data. Not only employees need training but also managers. Entrepreneurs and management often lack knowledge of emerging technology that can make or break important decisions(Bianchini og Michalkova, 2019).

2.2 Greiner's Growth model

Greiner wanted to create a model of the process of organizational development. At the time, most of the studies were empirical, but the factors influencing the growth were related to the organization's age and size, the growth rate of the industry, and the stages of evolution and revolution(Larry E. Greiner, 1998).

The evolution period is when the company maintains growth under the same management pattern. This means that some organizational practices will change throughout the organization's life span. More minor changes can be retained over an extended period, but increased employees, customers, and sales lead to coordination of management practices. Then a revolution occurs when the old patterns are not enough, and the organization needs something new to keep growing.

There will be a dominant management style for the evolutionary periods to achieve growth. Over time the management style will be a problem, so in the revolution periods, typical a crisis. The problem needed to be solved before the business continued to grow. In addition, will the industry growth rate determine how fast or slow these phases of evolution and revolution progresses.

Dividing the growth into 5 Phases				
Creativity	Direction	Delegation	Co-ordination	Collaboration

Tabell 2.1: Growth 5 stages

2.2.1 Phase 1 Creativity

Creativity Is focusing on the beginning of organization or a company that are launching new products. With a high technical or entrepreneurial spirit for the product, management activities are overlooked. The communication is informal and long hours. To get started, these creative activities are important, but when the grows these become a problem. The increasing in financial responsibilities emerge. New employees, will fell as connected to product. So the crisis of leadership occurs. A strong manger with the necessary knowledge is needed to break up old business technique. Founders is often resisting stepping aside even though they are unsuited to the job. So finding a business manager that is acceptable for the founders is crucial for moving forward.

2.2.2 Phase 2 Direction

With a more defined frames the communication becomes more formal, but some might find it hard to deal with a centralized hierarchy, and feel torn between following procedures and systems. Due to more complexity the organization can have trouble with fast change, and employees taking initiative on their own, being limited by the system. This will lead towards a crisis in autonomy.Giving employees more choices can be challenging for top-level managers who previous were successful at the top.

2.2.3 Phase 3 Delegation

Top management can easily overworked during more important decision making, therefore decision are sent downwards in the organization, that means the lower management making decision for them self. This is able to create a more decentralized company structure, but you loose at bit of the holistic view over processes. Leading to a control crisis.

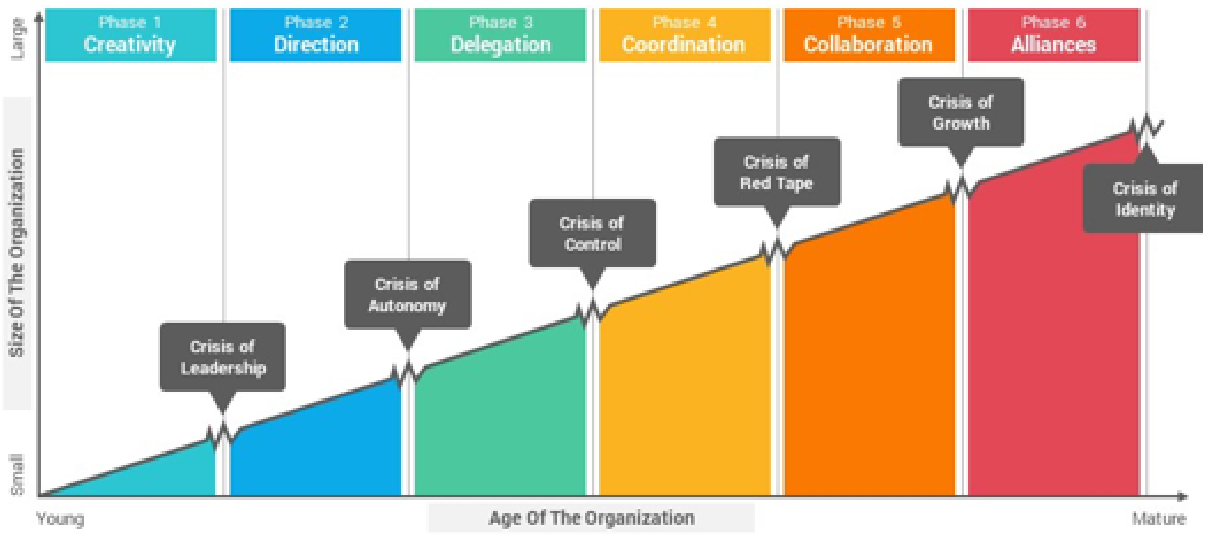
2.2.4 Phase 4 Coordination

Moving over to coordination phase, they might implementing new communication systems, there are a lot of paperwork and administration that can influence the business culture to be more bureaucratic for more documentation and control, in addition bureaucracy make it difficult to do innovation.

2.2.5 Phase 5 Collaboration

When the organization is growing organic, with increased sales and internal expansions. Assigning the right people to the job can give the organization a good direction, but it is easy to lose control over the organization.

Figur 2.1: Greiner's model of organizational growth



The growth model is shown in figure 2.1 where the age is represented by x-axis and size by y-axis. During the phase it is a steady growth until the crisis occur at the end. After some time new products and corporate venture happens so the *Alliances* phase is part of a new cycle of the growth model.

2.2.6 Considerations/ reflections

Figure 2.1 Above is a picture of how Greiner's model is represented. Even though this model is represented as linear, it can occur in a nonlinear fashion and different order of the phases, by reason of complex real-world problem are not seen in the same way as presented on paper. Some have criticized it for being hard to apply in practice since some organizations jump over crisis, that means that a company not necessarily did not have a crisis, but there are some external or internal environments that that influenced the business process. In worst case a business can fail the crisis and not move foreword.

The result of the previous phase and cause for the next. Some can also fail the crisis and not move foreword. Being able to identify where the organization lies and what is coming up in that face would, be easier to handle when it appears. Using historical data and live data

Using this model to identify where you Organization are / or headed, will give useful insight towards strategic planning and activities. Because this highlights that the growth just comes by it self. Growth is creating new problems to solve. Because is not possible to grow indefinitely without some sort of disruption. To continue to growth there is a need for change, and that is important to understand.

If it is a long time since an revolution, it may be tempted for managers to skip or avoid the revolution, but it brings with it tension, ideas awareness and choose the more comfortable decision you made earlier. but that make it hard for the new phase to evolve.

`Hypotesis 1: Can data Analytic help Business Growth?`

2.3 External and internal environment

Burke letwin External and internal environment is an simplification of 12 parts that influence an organization performance. With a vertical design the elements further up like leadership, culture, and strategy have an greater impact than lower elements such as tasks and motivation. Even though the model has a top down approach is not set in stone that you should always start at the top. due to a bi dirontional causal relationship between the different elements(Hayes, 2018).

To figure out which elements to start with is important to distinguishes between the 4 major parts in the bullet points. Because sum of these four will represent the organization performance.

- External Factors, like governments laws, competitors is often the reason for the need of change.
- Strategic Factors often referred to as Transformational is the hardest one to deal with and are key to success in change.
- Operating factors could also be called Transactional are easy to change, but do not have the power to sustain lasting change.
- Individual Factors focus on the individual, but these can be affected by a lot of different things.

Figur 2.2: Transformanal factors, source: (Hayes, 2018)

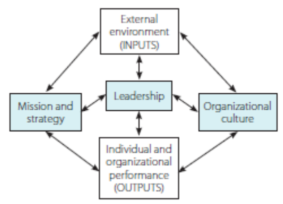


Figure 2.2 show the Transformational factors. When doing organizational diagnosis for improvements, this model is suited towards predictive and not prescriptive analysis with a holistic view over the different type of elements are influenced by change and keep attention to those. Because of a fast changing environment there is today with technology, wars, and supply and demand some of them can get misaligned. A complete picture of this model could be found in appendix A1.

```
Hypotesis 2: the transformational factors have biggest impact on performance?
```

2.4 Digital transformations

Digital transformation, is really a broad term and be associated the concept of the industrial revolutions.

- First industrial revolution with the introduction of industry and steam power

- Second industrial revolution with mass production, electrical energy.
- Third electronic technologies become more available such as PCs, internet and electronic equipment.
- Data governance and IoT, Machine learning, AI and data analytics.

The fourth industrial revolution, is closely connected to digital technology transformation. The pluming cost of computing capacity and stores going online. Going from the third to the forth will bring digital technology as the backbone for new product and services. At the same time creating new business's models based on data from various sources. Enabling workers to check for updated values instead for driving around and writing them down, making more time for higher value- added responsibilities. There are some crucial part to how this will affect us contra the prior industrial revolutions. The degree of good over the bad is still yet unknown, but we are improving it over time. Like the loss of privacy debate, earlier there were a lot less restriction compared to today.

Talking about digital transformation is to move from the third to forth industrial revolution. The transformation must help the organization to *Take Off* and *Stay Ahead*. "Therefore, the only logical end point of successful digital transformations must be to reach the stage of perpetual market leadership via innovation. This is Stage 5 digital transformation"(Saldanha, 2019).

2.4.1 5 stages road map for digital transformation success

The road map model have some steps that can not be skipped, however there is a possibility for steps to be combined. Native digital companies have an advantage over traditional businesses, since they already have an synchronized digital platform. Make it easier to reach stage 5.

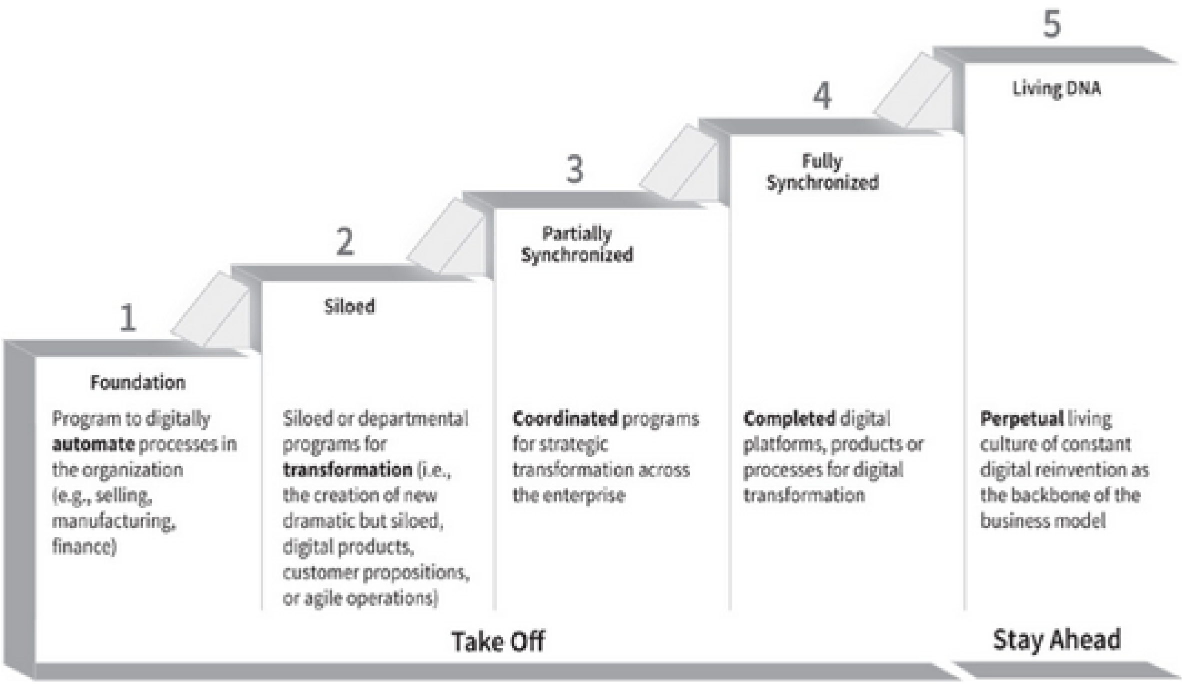
1. **Foundation:** Enterprises are automating internal process, maybe starting using new systems like Customer Relation Management (CRM), Enterprise Resource Planning (ERP), Microsoft Teams or other. Digitization of reports. Maybe outsourcing is the way to go? But the main goal is make manual effort into data.
2. **Siloed:** The changed workflow have functions that allow new business model, such as a digital measurement device for temperature or gas, that are send into a database,

instead of driving out and writing the value down. The are no overall strategy in the company that's driving transformation.

3. **Partially Synchronized:** the management, leader has recognized the power of digital technologies, and there is a define future state. The organization have not completed the transformation for new business model and digital backbone in addition the innovative culture haven't become sustainable.
4. **Fully Synchronous:** A new business model have settle, but there is still a chance for this to be disruptive, to survive the treats against it is to make an agile innovative culture with digital capabilities a part of the organization.
5. **Living DNA:** The transformation has become perpetual, and data is really the backbone of the business models and daily activities. Only only being a market leader, but setting industry trends as an disciplined innovator.

Down below figure 2.3 gives an overview over the model described in the enumerated section.

Figur 2.3: Digital transformation



2.4.2 Considerations

Many organizations working on this, but it can be difficult to see the way ahead, and could be the lack of Data Literacy. And that is the ability to read, write and communicate with data.(Nehme, 2022)

Employees with expertise over a business area are going to be the ones that are best able to act on data insights to create results".

Earlier in stage 1 the main objective was to digitize something physical, like a report or automating a process. During this stage the view is looked through a lens of technology, altogether this was an IT job working as an enabler with operational support. Now businesses are created based on data and digital technology, like Uber and their app for ordering transport compared to traditional taxis. This have continued to expand(Saldanha, 2019).

Moving further to next stages is not about the technology, but the culture, habits and people skills. This will make the commitment and ownership harder from the transformational factors mentioned earlier. In addition making sure data is being part of the organization habits is important. As a consequence of this, everybody that works with data is a part of it.

2.5 Technology skills, hybrid jobs and TW process

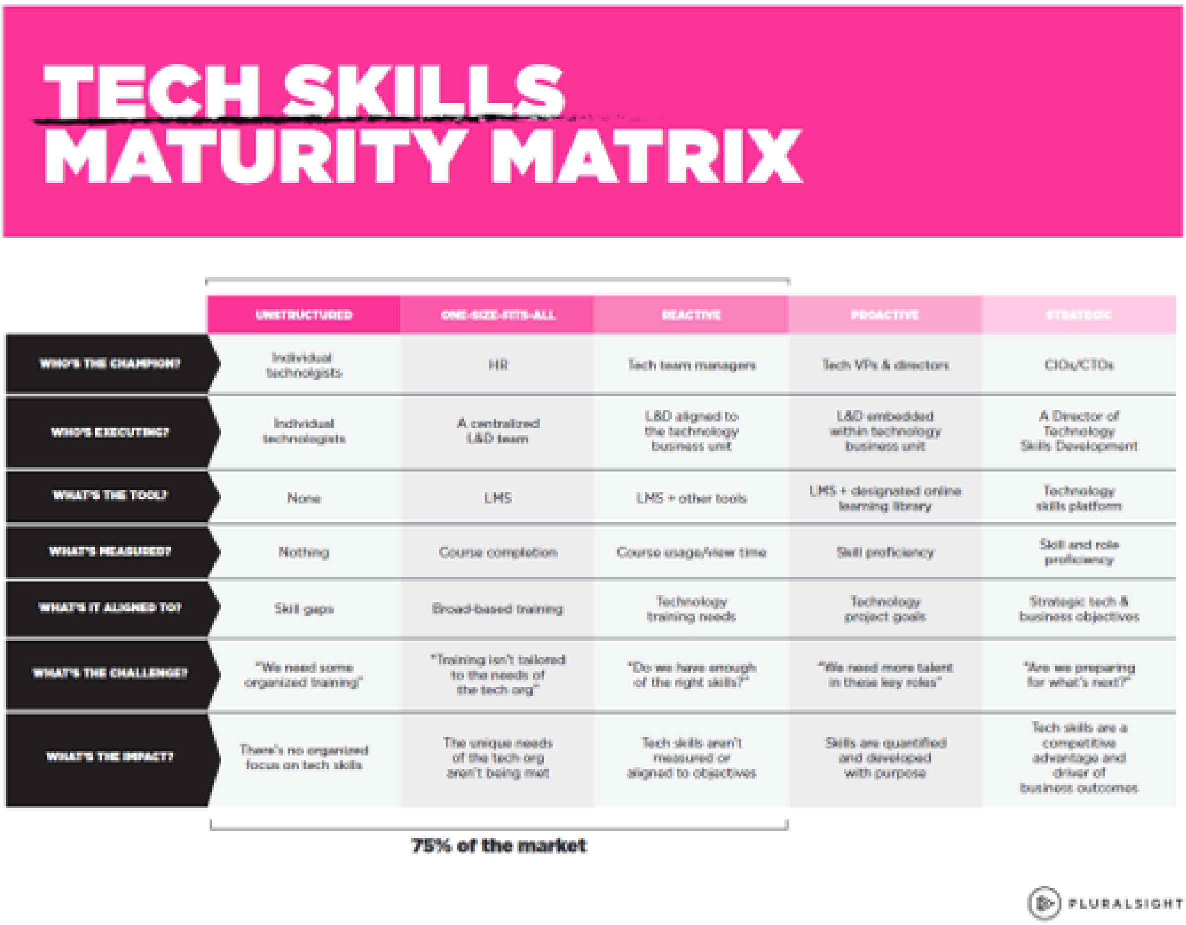
2.5.1 Skills and roles

To complement the model of digital transformation, the employees also need new skills. Because skill evolution is a necessity for every function in an organization"(PLURALSIGHT (2021)(p.13)). By not evolving on organizational platforms, knowledge base and employee base, the platform may become obsolete and big cost for complacency.

By developing technology skills the organization to use it to their competitive advantage, being a strategy approach, rather than an investment. Have the ability to drive business outcomes, by having right people and the right skills. To deliver innovation. If the tech skill development is not implemented as a priority, the pace and complexity will out pace your ability to capitalize on it(PLURALSIGHT , 2021).

When building a TSDS (tech skill development strategy) pluralsight PLURALSIGHT (2021) have made a matrix to help evaluate strategy stands shown in picture 2.4

Figur 2.4: Technology develop skills matrix, Source: Pluralsight


TECH SKILLS MATURITY MATRIX

	UNSTRUCTURED	ONE-SIZE-FITS-ALL	REACTIVE	PROACTIVE	STRATEGIC
WHO'S THE CHAMPION?	Individual technologists	HR	Tech team managers	Tech VPs & directors	CIOs/CTOs
WHO'S EXECUTING?	Individual technologists	A centralized L&D team	L&D aligned to the technology business unit	L&D embedded within technology business unit	A Director of Technology Skills Development
WHAT'S THE TOOL?	None	LMS	LMS + other tools	LMS + designated online learning library	Technology skills platform
WHAT'S MEASURED?	Nothing	Course completion	Course usage/view time	Skill proficiency	Skill and role proficiency
WHAT'S IT ALIGNED TO?	Skill gaps	Broad-based training	Technology training needs	Technology project goals	Strategic tech & business objectives
WHAT'S THE CHALLENGE?	"We need some organized training"	"Training isn't tailored to the needs of the tech org"	"Do we have enough of the right skills?"	"We need more talent in these key roles"	"Are we preparing for what's next?"
WHAT'S THE IMPACT?	There's no organized focus on tech skills	The unique needs of the tech org aren't being met	Tech skills aren't measured or aligned to objectives	Skills are quantified and developed with purpose	Tech skills are a competitive advantage and driver of business outcomes

75% of the market

PLURALSIGHT

It explains that 75% of the market can be found in the first three section, making an opportunity to bring them to the last two. For this it may be an idea to start a data program for everybody in the organization.Nehme (2022)

2.5.2 Technology watch and audit

IF the business is unsure what to look for(Zabala-Iturriagagoitia, 2014) illustrated how we can organize roles and activities when working with TW process. The TW (technology watch) is a process where companies looking for new technology that can be useful for the future. This can be done both outside and inside the organization. depending of the size of the company, typically SMEs outsource this process, but by outsourcing the connection to the data can be misaligned. with limited resources, priorities need to be made.

The TW can be divided into 3 different roles,

- The observer will look into new technologies that are emerging in a set scope of domain. It is important to have a good scope due to the amount of technology that can be created compared to business domains/other word?.
- The analyzer will analyze the observer's technology and evaluate more what could be used.
- The decision maker make the final word of what to focus on.

Involving both the observer and the analyst in the decision making process will give more fluid decision When a organization is In the role of observers, technology do when looking for new technology in TW, Selecting what to search for is vital part for the company, making them aliged with the corporate strategy. since the limited resources c.

It ca create a domino effect on internal processes, all the roles are generating new ideas, concepts that can be used for internal learning for potential theaths or possibilites. 2.1.5 Hybrid jobs and roles and skills (datacamp podcast, Pluarsight pdf complete) When looking at a org chart typically a role does one thing with certain set of skills. One of the things mr man addresses in datacamp podcast about th new emerging work force. Hybrid jobs, this consist of looking at skills used across the organization. Connecting skill and job roles to objectives, the overall

2.6 Story telling with data

When telling a story with data, the data is the main building block. A story will share insights in a more memorable and persuasive than just giving the facts. that because it has a narrative. combining the narrative with data and visuals you get a more complete picture. even though the elements could be strong alone. with a a good data story it can result in change(Dykes, 2020). Figure 2.5 showing all elements to a data story.

Figur 2.5: Combining the story elements

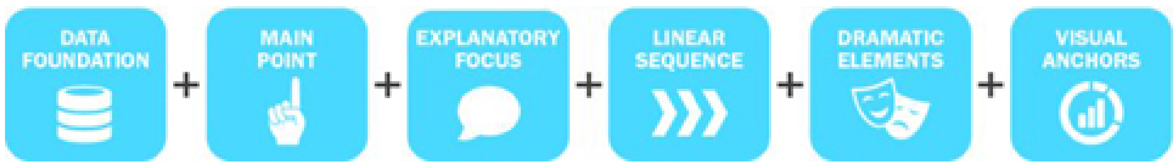
People react differently to facts and stories, with different activation parts of the brain.

- facts that are aligned with audience viewpoints are more accepting compared to facts they do not like. If the facts are visualized it is harder to reject them.
- A story has more unique connection between the storyteller and listener. Creating a less critical sphere and are more open to change.
- With visualizations it can reduce information deficits.
- Stories enhance our comprehension. Stories can help the audience to better comprehend difficult or complex concepts.

2.6.1 Elements of a story

what things should be included in a data story, figure 2.6 explain the elements

Figur 2.6: Combining the story elements

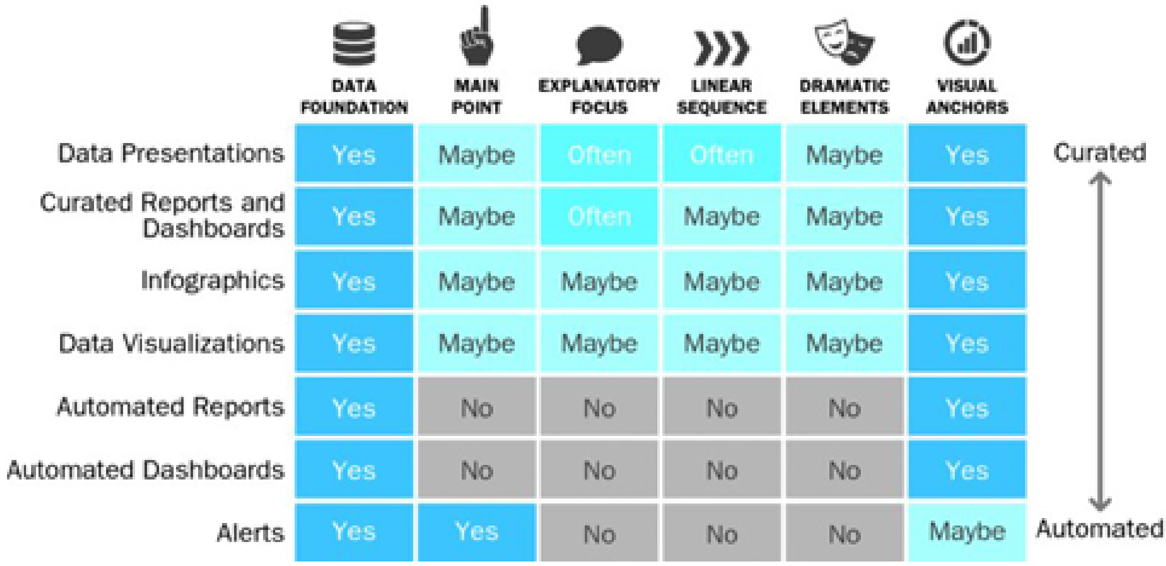


even though we have the elements for creating a good data story we need to look out for what we already have. Figure 2.7 show us how good story elements do not aligned good with automated data(Dykes, 2020).

2.6.2 Stories alliged with data

so where is a data story useful, it turn out

Figur 2.7: story and automated data

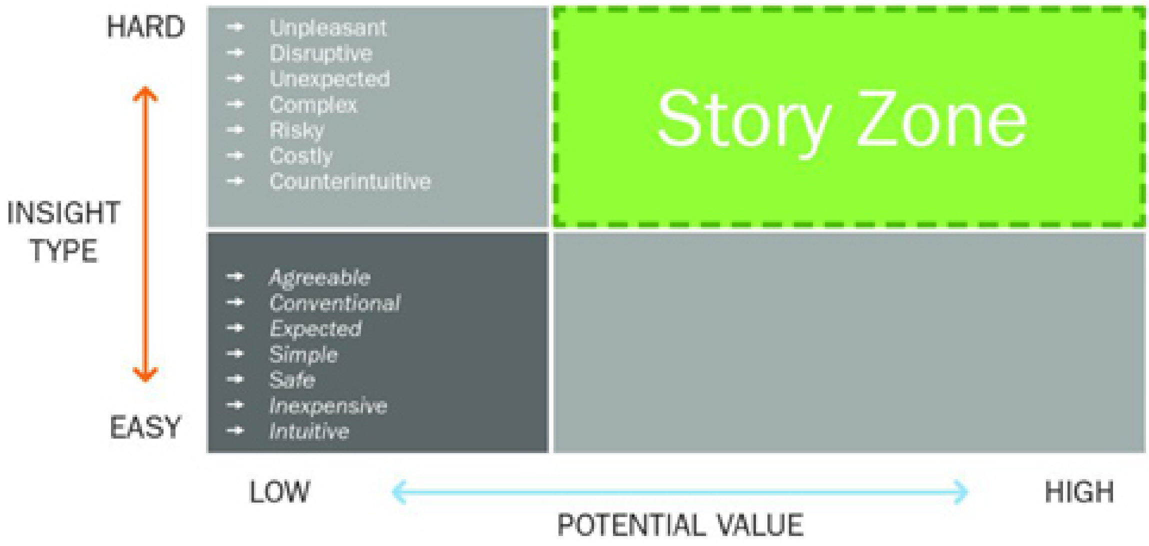


	DATA FOUNDATION	MAIN POINT	EXPLANATORY FOCUS	LINEAR SEQUENCE	DRAMATIC ELEMENTS	VISUAL ANCHORS
Data Presentations	Yes	Maybe	Often	Often	Maybe	Yes
Curated Reports and Dashboards	Yes	Maybe	Often	Maybe	Maybe	Yes
Infographics	Yes	Maybe	Maybe	Maybe	Maybe	Yes
Data Visualizations	Yes	Maybe	Maybe	Maybe	Maybe	Yes
Automated Reports	Yes	No	No	No	No	Yes
Automated Dashboards	Yes	No	No	No	No	Yes
Alerts	Yes	Yes	No	No	No	Maybe

2.6.3 The Manager that just/only want the facts

this is for discussion With an enviroment that changes fast, there can be some managers in a hurry and only want the facts. but sometimes listen to what they have to say can bring great insights as figure 2.8

Figur 2.8: Story telling zone



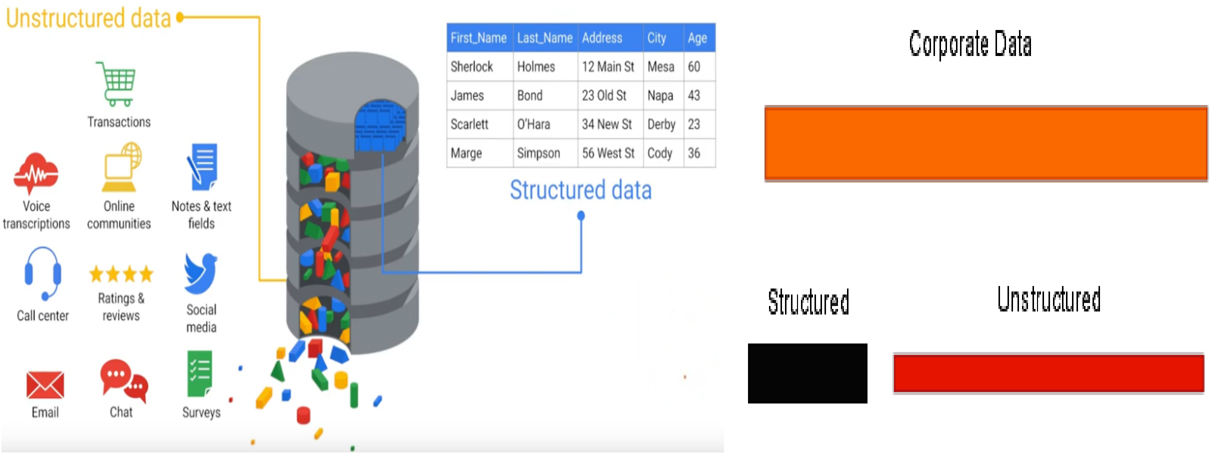
2.7 Data Science perspective Datatypes, Tools, and workflow

2.7.1 Data types and sources

As explained earlier, external and internal environments shape the business. The separate domains include corporate data inside the organization and cloud data outside. Both of these will create some data that can be used to enhance business performance. However, corporate data is the most essential.

The organization may produce a quantity of data so that the corporate data can be divided into structured and unstructured data. Usually, all the structured data is more predictable and have regularity. This is well-defined data and is stored in DBMS systems. The information can be retrieved quickly and effortlessly and is also suited for analytics. But not all data come in a structured form. Figure 2.9a shows the content view, and 2.13b shows a more flat distribution view. Around 80% of the data is unstructured. Yet are, the majority of decisions made by the minority structured data. The primary reason, it is easier to automate and analyze. Not taking advantage of the potential of unstructured data could lead to innovation failure(Inmon, 2015).

Figur 2.9: the distribution of structured and unstructured data source: (Inmon, 2015), refined by me

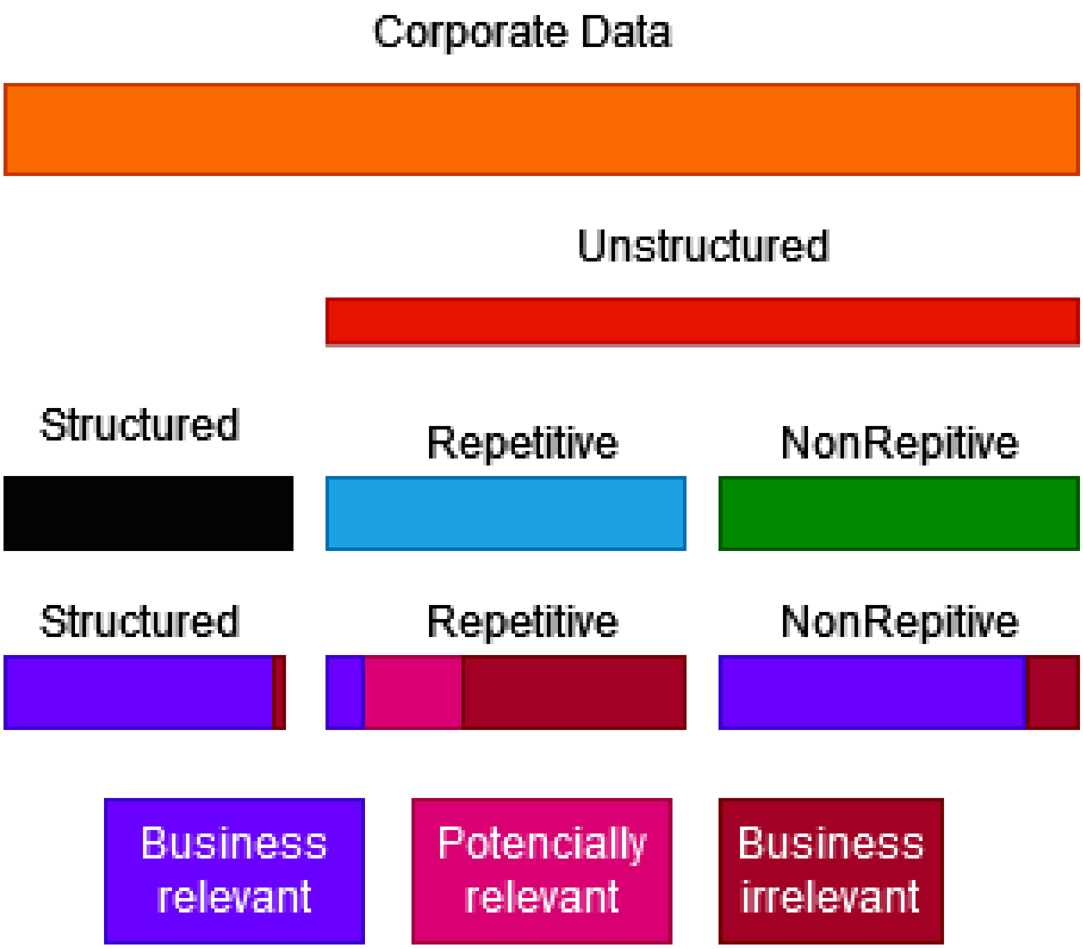


(a) Content based overview **(b)** Distribution overview over data

Further, unstructured data is usually referred to as Big Data due to its significant size. It consists of two groups repetitive tasks and non-repetitive. They have different characteristics, but because executives and managers lack data literacy knowledge. They might think the difference is trivial. It is more like a split of how you deal with them. Being more informed about data literacy, you know it is essential to access, monitor, and display unstructured repetitive material. Still, time and effort to find something useful is time-consuming, so an option would be to detect anomalies.

On the other hand, for non-repetitive parts is highly common to find business value and comes in formats such as email, call center information, and marketing. It is Connected to textual disambiguation. By reformatting, contextualize it. For business growth and development, managers and executives need to understand the differences between the data. How can you make the right decisions if you do not know what data you are working with? Figure 2.12 shows a complete view of the corporate data, including the business relevancy for the different data types.

Figur 2.10: what data type has most business value relevant(Inmon, 2015), refined by me



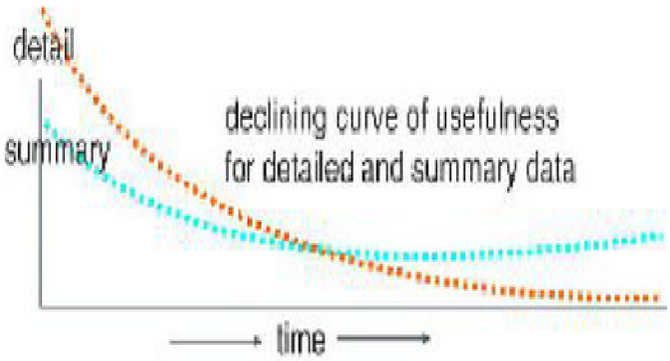
Structure data: Almost all structured business data is relevant Because the data is easy to analyze and gives business more meaning. During the project, an HR dataset was used to illustrate this data as applicable, more detailed information can be found in Appendix A

Unstructured Repetitive: Use the time to collect and sort out important values. Creating alerts about exceptions can help you to act when needed.

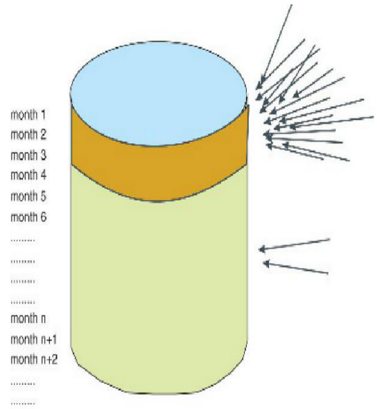
Unstructured non-repetitive: It gives meaning right away, but after a while, the data lose its value. Historical data will have much more business value by contextualizing it using NLP technology.

Data analysis of the corporate data can be an important asset, but you need find a balance between new an old data. Figure **??** describe that detailed new information is accessed often, older information are accessed less, as the time goes summary data will be more essential.

Figur 2.11: usefulness of new vs old data, Source: (Inmon, 2015)



(a) Details is more important at the begining



(b) New data accessed more then older data

The data outside an organization can be valuable for businesses (e.g.) weather forecasts, power prices, public records, or patents. In recent years, patent data have been more popular. It shows the ability to extract knowledge and put it into economic gains(why collect patent data. This data can be accessible and easily downloaded or behind a paywall. The most common way to get the data into your organization is to use web scraping, an automated way to extract a large amount of information from a website or Application programming interface(API), a set of rules on how to systems talk to each other.

2.7.2 Data science work flow and data pyramid

Before data analytics and machine learning, gathering data as information was the primary goal. However, the emergence of machine learning that uses data as input played a more critical role. Without accurate input data, the output would be impractical. When working with data, there is created a hierarchy pyramid. This pyramid got extended with data but also roles for the different levels. The enumerated section explains levels 1-6, from data collection to machine learning in production.(Datacamp Course)

1. Data acquisition: At the bottom is where data are generated or entered into the organization. This is the most crucial step for further data handling. garbage in = garbage out.
2. Data engineering: It is typically the data transformation phase. Data types will have different locations based on the required preprocessing steps, such as data

warehouse, data lake, or reporting.

3. Data analysis: Reporting an business intelligence: based on the data available, the analyst team creates automated reports and analyzes the data for new insights.

4. Data science: With clean and processed data, the next step would be to create machine learning models on the data.

5. ML: The machine learning models are running in production.

In addition, for the data science there is a framework called OSEMN this will be used during the study(Dr. Cher Han Lau, 2022).

- **O**btaining: Get the right data from the right place, is the data available inside the business or is it outside.

- **S**crubbing: Looking at the data format (rows and column) to if its clean, a lot of missing data? variables names? make sure the the data is clean and easy to work with.

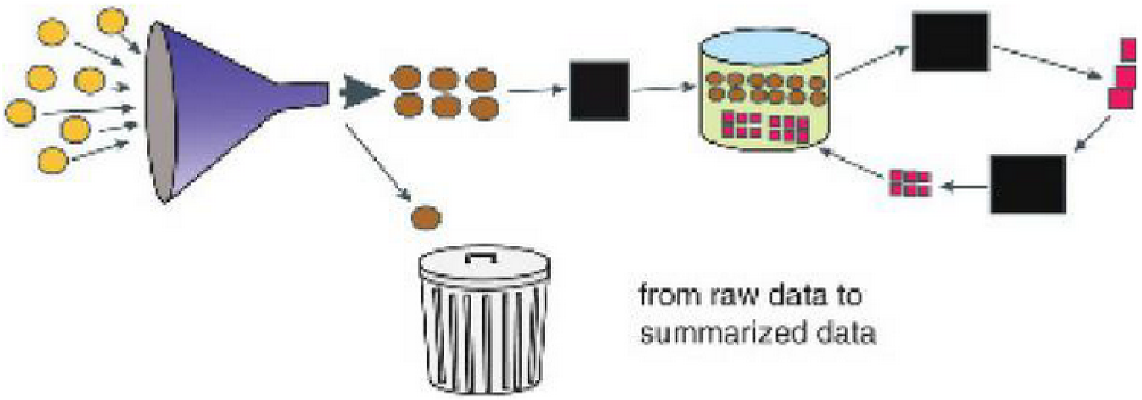
- **E**xtration: Going through the data with an EDA exploratory/explanatory data analysis creating some visuals, general statistics.

- **M**odel: Predicting something about the data, is it possible to do an classfication problem or regression analysis

- I**N**nterpert: Analysing the result from the model and maybe do some adjustments.

On a lower level in the pyramid typically data engineer, we find ETL/ ELT extract, load and transform and is more related to when data enters the organization and need to be put into the correct database. Figure shows how the data go through a funnel process.

Figur 2.12: what data type is most relevant(Inmon, 2015)



2.7.3 Different types of analytic

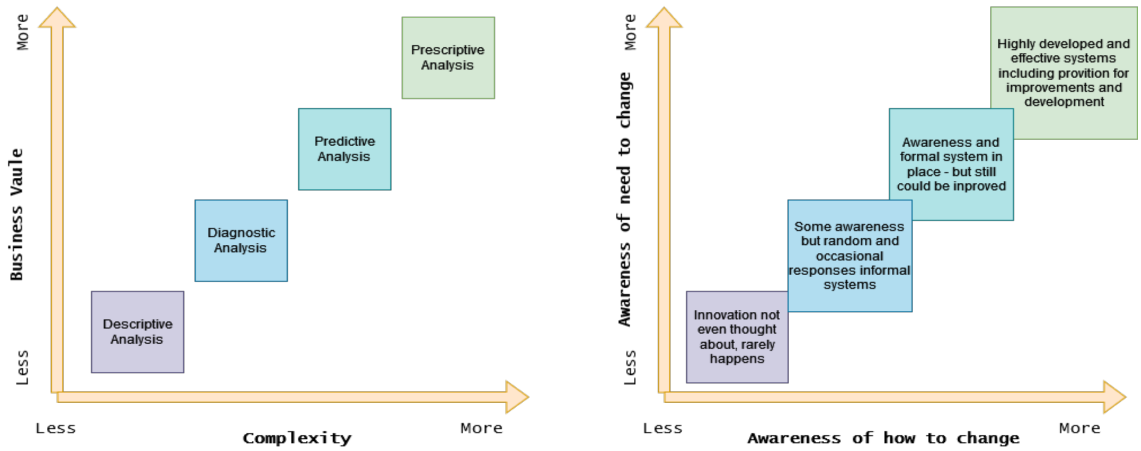
Data analytic can be divided into 4 separate types the part are explained in the bullet section (Catherine Cote, 2021)

- Descriptive Analytics: This is the foundation and easiest analysis to do, also referred to as reporting. Furthermore, answer the question "What happened" by looking at trends from raw data. (e.g.) What happened? A machine stopped.
- Diagnostic Analytic: When something important needs more investigation, to get to a business problem's root cause, compare different variables and relationships. The next step is to know "why did it happen"?(e.g.) Why did this machine stop? Power outage.
- Predictive Analytic: Considering future trends and "what might happen in the future" by comparing historical data with industry trends. (e.g.) What happens if more power outages are occurring? losing sales/ not having an operative business
- Prescriptive Analytic: "what should we do next" addressing actionable takeaways. What is the next step? Installing a UPS will run our systems while there is a power outage, and systems will still be online.

In some cases, descriptive can be enough. However, sometimes going further will provide more context to the problem. A weather subscription or SMS notification from the power operator can give helpful information about why the UPS is running.

Figure 2.13 under how I interpret data analytics with innovation. As the analytics get more complex, the organization has more knowledge about how to change. More information will enhance the need for change by moving up the y-axis about creating business value.

Figur 2.13: Similarity between data analytic types and awareness of innovation, source: own elaboration



(a) The four different types **(b)** innovation awareness

2.7.4 Tools

When working with data analytic there are different tools that can be used with different purposes.

- Python: Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. it includes libraries and packages, which encourages program modularity and code reuse for scalablilty.((Python, 2022))
- R: R is a language and environment for statistical computing and graphics. with a wide variety of statistical (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering()
- Excel: A spreadsheet where users can format, organize and calculate data.
- Power BI: Used on database or spreadsheet to create visuals and reports.
- Jupyter Notebook: A data science environment where you can code and write text blocks independent of each other. Compared to a traditional workflow for only code or text.
- SQL: structured way to store data in a table format, with rows and columns.

2.8 SME and benefits / literature gap, problem solving.

Working with business strategies for SMEs can be a daunting task in the industrial revolution 4.0. Businesses with a solid and reasonable culture for data literacy, data analytics, and understanding of the business stages are building up a more substantial competitive industry that is more sustainable and easier to work with in the future.

For native digital companies, it is much easier to see the use of data than for non-native companies.

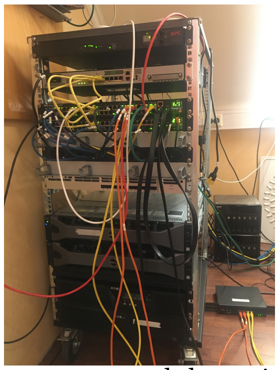
Further, the TW process can also be automated since the data generation, and collection can be a monumental task. With machine learning tools and data analytics, you have free intellectual capacity from employees, making it much more versatile in selecting your scope and the data roles in your team.

3 Methology review

3.1 Scope of the study

The study was created around a home lab scenario since the main idea is to make a proof of concept of how data analytics and information flow can give business value. This could either be done in the cloud from a cloud provider or with on-premises servers. We had on-prem equipment available, which was chosen because of our experience with those systems. When gathering valuable datasets, fictional/ dummy datasets were used to not worry about corporate data or regulations. In Figure 3.1 you can see the hardware with servers, networking and ups put in a custom made rack during building process. With possibility to expand upon in the future.

Figur 3.1: Server room, source: Own elaboration



Source: own elaboration

The datasets were found from open access, from places like Kaggle, Github, IBM and Microsoft, Some of the datacamp sources required an account. Most of the data had the CSV and Jupyter notebook format, except the database. With the homelab, it would be easier to take backup of the data and use ETL process.

3.2 Participants

In this study, the participants are my dad and me. Due to his experience as a system administrator, he knows organizational infrastructure and what considerations to make. He contributed with the hardware and installation of the operating system and hypervisor. I did the data collection and analysis. Being the owner of the equipment, it will be easy to scale up the home lab if that necessary without additional cost with an cloud solution.

3.3 Equipment used

The necessary software was installed on one server to simplify the business environment. I will not go into the technical detail about the specs to the server, but I have replaced HDD (Hard disk drives) using magnetic platters and moving parts to store data with SSDs (solid state drives) using semiconductor chips to store data and have no moving parts. To increase the read and write speed, have higher capacity on storage, use less power, and have better performance.

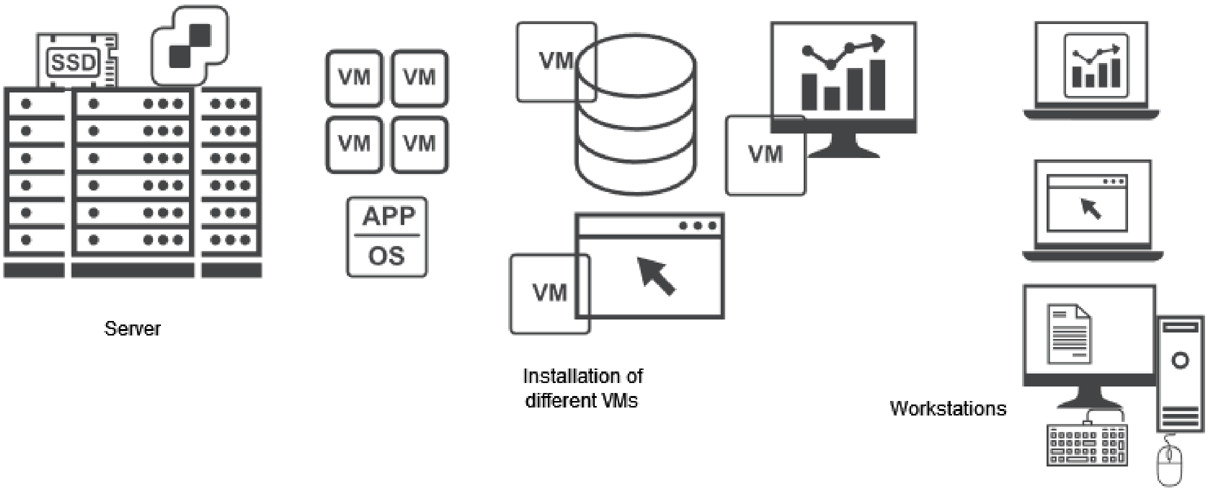
Firstly the operating system and a visualization technology needed to be installed on the hardware. For this purpose, Windows and Vmware were chosen because of prior user knowledge. Since the main goal was to have a server with multiple VMs, then a hypervisor (A hypervisor is a virtual machine monitor that runs virtual machines, on a host computer/server, sharing resources such as memory and processors and are closest to the hardware), like Vmware ESXI is needed.

After installing the hypervisor the next step where to make some VMs(virtual machine), in this case, a database, file server, and application server that have Windowws Server 2022 as the operating system.

- **Application server:**On this all the tools working with data were installed such as Python, Power BI, and Jupyter Notebook
- **Database server:** containing a database.
- **File server:** All the dataset files that were in csv or txt format lived here.

Figure 3.2 describes a schematic of the study, where we have our server with a group of VMs on the left and illustrate that the different VMs have different purposes in the middle. Lastly, the standard computer accessing the VMs for installation, coding, or viewing results is on the right.

Figur 3.2: Schematic overview, source: Own elaboration



3.3.1 Tools used

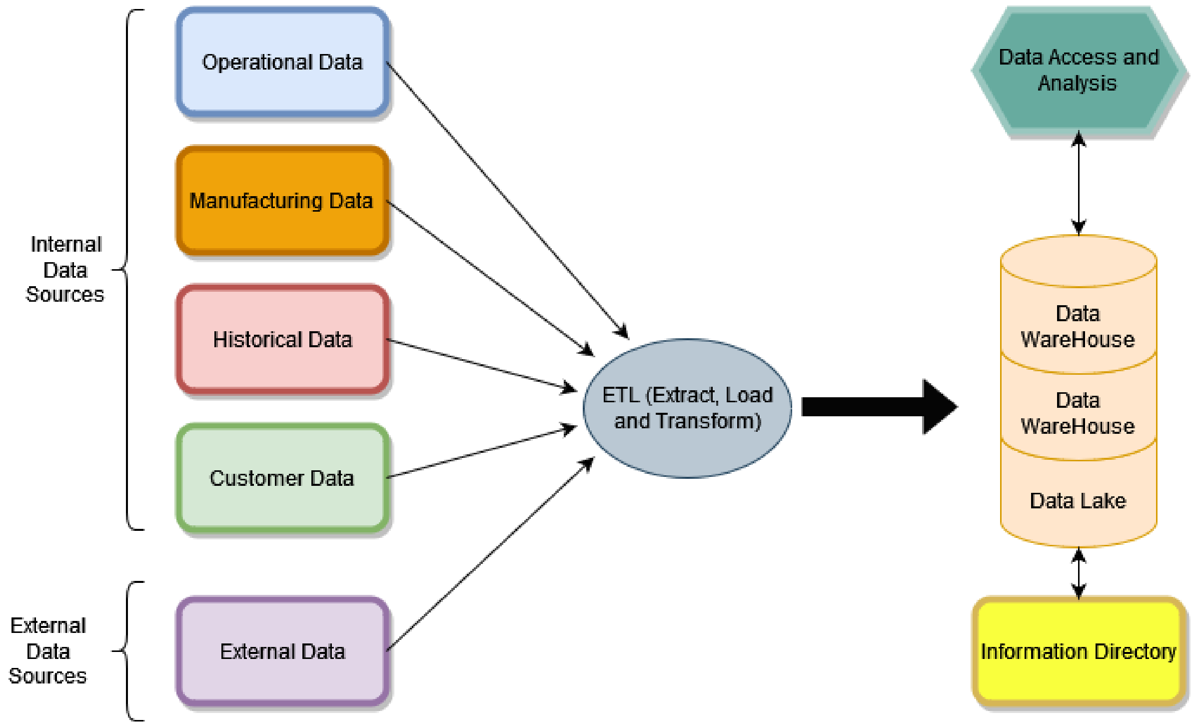
For the most part Python were the go to programming language together with its libraries and packages such as matplotlib, pandas, networkx and sklearn. To create, format, manipulate and visualize graphs using the OSEMN framework describes earlier. Together with jupyter notebook this was a intuitive working environment. Also SQL was used to querry the database.

3.4 How I did the analysis

First the goal was to get different types of data, both structured and unstructured as well as internal and external sources. Then apply the data science framework on the gathered data to extract new information about the data that was not there before just looking at the data. I will have more focus on internal data than external data.

Figure 3.3 describes the idea i had in mind. Collecting different data, do some analysis on them, and lastly display result in a report or a dashboard to share the insights. My data data warehouse is small with only one file server and a database. There are some parts that are excluded from this simplified figure is things like backup, security concerns, networking and users.

Figur 3.3: Data Flow, **Source**: Own elaboration



In this study mainly the data where downloaded as csv format and loaded into jupyter notebook using python. the using python to manipulate the data. If there were time left to explore the SQL database adventureworks, that would be great.

3.4.1 Internal Data Collection

For the internal data collection the focus was to find structured and unstructured data, most of the data a company has is internal so that will be most important to analyze. For the structured data a HR dataset is suitable for its clean format, easy to analyse and not many missing values.

3.4.1.1 HR Dataset (structured)

HR analytics by IBM were retrieved by downloading the CSV. A HR data is a set of employees data, that are typical really structured and clean. and easy to do analytic. this dataset include a label if the the employee left of not. The main goal here is to start with an Exploratory data analysis (EDA) which is a part of extracting phase in the OSEMN framework. Try to find the job roles with most attrition, and use machine learning to predict if a employee is leaving or not.

3.4.1.2 Email Data (unstructured)

A dataset retrieved from Kaggle downloaded the CSV with compromised email from enron scandal. To get some more unstructured data to look at. The reason for including a Email dataset is because this type of data is typically not data companies give out. and here lies important information, in this case the emails here in part of and bankruptcy scandal in early 2000, so the goal was to find the most important persons, who send the most emails.

3.4.1.3 Customer segmentation

Was retrieved from github. a small dataset with age, income and spending score. Trying to classify the person based spending score.

3.4.2 External data Collection

I wanted to included some of external data that are easy to collect and gather information from. even though the focus is on internal data is essential to see some options.

Patents have seen a growth recently as a valuable resource. The idea here is to create a own list of patents to look for. Then the machine learning model could collect similar pattern to the input list and plot interesting patents. Due to patent data being open it can be downloaded or queried.

Google trends search is an easy way to find something new. the algorithm will favor popular searches. By making a list of words related to this study to see if they are popular.

3.5 A novel method? or other methodologies more suitable?

The reason for this approach was tied down taking ownership of the data, standing freely to do what you wanted and use the datasets that were applicable, they may be different in terms of labels and variables. So, if one was not good just replace it. This could be done working with a company and using their data, however it can create a problem with what can be published and what is secret corporate data.

This could be a novel method because I identified a gap between working as a traditional business and the newer businesses. By looking at how to improve their situation to be better, if they not want to change the main business can be in danger. Learning new skills and take decision is something that is all day, so why do not make good ones.

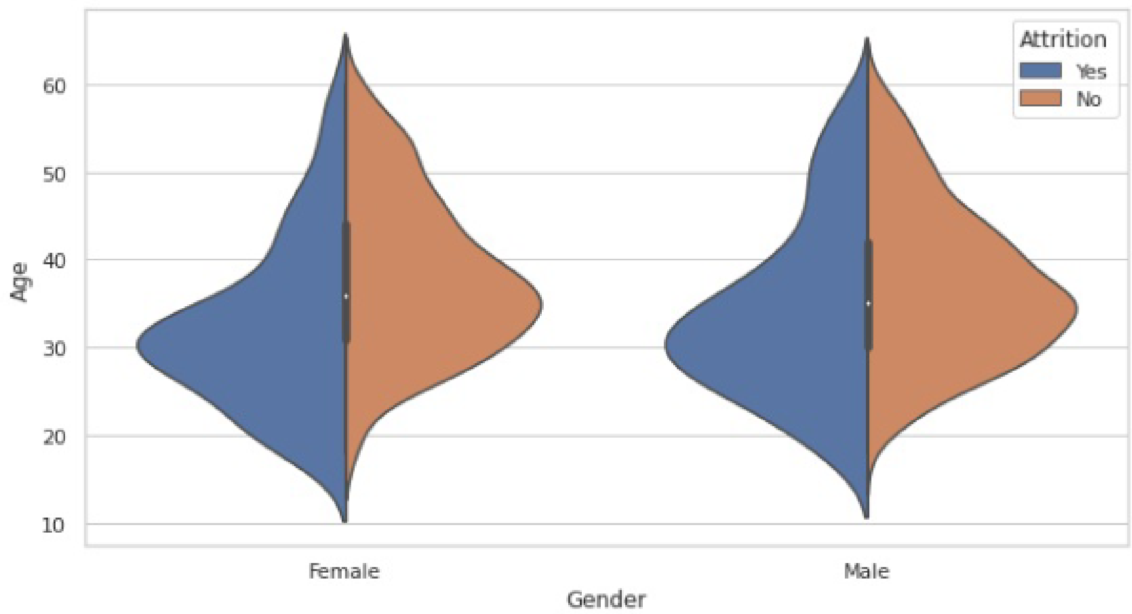
4 Results

4.1 Internal Data

4.1.1 Structured

The HR dataset, due to its clean format it was easy to get started analyzing the data. With a exploratory mindset finding as much information about the dataset as possible. In this case it turned out that young single men were the ones who left as shown on figure 4.1 below.

Figur 4.1: Attrition based on age and gender, source: own elaboration

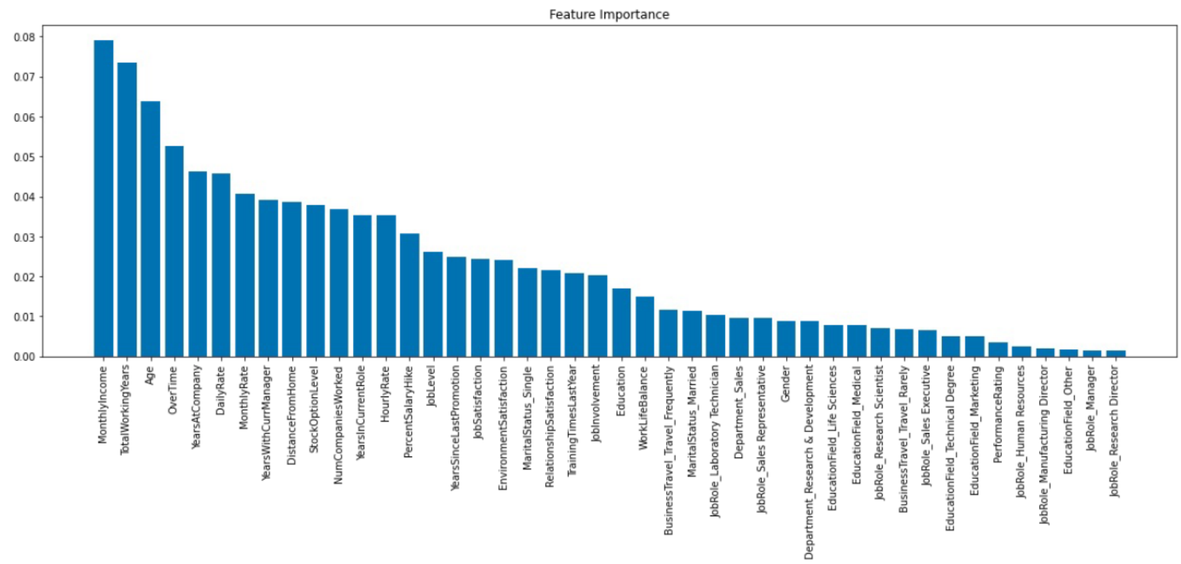


After exploring the some more it was possible to identify which job role had the highest attrition compared with monthly income shown in figure 4.2. With a more important role the person is less likely to leave.

Figur 4.2: Attrition compared to role and income, source: own elaboration

Going a little deeper into the data, Since this dataset included a label for attrition, this could be used to classify if an employee would leave of not. going through a machine learning model I could retrieve the most significant variables influencing attrition. Shown in figure 4.3. All the details about the HR dataset can be found in Appendix A5.

Figur 4.3: Attrition compared to role and income, source: own elaboration



For a company this is useful information due to their employee will be the most important asset, and keeping track of key employees. this will also work in a requiting phase when starting to develop a road map of a career path, this can then be updated when you need it in an interactive dashboard.

4.1.2 Unstructured

Moving over to the email data, this needed more work before it was usable. It contain all the elements in two columns, after splitting with this code

```
# With the function below, we can extract the information we want from the "message"
def get_field(field, messages):
    column = []
    for message in messages:
        e = emial.message_from_string(message)
        column.append(e.get(field))
    return column
```

Figure 4.4 show before and after cleaning.

Figur 4.4: Show both clean and non clean data, source: own elaboration

	file	message
0	allen-p/_sent_mail/1.	Message-ID: <18782981.1075855378110.JavaMail.e...
1	allen-p/_sent_mail/10.	Message-ID: <15464986.1075855378456.JavaMail.e...
2	allen-p/_sent_mail/100.	Message-ID: <24216240.1075855687451.JavaMail.e...
3	allen-p/_sent_mail/1000.	Message-ID: <13505866.1075863688222.JavaMail.e...
4	allen-p/_sent_mail/1001.	Message-ID: <30922949.1075863688243.JavaMail.e...

(a) not clean data

	index	file	message	date	subject	From	To	
0	0	allen-p/_sent_mail/1.	Message-ID: <18782981.1075855378110.JavaMail.e...	Mon, 14 May 2001 16:39:00 -0700 (PDT)		phillip.allen@enron.com	tim.belden@enron.com	\Philli
1	1	allen-p/_sent_mail/10.	Message-ID: <15464986.1075855378456.JavaMail.e...	Fri, 4 May 2001 13:51:00 -0700 (PDT)	Re:	phillip.allen@enron.com	john.lavorato@enron.com	\Philli
2	2	allen-p/_sent_mail/100.	Message-ID: <24216240.1075855687451.JavaMail.e...	Wed, 18 Oct 2000 03:00:00 -0700 (PDT)	Re: test	phillip.allen@enron.com	leah.arsdall@enron.com	\Ph

(b) Clean data

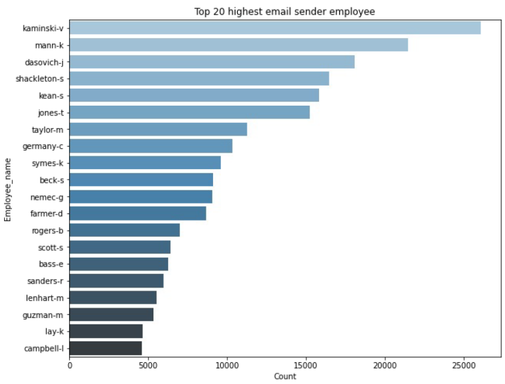
Starting with the data it looked something like figure 4.4a, difficult to work with and little to no information. Email contain a lot of headers and information that needed to be filtered out. After prepossessing 4.4b . After some pre processing. data looked like figurew

4.2.

now the data was more approachable, and since the data was unknown starting with an EDA(exploritory data anlysis) is an good option. from This EDA I retrived charlennght of the emails, to and also find the persons that send the amount of emails.

From figure 4.5 we can see the persons who have sent the most email. in a period of roughly one year the top person have 25000 emails. The persons with the most mentions was the top executives.

Figur 4.5: Send most email and most mentions, source: own elaboration



(a) persons who send the most email

```
Most mentioned people in mails

Name          -
Jeff          PERSON    47
Steve         PERSON    40
Kay Mann/C    PERSON    37
John          PERSON    35
Mark          PERSON    33
Kay           PERSON    26
Mike          PERSON    26
Mary          PERSON    22
Vince J Ka    PERSON    21
Rick          PERSON    21
dtype: int64
```

(b) Mention the most in email

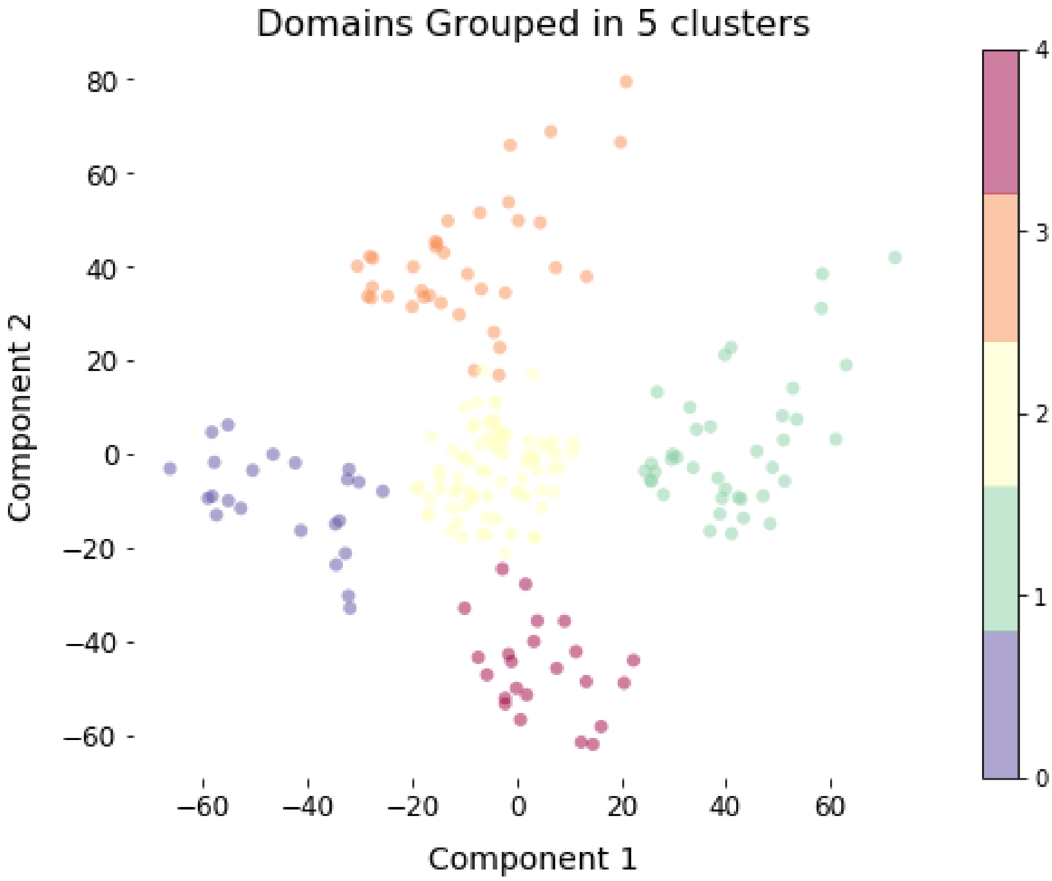
Making a reference point once in a while, you could get a trend over who is talking to who. and if the connections change over time.

Furthermore this connections between the persons were put into a network diagram, to really who is talking. In a network diagram like that supplychain, shareholders and stake holders could also be included to give a picture of the important connections. The appendix A2 describes this in more detail.

4.2 Customer segmentation

This example was created by (ayushsanghavi, 2022) and he classified the customer on spending score. based on the age, income, gender he was able to classify into 5 different groups. All information about the dataset is in Appendix A3

Figur 4.6: Customer classification, source:(ayushsanghavi, 2022)



4.3 External Data

I have not worked on external that much, so online help was much appreciated. Usually, the amount of data is enormous and hard to work with if you do not know what to do. I discovered that my competence was insufficient to work with this data.

4.3.1 Patent

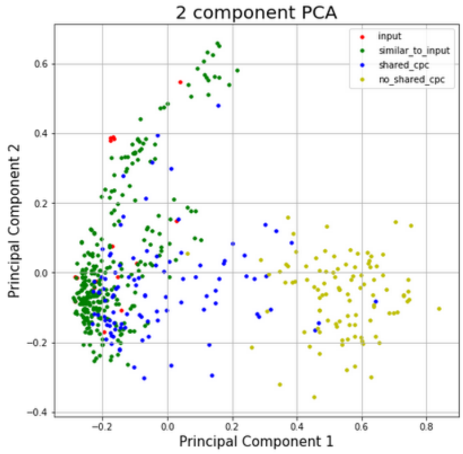
I have not worked with patent data before. All credit to (ostegm, 2022) for the example he provided. From his example, we can see that a list of exciting patents is transformed into a visual representation of the patents from the list in red. Similar patents colored blue and green. And patent with no relation in yellow. More information can be found in Appendix A6.

Analyzing like this will give a business a piece of crucial information about innovation and strategy. Putting them at the top of digital transformation.

Figur 4.7: Transforming patents list to visual, source:(ostegm, 2022)

```
# A list of patents we care about. In this case we're go:
input_patents = [
    'US-7292636-B2',
    'US-6115503-A',
    'US-6812873-B1',
    'US-6825782-B2',
    'US-6850175-B1',
    'US-6934331-B2',
    'US-6967601-B2',
    'US-7068721-B2',
    'US-7183951-B2',
    'US-7190289-B2',
    'US-7199836-B1',
    'US-7298303-B2',
    'US-7310372-B2',
    'US-7339991-B2',
    'US-7346216-B2',
    'US-7474699-B2',
]
# Converts input list into a string for use in queries.
input_patents_str = '(' + str(input_patents)[1:-1] + ')'
input_patents_str
```

(a) patents as a list

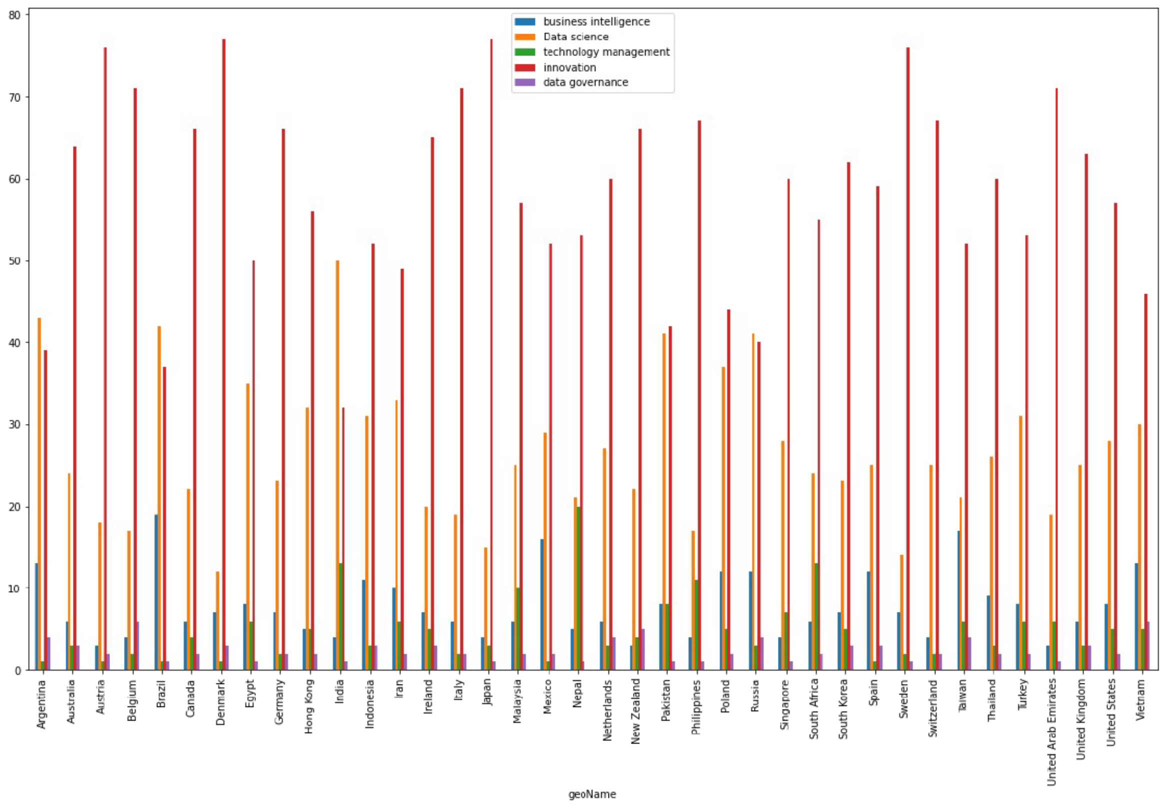


(b) Similar patents visualized

4.3.2 Trends search

For the trend search I provided my own list with words related to this study and sorted by region to see how popular the search terms where. Figure **??** show the output.

Figur 4.8: Attrition compared to role and income, source: own elaboration



Normally a search trend can change fast, but keeping a eye out for interesting and useful information is a good idea. The Google search trends are limited to only five words, both research topics and ordinary search terms can be combined, but for best result use one type of search.

4.4 Other findings

Is not only the technology that is part of an digital transformation success, it also comes down to business commitment / ownership on transformational factors. without it is hard to keep innovating.

The management style is changing throughout a company's lifecycle, and this will play a role business performance. Newer data native companies are already thinking how to get the most out of the data available. To keep up the older businesses need to find out that data is not dangerous and everybody use it.

When to use which type of data analytics or the storytelling is not easy. Experimenting with different use cases to is essential to get a feel for what works and not. However, with continuous skills development for employees and managers about data literacy, is crucial for success.

these are connected to the roles and skills, because that is one of the crucial parts of succeeding, continuous development of employees and managers about what data to use, telling good data stories and know where to look for data with the most business relevance data both internal or external.

5 Discussion

5.1 Future of data analytic

When the business have a mutual understanding of concepts like data governacne, data literacy and data analytics. it will help make better decisions, from the HR dataset the analysis tell us that the biggest workforce with sales representatives, human resource and research scientist have average working time of 1 year and are the employees with the highest attrition. Typical there is high turnover on these roles, but there is a research director that is also leaving that have been there for 15 years. That is more critical, if events like this are addressed then the organization, measures against it.

Let's say you are replacing a flow computer every 5 years, on a factory and you have 50 in total. short while after replacing them, there is problems with 5 and they need to be changed. But after the change you still have problems. With no meta data is hard to find the cause and every flow computer need to checked. However, with meta data such a geolocation, cabinet type, production data, timestamps, and other equipment in the cabinet. You may find out that the new flow computer have a different cabinet, some of the cabinet are exposed to sun. the cabinet heats up during the main peak production hour. Because of the heat up the cabinet a signal processor malfunction cause the flow computer to fail.

Dealing with data have it roots in structured internal data that are accessed often. With new tools and knowledge all employees have the ability to extract useful information from different sources. Using NLP technology to contextualize internal processes or have an eye on the external environment. Have better teamwork across the organization and employees that want to learn more.

I think that many SMEs are in the phase 3-4 of the growth model and the 2 stage in digital transformation. There is a clear external pressure that influence the business performance, it is hard to keep up with the fast change, and most of the vendors just want to sell. The management not necessary have direction of phase 2 if its a long time since a big change like this, the management practices have spouted solid and is hard to change.

5.2 Anwsering Research questions and hypothesis

Research questions

1. Data management and innovation management are closely related, 2.13 managers and entrepreneurs can get knowledge that was earlier unknown. There is still a lot of unexplored data through NLP technology.
2. Business culture is strongly connected to digital transformation, in the start phase it does not feel like like due to operational IT tasks. After that everybody can use data tools and vital that all have the same mutual understanding.
3. The skills needed is related to data literacy, how to communicate with data. Data governance, is the quality of data good. And, data analytics, what tools should I use in different situations.
4. I can see the potential use of the data, but I need to do some more investigation about this type of data.

Hypothesis

- Data analytics can help business growth with collecting the right data to the right time, having automated process that accessible on the computer instead of driving around. Doing a decision you are unsure of the outcome can be difficult. If this were backup by data, the decision would have been more confident.

5.3 Future improvements

Future improvements could be to investigate more about the unexplored external data that can give business potential. Also, learn more about SQL structured data analysis. This included just a small sample of business data. But there is much more data that be useful to look at such as supply chain, marketing, finance to mention a few.

However, there is some literature that could be interesting to look at that can complement this study.

- **Knowledge Management:** Is essential to the process of organizing, sharing, create, utilize collective knowledge within an organization. because it
- **Critical Thinking:** Beacause it
- **Dynamic Capabilities:** is about an companys's ability to proactively shape change. integrate, and recon figure internal and external resources to address rapidly changing business environments. proactivly shape change
- **Absorptive Capacity:** addresses the ability to apply knowledge for improving organizational learning. focusing on the outside of business learn

Now I am more aware need to check what I aquired if this is a real business problem that companies need help with.

5.4 Emerging Business Idea

To succeed in a digital transformation and stay at the top a business need culture, skills and technology. For a smaller business it can be beneficial to outsource some of this to get additional help, but for bigger businesses many processes should be integrated internally for better steadfast.

My idea is divided into three parts.

1. Firstly is to get started about is to have an internal audit about processes, competence, and skills to find the key employees. The main goal with this part is identify crucial employees that drive the business forward and classify them as vital assets.
2. Secondly developing a digital transformation strategy and stick to it. Explaining the different fundamentals of data literacy, governance, and analytics with benfits.
3. Finally, get some taught about useful external data like the patents, if applicable or power price, gas price.

If you have an employee in the upper quartile in competence and skill, but management always add new projects and the urgency is badly defined. Over time this will lead to resignation and under performing of the employee. Even if they say its to much, the only answer is why are you under performing? The real issue is management have not identified this as a issue.

Knowing what you are working with of competence you can then make strategic plans with data in its core. addressing the benefits and problems with a digital transformation.

Are the organization on top of the digital transformation, then some external data could bring in some challenges and new information.

Altogether people trumps technology, so that means if the culture, strategies and commitment is not in place, there is no need for advanced ELT, datawarehouses or data analytics. That said, when you automate tedious processes, you release intellectual capacity to employees, making them more robust to work with what matters. Let people be more loose about roles and structure, a finance person can help select the best new IT equipment with a spec list from IT. And IT can help with data analysis for HR until they get their own person.

6 Conclusion

From this study we can see that the 4 industrial revolution is here and will continue to be important. Getting more insights about your own business data will help to stay more competitive, have better internal processes, and information about what to do next. Even though good insights are helpful you also need new business culture and strategies to succeed in a digital transformation process.

With a business culture that have data in its core and continuous learning about how the data and business interact. This will benefit the organization in the long run. In, addition, the most significant data is inside the organization, so a solid data infrastructure is essential for good data analysis.

I think from my experience that many SMEs can be out of business if they not capture the opportunity to enhance data governance, data literacy, and data analytic. For the reason being that managers and executives in non-native businesses do not understand, the usefulness of the data and how proceed with strategies with data as a core. They do not have the same relationship to Agile method of working.

If I would give an estimate based on experience it would be stage 2 in digital transformation and and delegation phase for the business growth. There are no overall strategy for data oriented business models and lack of learning development for employees.

Being proactive about how you view data as a resource instead of just information, the business will gain great results. If you have the data in front of you, just get started doing some small analysis and tell about it with some good story telling techniques and review the feedback. Under are three takeaways I think is essential for success. Because there is a potential business idea to this.

1. Find out were the organization are. Mainly descriptive and diagnosis analysis on internal data. address key data, persons, and competence and align it in a skill map.

2. To get more commitment towards data as a resource create a data literacy program, that everybody can participate in, then everybody will feel connected and engaging using data.

3. Look externally to get more info, typically unknown extract data from patents, APIs

or other places that is relevant.

Finally this is just a proof of concept that show how data analytics could be used to find hidden insights, and how to relate it to business growth. In the real world other problems can occur like stress, unexpected perception or resistance from executives that can dismantle what is already build up.

References

ayushsanghavi (2022). customer segmentation. retrived 30. january 2022, from: https://github.com/ayushsanghavi/Customer-Segmentation-using-K-means.

Besker, T., Martini, A., og Bosch, J. (2019). Software developer productivity loss due to technical debt—a replication and extension study examining developers' development work. *Journal of Systems and Software*, 156:41–61.

Bianchini, M. og Michalkova, V. (2019). Data analytics in smes. (15).

Blackburn, M., Alexander, J., Legan, J. D., og Klabjan, D. (2017). Big data and the future of rd management: The rise of big data and big data analytics will have significant implications for rd and innovation management in the next decade. *Research technology management*, 60(5):43–51.

Burke W.W , Litwin, G.H (1992). *A causual model of organizational performance and change.* Journal of Management, 18(3), Vienna, Austria.

Catherine Cote (2021). 4 types of data analytics to improve decision-making. Retrived 12. April 2022, from: https://online.hbs.edu/blog/post/types-of-data-analysis.

Clear, J. (2018). *Atomic habits: An easy & proven way to build good habits & break bad ones.* Penguin.

David Reinsel, John Gantz, John Rydning (2018). The digitization of the world. Retrived 12. April 2022, from: https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf.

Dr. Cher Han Lau (2022). 5 steps of a data science project lifecycle. retrived 10. march 2022, fra: https://towardsdatascience.com/5-steps-of-a-data-science-project-lifecycle-26c50372b492.

Dykes, B. (2020). *Effective data storytelling : how to drive change with data, narrative, and visuals.* Wiley, Hoboken, New Jersey, 1st edition. edition.

Hayes, J. (2018). *The Theory and Practice of Change Management.* Bloomsbury Publishing Plc, London.

Inmon, W. H. (2015). *Data architecture : a primer for the data scientist : big data, data warehouse and data vault.* Morgan Kaufmann, Amsterdam, Netherlands, 1st edition. edition.

Knaflic, C. N. (2015). *Storytelling with data : a data visualization guide for business professionals.* Wiley, Hoboken, New Jersey, 1st edition. edition.

Knaflic, C. N. (2020). *Storytelling with data : let's practice!* Wiley, Hoboken, New Jersey, 1st edition. edition.

Larry E. Greiner (1998). Evolution and revolution as organizations grow. Retrived 12. April 2022, from: https://hbr.org/1998/05/evolution-and-revolution-as-organizations-grow.

LaValle, S., Lesser, E., Shockley, R., Hopkins, M. S., og Kruschwitz, N. (2011). Big data, analytics and the path from insights to value. *MIT sloan management review*, 52(2):21–32.

Nehme, A. (2022). The rise of hybrid jobs the future of data skills[ep 80].

ostegm (2022). plotting similar pattens. retrived 30. january 2022, from: https://www.kaggle.com/code/ostegm/plotting-similar-patents/notebook.

Paul Leinwand P. and Mahadeva M. Matt Mani (2021). Digitizing isn't the same as digital transformation. Retrived 12. April 2022, from: https://hbr.org/2021/03/digitizing-isnt-the-same-as-digital-transformation.

PLURALSIGHT (2021). Perspective on technology skills development. Retrived 12. April 2022, from: https://www.pluralsight.com/resource-center/tech-anthology/technology-skill-development.

Python (2022). what is python.

R (2022). what is r.

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Saldanha, T. (2019). *Why Digital Transformations Fail*. Berrett-Koehler Publishers, 1st edition. edition.

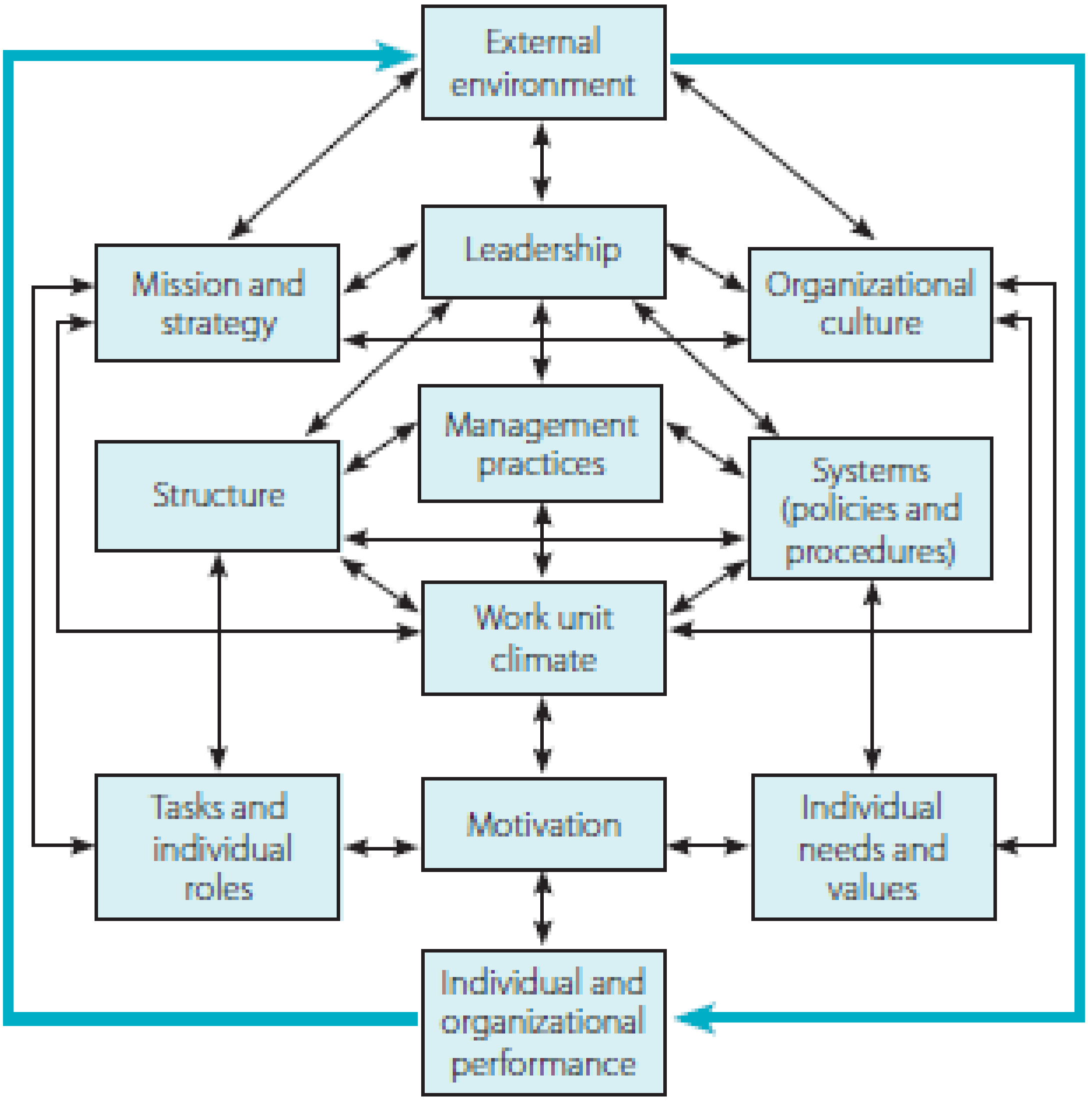
Srdjan Dragutinovic (2021). The role of data analytics in smes and why it shouldn't be dismissed. Retrived 12. April 2022, from: https://www.bit.com.au/guide/the-role-of-data-analytics-in-smes-and-why-it-shouldnt-be-dismissed-563921.

Zabala-Iturriagagoitia, J. M. (2014). Innovation management tools: implementing technology watch as a routine for adaptation. *Technology Analysis & Strategic Management*, 26(9):1073–1089.

Appendix

A1 Burke and litwin organizational model

Figur A1.1: Burke and litwin organizational model, Source:



A2 Search Trends

Added as html file A2

A3 Customer segmentation

In [1]:

```
#Numpy and pandas easy-to-use data structures and data analysis tools for the Python prog
ramming language.
#It allows for fast analysis and data cleaning and preparation
import numpy as np
import pandas as pd
# For visualizations
import matplotlib.pyplot as plt
# For regular expressions
import re
# For handling string
import string
# For performing mathematical operations
import math
#data visualization library based on matplotlib.
import seaborn as sns
import sklearn
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.preprocessing import MinMaxScaler
```

In [2]:

```
# Importing dataset
customers=pd.read_csv('/Users/ayushsanghavi/Downloads/customer-segmentation-dataset/Mall_
Customers.csv')
print("Shape of data=>",customers.shape)
```

```
Shape of data=> (200, 5)
```

In [3]:

```
customers.head() #printing top 5 lines of the dataset
```

Out[3]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

/Exploring the data

Now, we check the quality of the data and the distribution of the variables.

First, we check that if there is any missing value in the dataset. K-means algorithm is not able to deal with missing values.

In [4]:

```
print(f"Missing values in each variable: \n{customers.isnull().sum()}")
```

```
Missing values in each variable:
CustomerID                0
Gender                    0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

We have no missing data. Next, we check if there is any duplicate rows or no.

In [5]:

```
print(f"Duplicated rows: {customers.duplicated().sum()}")
```

```
Duplicated rows: 0
```

We check the data type of each variable in the DataFrame. Categorical variables cannot be handled directly. K-means is based on distances. The approach for converting those variables depend on the type of categorical variables.

In [6]:

```
print(f"Variable:                         Type: \n{customers.dtypes}")
```

```
Variable:                          Type:
CustomerID                         int64
Gender                            object
Age                                int64
Annual Income (k$)                 int64
Spending Score (1-100)             int64
dtype: object
```

Observing the distribution of variables. Define two functions. The first one will retrieve descriptive statistics of the variables. The second one will help us graph the variable distribution.

In [7]:

```
def statistics(variable):
    if variable.dtype == "int64" or variable.dtype == "float64":
        return pd.DataFrame([[variable.name, np.mean(variable), np.std(variable), np.med
ian(variable), np.var(variable)]],
                            columns = ["Variable", "Mean", "Standard Deviation", "Median
", "Variance"]).set_index("Variable")
    else:
        return pd.DataFrame(variable.value_counts())
```

In [8]:

```
def graph_histo(x):
    if x.dtype == "int64" or x.dtype == "float64":
        # Select size of bins by getting maximum and minimum and divide the substraction
by 10
        size_bins = 10
        # Get the title by getting the name of the column
        title = x.name
        #Assign random colors to each graph
        color_kde = list(map(float, np.random.rand(3,)))
        color_bar = list(map(float, np.random.rand(3,)))

        # Plot the displot
        sns.distplot(x, bins=size_bins, kde_kws={"lw": 1.5, "alpha":0.8, "color":color_k
de},
                              hist_kws={"linewidth": 1.5, "edgecolor": "grey",
                                        "alpha": 0.4, "color":color_bar})
        # Customize ticks and labels
        plt.xticks(size=14)
        plt.yticks(size=14);
        plt.ylabel("Frequency", size=16, labelpad=15);
        # Customize title
        plt.title(title, size=18)
        # Customize grid and axes visibility
        plt.grid(False);
        plt.gca().spines["top"].set_visible(False);
        plt.gca().spines["right"].set_visible(False);
        plt.gca().spines["bottom"].set_visible(False);
        plt.gca().spines["left"].set_visible(False);
    else:
```

```
        x = pd.DataFrame(x)
        # Plot
        sns.catplot(x=x.columns[0], kind="count", palette="spring", data=x)
        # Customize title
        title = x.columns[0]
        plt.title(title, size=18)
        # Customize ticks and labels
        plt.xticks(size=14)
        plt.yticks(size=14);
        plt.xlabel("")
        plt.ylabel("Counts", size=16, labelpad=15);
        # Customize grid and axes visibility
        plt.gca().spines["top"].set_visible(False);
        plt.gca().spines["right"].set_visible(False);
        plt.gca().spines["bottom"].set_visible(False);
        plt.gca().spines["left"].set_visible(False);
```

In [9]:

```
spending = customers["Spending Score (1-100)"]
```

In [10]:

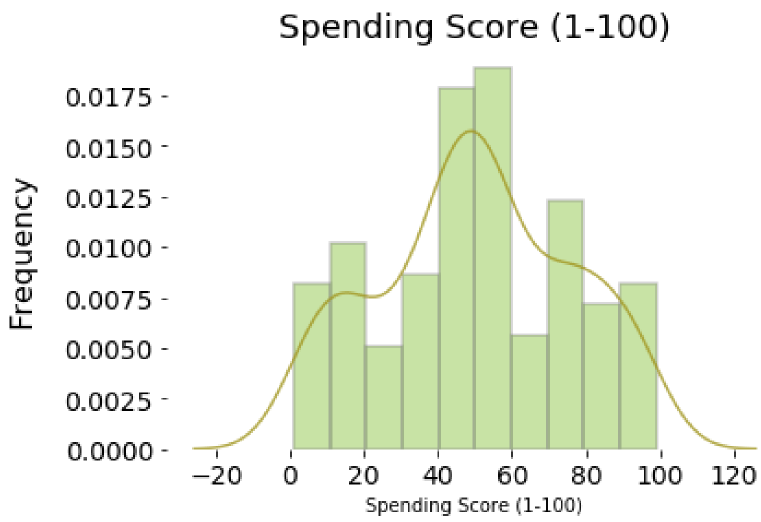
```
statistics(spending)
```

Out[10]:

Variable	Mean	Standard Deviation	Median	Variance
Spending Score (1-100)	50.2	25.758882	50.0	663.52

In [11]:

```
graph_histo(spending)
```



Now lets check age

In [12]:

```
age = customers["Age"]
statistics(age)
```

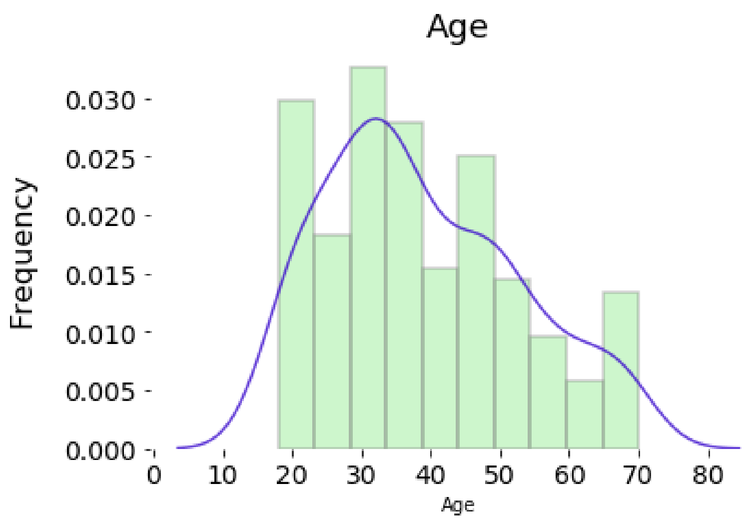
Out[12]:

Variable	Mean	Standard Deviation	Median	Variance
Age	38.85	13.934041	36.0	194.1575

In [13]:

In [13]:

```
graph_histo(age)
```



Let's explore annual income

In [14]:

```
income = customers["Annual Income (k$)"]
```

In [15]:

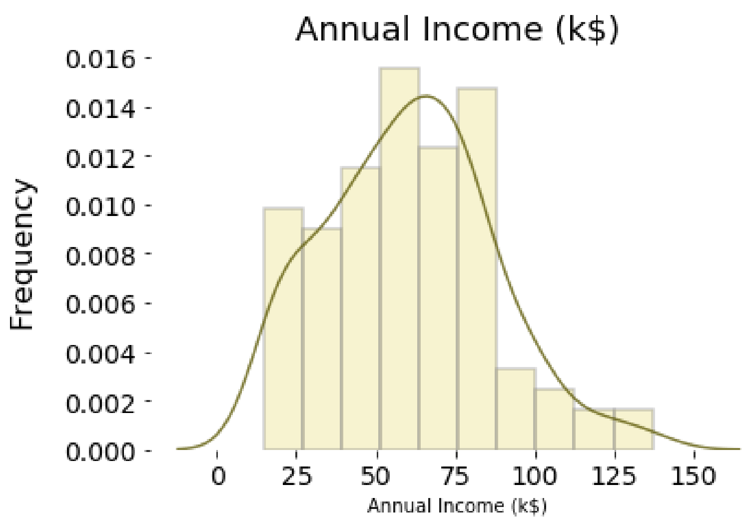
```
statistics(income)
```

Out[15]:

	Mean	Standard Deviation	Median	Variance
Variable				
Annual Income (k$)	60.56	26.198977	61.5	686.3864

In [16]:

```
graph_histo(income)
```



Gender now

In [17]:

```
gender = customers["Gender"]
```

In [18]:

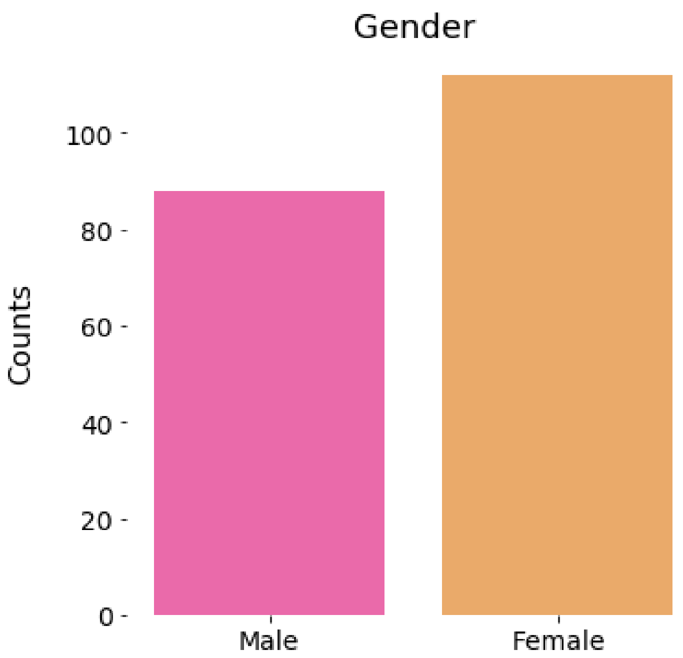
```
statistics(gender)
```

Out[18]:

	Gender
Female	112
Male	88

In [19]:

```
graph_histo(gender)
```

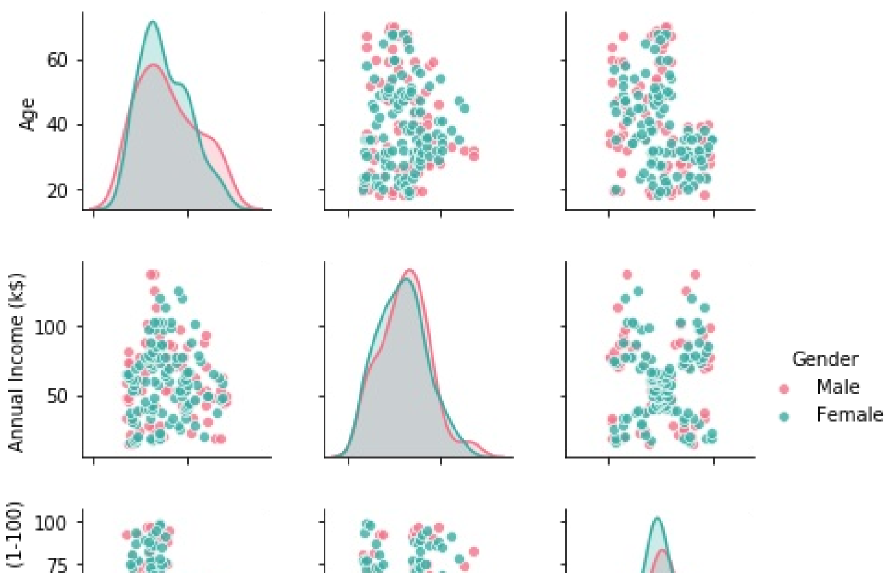


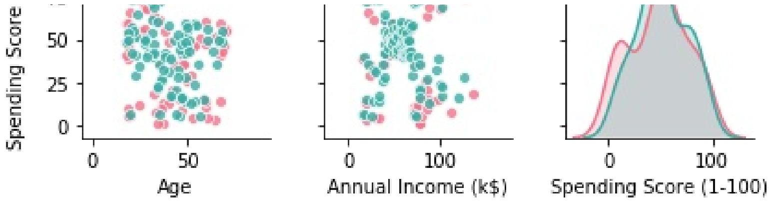
Correlation between parameteres:

We will analyze the correlation between the numeric parameters. For that, we'll use the pairplot seaborn function. We want to see whether there is a difference between gender. So, we are going to set the hue parameter to get different colors for points belonging to female or customers.

In [20]:

```
sns.pairplot(customers, x_vars = ["Age", "Annual Income (k$)", "Spending Score (1-100)"]
,
               y_vars = ["Age", "Annual Income (k$)", "Spending Score (1-100)"],
               hue = "Gender",
               kind= "scatter",
               palette = "husl",
               height = 2,
               plot_kws={"s": 35, "alpha": 0.8});
```





In order to apply K-means, we need to meet the algorithm assumptions.

K-means assumes:

Cluster's shape: The variance of the distribution is spherical meaning that clusters have a spherical shape. In order for this to be true, all variables should be normally distributed and have the same variance. Clusters' Size: All clusters have the same number of observations. Relationship between variables: There is little or no correlation between the variables. In our dataset, our variables are normally distributed. Variances are quite close to each other. Except for age that has a lower variance that the rest of the variables. We could find a proper transformation to solve this issue. We could apply the logarithm or Box-Cox transformation. Box-Cox is a family of transformations which allows us to correct non-normal distributed variables or non-equal variances.

Dimensionality reduction After we checked that we can apply k-means, we can apply Principal Component Analysis (PCA) to discover which dimensions best maximize the variance of features involved.

Principal Component Analysis (PCA) First, we'll transform the categorical variable into two binary variables.

In [21]:

```
customers["Male"] = customers.Gender.apply(lambda x: 0 if x == "Male" else 1)
```

In [22]:

```
customers["Female"] = customers.Gender.apply(lambda x: 0 if x == "Female" else 1)
```

In [23]:

```
X = customers.iloc[:, 2:]
```

In [24]:

```
X.head()
```

Out[24]:

	Age	Annual Income (k$)	Spending Score (1-100)	Male	Female
0	19	15	39	0	1
1	21	15	81	0	1
2	20	16	6	1	0
3	23	16	77	1	0
4	31	17	40	1	0

In [25]:

```
# Apply PCA and fit the features selected
pca = PCA(n_components=2).fit(X)
```

During the fitting process, the model learns some quantities from the data: the "components" and "explained variance".

In [26]:

```
print(pca.components_)
```

```
[[-1.88980385e-01  5.88604475e-01  7.86022241e-01  3.32880772e-04
  -3.32880772e-04]
 [ 1.30957602e-01  8.08400899e-01 -5.73875514e-01 -1.57927017e-03
```

```
   1.57927017e-03]]
```

In [27]:

```
print(pca.explained_variance_)
```

```
[700.26450987 684.33354753]
```

The vectors represent the principal axes of the data. The length of the vector indicates the importance of that axis in describing the distribution of the data. The projection of each data point onto the principal axes are the principal components of the data.

In [28]:

```
# Transform samples using the PCA fit
pca_2d = pca.transform(X)
```

We can represent this using a type of scatter plot called biplot. Each point is represented by its score regarding the principal components. It is helpful to understand the reduced dimensions of the data. It also helps us discover relationships between the principal components and the original variables.

K-means clustering:

In order to cluster data, we need to determine how to tell if two data points are similar. A proximity measure characterizes the similarity or dissimilarity that exists between objects.

We can choose to determine if two points are similar. So if the value is large, the points are very similar. Or choose to determine if they are dissimilar. If the value is small, the points are similar. This is what we know as "distance".

There are various distances that a clustering algorithm can use: Manhattan distance, Minkowski distance, Euclidean distance, among others.

$$\sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

K-means typically uses Euclidean distance to determine how similar (or dissimilar) two points are.

First, we need to fix the numbers of clusters to use.

There are several direct methods to perform this. Among them, we find the elbow and silhouette methods.

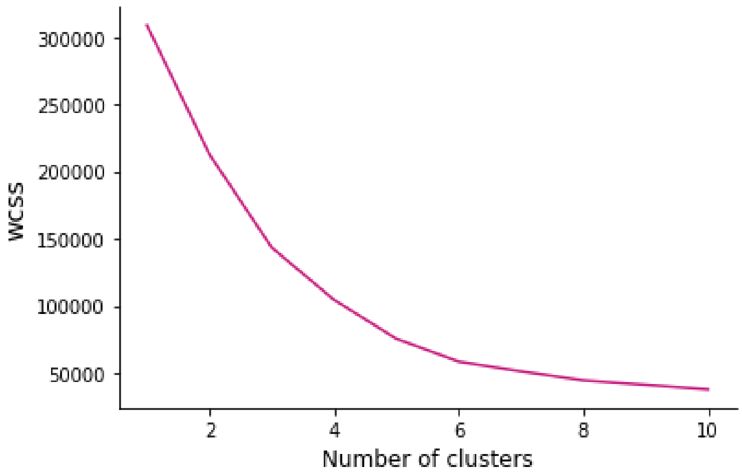
We'll consider the total intra-cluster variation (or total within-cluster sum of square (WSS)). The goal is to minimize WSS.

The Elbow method looks at how the total WSS varies with the number of clusters. For that, we'll compute k-means for a range of different values of k. Then, we calculate the total WSS. We plot the curve WSS vs. number of clusters. Finally, we locate the elbow or bend of the plot. This point is considered to be the appropriate number of clusters.

In [29]:

```
wcss = []
for i in range(1,11):
    km = KMeans(n_clusters=i,init='k-means++', max_iter=300, n_init=10, random_state=0)
    km.fit(X)
    wcss.append(km.inertia_)
plt.plot(range(1,11),wcss, c="#c51b7d")
plt.gca().spines["top"].set_visible(False)
plt.gca().spines["right"].set_visible(False)
plt.title('Elbow Method', size=14)
plt.xlabel('Number of clusters', size=12)
plt.ylabel('wcss', size=14)
plt.show()
```

Elbow Method



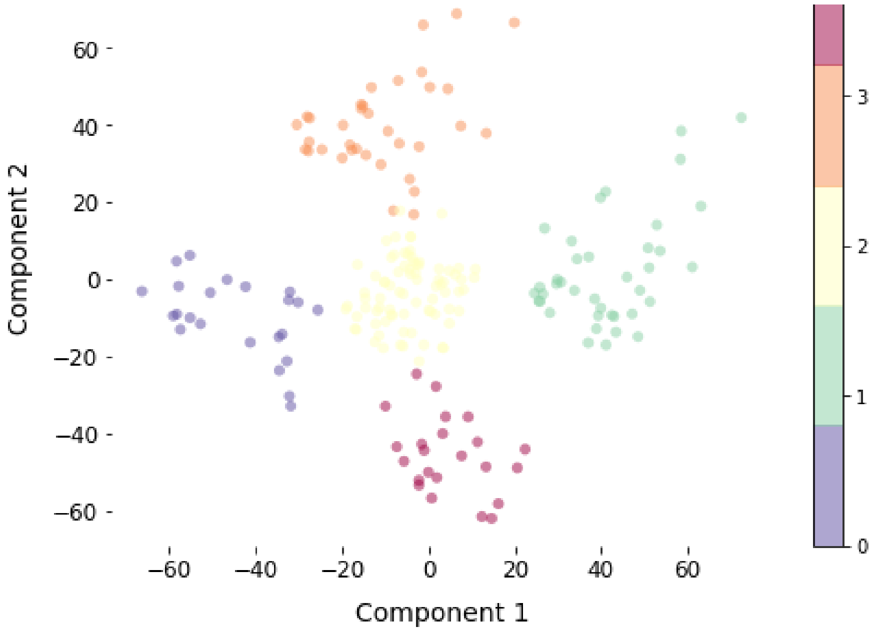
How does k-means clustering works? The main idea is to select k centers, one for each cluster. There are several ways to initialize those centers. We can do it randomly, pass certain points that we believe are the center or place them in a smart way (e.g. as far away from each other as possible). Then, we calculate the Euclidean distance between each point and the cluster centers. We assign the points to the cluster center where the distance is minimum. After that, we recalculate the new cluster center. We select the point that is in the middle of each cluster as the new center. And we start again, calculate distance, assign to cluster, calculate new centers. When do we stop? When the centers do not move anymore.

In [30]:

```
# Kmeans algorithm
# n_clusters: Number of clusters. In our case 5
# init: k-means++. Smart initialization
# max_iter: Maximum number of iterations of the k-means algorithm for a single run
# n_init: Number of time the k-means algorithm will be run with different centroid seeds.

# random_state: Determines random number generation for centroid initialization.
kmeans = KMeans(n_clusters=5, init='k-means++', max_iter=10, n_init=10, random_state=0)

# Fit and predict
y_means = kmeans.fit_predict(X)
```

In [31]:

```
fig, ax = plt.subplots(figsize = (8, 6))

plt.scatter(pca_2d[:, 0], pca_2d[:, 1],
            c=y_means,
            edgecolor="none",
            cmap=plt.cm.get_cmap("Spectral_r", 5),
            alpha=0.5)

plt.gca().spines["top"].set_visible(False)
plt.gca().spines["right"].set_visible(False)
plt.gca().spines["bottom"].set_visible(False)
plt.gca().spines["left"].set_visible(False)

plt.xticks(size=12)
plt.yticks(size=12)

plt.xlabel("Component 1", size = 14, labelpad=10)
plt.ylabel("Component 2", size = 14, labelpad=10)

plt.title('Domains Grouped in 5 clusters', size=16)


plt.colorbar(ticks=[0, 1, 2, 3, 4]);

plt.show()
```



In [32]:

```
centroids = pd.DataFrame(kmeans.cluster_centers_, columns = ["Age", "Annual Income", "Sp
ending", "Male", "Female"])
```

In [33]:

```
centroids.index_name = "ClusterID"

centroids["ClusterID"] = centroids.index
centroids = centroids.reset_index(drop=True)
```

In [34]:

```
centroids
```

Out[34]:

	Age	Annual Income	Spending	Male	Female	ClusterID
0	45.217391	26.304348	20.913043	0.608696	0.391304	0
1	32.692308	86.538462	82.128205	0.538462	0.461538	1
2	43.088608	55.291139	49.569620	0.582278	0.417722	2
3	40.666667	87.750000	17.583333	0.472222	0.527778	3
4	25.521739	26.304348	78.565217	0.608696	0.391304	4

The most important features appear to be Annual Income and Spending score. We have people whose income is low but spend in the same range-segment 0. People whose earnings a high and spend a lot-segment 1. Customers whose income is middle range but also spend at the same level-segment 2. Then we have customers whose income is very high but they have most spendings-segment 3. And last, people whose earnings are little but they spend a lot- segment 4.

Imagine that tomorrow we have a new member. And we want to know which segment that person belongs. We can predict this.

In [35]:

```
X_new = np.array([[22,10,9,0,2]])

new_customer = kmeans.predict(X_new)
print(f"The new customer belongs to segment {new_customer[0]}")
```

```
The new customer belongs to segment 0
```

In [ ]:

A4 Email Notebook

In [3]:

```
# install kaggle package
! pip install -q kaggle
```

upload `kaggle.json` API key

In [4]:

```
# make folder for api key
! mkdir ~/.kaggle
```

```
mkdir: cannot create directory ‘/root/.kaggle’: File exists
```

In [5]:

```
# copy key into folder
! cp kaggle.json ~/.kaggle/
```

In [6]:

```
# change access permissions
! chmod 600 ~/.kaggle/kaggle.json
```

for getting data go to kaggle page and ... and copy API command

In [7]:

```
! kaggle datasets download -d wcukierski/enron-email-dataset
```

```
enron-email-dataset.zip: Skipping, found more recently modified local copy (use --force to force dow
nload)
```

In [10]:

```
! unzip /content/enron-email-dataset.zip
```

```
Archive:  /content/enron-email-dataset.zip
replace emails.csv? [y]es, [n]o, [A]ll, [N]one, [r]ename: n
```

In [9]:

```
!pip install datashader -q
!pip install -qq -U gensim
```

In [11]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
% matplotlib inline
import email
import datetime

import nltk
import re
nltk.download('punkt')
nltk.download('stopwords')
from nltk.tokenize import word_tokenize
from string import punctuation

import spacy
nlp = spacy.load("en_core_web_sm")
import re
import networkx as nx

from os import path
from PIL import Image
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator


import holoviews as hv
from holoviews import opts
hv.extension('bokeh')
from bokeh.plotting import show
kwargs = dict(width=800, height=800, xaxis=None, yaxis=None)
opts.defaults(opts.Nodes(**kwargs), opts.Graph(**kwargs))


from gensim.corpora.dictionary import Dictionary
from gensim.models.tfidfmodel import TfidfModel
from gensim.models.lsimodel import LsiModel
from gensim.similarities import MatrixSimilarity
from sklearn.decomposition import PCA


from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from mlxtend.plotting import plot_confusion_matrix
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

In [12]:

```
#Loading the dataset.
df_emails = pd.read_csv("emails.csv")
#And inspects it:
df_emails.head()
```

Out[12]:

	file	message
0	allen-p/_sent_mail/1.	Message-ID: <18782981.1075855378110.JavaMail.e...
1	allen-p/_sent_mail/10.	Message-ID: <15464986.1075855378456.JavaMail.e...
2	allen-p/_sent_mail/100.	Message-ID: <24216240.1075855687451.JavaMail.e...
3	allen-p/_sent_mail/1000.	Message-ID: <13505866.1075863688222.JavaMail.e...
4	allen-p/_sent_mail/1001.	Message-ID: <30922949.1075863688243.JavaMail.e...

In [13]:

```
# Drops rows containing messages without some specified value in the expected locations.
def standard_format(df, Series, string, slicer):
    rows = []
    for row, message in enumerate(Series):
        message_words = message.split('\n')
        if string not in message_words[slicer]:
            rows.append(row)
    df = df.drop(df.index[rows])
    return df

# Applying the cleansing.
x = len(df_emails.index)
headers = ['Message-ID: ', 'Date: ', 'From: ', 'To: ', 'Subject: ']
for i, v in enumerate(headers):
    df_emails = standard_format(df_emails, df_emails.message, v, i)
df_emails = df_emails.reset_index()
print("Got rid of {} useless emails! That's {}% of the total number of messages in this dataset.".format(x - len(
df_emails.index), np.round(((x - len(df_emails.index)) / x) * 100, decimals=2)))
```

```
Got rid of 111433 useless emails! That's 21.54% of the total number of messages in this dataset.
```

In [14]:

```
# With the function below, we can subtract the information we want from the "message" column.
def get_field(field, messages):
    column = []
    for message in messages:
        e = email.message_from_string(message)
        column.append(e.get(field))
    return column
```

In [15]:

```
# Now using the function above.
df_emails["date"] = get_field("Date", df_emails["message"])
df_emails["subject"] = get_field("Subject", df_emails["message"])
df_emails["From"] = get_field("From", df_emails["message"])
df_emails["To"] = get_field("To", df_emails["message"])
df_emails["X-Folder"] = get_field("X-Folder", df_emails["message"])
df_emails["X-From"] = get_field("X-From", df_emails["message"])
df_emails["X-To"] = get_field("X-To", df_emails["message"])
df_emails.head(3)
```

Out[15]:

	index	file	message	date	subject	From	To	
0	0	allen-p/_sent_mail/1.	Message-ID: <18782981.1075855378110.JavaMail.e...	Mon, 14 May 2001 16:39:00 -0700 (PDT)		phillip.allen@enron.com	tim.belden@enron.com	\Philli
1	1	allen-p/_sent_mail/10.	Message-ID: <15464986.1075855378456.JavaMail.e...	Fri, 4 May 2001 13:51:00 -0700 (PDT)	Re:	phillip.allen@enron.com	john.lavorato@enron.com	\Philli
2	2	allen-p/_sent_mail/100.	Message-ID: <24216240.1075855687451.JavaMail.e...	Wed, 18 Oct 2000 03:00:00 -0700 (PDT)	Re: test	phillip.allen@enron.com	leah.arsdall@enron.com	\Ph

In [16]:

```
#Changes date column to datetime format, using the datetime package
df_emails['date'] = pd.to_datetime(df_emails['date'], infer_datetime_format=True, utc=True)
```

In [17]:

```
#The function below extracts the body/actual mail from the "message" column.

def body(messages):
    column = []
    for message in messages:
        e = email.message_from_string(message)
        column.append(e.get_payload())
    return column

df_emails["body"] = body(df_emails["message"])
```

In [18]:

```
#This function extracts the employee name of the sender.

def employee(file):
    column = []
    for string in file:
        column.append(string.split("/")[0])
    return column

df_emails["employee"] = employee(df_emails["file"])
df_emails.head(3)
```

Out[18]:

	index	file	message	date	subject	From	To
0	0	allen-p/_sent_mail/1.	Message-ID: <18782981.1075855378110.JavaMail.e...	2001-05-14 23:39:00+00:00		phillip.allen@enron.com	tim.belden@enron.com
1	1	allen-p/_sent_mail/10.	Message-ID: <15464986.1075855378456.JavaMail.e...	2001-05-04 20:51:00+00:00	Re:	phillip.allen@enron.com	john.lavorato@enron.com
2	2	allen-p/_sent_mail/100.	Message-ID: <24216240.1075855687451.JavaMail.e...	2000-10-18 10:00:00+00:00	Re: test	phillip.allen@enron.com	leah.arsdall@enron.com

EDA

In [19]:

```
#First, we take a look at the number of employees, thats included in the dataset.
print("Number of employees:",df_emails['employee'].nunique())
```

```
Number of employees: 150
```

We where interested in how many charcheter where used in the emails to get an understand, how there could be so many. By going through the email body we could find the lenght. the first graph show that the density all most of the emails contains few or no words, and the other scale some messages containing a lot for words.

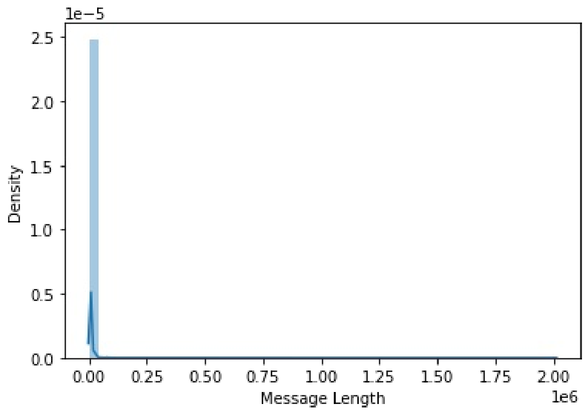
In [20]:

```
#Maybe include
df_emails['Message Length'] = df_emails['body'].apply(lambda x: len(x))

sns.distplot(df_emails['Message Length'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a
deprecated function and will be removed in a future version. Please adapt your code to use either `d
isplot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for
histograms).
  warnings.warn(msg, FutureWarning)
```

Out[20]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f95058cd7d0>
```

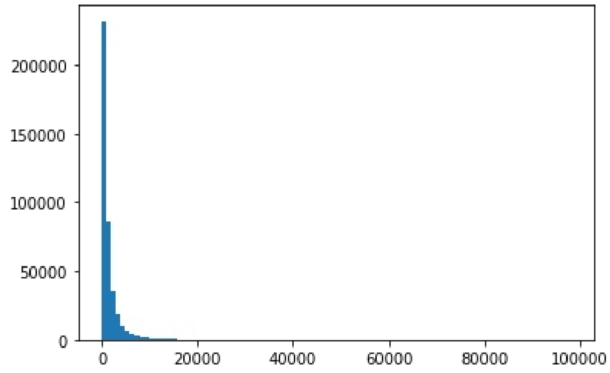


Then we wanted to see the distribution of different size of characters, by making a range for characters and set a bin size with 100 gave an good indication as the characters decreased

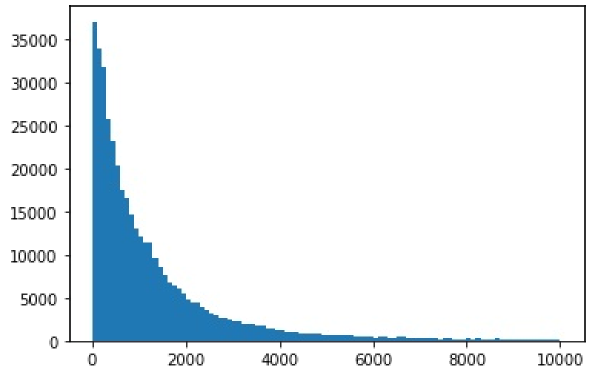
In [21]:

```
#The for loop below, plots and visualize the length of the emails in the dataset.

for order_of_magnitude in reversed(range(2,6)):
    max_ = 10**order_of_magnitude
    print("Messages not longer than %i characters:"%max_)
    plt.hist(df_emails.query('`Message Length`<@max_')['Message Length'], bins=100)
    #histplot(email_df.query('`Message Length`<@max_'), x='Message Length')
    plt.show()
```

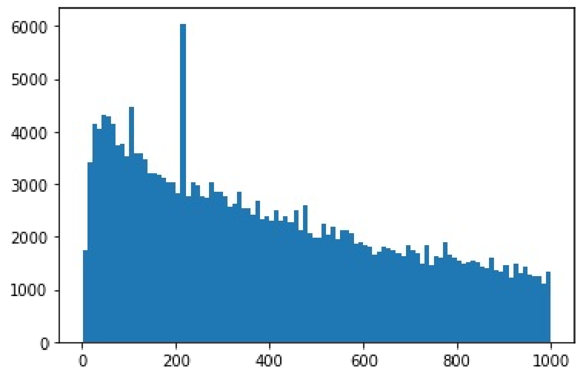
Messages not longer than 100000 characters:



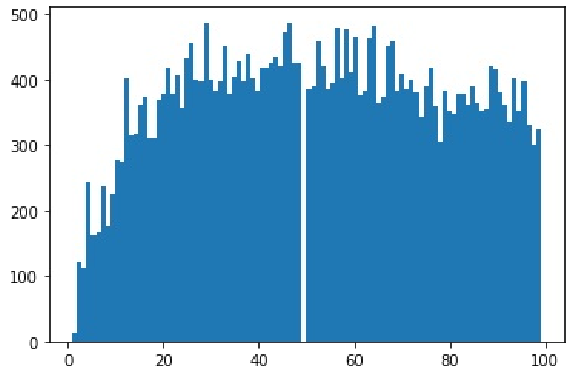
Messages not longer than 10000 characters:



Messages not longer than 1000 characters:



Messages not longer than 100 characters:

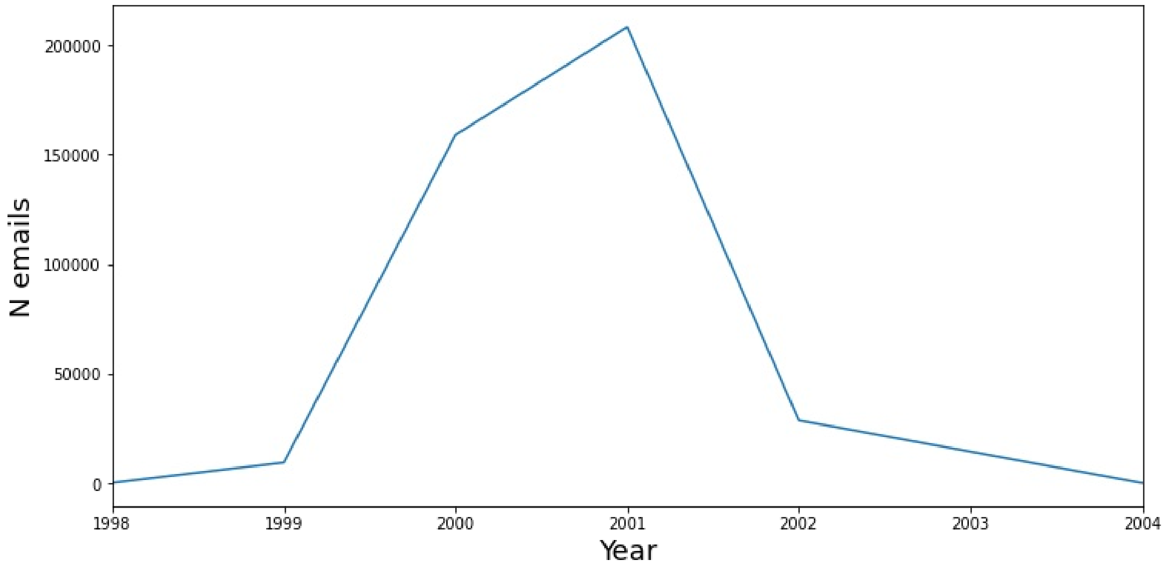


In [22]:

```
#The function below shows in which period the emails from the dataset have been sent.
plt.figure(figsize=(12,6))
ax = df_emails.groupby(df_emails['date'].dt.year)['body'].count().plot()
ax.set_xlabel('Year', fontsize=18)
ax.set_ylabel('N emails', fontsize=18)
ax.set_xlim(1998,2004)
```

Out[22]:

```
(1998.0, 2004.0)
```



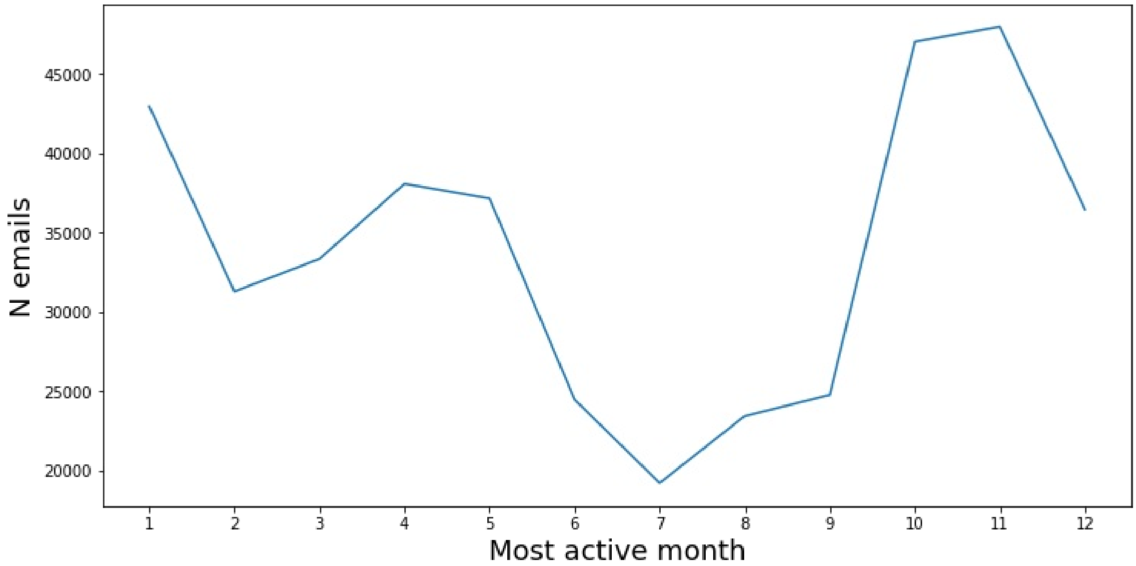
From the data set we have we can clearly see on the year email that therre is a ramp up of send emails, for the month graph show the year with the most emails, year 2001. Enron changed their CEO, in february march time, so more emails where send, going of for summer vacation, the amount of emails dropped. During august and october there was a dicussion going on for the employees of enron to buy enron stock and the company was still doing fine. short after this, accorind to the stock price. Enron have lot a lost of money, resulting the decreaseing emails.

In [23]:

```
# Next, we illustrate the most active email-months.
# Clearly, theres less emails being send doing the summer.
plt.figure(figsize=(12,6))
ax = df_emails.groupby(df_emails['date'].dt.month)['body'].count().plot()
ax.set_xlabel('Most active month', fontsize=18)
ax.set_ylabel('N emails', fontsize=18)
ax.set_xticks(range(1,13))
```

Out[23]:

```
[<matplotlib.axis.XTick at 0x7f95060e3510>,
 <matplotlib.axis.XTick at 0x7f95060e3a50>,
 <matplotlib.axis.XTick at 0x7f94fbf8a2d0>,
 <matplotlib.axis.XTick at 0x7f94fbf58e50>,
 <matplotlib.axis.XTick at 0x7f95061b0f10>,
 <matplotlib.axis.XTick at 0x7f950609fd50>,
 <matplotlib.axis.XTick at 0x7f950609ffd0>,
 <matplotlib.axis.XTick at 0x7f94fbf585d0>,
 <matplotlib.axis.XTick at 0x7f95060e3690>,
 <matplotlib.axis.XTick at 0x7f95060b2210>,
 <matplotlib.axis.XTick at 0x7f95060b2090>,
 <matplotlib.axis.XTick at 0x7f9505ff8950>]
```

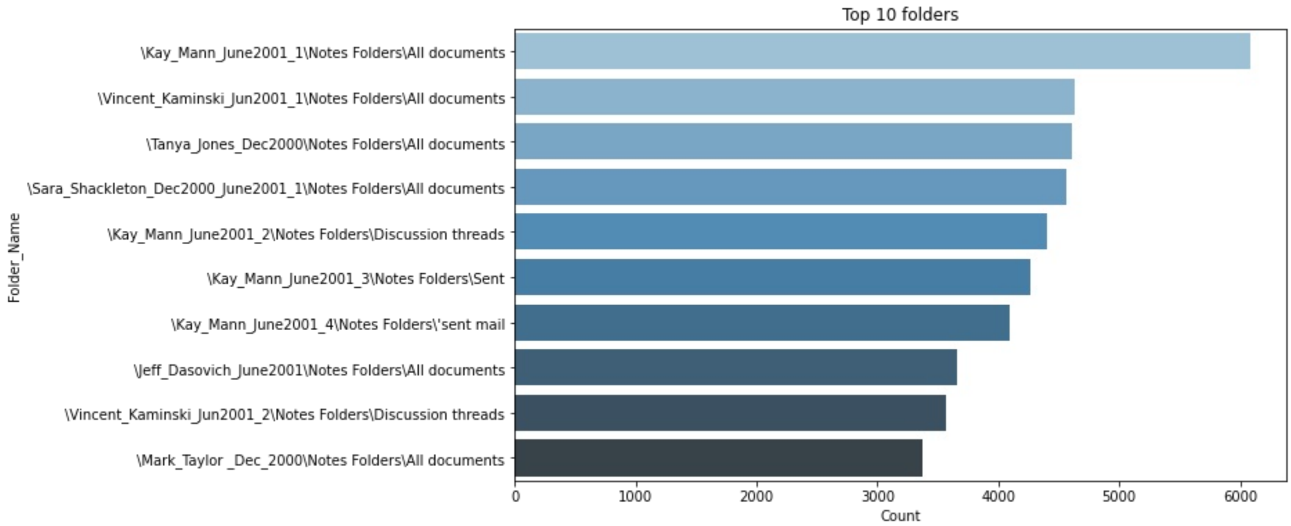


We wanted to see which of the folder where most used, and this turn out to be Kay Mann's folder this is a person in charge of the legal in Enron, so many of the emails contain information about legal issues.

In [24]:

```
unique_emails = pd.DataFrame(df_emails['X-Folder'].value_counts())
unique_emails.reset_index(inplace=True)


unique_emails.columns = ['folder_name', 'count']
# Top 20 folders
print(unique_emails.iloc[:10,:])

#Plots the figure
plt.figure(figsize=(10,6))
sns.barplot(x='count', y='folder_name', data=unique_emails.iloc[:10, :], palette="Blues_d")
plt.title("Top 10 folders")
plt.xlabel("Count")
plt.ylabel("Folder_Name")
plt.show()
```

```
                                      folder_name  count
0   \Kay_Mann_June2001_1\Notes Folders\All documents   6081
1  \Vincent_Kaminski_Jun2001_1\Notes Folders\All ...   4635
2   \Tanya_Jones_Dec2000\Notes Folders\All documents   4606
3  \Sara_Shackleton_Dec2000_June2001_1\Notes Fold...   4560
4  \Kay_Mann_June2001_2\Notes Folders\Discussion ...   4405
5            \Kay_Mann_June2001_3\Notes Folders\Sent   4269
6      \Kay_Mann_June2001_4\Notes Folders\'sent mail   4089
7  \Jeff_Dasovich_June2001\Notes Folders\All docu...   3657
8  \Vincent_Kaminski_Jun2001_2\Notes Folders\Disc...   3567
9  \Mark_Taylor _Dec_2000\Notes Folders\All docum...   3378
```



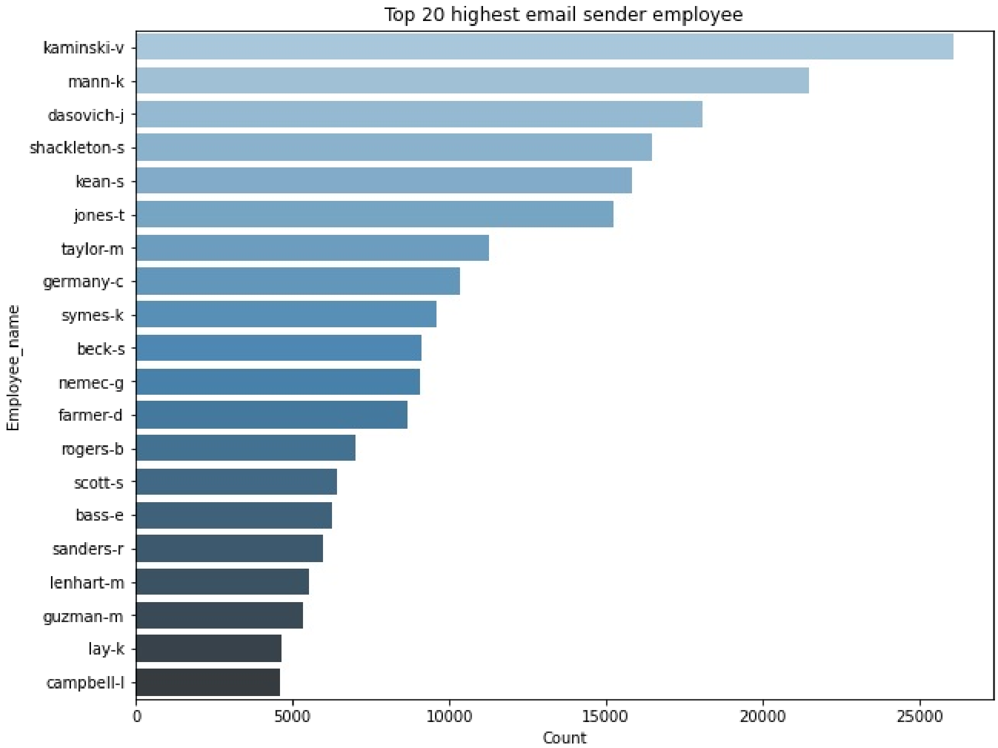
Then we wanted to find out who is person who send the most emails.

- Kaminski was the director of reseach, so that's why he send a lot of emails
- Dasovich was the governmental affairs executive Some of the others ware secretaies and traders

the CEO, Lay, is at only ranked as 20, so he was sending not so much emails

In [25]:

```
# The plot below shows the most frequent sender.
top_20 = pd.DataFrame(df_emails['employee'].value_counts()[:20])
top_20.reset_index(inplace=True)
top_20.columns = ["Employee_name", "Counts"]

#Plots the figure
plt.figure(figsize=(10,8))
sns.barplot(y="Employee_name", x="Counts", data=top_20, palette="Blues_d")
plt.title("Top 20 highest email sender employee")
plt.xlabel("Count")
plt.ylabel("Employee_name")
plt.show()
```



Network

In [26]:

```
#Setting up the edges
edges = df_emails[["From","To"]]

edges = edges[edges.From != edges.To]
edges.head()
```

Out[26]:

	From	To
0	phillip.allen@enron.com	tim.belden@enron.com
1	phillip.allen@enron.com	john.lavorato@enron.com
2	phillip.allen@enron.com	leah.arsdall@enron.com
3	phillip.allen@enron.com	randall.gay@enron.com
4	phillip.allen@enron.com	greg.piper@enron.com

In [27]:

```
# Grouping to aggregate multiple co-occurences and to generate a weight:
# Based on how many times one sender sends a mail to a reciever.
# Lastly, reset_index makes everytging from a multi-index-series into a dataframe
edges = edges.groupby(['From', 'To']).size().reset_index()
```

In [28]:

```
#Renaming the column 0 to weight.
edges.rename({0:'weight'}, axis = 1, inplace=True)
```

In [29]:

```
# Creates network object from pandas edgelist
G = nx.from_pandas_edgelist(edges, source='From', target='To', edge_attr='weight', create_using=nx.Graph())
```

In [30]:

```
#Then sets the node attributes.
nx.set_node_attributes(G, {G.degree(): 'degree'})
```

In [31]:

```
# Subset the graph keeping only nodes with degree > 1
G = nx.subgraph(G, [n for n,d in G.degree() if d > 1])

# Here we can calculate different centrality indicators.
centrality_dgr = nx.degree_centrality(G)
centrality_eig = nx.eigenvector_centrality_numpy(G)
degree = G.degree()

# All these indicators can now be set as attribute of the Graph
nx.set_node_attributes(G, centrality_dgr, 'dgr')
nx.set_node_attributes(G, centrality_eig, 'eig')
nx.set_node_attributes(G, dict(degree), 'degree_basic')
```

In [32]:

```
# Turns the Graph object (NetworkX) to a Dataframe
nodes_df = pd.DataFrame.from_dict(dict(G.nodes(data=True)), orient='index')
```

In [33]:

```
# Since the dataset contains of 150 people, we'll find the top 25 most central people.
top_10_eig = nodes_df.sort_values('eig', ascending=False)[:10]
```

In [34]:

```
#Creates nodes for plot
emails_eig = top_10_eig.eig.index

#Create subset graph
g_sub_emails = nx.subgraph(G,emails_eig)

# Create and save a layout.
g_layout = nx.layout.spring_layout(g_sub_emails)
g_plot = hv.Graph.from_networkx(g_sub_emails, g_layout).opts(tools=['hover'], node_color='partition')
labels = hv.Labels(g_plot.nodes, ['x', 'y'])

# Makes the plot
from holoviews.operation.datashader import datashade, bundle_graph
bundled = bundle_graph(g_plot)

# Show the plot
show(hv.render(bundled * labels.opts(text_font_size='6pt', text_color='white', bgcolor='lightblue')))

print(top_10_eig[:10])
```

```
bokeh.core.validation.check - ERROR - E-1001 (BAD_COLUMN_NAME): Glyph refers to nonexistent column n
ame. This could either be due to a misspelling or typo, or due to an expected column being missing.
: key "fill_color" value "partition" [renderer: GlyphRenderer(id='1218', ...)]


                                dgr       eig  degree_basic
tana.jones@enron.com       0.059611  0.206684           597
louise.kitchen@enron.com   0.044733  0.199194           448
sara.shackleton@enron.com  0.058612  0.191055           587
mark.taylor@enron.com      0.045931  0.177937           460
sally.beck@enron.com       0.043635  0.140461           437
mark.haedicke@enron.com    0.027659  0.139406           277
vince.kaminski@enron.com   0.060110  0.133938           602
john.lavorato@enron.com    0.029855  0.133050           299
elizabeth.sager@enron.com  0.028557  0.131349           286
richard.sanders@enron.com  0.032651  0.125268           327
```

NLP

Subject wordcloud

In [35]:

```
subject_text = " ".join(message for message in df_emails.subject)
print ("There are {} words in the subjects in combination of all mails.".format(len(subject_text)))
```

```
There are 11396608 words in the subjects in combination of all mails.
```

In [36]:

```
# Create stopword list:
stopwords = set(STOPWORDS)
stopwords.update(["RE", "FW","Fwd","Date","Hour","HourAhead","Reminder","PLEASEREAD"])

# Generate a word cloud image
wordcloud = WordCloud(stopwords=stopwords, background_color="white").generate(subject_text)

# Display the generated wordcloud:
plt.figure(figsize=(20,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

Full text

In [44]:

```
#Takes a sample of 10.000 emails
emails_nlp = nlp(df_emails.body.sample(n=10000,random_state=42).to_string())
```

In [45]:

```
# Most mentioned people in the e-mails
persons_entity = [(ent.text, ent.label_) for ent in emails_nlp.ents if ent.label_ == "PERSON" ]
persons_entity = pd.DataFrame(persons_entity,columns=["Name","-"])

print("Most mentioned people in mails")
print()
print(persons_entity.value_counts()[:10])
print()
print("Top 10 central people accoring to Network Analysis")
print(top_10_eig[:10])
```

```
Most mentioned people in mails

Name        -
Jeff        PERSON    47
Steve       PERSON    40
Kay Mann/C  PERSON    37
John        PERSON    35
Mark        PERSON    33
Kay         PERSON    26
Mike        PERSON    26
Mary        PERSON    22
Vince J Ka  PERSON    21
Rick        PERSON    21
dtype: int64

Top 10 central people accoring to Network Analysis
                                dgr       eig  degree_basic
tana.jones@enron.com       0.059611  0.206684           597
louise.kitchen@enron.com   0.044733  0.199194           448
sara.shackleton@enron.com  0.058612  0.191055           587
mark.taylor@enron.com      0.045931  0.177937           460
sally.beck@enron.com       0.043635  0.140461           437
mark.haedicke@enron.com    0.027659  0.139406           277
vince.kaminski@enron.com   0.060110  0.133938           602
john.lavorato@enron.com    0.029855  0.133050           299
elizabeth.sager@enron.com  0.028557  0.131349           286
richard.sanders@enron.com  0.032651  0.125268           327
```

In [46]:

```python
#Most used adjectives in the emails
adj_entity = [(token.lemma_, token.pos_) for token in emails_nlp if token.pos_ == "ADJ" and not token.is_stop]
adj_entity = pd.DataFrame(adj_entity,columns=["Adjective","-"])

print("Most used adjectives in mails")
print()
print(adj_entity.value_counts()[:10])


# Generate a word cloud image
wordcloud = WordCloud(stopwords=stopwords, background_color="white").generate(adj_entity.Adjective.to_string())

# Display the generated image:
# the matplotlib way:
plt.figure(figsize=(20,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

```
Most used adjectives in mails

Adjective  -
good       ADJ    167
new        ADJ     90
dear       ADJ     86
great      ADJ     71
late       ADJ     57
sure       ADJ     52
fine       ADJ     48
sorry      ADJ     39
able       ADJ     32
glad       ADJ     27
dtype: int64
```

In [47]:

```
# The function below returns a string that is cleaned.

def clean_text(Series):

    result = []
    strings = Series.str.lower()

    for string in strings:
        new_string = []
        words = string.split(" ")

        for word in words:
            word = word.strip(punctuation)

            if word in stopwords:
                continue
            if re.search(r'[\W\d]',word):
                continue

            new_string.append(word)

        new_string = " ".join(new_string)

        result.append(new_string)

    return result
```

In [48]:

```
#Next, we'll generate a wordcloud on the "body" of every email.
#And firstly, we create a new dataframe.
enorn_person_email = df_emails[["employee","body"]]

#Then cleans the data
enorn_person_email['body'] = clean_text(enorn_person_email['body'])

#Grouping every word a person has mailed into one dataframe.
df_grouped = enorn_person_email.groupby("employee")['body'].apply(' '.join).reset_index()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexi
ng.html#returning-a-view-versus-a-copy
```

In [52]:

```
# Generate a word cloud image
stopwords.update(["CC","subject","forwarded","seeattached","please see","pleasefind","fyi"])
wordcloud = WordCloud(stopwords=stopwords, background_color="white").generate(enorn_person_email.body.to_string()
)

# Display the generated image:
# the matplotlib way:
plt.figure(figsize=(20,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

Unsupervised Machine Learning

In [53]:

```
#For the UML, we first take a sample, since our Collab cannot run the full dataset.
uml_emails = df_emails.sample(n=4000,random_state=42)
```

In [54]:

```
#We interested to see, if theres a certain pattern in terms of whether the sender is from Enron or not.
#Therefore, we firstly split the "From" emails by the "@", and afterwards we categorize this into a new column.
uml_emails["Enron_or_not"] = uml_emails.From.apply(lambda t: t.split("@")[1])
```

In [55]:

```
filter = uml_emails["Enron_or_not"] == "enron.com"
```

In [56]:

```
uml_emails["Enron_or_not"] = uml_emails["Enron_or_not"].replace("enron.com","Enron")
uml_emails["Enron_or_not"] = uml_emails.Enron_or_not.where(filter,"Not_Enron")
```

In [57]:

```
uml_emails[["Enron_or_not","From"]].head()
```

Out[57]:

	Enron_or_not	From
286099	Enron	joe.quenet@enron.com
328179	Enron	sara.shackleton@enron.com
1550	Enron	phillip.allen@enron.com
134351	Not_Enron	lofeco@ev1.net
377751	Enron	justin.boyd@enron.com

In [58]:

```
#After we've classified the different emails, we create a list of our tokens, called tokens.
#Here, we both use lemmatization and lower the tokens. Moreover, are we only interesseted in a certain types of w
ords.

tokens = []

for summary in nlp.pipe(uml_emails.body):
  proj_tok = [token.lemma_.lower() for token in summary if token.pos_ in ['NOUN', 'PROPN', 'ADJ', 'ADV'] and not
token.is_stop]
  tokens.append(proj_tok)
```

In [59]:

```
#The, we sets the index
uml_emails.index = range(len(uml_emails))

#And takes the tokens back into our sample of emails
uml_emails["tokens"] = tokens
```

In [60]:

```
# Create a Dictionary from the emails, called: dictionary
dictionary = Dictionary(uml_emails['tokens'])

# And based on our dictionary, we can construct our corpus.
corpus = [dictionary.doc2bow(doc) for doc in uml_emails['tokens']]
```

In [61]:

```
# We now sets up our Tfidf model, using our corpus from above.
# Create and fit a new TfidfModel using the corpus: tfidf
tfidf = TfidfModel(corpus)
# Now we can transform the whole corpus
tfidf_corpus = tfidf[corpus]
```

In [62]:

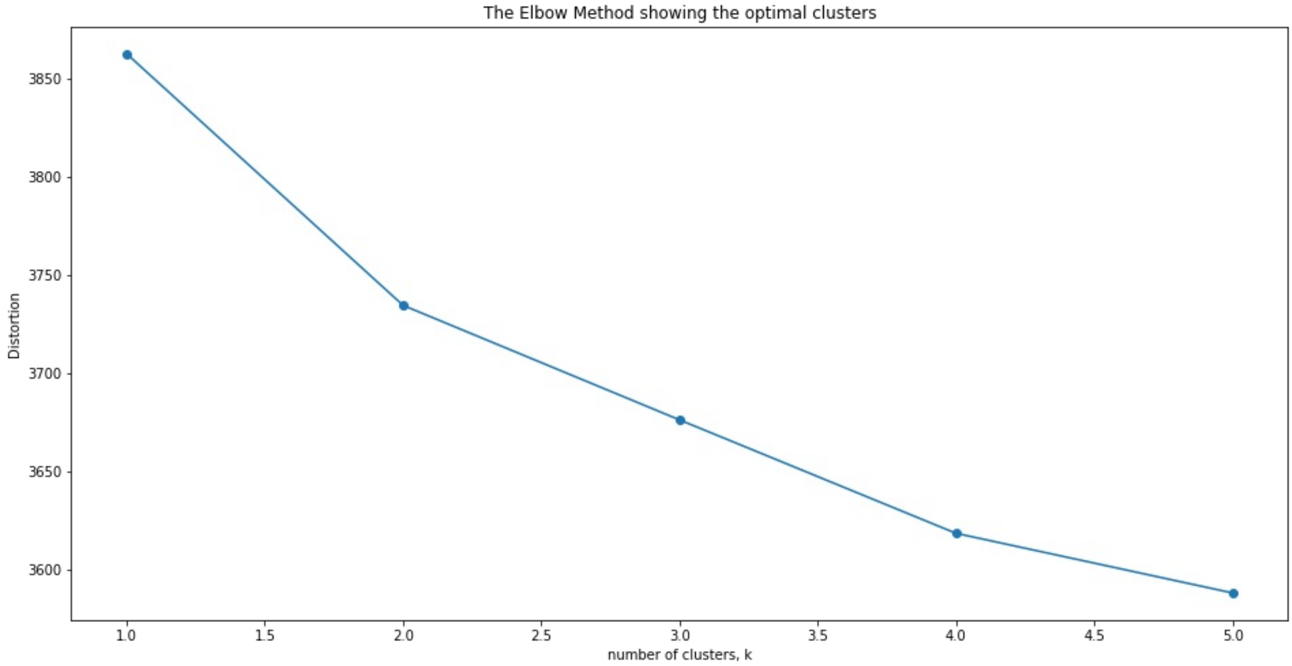
```
#We then trains our model
lsi = LsiModel(tfidf_corpus, id2word=dictionary, num_topics=400)

# And our trained model can then be used to transform our corpus
lsi_corpus = lsi[tfidf_corpus]
```

In [63]:

```
# Creates the email-topic-matrix
email_topic_matrix = MatrixSimilarity(lsi_corpus)
email_topic_matrix_ix = email_topic_matrix.index
```

In [64]:

```
from sklearn.cluster import KMeans
# Creates a "for loop", to determine number of clusters.
distortions = []
K = range(1,6)
for k in K:
    kmeanModel = KMeans(n_clusters=k)
    kmeanModel.fit(email_topic_matrix_ix)
    distortions.append(kmeanModel.inertia_)


plt.figure(figsize=(16,8))
plt.plot(K, distortions, 'o-')
plt.xlabel('number of clusters, k')
plt.ylabel('Distortion')
plt.title('The Elbow Method showing the optimal clusters')
plt.show()
```



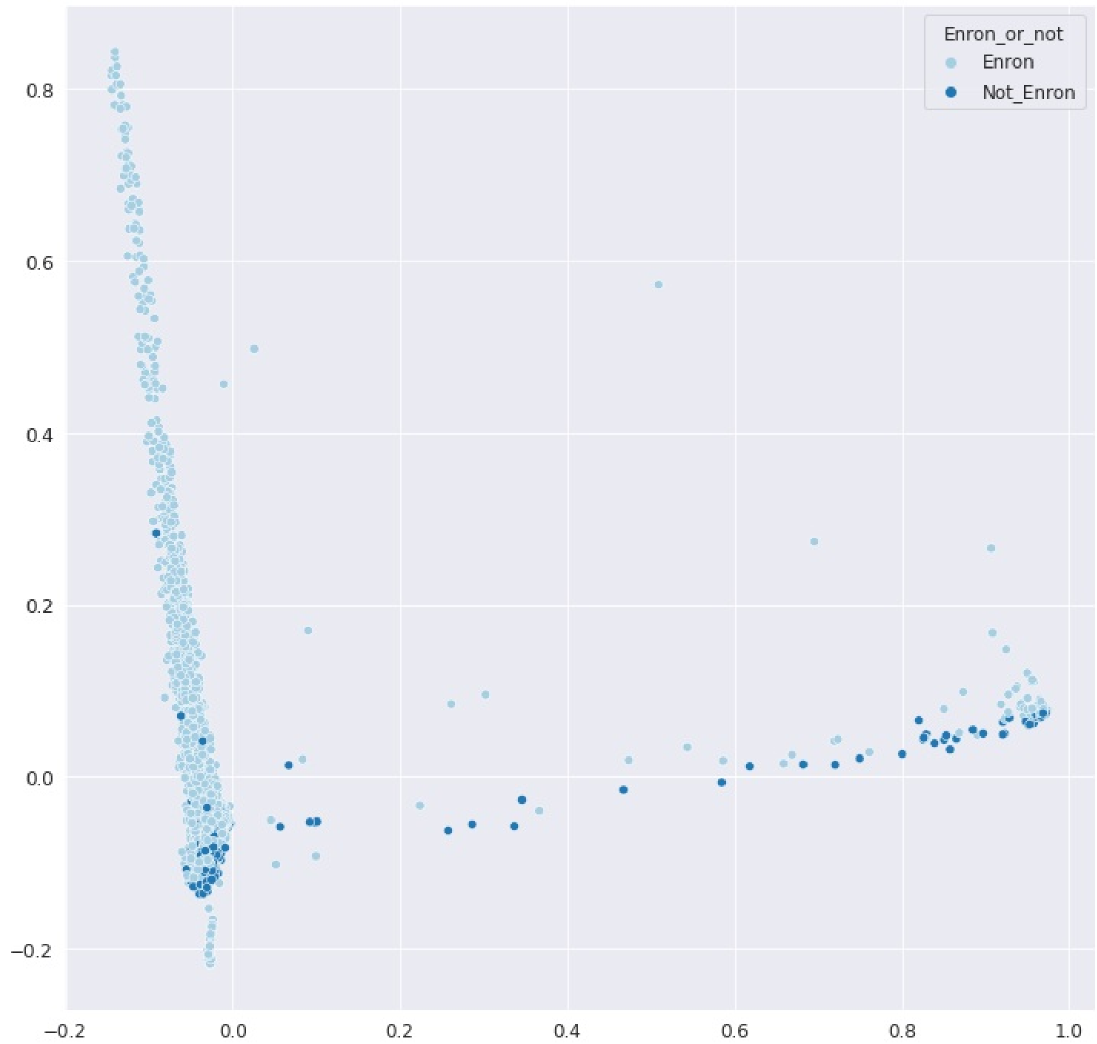
In [70]:

```
reduced = PCA(n_components = 2).fit_transform(email_topic_matrix_ix)

# Sets up the clustrs, which is 2
clusterer = KMeans(n_clusters = 2)
clusterer.fit(email_topic_matrix_ix)

# Plotting things
sns.set_style("darkgrid")

plt.rcParams.update({'font.size': 12})
plt.figure(figsize=(12,12))
g = sns.scatterplot(reduced[:,0],reduced[:,1],
                   hue=uml_emails["Enron_or_not"],
                    palette="Paired",
                   legend='full')

#Illustrating this in a crosstab as well, to get a better picture of the clustering
pd.crosstab(clusterer.labels_, uml_emails['Enron_or_not'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following
variables as keyword args: x, y. From version 0.12, the only valid positional argument will be `data
`, and passing other arguments without an explicit keyword will result in an error or misinterpretat
ion.
  FutureWarning
```

Out[70]:

Enron_or_not	Enron	Not_Enron
row_0		
0	3257	595
1	73	75



Supervised machine learning

In [78]:

```
#We import a set of fraudlent emails, to mix with our EnronEmails, to train our model
! unzip /content/fradulent_emails.txt.zip

#Reads fraudlent emails:
with open("fradulent_emails.txt", 'r',encoding="latin1") as file:
    fraudlent_mails = file.read()

#And inspects the dataset
print(fraudlent_mails[:5000]) #Clearly, every mail starts with "From r",
# - this will therefore be used to split the emails into seperate strings.
```

```
Archive:  /content/fradulent_emails.txt.zip
replace fradulent_emails.txt? [y]es, [n]o, [A]ll, [N]one, [r]ename: n
From r  Wed Oct 30 21:41:56 2002
Return-Path: <james_ngola2002@maktoob.com>
X-Sieve: cmu-sieve 2.0
Return-Path: <james_ngola2002@maktoob.com>
Message-Id: <200210310241.g9V2fNm6028281@cs.CU>
From: "MR. JAMES NGOLA." <james_ngola2002@maktoob.com>
Reply-To: james_ngola2002@maktoob.com
To: webmaster@aclweb.org
```

Date: Thu, 31 Oct 2002 02:38:20 +0000
Subject: URGENT BUSINESS ASSISTANCE AND PARTNERSHIP
X-Mailer: Microsoft Outlook Express 5.00.2919.6900 DM
MIME-Version: 1.0
Content-Type: text/plain; charset="us-ascii"
Content-Transfer-Encoding: 8bit
X-MIME-Autoconverted: from quoted-printable to 8bit by sideshowmel.si.UM id g9V2foW24311
Status: O

FROM:MR. JAMES NGOLA.
CONFIDENTIAL TEL: 233-27-587908.
E-MAIL: (james_ngola2002@maktoob.com).

URGENT BUSINESS ASSISTANCE AND PARTNERSHIP.

DEAR FRIEND,

I AM (DR.) JAMES NGOLA, THE PERSONAL ASSISTANCE TO THE LATE CONGOLESE (PRESIDENT LAURENT KABILA) WH
O WAS ASSASSINATED BY HIS BODY GUARD ON 16TH JAN. 2001.

THE INCIDENT OCCURRED IN OUR PRESENCE WHILE WE WERE HOLDING MEETING WITH HIS EXCELLENCY OVER THE FIN
ANCIAL RETURNS FROM THE DIAMOND SALES IN THE AREAS CONTROLLED BY (D.R.C.) DEMOCRATIC REPUBLIC OF CON
GO FORCES AND THEIR FOREIGN ALLIES ANGOLA AND ZIMBABWE, HAVING RECEIVED THE PREVIOUS DAY (USD$100M)
ONE HUNDRED MILLION UNITED STATES DOLLARS, CASH IN THREE DIPLOMATIC BOXES ROUTED THROUGH ZIMBABWE.

MY PURPOSE OF WRITING YOU THIS LETTER IS TO SOLICIT FOR YOUR ASSISTANCE AS TO BE A COVER TO THE FUND
AND ALSO COLLABORATION IN MOVING THE SAID FUND INTO YOUR BANK ACCOUNT THE SUM OF (USD$25M) TWENTY FI
VE MILLION UNITED STATES DOLLARS ONLY, WHICH I DEPOSITED WITH A SECURITY COMPANY IN GHANA, IN A DIPL
OMATIC BOX AS GOLDS WORTH (USD$25M) TWENTY FIVE MILLION UNITED STATES DOLLARS ONLY FOR SAFE KEEPING
IN A SECURITY VAULT FOR ANY FURTHER INVESTMENT PERHAPS IN YOUR COUNTRY.

YOU WERE INTRODUCED TO ME BY A RELIABLE FRIEND OF MINE WHO IS A TRAVELLER,AND ALSO A MEMBER OF CHAMB
ER OF COMMERCE AS A RELIABLE AND TRUSTWORTHY PERSON WHOM I CAN RELY ON AS FOREIGN PARTNER, EVEN THOU
GH THE NATURE OF THE TRANSACTION WAS NOT REVEALED TO HIM FOR SECURITY REASONS.

THE (USD$25M) WAS PART OF A PROCEEDS FROM DIAMOND TRADE MEANT FOR THE LATE PRESIDENT LAURENT KABILA
WHICH WAS DELIVERED THROUGH ZIMBABWE IN DIPLOMATIC BOXES. THE BOXES WERE KEPT UNDER MY CUSTODY BEFOR
E THE SAD EVENT THAT TOOK THE LIFE OF (MR. PRESIDENT).THE CONFUSION THAT ENSUED AFTER THE ASSASSINAT
ION AND THE SPORADIC SHOOTING AMONG THE FACTIONS, I HAVE TO RUN AWAY FROM THE COUNTRY FOR MY DEAR LI
FE AS I AM NOT A SOLDIER BUT A CIVIL SERVANT I CROSSED RIVER CONGO TO OTHER SIDE OF CONGO LIBREVILLE
FROM THERE I MOVED TO THE THIRD COUNTRY GHANA WHERE I AM PRESENTLY TAKING REFUGE.

AS A MATTER OF FACT, WHAT I URGENTLY NEEDED FROM YOU IS YOUR ASSISTANCE IN MOVING THIS MONEY INTO YO
UR ACCOUNT IN YOUR COUNTRY FOR INVESTMENT WITHOUT RAISING EYEBROW. FOR YOUR ASSISTANCE I WILL GIVE Y
OU 20% OF THE TOTAL SUM AS YOUR OWN SHARE WHEN THE MONEY GETS TO YOUR ACCOUNT, WHILE 75% WILL BE FOR
ME, OF WHICH WITH YOUR KIND ADVICE I HOPE TO INVEST IN PROFITABLE VENTURE IN YOUR COUNTRY IN OTHER T
O SETTLE DOWN FOR MEANINGFUL LIFE, AS I AM TIRED OF LIVING IN A WAR ENVIRONMENT.

THE REMAINING 5% WILL BE USED TO OFFSET ANY COST INCURRED IN THE CAUSE OF MOVING THE MONEY TO YOUR A
CCOUNT. IF THE PROPOSAL IS ACCEPTABLE TO YOU PLEASE CONTACT ME IMMEDIATELY THROUGH THE ABOVE TELEPHO
NE AND E-MAIL, TO ENABLE ME ARRANGE FACE TO FACE MEETING WITH YOU IN GHANA FOR THE CLEARANCE OF THE
FUNDS BEFORE TRANSFRING IT TO YOUR BANK ACCOUNT AS SEEING IS BELIEVING.

FINALLY, IT IS IMPORTANT ALSO THAT I LET YOU UNDERSTAND THAT THERE IS NO RISK INVOLVED WHATSOEVER AS
THE MONEY HAD NO RECORD IN KINSHASA FOR IT WAS MEANT FOR THE PERSONAL USE OF (MR. PRESIDEND) BEFORE
THE NEFARIOUS INCIDENT OCCURRED, AND ALSO I HAVE ALL THE NECESSARY DOCUMENTS AS REGARDS TO THE FUNDS
INCLUDING THE (CERTIFICATE OF DEPOSIT), AS I AM THE DEPOSITOR OF THE CONSIGNMENT.

LOOKING FORWARD TO YOUR URGENT RESPONSE.

YOUR SINCERELY,

MR. JAMES NGOLA.

From r Thu Oct 31 08:11:39 2002
Return-Path: <bensul2004nng@spinfinder.com>
X-Sieve: cmu-sieve 2.0
Return-Path: <bensul2004nng@spinfinder.com>
Message-Id: <200210311310.g9VDANt24674@bloodwork.mr.itd.UM>
From: "Mr. Ben Suleman" <bensul2004nng@spinfinder.com>

```
Date: Thu, 31 Oct 2002 05:10:00
To: R@M
Subject: URGENT ASSISTANCE /RELATIONSHIP (P)
MIME-Version: 1.0
Content-Type: text/plain;charset="iso-8859-1"
Content-Transfer-Encoding: 7bit
Status: O

Dear Friend,

I am Mr. Ben Suleman a custom officer and work as Assistant controller of the Customs and Excise dep
artment Of the Federal Ministry of Internal Affairs stationed at the Murtala Mohammed International
Airport, Ikeja, Lagos-Nigeria.

After the sudden death of the former Head of state of Nigeria General Sanni Abacha on June 8th 1998
his aides and immediate members of his family were arrested while trying to escape from Nigeria in a
Chartered jet to Saudi Arabia with 6 trunk boxes Marked "Diplomatic B
```

In [81]:

```
#Then, we use our body function, to extract the body of the emails.
fraudlent_mails_body = body(fraudlent_mails)

#And afterwards we put it into a new DataFrame.
fraudlent_mails_body = pd.DataFrame(fraudlent_mails_body,columns={"body"})
fraudlent_mails_body.drop([0,0],inplace=True)
fraudlent_mails_body.head()
```

Out[81]:

	body
1	FROM:MR. JAMES NGOLA.\nCONFIDENTIAL TEL: 233-2...
2	Dear Friend,\n\nI am Mr. Ben Suleman a custom ...
3	FROM HIS ROYAL MAJESTY (HRM) CROWN RULER OF EL...
4	FROM HIS ROYAL MAJESTY (HRM) CROWN RULER OF EL...
5	Dear sir, \n \nIt is with a heart full of hope...

In [82]:

```
# For the next cleaning part, we define a function to remove punctuation marks and other nonword characters using
regex library.

def reg_expressions(row):
    tokens = []
    try:
        for token in row:
            token = token.lower()
            token = re.sub(r'[\W\d]', "", token)
            tokens.append(token)
    except:
        token = ""
        tokens.append(token)
    return tokens
```

In [83]:

```
def stop_word_removal(row):
    token = [token for token in row if token not in stopwords]
    return token
```

Bag-of-words model

For the computer to make inferences of the e-mails, it has to be able to interpret the text by making a numerical representation of it. One way to do this is by using something called a "bag-of-words" model. This model simply counts the frequency of word tokens for each email and thereby represents it as a vector of these counts.

In [84]:

```
#Firstly, we'll take out a random 10.000 sample of the Enron-emails body
EnronEmails = df_emails.body.sample(n=10000,random_state=42)
#Then uses word_tokenizer to tokenize the words
EnronEmails = EnronEmails.apply(lambda w: word_tokenize(w))
#Now, we remove stopwords from the mails
EnronEmails = EnronEmails.apply(lambda w: stop_word_removal(w))
#Then, we use our Reg_expres fucntion to delete punctuation marks and other nonword characters
EnronEmails = EnronEmails.apply(reg_expressions)
```

Next, we do all the same steps on our "Fraud/spam" mails

In [85]:

```
SpamEmails = fraudlent_mails_body.body.sample(n=3977).astype(str)
SpamEmails = SpamEmails.apply(lambda w: word_tokenize(w))
SpamEmails = SpamEmails.apply(lambda w: stop_word_removal(w))
SpamEmails = SpamEmails.apply(reg_expressions)
```

In [86]:

```
#Lastly, we take a sample of 1000 mails from both Enrom and Spam
nsamples = 1000

SpamEmails = SpamEmails.sample(n=nsamples,random_state=42)
EnronEmails = EnronEmails.sample(n=nsamples,random_state=42)

#Then we contat these, and name the columns: SpamEmails and EnronEmails
concat_mails = pd.concat([SpamEmails,EnronEmails], axis=0).values
#Checks the shape
concat_mails.shape #Which is correct, since 1000*2 = 4000 ;).
```

Out[86]:

```
(2000,)
```

In [87]:

```
#The function below assembles a new dataframe containing all the unique words found in the input.
# - it then counts the word frequency and then returns the new dataframe.

def assemble_bag(data):
    used_tokens = []
    all_tokens = []

    for item in data:
        for token in item:
            if token in all_tokens:
                if token not in used_tokens:
                    used_tokens.append(token)
            else:
                all_tokens.append(token)

    df = pd.DataFrame(0, index = np.arange(len(data)), columns = used_tokens)

    for i, item in enumerate(data):
        for token in item:
            if token in used_tokens:
                df.iloc[i][token] += 1
    return df
```

In [88]:

```python
# We then uses our funtion above, to create a bag-of-words model
EnronSpamBag = assemble_bag(concat_mails)
# This is the list of words in our bag-of-words model
predictors = [column for column in EnronSpamBag.columns]

#And lastly shows the model
EnronSpamBag
```

Out[88]:

		mr	zuma	barley	auditing	accounting	dept	african	development	bank	benin	adb	will	i	contact	fund	end	us	acco
0	75	3	3	3	3	2	2	2	2	6	6	2	16	9	2	9	4	7	
1	51	0	0	0	0	0	0	0	0	0	0	0	4	4	0	1	0	2	
2	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
3	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1995	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1996	7	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	
1997	38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1998	109	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1999	61	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	

2000 rows × 16508 columns

In [89]:

```python
#Sets up the header, before we can shuffle and mix the data.
header = ([1]*nsamples)
header.extend(([0]*nsamples))
```

In [90]:

```python
#This function mixes our data, so we can split it into a training and test set.
def shuffle_data(data, header):
    p = np.random.permutation(len(header))
    data = data[p,:]
    header = np.asarray(header)[p]
    return data, header
```

In [91]:

```python
data, header = shuffle_data(EnronSpamBag.values, header)
print(header.shape)
print(data.shape)
```

```
(2000,)
(2000, 16508)
```

In [92]:

```python
# Splits into independent 70% training and 30% testing sets
idx = int(0.7*data.shape[0])

# 70% of data for training
train_x = data[:idx,:]
train_y = header[:idx]
# Remaining 30% for testing
test_x = data[idx:,:]
test_y = header[idx:]
```

In [93]:

```
logreg = LogisticRegression()
logreg.fit(train_x,train_y)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: ConvergenceWarning: lb
fgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

Out[93]:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [94]:

```
#Uses the Logistic Regression to predict
y_pred = logreg.predict(test_x)

#Evaluates the score
print("The logistic regression accuracy score is:")
print(accuracy_score(test_y,y_pred))
```

```
The logistic regression accuracy score is:
0.9916666666666667
```

In [ ]:

```
!jupyter nbconvert --to html ""
```

A5 HR Notebook

M4 PROJECT: What are the most influencing factors for employee attrition and whom are those leaveing.

In [1]:

```
#Importing the libriaries needed
# Ignore the warnings
import warnings
warnings.filterwarnings('always')
warnings.filterwarnings('ignore')

# data visualisation and manipulation
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib import style
import seaborn as sns
#import missingno as msno

#configure
# sets matplotlib to inline and displays graphs below the corressponding cell.
%matplotlib inline
style.use('fivethirtyeight')
sns.set(style='whitegrid',color_codes=True)

#import the necessary modelling algos.
#from sklearn.linear_model import LogisticRegression
#from sklearn.svm import LinearSVC
#from sklearn.svm import SVC
#from sklearn.neighbors import KNeighborsClassifier
#from sklearn.ensemble import RandomForestClassifier
#from sklearn.tree import DecisionTreeClassifier
#from sklearn.ensemble import GradientBoostingClassifier
#from sklearn.naive_bayes import GaussianNB

#model selection
#from sklearn.model_selection import train_test_split
#from sklearn.model_selection import KFold
#from sklearn.metrics import accuracy_score,precision_score,recall_score,confusion_matrix,roc_curve,roc_auc_score
#from sklearn.model_selection import GridSearchCV

#from imblearn.over_sampling import SMOTE

#preprocess.
#from sklearn.preprocessing import MinMaxScaler,StandardScaler,LabelEncoder,OneHotEncoder

# Common sklearn Model Helpers
#from sklearn import feature_selection
#from sklearn import model_selection
#from sklearn import metrics
# from sklearn.datasets import make_classification

# sklearn modules for performance metrics
#from sklearn.metrics import confusion_matrix, classification_report, precision_recall_curve
#from sklearn.metrics import auc, roc_auc_score, roc_curve, recall_score, log_loss
#from sklearn.metrics import f1_score, accuracy_score, roc_auc_score, make_scorer
#from sklearn.metrics import average_precision_score
# ann and dl libraraies
#from keras import backend as K
#from keras.models import Sequential
#from keras.layers import Dense
#from keras.optimizers import Adam,SGD,Adagrad,Adadelta,RMSprop
#from keras.utils import to_categorical

#import tensorflow as tf
import random as rn
```

---

1. Dataset Explaination

This dataset was a fictional dataset created by IBM to indentify important factors that may be influencing attrition for an employee. The dataset contains 1470 rows and 35 coulmns. Our project is focusing on to find the most importance metrics that influence attrition. Firstly we need to do some general statistics to get insight into the dataset, Second using machine learning to predict attrition. Maybe it will give findings that people do not usally think about regarding employee attrition.

Haveing a understanding of what make employees leave is important to know, if a person is leaving replacement cost could be high. Being aware of it will be easier to take action to improve to the employee attrition.

Some of the questions we want to cover during this project

- What is the likelihood of an active employee leaving the company?
- What are the key indicators of an employee leaving the company?
- What policies or strategies can be adopted based on the results to improve employee retention?

Given that we have data on former employees, this is a standard supervised classification problem where the label is a binary variable, 0 (active employee), 1 (former employee). In this study, our target variable Y is the probability of an employee leaving the company.

Some important columns in the dataset with information about personal and employment details, explained in more:

Some important columns:

1. Attrition: Whether employees are still with the company or whether they’ve gone to work somewhere else.
2. Age: 18 to 60 years old
3. Gender: Female or Male
4. Department: Research & Development, Sales, Human Resources.
5. BusinessTravel: Travel_Rarely, Travel_Frequently, Non-Travel.
6. DistanceFromHome: Distance between the company and their home in miles.
7. MonthlyIncome: Employees' numeric monthly income.
8. MaritalStatus: Married, Single, Divorced.
9. Education: 1 'Below College' 2 'College' 3 'Bachelor' 4 'Master' 5 'Doctor'.
10. EducationField: Life Sciences， Medical， Marketing，Technical Degree，Other.
11. EnvironmentSatisfaction: 1 'Low' 2 'Medium' 3 'High' 4 'Very High'.
12. RelationshipSatisfaction: 1 'Low' 2 'Medium' 3 'High' 4 'Very High'.
13. JobInvolvement: 1 'Low' 2 'Medium' 3 'High' 4 'Very High'.
14. JobRole: Sales Executive，Research Science, Laboratory Tec, Manufacturing, Healthcare Rep, etc
15. JobSatisfaction: 1 'Low' 2 'Medium' 3 'High' 4 'Very High'.
16. OverTime: Whether they work overtime or not.
17. NumCompaniesWorked: Number of companies they worked for before joinging IBM.
18. PerformanceRating: 1 'Low' 2 'Good' 3 'Excellent' 4 'Outstanding'.
19. YearsAtCompany: Years they worked for IBM.
20. WorkLifeBalance: 1 'Bad' 2 'Good' 3 'Better' 4 'Best'.
21. YearsSinceLastPromotion: Years passed since their last promotion.

---

2. Data Preparation

In [2]:

```
df_employee = pd.read_csv('WA_Fn-UseC_-HR-Employee-Attrition.csv')
```

After loading the dataset into a dataframe, using the command under we can get a good understadning how dataset is put together. Some of functions used is listed under.

```
df_employee.head()
df_employee.columns
df_employee.decribe()
df_employee.shape()
df_employee.info()
```

From the those commands we can see that the dataset contains no missing values. It is several numerical and categorical variables. From a HR prepective,these type of data about empoyees is unlikely to feature huge amount of missing data

In [3]:

```
#Taking a look at the data set
df_employee.head()
```

Out[3]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNum
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	

5 rows × 35 columns

In [4]:

```
df_employee.columns
```

Out[4]:

```
Index(['Age', 'Attrition', 'BusinessTravel', 'DailyRate', 'Department',
       'DistanceFromHome', 'Education', 'EducationField', 'EmployeeCount',
       'EmployeeNumber', 'EnvironmentSatisfaction', 'Gender', 'HourlyRate',
       'JobInvolvement', 'JobLevel', 'JobRole', 'JobSatisfaction',
       'MaritalStatus', 'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked',
       'Over18', 'OverTime', 'PercentSalaryHike', 'PerformanceRating',
       'RelationshipSatisfaction', 'StandardHours', 'StockOptionLevel',
       'TotalWorkingYears', 'TrainingTimesLastYear', 'WorkLifeBalance',
       'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion',
       'YearsWithCurrManager'],
      dtype='object')
```

In [5]:

```
df_employee.shape
```

Out[5]:

```
(1470, 35)
```

In [6]:

```
df_employee.describe()
```

Out[6]:

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	Hourly
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	1470.000000	1470.000000	1470.00
mean	36.923810	802.485714	9.192517	2.912925	1.0	1024.865306	2.721769	65.89
std	9.135373	403.509100	8.106864	1.024165	0.0	602.024335	1.093082	20.32
min	18.000000	102.000000	1.000000	1.000000	1.0	1.000000	1.000000	30.00
25%	30.000000	465.000000	2.000000	2.000000	1.0	491.250000	2.000000	48.00
50%	36.000000	802.000000	7.000000	3.000000	1.0	1020.500000	3.000000	66.00
75%	43.000000	1157.000000	14.000000	4.000000	1.0	1555.750000	4.000000	83.75
max	60.000000	1499.000000	29.000000	5.000000	1.0	2068.000000	4.000000	100.00

8 rows × 26 columns

In [7]:

```
df_employee.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 35 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   Age                       1470 non-null   int64 
 1   Attrition                 1470 non-null   object
 2   BusinessTravel            1470 non-null   object
 3   DailyRate                 1470 non-null   int64 
 4   Department                1470 non-null   object
 5   DistanceFromHome          1470 non-null   int64 
 6   Education                 1470 non-null   int64 
 7   EducationField            1470 non-null   object
 8   EmployeeCount             1470 non-null   int64 
 9   EmployeeNumber            1470 non-null   int64 
 10  EnvironmentSatisfaction   1470 non-null   int64 
 11  Gender                    1470 non-null   object
 12  HourlyRate                1470 non-null   int64 
 13  JobInvolvement            1470 non-null   int64 
 14  JobLevel                  1470 non-null   int64 
 15  JobRole                   1470 non-null   object
 16  JobSatisfaction           1470 non-null   int64 
 17  MaritalStatus             1470 non-null   object
 18  MonthlyIncome             1470 non-null   int64 
 19  MonthlyRate               1470 non-null   int64 
 20  NumCompaniesWorked        1470 non-null   int64 
 21  Over18                    1470 non-null   object
 22  OverTime                  1470 non-null   object
 23  PercentSalaryHike         1470 non-null   int64 
 24  PerformanceRating         1470 non-null   int64 
 25  RelationshipSatisfaction  1470 non-null   int64 
 26  StandardHours             1470 non-null   int64 
 27  StockOptionLevel          1470 non-null   int64 
 28  TotalWorkingYears         1470 non-null   int64 
 29  TrainingTimesLastYear     1470 non-null   int64 
 30  WorkLifeBalance           1470 non-null   int64 
 31  YearsAtCompany            1470 non-null   int64 
 32  YearsInCurrentRole        1470 non-null   int64 
 33  YearsSinceLastPromotion   1470 non-null   int64 
 34  YearsWithCurrManager      1470 non-null   int64 
dtypes: int64(26), object(9)
memory usage: 402.1+ KB
```

In [8]:

```
df_employee.isnull().sum()
```

Out[8]:

```
Age                         0
Attrition                   0
BusinessTravel              0
DailyRate                   0
Department                  0
DistanceFromHome            0
Education                   0
EducationField              0
EmployeeCount               0
EmployeeNumber              0
EnvironmentSatisfaction     0
Gender                      0
HourlyRate                  0
JobInvolvement              0
JobLevel                    0
JobRole                     0
JobSatisfaction             0
MaritalStatus               0
MonthlyIncome               0
MonthlyRate                 0
NumCompaniesWorked          0
Over18                      0
OverTime                    0
PercentSalaryHike           0
PerformanceRating           0
RelationshipSatisfaction    0
StandardHours               0
StockOptionLevel            0
TotalWorkingYears           0
TrainingTimesLastYear       0
WorkLifeBalance             0
YearsAtCompany              0
YearsInCurrentRole          0
YearsSinceLastPromotion     0
YearsWithCurrManager        0
dtype: int64
```

Now is time to check if all variables will give some useful insights or some of them could be deleted. To check this it is possible to loop thought and check if unique value is 1, and them drop the columns.

In [9]:

```
#this fuction is not test out yet. but will be
notneeded = []
for col in df_employee.columns:
    if len(df_employee[col].unique()) == 1:
        notneeded.append(col)
        df_employee.drop(col,inplace=True,axis=1)
```

In [10]:

```
print(notneeded)
```

```
['EmployeeCount', 'Over18', 'StandardHours']
```

In [11]:

```
df_employee.drop(['EmployeeNumber'], axis = 1, inplace = True)
```

In [12]:

```
print(df_employee.shape)
df_employee.head()
```

(1470, 31)

Out[12]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EnvironmentSatisfaction	Gend
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	2	Fema
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	3	Ma
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	4	Ma
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	4	Fema
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	Ma

5 rows × 31 columns

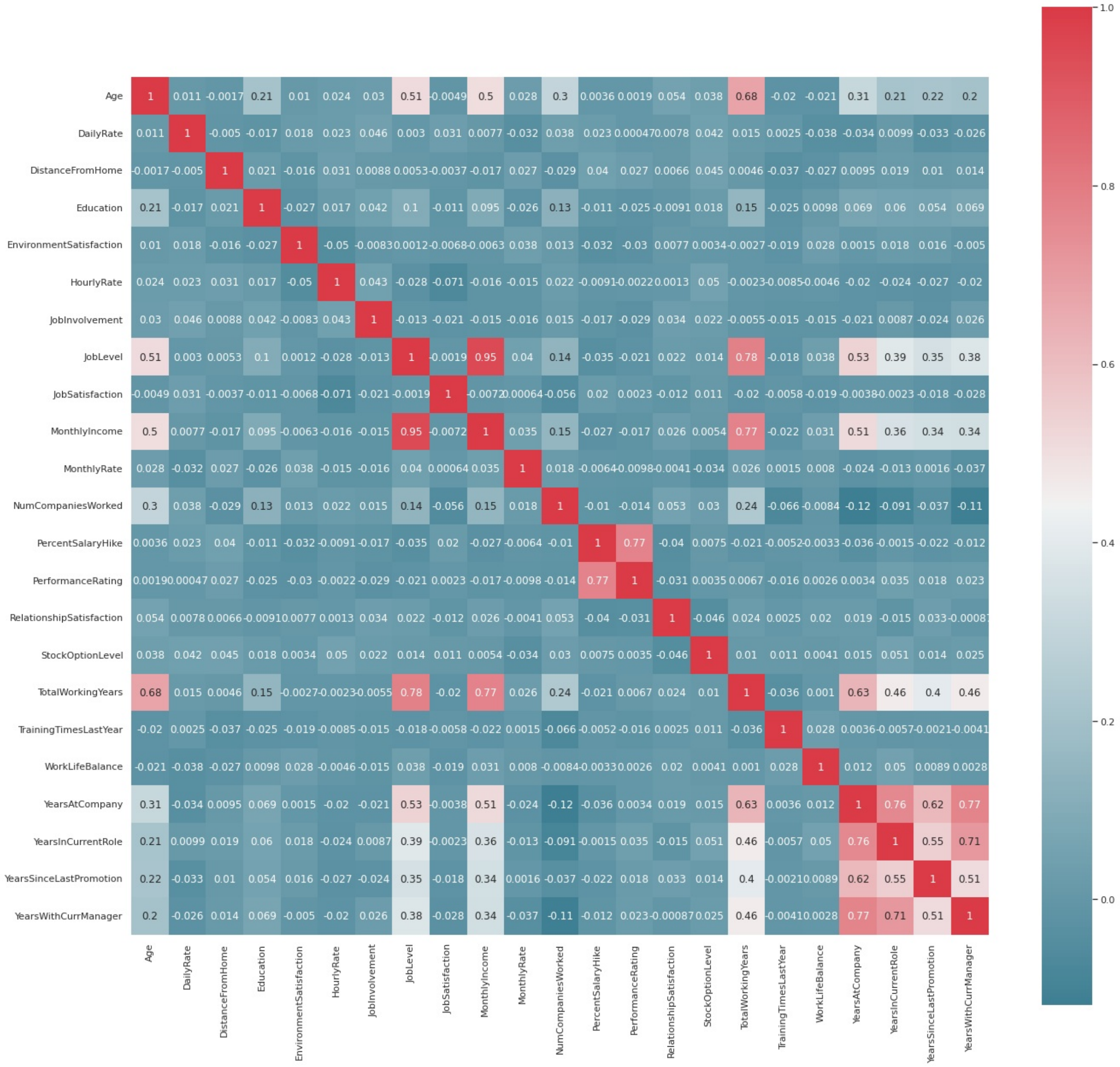
After running this loop, the columns dropped where EmployeeCount, Over18 and StandardHours Also the EmployeeNumber is just a number increaseing so that column is dropped. The Next step would be to look at how the different variables are correlated.

In [13]:

```
f, ax = plt.subplots(figsize=(20, 20))
corr = df_employee.corr()
sns.heatmap(corr, mask=np.zeros_like(corr, dtype=np.bool), cmap=sns.diverging_palette(220, 10, as_cmap=True),
            square=True, ax=ax, annot = True)
```

Out[13]:

<AxesSubplot:>



Conclutions from Data preparing:

- Several numerical and categorical columns with information about employee's personal and employment details
- There are no missing values in this dataset, hence HR do not tend to have huge amout of missing data
- Dropped the not useful cloumns
- Important correlations:
  - Age - TotalworkingYears
  - Age - JobLevel
  - JobLevel - MonthlyIncome
  - JobLevel - TotalWorkingYears
  - YearsAtCompany -YearsInCurrentRole
  - MonthlyIncome - TotalWorkingYears

---

3. EDA (Exploratory Data Analysis)

3.1 General feature statistics

Starting the EDA of with some histograms of the for numerical features.

In [14]:

```
df_hrs = df_employee.copy()
df_hr_cat_name = df_employee.copy()
df_Anumber = df_employee.copy()
```

In [15]:

```
df_employee.hist(figsize=(20,20))
plt.show()
```

- What we can see from the histograms is that many of them is right skwed
- Age distribution is more towards the younger genration between 25 and 45 years old
- A lot of people are in the compnay less than 10 years

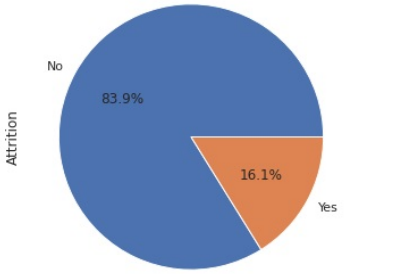
Attrtion Rate

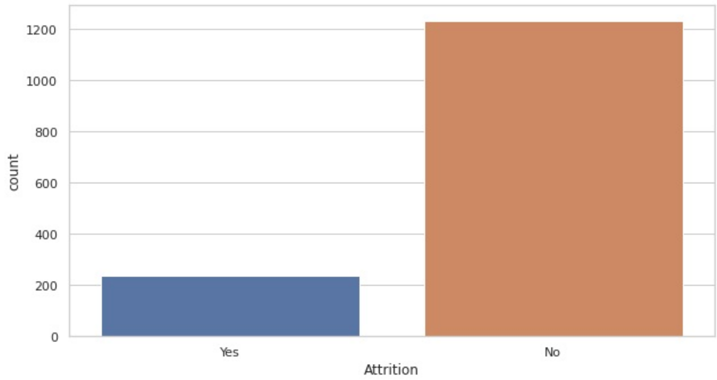
In [16]:

```
plt.figure(figsize=(20,5))
plt.subplot(1,2,1)
df_employee['Attrition'].value_counts().plot.pie(autopct='%1.1f%%')
plt.subplot(1,2,2)
sns.countplot(df_employee['Attrition'])
df_employee['Attrition'].value_counts().to_frame()
```

Out[16]:

	Attrition
No	1233
Yes	237





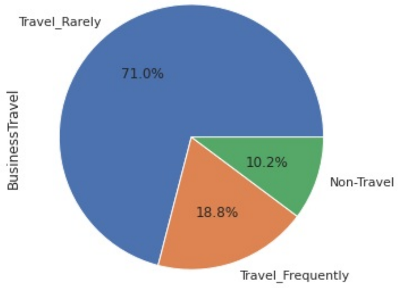
From the attrition rate graphs we can see that the majority is still there. Important mention, piechart and bar chart interpert the colors differently.

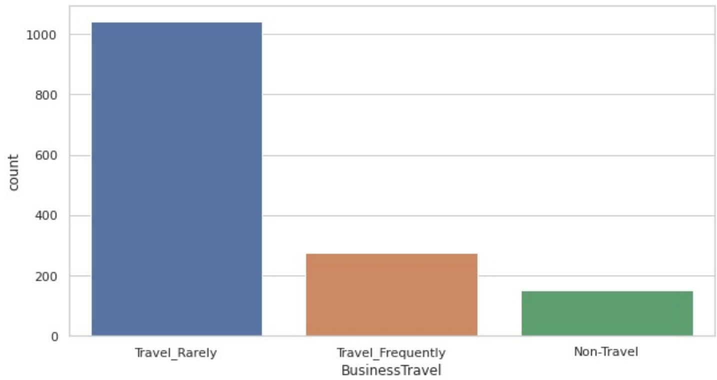
Travelling

In [17]:

```
plt.figure(figsize=(20,5))
plt.subplot(1,2,1)
df_employee['BusinessTravel'].value_counts().plot.pie(autopct='%1.1f%%')
plt.subplot(1,2,2)
sns.countplot(df_employee['BusinessTravel'])
#df_employee['BusinessTravel'].value_counts().to_frame()
print(df_employee.groupby('BusinessTravel')['Attrition'].value_counts())
#df_employee.groupby('BusinessTravel')['Attrition'].value_counts
#plt.subplot(1,3,3)
#df_employee.groupby('Attrition')['BusinessTravel'].value_counts(df_employee.Attrition.all()).plot.pie(autopct='%
1.1f%%',figsize=(11,6))
#print(df_employee.groupby(['BusinessTravel','Gender'])['Attrition'].value_counts(100. * df_employee.Attrition.va
lue_counts() / len(df_employee.Attrition)))
```

```
BusinessTravel     Attrition
Non-Travel         No           138
                   Yes           12
Travel_Frequently  No           208
                   Yes           69
Travel_Rarely      No           887
                   Yes          156
Name: Attrition, dtype: int64
```





The business travels have a clear amount that travel rarely, the person travel frequently or do not travel is small

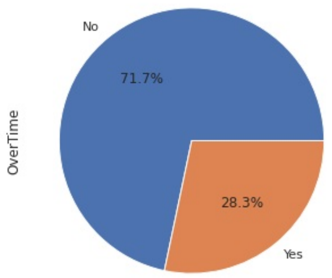
OverTime

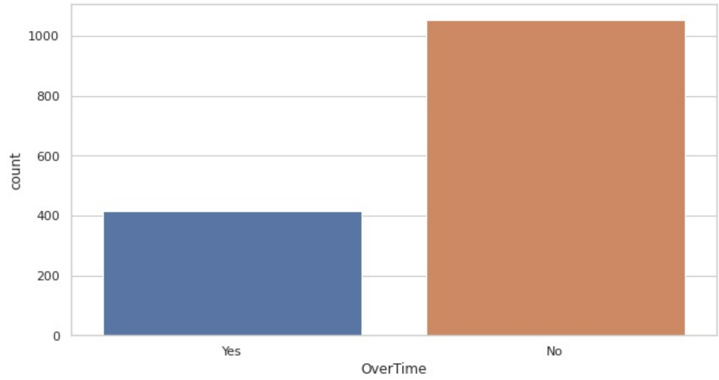
In [18]:

```
plt.figure(figsize=(20,5))
plt.subplot(1,2,1)
df_employee['OverTime'].value_counts().plot.pie(autopct='%1.1f%%')
plt.subplot(1,2,2)
sns.countplot(df_employee['OverTime'])
#df_employee['OverTime'].value_counts().to_frame()
df_employee.groupby('OverTime')['Attrition'].value_counts()
```

Out[18]:

```
OverTime  Attrition
No        No           944
          Yes          110
Yes       No           289
          Yes          127
Name: Attrition, dtype: int64
```





1/3 of the people tend to have overtime

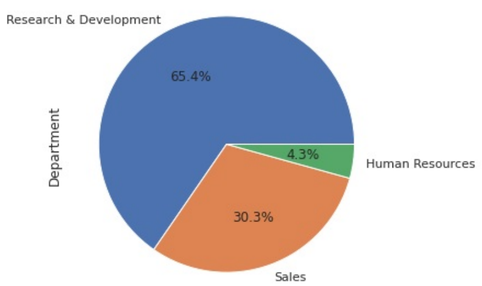
Department

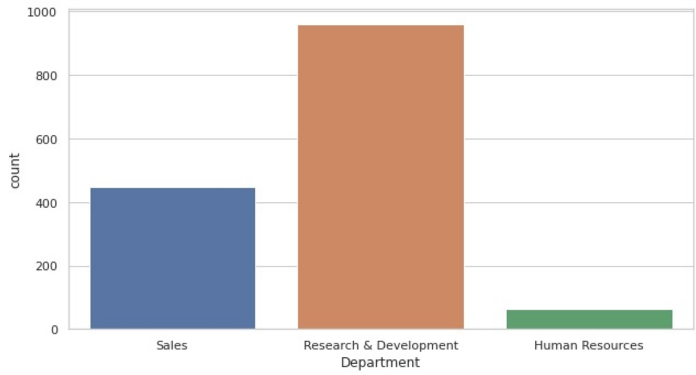
In [19]:

```
plt.figure(figsize=(20,5))
plt.subplot(1,2,1)
df_employee['Department'].value_counts().plot.pie(autopct='%1.1f%%')
plt.subplot(1,2,2)
sns.countplot(df_employee['Department'])
#print(df_employee.groupby('Department')['Attrition'].value_counts())
df_employee.groupby('Department')['Attrition'].value_counts()
#pd.pivot_table(df_employee, values = 'Department', index='Attrition').reset_index()
#sns.countplot(df_employee.groupby('Department',)['Attrition'])
#df_employee['Department'].value_counts().to_frame()
```

Out[19]:

```
Department              Attrition
Human Resources         No            51
                        Yes           12
Research & Development  No           828
                        Yes          133
Sales                   No           354
                        Yes           92
Name: Attrition, dtype: int64
```





The majority of the people are part of the research & Development department with 65%, also sales department are big with 30%

Education Level

In [20]:

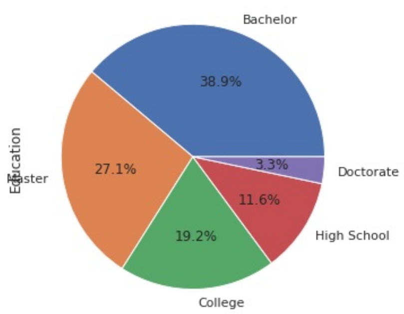
```
#need to change the value to the column to get a better understanding of what the graph says
df_employee.Education.replace({1: 'High School', 2:'College', 3:'Bachelor', 4:'Master', 5:'Doctorate'},inplace=True)
df_hrs.Education.replace({1: 'High School', 2:'College', 3:'Bachelor', 4:'Master', 5:'Doctorate'},inplace=True)
#df_employee.Education.replace({'High School':1, 'Undergrad':2,'Graduate':3, 'Post Graduate':4, 'Doctorate':5},inplace=True)
```

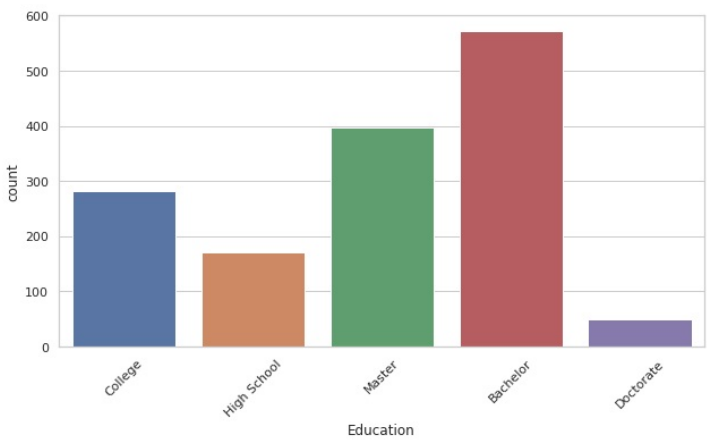
In [21]:

```
plt.figure(figsize=(20,5))
plt.subplot(1,2,1)
df_employee['Education'].value_counts().plot.pie(autopct='%1.1f%%')
plt.subplot(1,2,2)
sns.countplot(df_employee['Education'])
plt.xticks(rotation=45)
#df_employee['Education'].value_counts().to_frame()
df_employee.groupby('Education')['Attrition'].value_counts()
```

Out[21]:

```
Education    Attrition
Bachelor     No           473
             Yes           99
College      No           238
             Yes           44
Doctorate    No            43
             Yes            5
High School  No           139
             Yes           31
Master       No           340
             Yes           58
Name: Attrition, dtype: int64
```





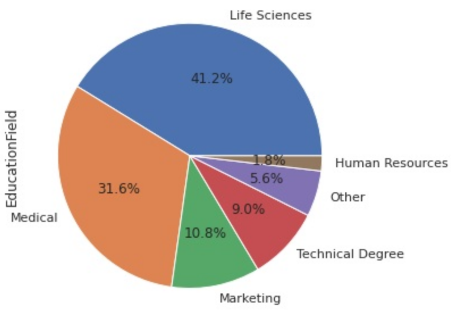
in the education level is a clear for persons having bacheolor andmaster degree

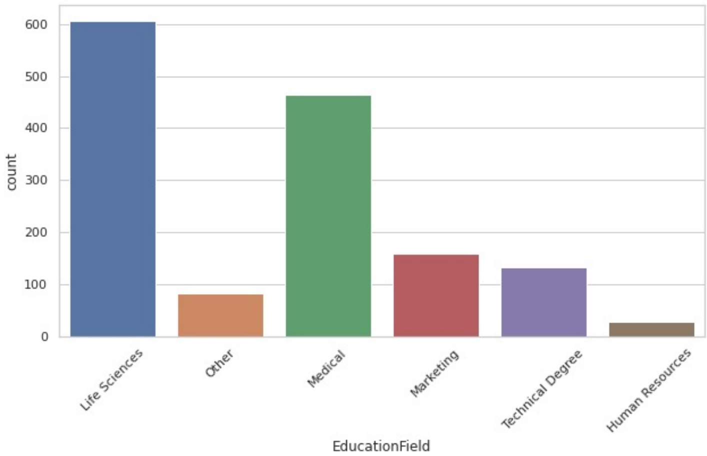
Education Field

In [22]:

```
plt.figure(figsize=(20,5))
plt.subplot(1,2,1)
df_employee['EducationField'].value_counts().plot.pie(autopct='%1.1f%%')
plt.subplot(1,2,2)
sns.countplot(df_employee['EducationField'])
plt.xticks(rotation=45)
#df_employee['EducationField'].value_counts().to_frame()
print(df_employee.groupby('EducationField')['Attrition'].value_counts().to_frame())
```

```
                              Attrition
EducationField   Attrition
Human Resources  No                  20
                 Yes                  7
Life Sciences    No                 517
                 Yes                 89
Marketing        No                 124
                 Yes                 35
Medical          No                 401
                 Yes                 63
Other            No                  71
                 Yes                 11
Technical Degree No                 100
                 Yes                 32
```





There are two field that are dominant both medical and life science is

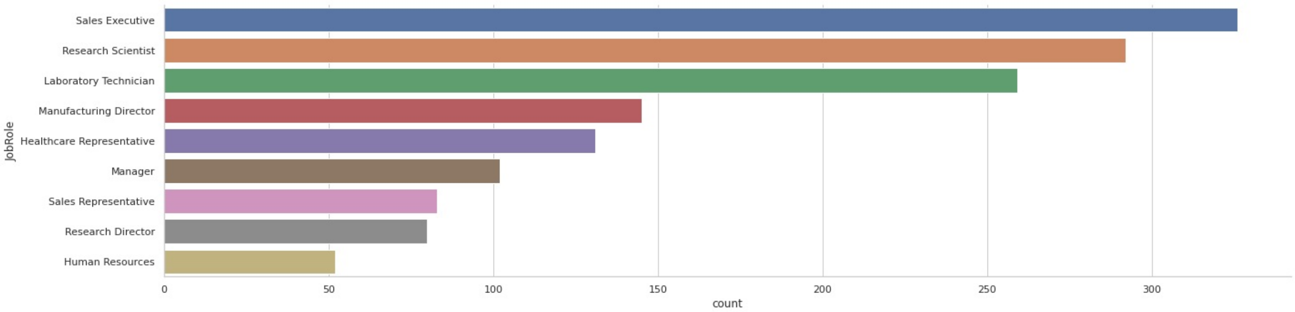
JobrRole

In [23]:

```
plt.figure(figsize=(20,10))
sns.catplot(y='JobRole', kind='count', aspect=4, data=df_employee)
print(df_employee['JobRole'].value_counts())
print(df_employee.groupby('JobRole')['Attrition'].value_counts())
```

```
Sales Executive              326
Research Scientist           292
Laboratory Technician        259
Manufacturing Director       145
Healthcare Representative    131
Manager                      102
Sales Representative          83
Research Director             80
Human Resources               52
Name: JobRole, dtype: int64
JobRole                    Attrition
Healthcare Representative  No           122
                           Yes            9
Human Resources            No            40
                           Yes           12
Laboratory Technician      No           197
                           Yes           62
Manager                    No            97
                           Yes            5
Manufacturing Director     No           135
                           Yes           10
Research Director          No            78
                           Yes            2
Research Scientist         No           245
                           Yes           47
Sales Executive            No           269
                           Yes           57
Sales Representative       No            50
                           Yes           33
Name: Attrition, dtype: int64
```

```
<Figure size 1440x720 with 0 Axes>
```



We can see the divition of different job roles, there are most sales excutive and least working in human resources.

In [24]:

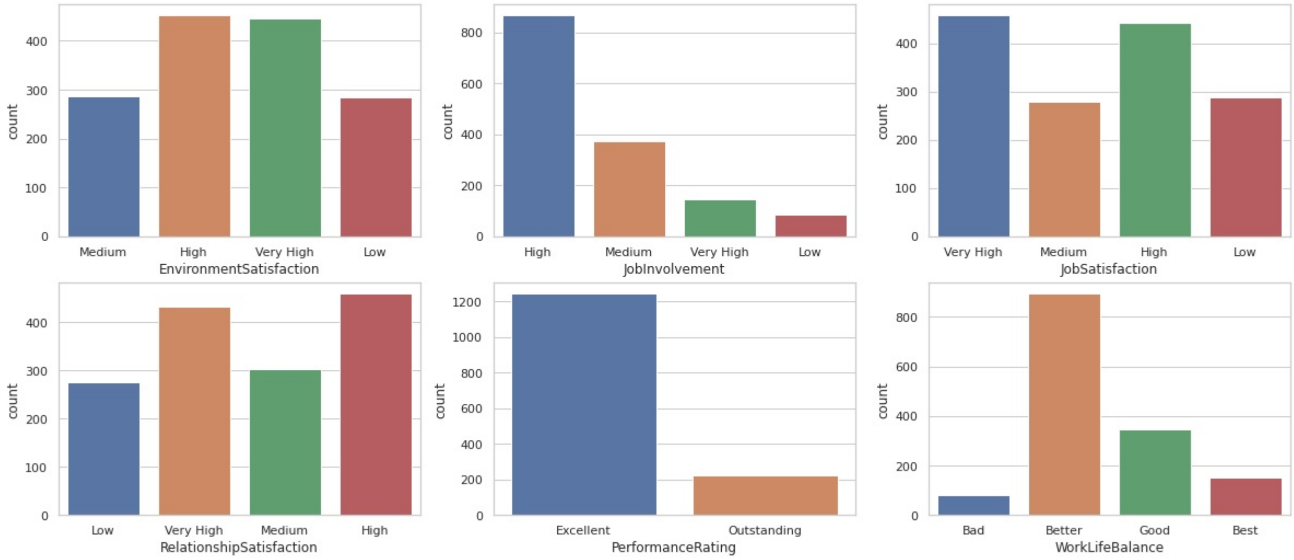
```
# Changing numeric values to corresponding categorical values
df_employee['EnvironmentSatisfaction'] = df_employee['EnvironmentSatisfaction'].map({1: 'Low', 2: 'Medium', 3: 'High', 4: 'Very High'})
df_employee['JobInvolvement'] = df_employee['JobInvolvement'].map({1: 'Low', 2: 'Medium', 3: 'High', 4: 'Very High'})
df_employee['JobSatisfaction'] = df_employee['JobSatisfaction'].map({1: 'Low', 2: 'Medium', 3: 'High', 4: 'Very High'})
df_employee['RelationshipSatisfaction'] = df_employee['RelationshipSatisfaction'].map({1: 'Low', 2: 'Medium', 3: 'High', 4: 'Very High'})
df_employee['PerformanceRating'] = df_employee['PerformanceRating'].map({1: 'Low', 2: 'Good', 3: 'Excellent', 4: 'Outstanding'})
df_employee['WorkLifeBalance'] = df_employee['WorkLifeBalance'].map({1: 'Bad', 2: 'Good', 3: 'Better', 4: 'Best'})
```

In [25]:

```
plt.figure(figsize=(18,8))
plt.subplot(2,3,1)
sns.countplot(df_employee['EnvironmentSatisfaction'])
plt.subplot(2,3,2)
sns.countplot(df_employee['JobInvolvement'])
plt.subplot(2,3,3)
sns.countplot(df_employee['JobSatisfaction'])
plt.subplot(2,3,4)
sns.countplot(df_employee['RelationshipSatisfaction'])
plt.subplot(2,3,5)
sns.countplot(df_employee['PerformanceRating'])
plt.subplot(2,3,6)
sns.countplot(df_employee['WorkLifeBalance'])
```

Out[25]:

```
<AxesSubplot:xlabel='WorkLifeBalance', ylabel='count'>
```



- EnvironmentSatisfaction: 1 'Low' 2 'Medium' 3 'High' 4 'Very High'.
- RelationshipSatisfaction: 1 'Low' 2 'Medium' 3 'High' 4 'Very High'.
- JobInvolvement: 1 'Low' 2 'Medium' 3 'High' 4 'Very High'.
- JobSatisfaction: 1 'Low' 2 'Medium' 3 'High' 4 'Very High'.
- PerformanceRating: 1 'Low' 2 'Good' 3 'Excellent' 4 'Outstanding'.
- WorkLifeBalance: 1 'Bad' 2 'Good' 3 'Better' 4 'Best'.

Overall is the metrics, mostly from maybe surveys show that people are happy and scoring high on the metrics.

Wonder why the perfomance rating is only 3 and 4. More people are overall more satified with their situation.

We can see the divition of different job roles, there are most sales excutive and least working in human resources.

3.2 Attrition effected by Age and MaritalStatus?

1. Hypothesis: Single people tend to leave more often than married people?

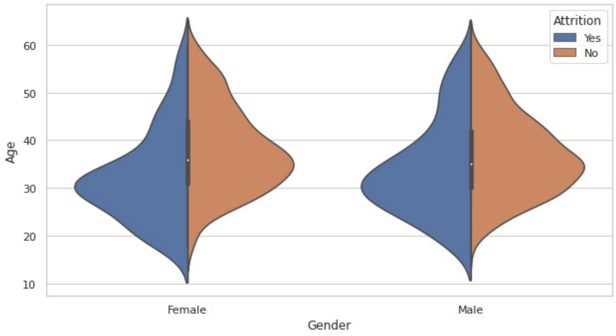
2. Hypothesis: Male are more active leavers?

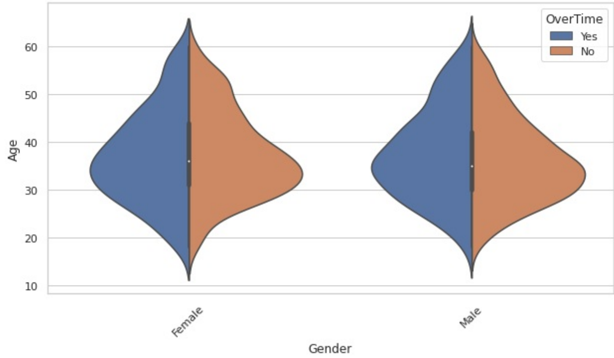
In [26]:

```
plt.figure(figsize=(20,5))
plt.subplot(1,2,1)
sns.violinplot(x ="Gender", y ="Age", hue ="Attrition",
data = df_hrs, split = True)
plt.subplot(1,2,2)
sns.violinplot(x ="Gender", y ="Age", hue ="OverTime",
data = df_hrs, split = True)
plt.xticks(rotation=45)
#df_employee['OverTime'].value_counts()
```

Out[26]:

```
(array([0, 1]), [Text(0, 0, 'Female'), Text(1, 0, 'Male')])
```



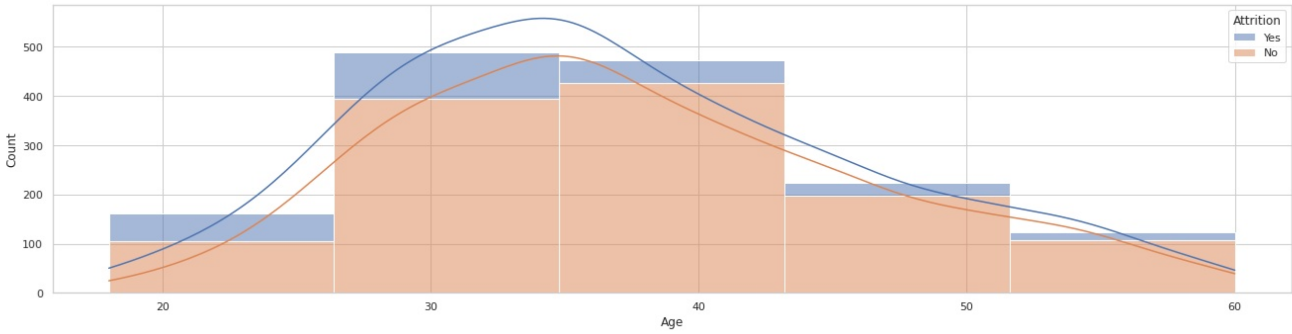


In [27]:

```
plt.subplots(figsize=(20,5))
sns.histplot(data=df_employee, x="Age", hue="Attrition", multiple="stack",bins=5,kde=True)
#df_employee['Gender'].value_counts(normalize=True)
df_employee.groupby('Gender')['Attrition'].value_counts()
```

Out[27]:

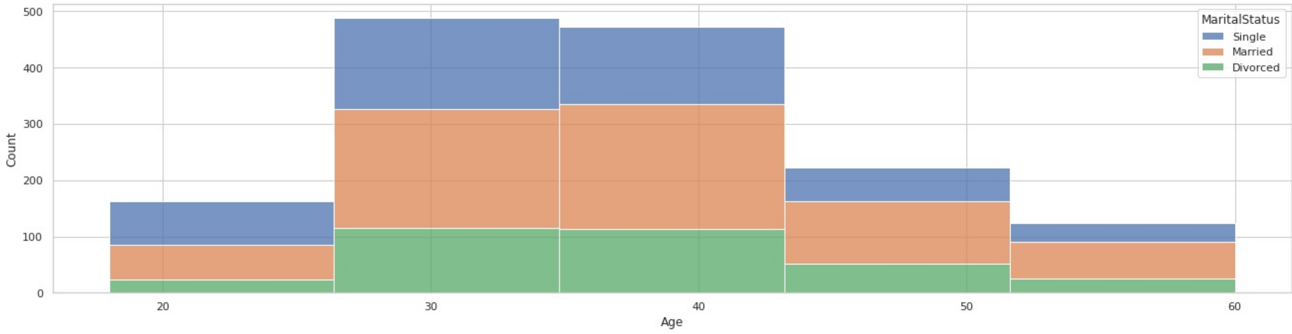
```
Gender  Attrition
Female  No           501
        Yes           87
Male    No           732
        Yes          150
Name: Attrition, dtype: int64
```



In [28]:

```
plt.subplots(figsize=(20,5))
sns.histplot(data=df_employee, x="Age", hue="MaritalStatus", multiple="stack",bins=5)
print(df_employee['MaritalStatus'].value_counts(normalize=True))
print(df_employee.groupby(['MaritalStatus','Gender'])['Attrition'].value_counts())
```

```
Married     0.457823
Single      0.319728
Divorced    0.222449
Name: MaritalStatus, dtype: float64
MaritalStatus  Gender  Attrition
Divorced       Female  No           108
                       Yes            9
               Male    No           186
                       Yes           24
Married        Female  No           241
                       Yes           31
               Male    No           348
                       Yes           53
Single         Female  No           152
                       Yes           47
               Male    No           198
                       Yes           73
Name: Attrition, dtype: int64
```

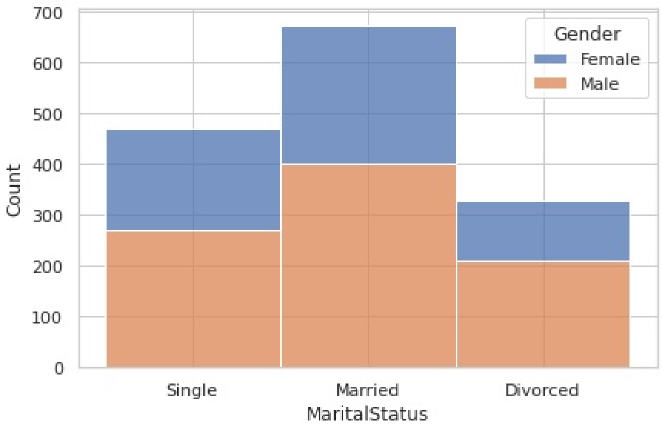


In [29]:

```
sns.histplot(data=df_employee, x="MaritalStatus", hue="Gender", multiple="stack",bins=5)
```

Out[29]:

```
<AxesSubplot:xlabel='MaritalStatus', ylabel='Count'>
```



From the part 3.2 we can see:

- that there are more men (150) leaving than women (87): *Hypothesis 2 confirmed*
- 45% Married Workers, 22% Singles and 31% Divorced.
- More single leave more Married: *Hypothesis 1 confirmed*
- Divorced women are stayers only 9 people leaving.

In [30]:

```
df_Anumber['Attrition'] = df_Anumber['Attrition'].map({'Yes': 1, 'No': 0})
#df_hr_cat_name['Attrition'] = df_hr_cat_name['Attrition'].map({'Yes': 1, 'No': 0})
```

3.3 Education, EducationField, JobRole and JobLevel

In this Chapther we are taking a look at the different JobRoles and Joblevels and connect that to Education and Education Field. And see who are more likely to leave **3. Hypothesis**: Workers in lower JobLevel are morelikely to leave

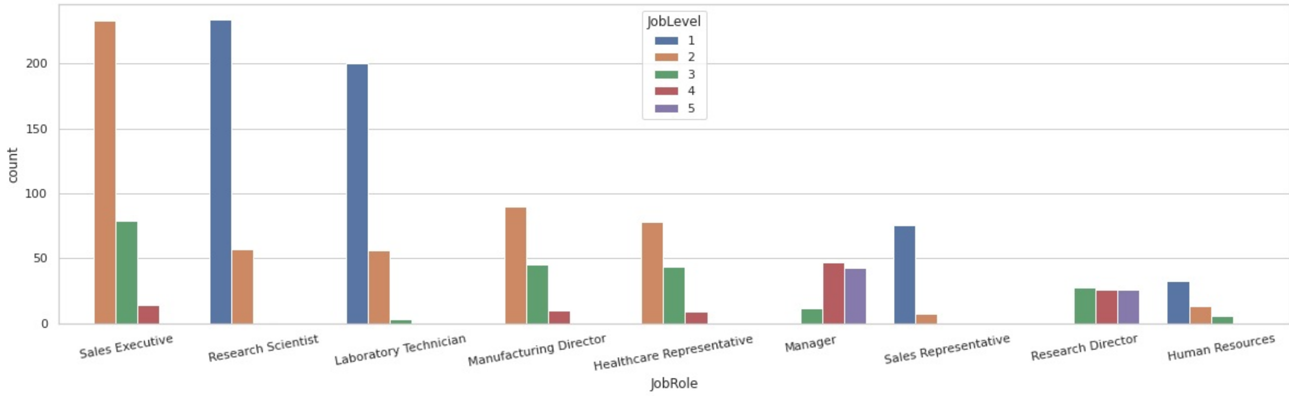
4. Hypothesis: Lower Education get more training

In [31]:

```
plt.subplots(figsize=(18,5))
sns.countplot(df_employee.JobRole, hue=df_employee.JobLevel)
plt.xticks(rotation=10)
```

Out[31]:

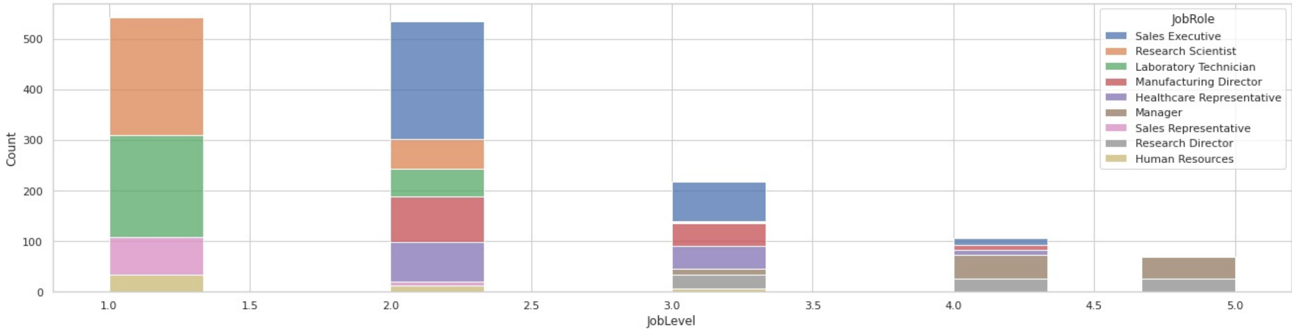
```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8]),
 [Text(0, 0, 'Sales Executive'),
  Text(1, 0, 'Research Scientist'),
  Text(2, 0, 'Laboratory Technician'),
  Text(3, 0, 'Manufacturing Director'),
  Text(4, 0, 'Healthcare Representative'),
  Text(5, 0, 'Manager'),
  Text(6, 0, 'Sales Representative'),
  Text(7, 0, 'Research Director'),
  Text(8, 0, 'Human Resources')])
```



In [32]:

```
plt.subplots(figsize=(20,5))
sns.histplot(data=df_employee, x="JobLevel" ,hue="JobRole", multiple="stack")
print(df_employee.groupby('JobLevel',)['JobRole'].value_counts(normalize=True))
print(df_employee.groupby(['JobLevel', 'JobRole'])['Attrition'].value_counts().sort_index())
```

```
JobLevel  JobRole
1         Research Scientist           0.430939
          Laboratory Technician        0.368324
          Sales Representative         0.139963
          Human Resources              0.060773
2         Sales Executive              0.436330
          Manufacturing Director       0.168539
          Healthcare Representative    0.146067
          Research Scientist           0.106742
          Laboratory Technician        0.104869
          Human Resources              0.024345
          Sales Representative         0.013109
3         Sales Executive              0.362385
          Manufacturing Director       0.206422
          Healthcare Representative    0.201835
          Research Director            0.128440
          Manager                      0.055046
          Human Resources              0.027523
          Laboratory Technician        0.013761
          Research Scientist           0.004587
4         Manager                      0.443396
          Research Director            0.245283
          Sales Executive              0.132075
          Manufacturing Director       0.094340
          Healthcare Representative    0.084906
5         Manager                      0.623188
          Research Director            0.376812
Name: JobRole, dtype: float64
JobLevel  JobRole                    Attrition
1         Human Resources            No            23
                                     Yes           10
          Laboratory Technician      No           144
                                     Yes           56
          Research Scientist         No           189
                                     Yes           45
          Sales Representative       No            44
                                     Yes           32
2         Healthcare Representative  No            75
                                     Yes            3
          Human Resources            No            13
          Laboratory Technician      No            51
                                     Yes            5
          Manufacturing Director     No            85
                                     Yes            5
          Research Scientist         No            55
                                     Yes            2
          Sales Executive            No           197
                                     Yes           36
          Sales Representative       No             6
                                     Yes            1
3         Healthcare Representative  No            39
                                     Yes            5
          Human Resources            No             4
                                     Yes            2
          Laboratory Technician      No             2
                                     Yes            1
          Manager                    No            10
                                     Yes            2
          Manufacturing Director     No            40
                                     Yes            5
          Research Director          No            28
          Research Scientist         No             1
          Sales Executive            No            62
                                     Yes           17
4         Healthcare Representative  No             8
                                     Yes            1
          Manager                    No            47
          Manufacturing Director     No            10
          Research Director          No            26
          Sales Executive            No            10
                                     Yes            4
5         Manager                    No            40
                                     Yes            3
          Research Director          No            24
                                     Yes            2
Name: Attrition, dtype: int64
```



From JobLevel and JobRole we can see that:

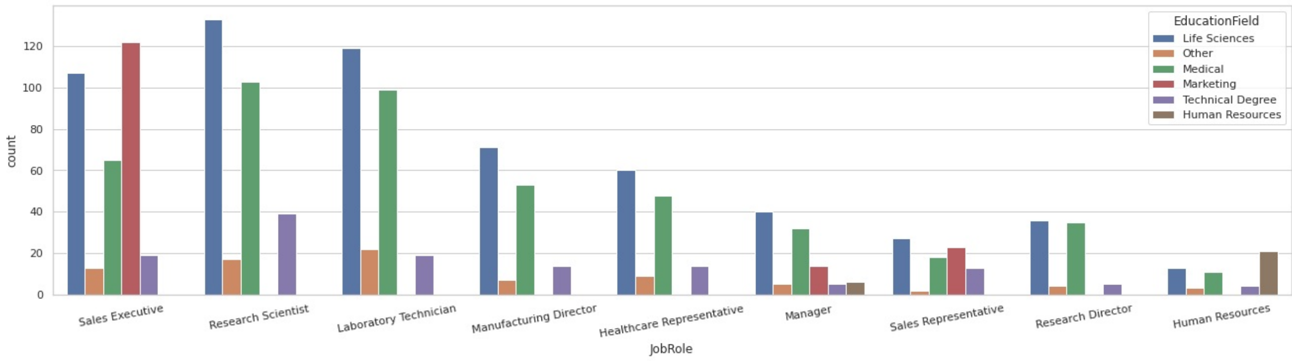
1. There are only Managers and Research Directors at level 5
2. The majority of people in Level 1 are only are low level workers
3. Only 2 roles Total Nr: 69, Roles: 62% Manager, 38% Resea

In [33]:

```
plt.subplots(figsize=(20,5))
sns.countplot(df_employee.JobRole, hue=df_employee.EducationField)
plt.xticks(rotation=10)
```

Out[33]:

```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8]),
 [Text(0, 0, 'Sales Executive'),
  Text(1, 0, 'Research Scientist'),
  Text(2, 0, 'Laboratory Technician'),
  Text(3, 0, 'Manufacturing Director'),
  Text(4, 0, 'Healthcare Representative'),
  Text(5, 0, 'Manager'),
  Text(6, 0, 'Sales Representative'),
  Text(7, 0, 'Research Director'),
  Text(8, 0, 'Human Resources')])
```



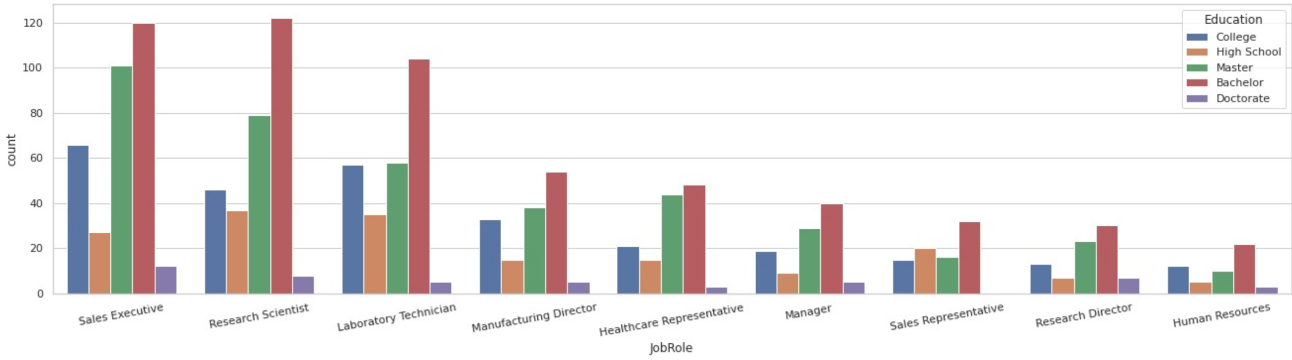
The dominant Education field are Life science and medical. for sales executives marking are important

In [34]:

```
plt.subplots(figsize=(20,5))
sns.countplot(df_employee.JobRole, hue=df_employee.Education)
plt.xticks(rotation=10)
```

Out[34]:

```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8]),
 [Text(0, 0, 'Sales Executive'),
  Text(1, 0, 'Research Scientist'),
  Text(2, 0, 'Laboratory Technician'),
  Text(3, 0, 'Manufacturing Director'),
  Text(4, 0, 'Healthcare Representative'),
  Text(5, 0, 'Manager'),
  Text(6, 0, 'Sales Representative'),
  Text(7, 0, 'Research Director'),
  Text(8, 0, 'Human Resources')])
```



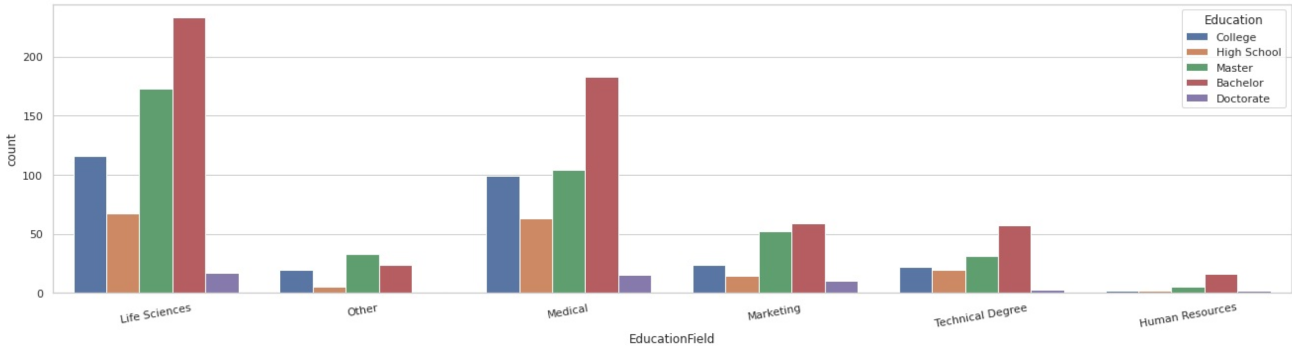
more people have bachelors ad masters degree

In [35]:

```
plt.subplots(figsize=(20,5))
sns.countplot(df_employee.EducationField, hue=df_employee.Education)
plt.xticks(rotation=10)
```

Out[35]:

```
(array([0, 1, 2, 3, 4, 5]),
 [Text(0, 0, 'Life Sciences'),
  Text(1, 0, 'Other'),
  Text(2, 0, 'Medical'),
  Text(3, 0, 'Marketing'),
  Text(4, 0, 'Technical Degree'),
  Text(5, 0, 'Human Resources')])
```



Moving over to Traing for the employees

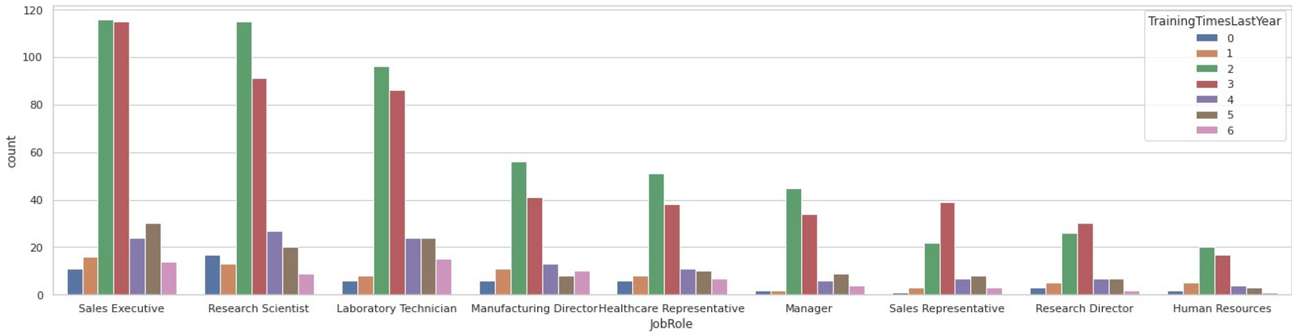
In [36]:

```
print(df_employee['TrainingTimesLastYear'].value_counts(['JobRole']))
print(df_employee.groupby(['TrainingTimesLastYear'])['Attrition'].value_counts(normalize=True).sort_index())
print(df_employee['TrainingTimesLastYear'].value_counts(['Attrition']))
plt.subplots(figsize=(20,5))
sns.countplot(df_employee.JobRole, hue=df_employee.TrainingTimesLastYear)
#print(df_employee['TrainingTimesLastYear'].value_counts(['JobRole']))
```

```
2    0.372109
3    0.334014
4    0.083673
5    0.080952
1    0.048299
6    0.044218
0    0.036735
Name: TrainingTimesLastYear, dtype: float64
TrainingTimesLastYear  Attrition
0                      No           0.722222
                       Yes          0.277778
1                      No           0.873239
                       Yes          0.126761
2                      No           0.820841
                       Yes          0.179159
3                      No           0.859470
                       Yes          0.140530
4                      No           0.788618
                       Yes          0.211382
5                      No           0.882353
                       Yes          0.117647
6                      No           0.907692
                       Yes          0.092308
Name: Attrition, dtype: float64
2    0.372109
3    0.334014
4    0.083673
5    0.080952
1    0.048299
6    0.044218
0    0.036735
Name: TrainingTimesLastYear, dtype: float64
```

Out[36]:

```
<AxesSubplot:xlabel='JobRole', ylabel='count'>
```

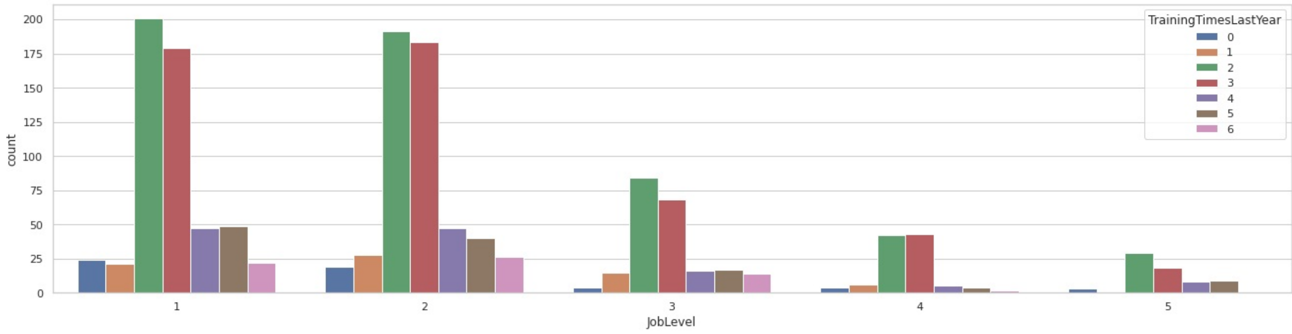


In [37]:

```
plt.subplots(figsize=(20,5))
sns.countplot(df_employee.JobLevel, hue=df_employee.TrainingTimesLastYear)
```

Out[37]:

```
<AxesSubplot:xlabel='JobLevel', ylabel='count'>
```



In [38]:

```
#df_employee.groupby(["JobRole"]).count().sort_values(["TrainingTimesLastYear"]==0)
#df1 = df_employee.melt(var_name='JobRole', value_name='TrainingTimesLastYear')
#df1
#df1 = df.melt(var_name='columns', value_name='index')
#df.apply(lambda x: x.value_counts())
print(df_employee['TrainingTimesLastYear'].value_counts()[0])
#df2
print(df_employee.JobRole[df_employee.TrainingTimesLastYear == 0].count())
#print(df_employee.groupby('JobLevel')['TrainingTimesLastYear'].value_counts()[0])
```

```
54
54
```

Training the employees have avarage on 2,3 times per year, indepentent on the JobLevel and JobRole

In [39]:

```
def highlight_max(s):
    is_max = s == s.max()
    return ['background-color: lightgreen' if v else '' for v in is_max]

#df.style.apply(highlight_max)

def highlight_min(s):
    is_min = s == s.min()
    return ['background-color: lightblue' if v else '' for v in is_min]

#df.style.apply(highlight_max)
```

In [40]:

```
TrainAtWork = df_hrs.groupby(['JobRole','Attrition'],as_index=False)[['JobInvolvement','JobSatisfaction','TrainingTimesLastYear','YearsInCurrentRole']].mean().sort_values(by=['TrainingTimesLastYear','JobSatisfaction'])
TrainAtWork.style.apply(highlight_max).apply(highlight_min)
```

Out[40]:

	JobRole	Attrition	JobInvolvement	JobSatisfaction	TrainingTimesLastYear	YearsInCurrentRole
11	Research Director	Yes	3.000000	2.500000	1.000000	15.000000
3	Human Resources	Yes	2.500000	2.166667	2.083333	2.000000
7	Manager	Yes	2.200000	2.400000	2.200000	7.800000
1	Healthcare Representative	Yes	2.666667	2.777778	2.222222	5.000000
9	Manufacturing Director	Yes	2.600000	2.600000	2.600000	3.500000
15	Sales Executive	Yes	2.526316	2.526316	2.649123	4.192982
13	Research Scientist	Yes	2.510638	2.425532	2.659574	2.191489
5	Laboratory Technician	Yes	2.532258	2.435484	2.661290	2.129032
12	Research Scientist	No	2.853061	2.840816	2.665306	3.481633
2	Human Resources	No	2.775000	2.675000	2.700000	3.475000
8	Manufacturing Director	No	2.688889	2.688889	2.755556	5.081481
0	Healthcare Representative	No	2.737705	2.786885	2.786885	4.852459
10	Research Director	No	2.769231	2.705128	2.820513	6.064103
6	Manager	No	2.804124	2.721649	2.845361	6.381443
14	Sales Executive	No	2.754647	2.802974	2.869888	4.996283
17	Sales Representative	Yes	2.454545	2.484848	2.939394	1.242424
4	Laboratory Technician	No	2.746193	2.771574	3.040609	3.538071
16	Sales Representative	No	2.780000	2.900000	3.060000	2.520000

In [41]:

```
pd.pivot_table(TrainAtWork, values = 'TrainingTimesLastYear', index='Attrition', columns = 'JobRole').reset_index
()
```

Out[41]:

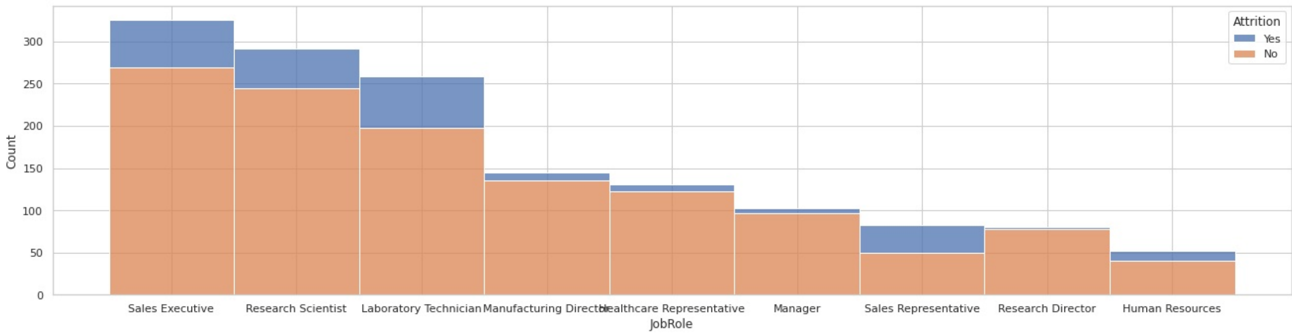
JobRole	Attrition	Healthcare Representative	Human Resources	Laboratory Technician	Manager	Manufacturing Director	Research Director	Research Scientist	Sales Executive	Sales Representative
0	No	2.786885	2.700000	3.040609	2.845361	2.755556	2.820513	2.665306	2.869888	3.060000
1	Yes	2.222222	2.083333	2.661290	2.200000	2.600000	1.000000	2.659574	2.649123	2.939394

In [42]:

```
plt.subplots(figsize=(20,5))
sns.histplot(data=df_employee, x="JobRole", hue="Attrition", multiple="stack")
```

Out[42]:

```
<AxesSubplot:xlabel='JobRole', ylabel='Count'>
```



In [43]:

```
print(df_employee.groupby('Attrition')['JobRole'].value_counts(normalize=True).sort_index())
print(df_employee.groupby('Attrition')['JobLevel'].value_counts(normalize=True).sort_index())
print(df_employee.groupby(['Attrition','JobLevel'])['Education'].value_counts().sort_index().sort_values())
print(df_employee.groupby('Attrition')['EducationField'].value_counts(normalize=True).sort_index())
```

```
Attrition  JobRole
No         Healthcare Representative    0.098946
           Human Resources              0.032441
           Laboratory Technician        0.159773
           Manager                      0.078670
           Manufacturing Director       0.109489
           Research Director            0.063260
           Research Scientist           0.198702
           Sales Executive              0.218167
           Sales Representative         0.040552
Yes        Healthcare Representative    0.037975
           Human Resources              0.050633
           Laboratory Technician        0.261603
           Manager                      0.021097
           Manufacturing Director       0.042194
           Research Director            0.008439
           Research Scientist           0.198312
           Sales Executive              0.240506
           Sales Representative         0.139241
Name: JobRole, dtype: float64
Attrition  JobLevel
No         1           0.324412
           2           0.390916
           3           0.150852
           4           0.081914
           5           0.051906
Yes        1           0.603376
           2           0.219409
           3           0.135021
           4           0.021097
           5           0.021097
Name: JobLevel, dtype: float64
Attrition  JobLevel  Education
Yes        5         Master          1
           4         Master          1
           2         Doctorate       1
           5         High School     1
           3         Doctorate       2
           1         Doctorate       2
No         5         Doctorate       2
```

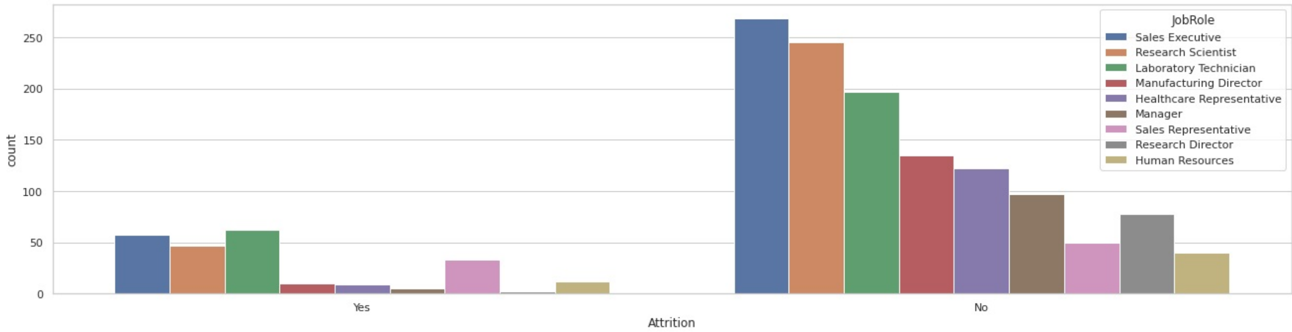
```
Yes        2         High School      3
           5         Bachelor         3
           4         Bachelor         4
           3         College          4
                     High School      4
No         5         High School      5
           1         Doctorate        6
           3         Doctorate        7
           4         High School      8
Yes        3         Master           9
No         4         Doctorate        9
           5         College         13
Yes        3         Bachelor        13
           2         Bachelor        14
                     College         15
No         3         High School     16
           4         College         17
Yes        2         Master          19
No         5         Master          19
           2         Doctorate       19
Yes        1         High School     23
No         5         Bachelor        25
Yes        1         College         25
No         4         Master          27
Yes        1         Master          28
No         3         College         29
           4         Bachelor        40
           2         High School     44
           3         Master          49
Yes        1         Bachelor        65
No         1         High School     66
                     College         69
           3         Bachelor        85
           1         Master          93
           2         College        110
                     Master         152
                     Bachelor       157
           1         Bachelor       166
Name: Education, dtype: int64
Attrition  EducationField
No         Human Resources     0.016221
           Life Sciences       0.419303
           Marketing           0.100568
           Medical             0.325223
           Other               0.057583
           Technical Degree    0.081103
Yes        Human Resources     0.029536
           Life Sciences       0.375527
           Marketing           0.147679
           Medical             0.265823
           Other               0.046414
           Technical Degree    0.135021
Name: EducationField, dtype: float64
```

In [44]:

```
plt.subplots(figsize=(20,5))
sns.countplot(df_employee.Attrition, hue=df_employee.JobRole)
df_employee.groupby('Attrition')['JobRole'].value_counts(normalize=True).sort_index()
```

Out[44]:

```
Attrition  JobRole
No         Healthcare Representative    0.098946
           Human Resources              0.032441
           Laboratory Technician        0.159773
           Manager                      0.078670
           Manufacturing Director       0.109489
           Research Director            0.063260
           Research Scientist           0.198702
           Sales Executive              0.218167
           Sales Representative         0.040552
Yes        Healthcare Representative    0.037975
           Human Resources              0.050633
           Laboratory Technician        0.261603
           Manager                      0.021097
           Manufacturing Director       0.042194
           Research Director            0.008439
           Research Scientist           0.198312
           Sales Executive              0.240506
           Sales Representative         0.139241
Name: JobRole, dtype: float64
```

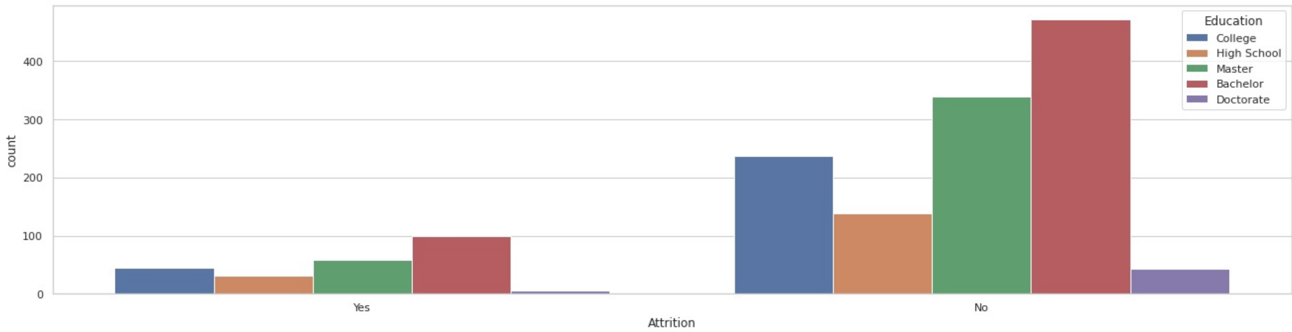


In [45]:

```
plt.subplots(figsize=(20,5))
sns.countplot(df_employee.Attrition, hue=df_employee.Education)
```

Out[45]:

```
<AxesSubplot:xlabel='Attrition', ylabel='count'>
```



In [46]:

```
#plt.subplots(figsize=(20,5))
#sns.countplot(df_employee.Attrition, hue=df_employee.EducationField)
```

In [47]:

```
#plt.subplots(figsize=(20,5))
#sns.countplot(df_employee.MaritalStatus, hue=df_employee.JobRole)
```

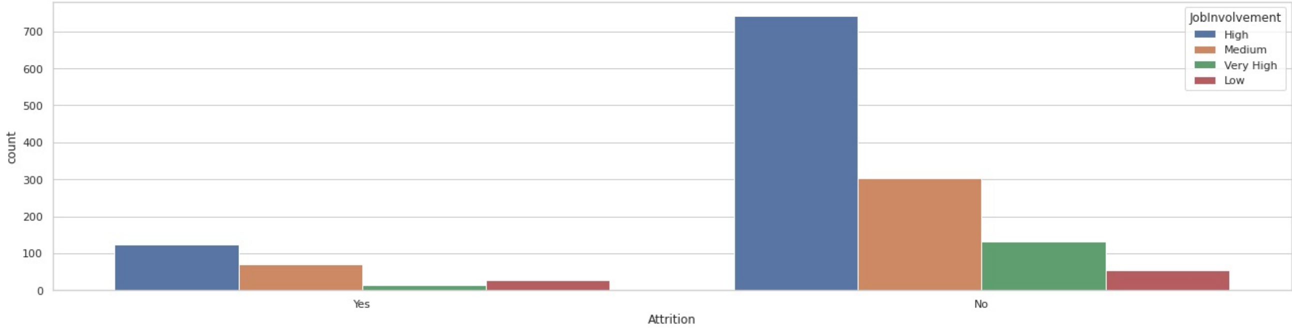
3.4 JobSatisfaction and Commitment

In [48]:

```
plt.subplots(figsize=(20,5))
sns.countplot(df_employee.Attrition, hue=df_employee.JobInvolvement)
```

Out[48]:

```
<AxesSubplot:xlabel='Attrition', ylabel='count'>
```

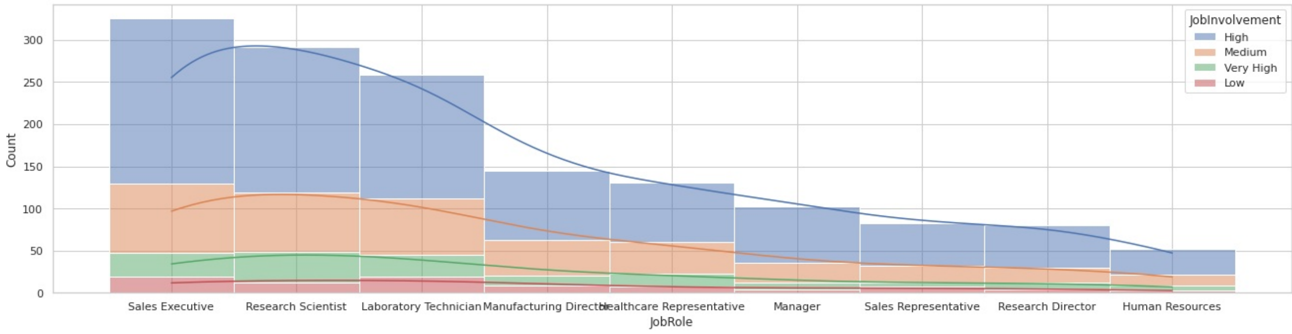


In [49]:

```
plt.subplots(figsize=(20,5))
sns.histplot(data=df_employee, x="JobRole", hue="JobInvolvement", multiple="stack",bins=5, kde=True)
```

Out[49]:

```
<AxesSubplot:xlabel='JobRole', ylabel='Count'>
```



In [50]:

```
df_employee.groupby('JobLevel')['Age'].value_counts(bins=6).sort_index()
```

Out[50]:

```
JobLevel
1         (17.958, 24.833]               87
          (24.833, 31.667]              192
          (31.667, 38.5]                157
          (38.5, 45.333]                 68
          (45.333, 52.167]               24
          (52.167, 59.0]                 15
2         (21.961, 28.333]               65
          (28.333, 34.667]              172
          (34.667, 41.0]                174
          (41.0, 47.333]                 75
          (47.333, 53.667]               28
          (53.667, 60.0]                 20
3         (26.965999999999998, 32.5]     40
          (32.5, 38.0]                   72
          (38.0, 43.5]                   34
          (43.5, 49.0]                   38
          (49.0, 54.5]                   22
          (54.5, 60.0]                   12
4         (28.968999999999998, 34.0]      5
          (34.0, 39.0]                    6
          (39.0, 44.0]                   23
          (44.0, 49.0]                   25
          (49.0, 54.0]                   31
          (54.0, 59.0]                   16
5         (38.978, 42.5]                 18
          (42.5, 46.0]                   14
          (46.0, 49.5]                    8
          (49.5, 53.0]                   15
          (53.0, 56.5]                   10
          (56.5, 60.0]                    4
Name: Age, dtype: int64
```

In [51]:

```
workForce =df_hrs.groupby(['JobRole','Attrition'], as_index=False)[['PerformanceRating','JobSatisfaction','Enviro
nmentSatisfaction','WorkLifeBalance']].mean().sort_values(by=['JobRole'])
workForce.style.apply(highlight_max).apply(highlight_min)
```

Out[51]:

	JobRole	Attrition	PerformanceRating	JobSatisfaction	EnvironmentSatisfaction	WorkLifeBalance
0	Healthcare Representative	No	3.155738	2.786885	2.819672	2.704918
1	Healthcare Representative	Yes	3.111111	2.777778	2.111111	2.666667
2	Human Resources	No	3.150000	2.675000	2.675000	2.925000
3	Human Resources	Yes	3.083333	2.166667	2.333333	2.916667
4	Laboratory Technician	No	3.147208	2.771574	2.822335	2.817259
5	Laboratory Technician	Yes	3.209677	2.435484	2.387097	2.403226
6	Manager	No	3.206186	2.721649	2.814433	2.762887
7	Manager	Yes	3.000000	2.400000	1.800000	3.000000
9	Manufacturing Director	Yes	3.000000	2.600000	2.600000	2.700000
8	Manufacturing Director	No	3.200000	2.688889	2.940741	2.770370
10	Research Director	No	3.102564	2.705128	2.487179	2.858974
11	Research Director	Yes	3.000000	2.500000	3.000000	3.000000
12	Research Scientist	No	3.151020	2.840816	2.746939	2.669388
13	Research Scientist	Yes	3.255319	2.425532	2.617021	2.723404
14	Sales Executive	No	3.130112	2.802974	2.732342	2.858736
15	Sales Executive	Yes	3.105263	2.526316	2.385965	2.543860
16	Sales Representative	No	3.160000	2.900000	2.760000	2.780000
17	Sales Representative	Yes	3.121212	2.484848	2.696970	3.060606

In [52]:

```
workForceD =df_hrs.groupby(['Department','Attrition'], as_index=False)[['PerformanceRating','JobSatisfaction','En
vironmentSatisfaction','WorkLifeBalance']].mean().sort_values(by=['Department'])
workForceD.style.apply(highlight_max).apply(highlight_min)
```

Out[52]:

	Department	Attrition	PerformanceRating	JobSatisfaction	EnvironmentSatisfaction	WorkLifeBalance
0	Human Resources	No	3.156863	2.705882	2.764706	2.921569
1	Human Resources	Yes	3.083333	2.166667	2.333333	2.916667
2	Research & Development	No	3.157005	2.769324	2.787440	2.748792
3	Research & Development	Yes	3.195489	2.458647	2.473684	2.578947
4	Sales	No	3.144068	2.810734	2.734463	2.836158
5	Sales	Yes	3.108696	2.521739	2.467391	2.739130

In [53]:

```
workForceG =df_hrs.groupby(['Gender','Attrition'], as_index=False)[['PerformanceRating','JobSatisfaction','Enviro
nmentSatisfaction','WorkLifeBalance']].mean().sort_values(by=['Gender'])
workForceG.style.apply(highlight_max).apply(highlight_min)
```

Out[53]:

	Gender	Attrition	PerformanceRating	JobSatisfaction	EnvironmentSatisfaction	WorkLifeBalance
0	Female	No	3.157685	2.728543	2.782435	2.760479
1	Female	Yes	3.172414	2.425287	2.367816	2.781609
2	Male	No	3.150273	2.812842	2.763661	2.795082
3	Male	Yes	3.146667	2.493333	2.520000	2.586667

In [54]:

```
workForceE =df_hrs.groupby(['Education','Attrition'], as_index=False)[['PerformanceRating','JobSatisfaction','Env
ironmentSatisfaction','WorkLifeBalance']].mean().sort_values(by=['Education'])
workForceE.style.apply(highlight_max).apply(highlight_min)
```

Out[54]:

	Education	Attrition	PerformanceRating	JobSatisfaction	EnvironmentSatisfaction	WorkLifeBalance
0	Bachelor	No	3.135307	2.701903	2.864693	2.737844
1	Bachelor	Yes	3.181818	2.414141	2.353535	2.686869
2	College	No	3.172269	2.823529	2.785714	2.785714
3	College	Yes	3.159091	2.477273	2.340909	2.659091
4	Doctorate	No	3.209302	2.744186	2.604651	2.767442
5	Doctorate	Yes	3.000000	2.000000	3.000000	3.200000
6	High School	No	3.187050	2.820144	2.726619	2.755396
7	High School	Yes	3.129032	2.709677	2.838710	2.870968
8	Master	No	3.144118	2.841176	2.670588	2.850000
9	Master	Yes	3.137931	2.465517	2.500000	2.448276

In [55]:

```
#workForcelevel =df_hrs.groupby(['JobLevel','Education','Attrition'], as_index=False)[['PerformanceRating','JobSa
tisfaction','EnvironmentSatisfaction','WorkLifeBalance']].mean().sort_values(by=['JobLevel'])
#workForcelevel
```

3.5 Enpowerment at Work

In [56]:

```
enpowerments =df_hrs.groupby(['JobRole','Attrition'], as_index=False)[['PerformanceRating','StockOptionLevel','Jo
bInvolvement','MonthlyIncome','YearsSinceLastPromotion','YearsAtCompany']].mean().sort_values(by=['JobRole'])
enpowerments.style.apply(highlight_max).apply(highlight_min)
```

Out[56]:

	JobRole	Attrition	PerformanceRating	StockOptionLevel	JobInvolvement	MonthlyIncome	YearsSinceLastPromotion	YearsAtCon
0	Healthcare Representative	No	3.155738	0.844262	2.737705	7453.557377	2.885246	8.1
1	Healthcare Representative	Yes	3.111111	0.666667	2.666667	8548.222222	4.111111	10.8
2	Human Resources	No	3.150000	0.725000	2.775000	4391.750000	1.400000	5.6
3	Human Resources	Yes	3.083333	0.833333	2.500000	3715.750000	0.833333	4.1
4	Laboratory Technician	No	3.147208	0.913706	2.746193	3337.223350	1.548223	5.6
5	Laboratory Technician	Yes	3.209677	0.516129	2.532258	2919.258065	1.016129	3.1
6	Manager	No	3.206186	0.752577	2.804124	17201.484536	4.835052	14.3
7	Manager	Yes	3.000000	0.600000	2.200000	16797.400000	4.800000	15.6
9	Manufacturing Director	Yes	3.000000	0.800000	2.600000	7365.500000	1.700000	8.7
8	Manufacturing Director	No	3.200000	0.814815	2.688889	7289.925926	2.148148	7.5
10	Research Director	No	3.102564	0.871795	2.769231	15947.346154	2.910256	10.5
11	Research Director	Yes	3.000000	0.000000	3.000000	19395.500000	14.000000	26.5
12	Research Scientist	No	3.151020	0.836735	2.853061	3328.122449	1.440816	5.2
13	Research Scientist	Yes	3.255319	0.446809	2.510638	2780.468085	1.851064	4.3
14	Sales Executive	No	3.130112	0.881041	2.754647	6804.617100	2.360595	7.6
15	Sales Executive	Yes	3.105263	0.526316	2.526316	7489.000000	3.070175	6.7
16	Sales Representative	No	3.160000	0.740000	2.780000	2798.440000	1.360000	3.4
17	Sales Representative	Yes	3.121212	0.454545	2.454545	2364.727273	0.606061	2.0

In [57]:

```
enpowermentsD =df_hrs.groupby(['Department','Attrition'], as_index=False)[['PerformanceRating','JobInvolvement','
MonthlyIncome','YearsSinceLastPromotion','YearsAtCompany']].mean().sort_values(by=['Department'])
enpowermentsD.style.apply(highlight_max).apply(highlight_min)
```

Out[57]:

	Department	Attrition	PerformanceRating	JobInvolvement	MonthlyIncome	YearsSinceLastPromotion	YearsAtCompany
0	Human Resources	No	3.156863	2.803922	7345.980392	2.000000	7.960784
1	Human Resources	Yes	3.083333	2.500000	3715.750000	0.833333	4.166667
2	Research & Development	No	3.157005	2.771739	6630.326087	2.179952	7.171498
3	Research & Development	Yes	3.195489	2.556391	4108.075188	1.872180	4.954887
4	Sales	No	3.144068	2.762712	7232.240113	2.395480	7.745763
5	Sales	Yes	3.108696	2.467391	5908.456522	2.195652	5.510870

In [58]:

```
enpowermentsG =df_hrs.groupby(['Gender','Attrition'], as_index=False)[['PerformanceRating','JobInvolvement','Mont
hlyIncome','YearsSinceLastPromotion','YearsAtCompany']].mean().sort_values(by=['Gender'])
enpowermentsG.style.apply(highlight_max).apply(highlight_min)
```

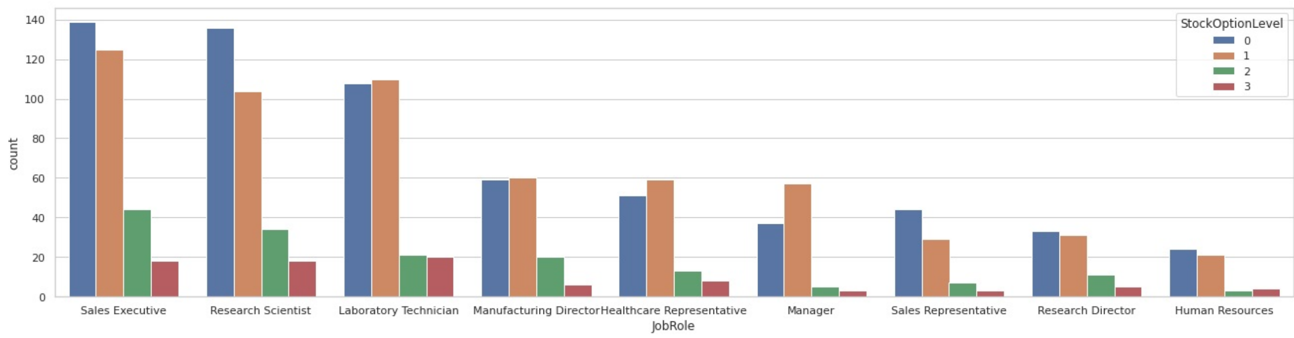
Out[58]:

	Gender	Attrition	PerformanceRating	JobInvolvement	MonthlyIncome	YearsSinceLastPromotion	YearsAtCompany
0	Female	No	3.157685	2.746507	7019.429142	2.339321	7.459082
1	Female	Yes	3.172414	2.528736	4769.735632	2.034483	5.919540
2	Male	No	3.150273	2.786885	6704.964481	2.162568	7.307377
3	Male	Yes	3.146667	2.513333	4797.160000	1.893333	4.673333

In [59]:

```
plt.subplots(figsize=(20,5))
sns.countplot(df_employee.JobRole, hue=df_employee.StockOptionLevel)
print(df_employee['JobRole'].value_counts(['StockOptionLevel']))
print(df_employee.groupby('StockOptionLevel')['JobRole'].value_counts().sort_index(ascending=True))
```

```
Sales Executive              0.221769
Research Scientist           0.198639
Laboratory Technician        0.176190
Manufacturing Director       0.098639
Healthcare Representative    0.089116
Manager                      0.069388
Sales Representative         0.056463
Research Director            0.054422
Human Resources              0.035374
Name: JobRole, dtype: float64
StockOptionLevel  JobRole
0                 Healthcare Representative     51
                  Human Resources               24
                  Laboratory Technician        108
                  Manager                       37
                  Manufacturing Director        59
                  Research Director             33
                  Research Scientist           136
                  Sales Executive              139
                  Sales Representative          44
1                 Healthcare Representative     59
                  Human Resources               21
                  Laboratory Technician        110
                  Manager                       57
                  Manufacturing Director        60
                  Research Director             31
                  Research Scientist           104
                  Sales Executive              125
                  Sales Representative          29
2                 Healthcare Representative     13
                  Human Resources                3
                  Laboratory Technician         21
                  Manager                        5
                  Manufacturing Director        20
                  Research Director             11
                  Research Scientist            34
                  Sales Executive               44
                  Sales Representative           7
3                 Healthcare Representative      8
                  Human Resources                4
                  Laboratory Technician         20
                  Manager                        3
                  Manufacturing Director         6
                  Research Director              5
                  Research Scientist            18
                  Sales Executive               18
                  Sales Representative           3
Name: JobRole, dtype: int64
```



In [60]:

```
pd.pivot_table(df_employee, values = 'StockOptionLevel', index='Attrition', columns = 'JobRole').reset_index()
```

Out[60]:

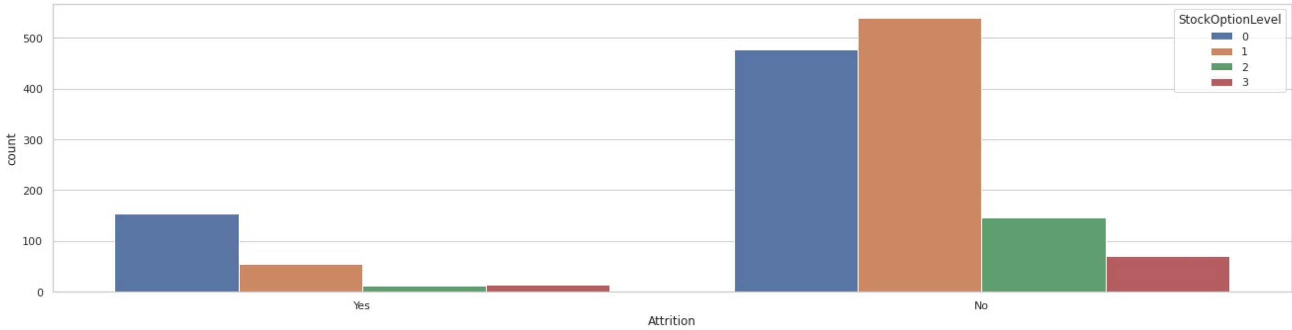
JobRole	Attrition	Healthcare Representative	Human Resources	Laboratory Technician	Manager	Manufacturing Director	Research Director	Research Scientist	Sales Executive	Sales Representative
0	No	0.844262	0.725000	0.913706	0.752577	0.814815	0.871795	0.836735	0.881041	0.740000
1	Yes	0.666667	0.833333	0.516129	0.600000	0.800000	0.000000	0.446809	0.526316	0.454545

In [61]:

```
plt.subplots(figsize=(20,5))
sns.countplot(df_employee.Attrition, hue=df_employee.StockOptionLevel)
df_employee.groupby('Attrition',)['StockOptionLevel'].value_counts(normalize=True).sort_index()
```

Out[61]:

```
Attrition  StockOptionLevel
No         0                   0.386861
           1                   0.437956
           2                   0.118410
           3                   0.056772
Yes        0                   0.649789
           1                   0.236287
           2                   0.050633
           3                   0.063291
Name: StockOptionLevel, dtype: float64
```



Have you reached Stock OptionLevel 2 then the chance for attirion is much lower

In [62]:

```
df_hrs[df_hrs.JobLevel ==3].groupby('Education', as_index=False)[['JobSatisfaction']].mean().sort_values(by=['Job
Satisfaction'])
```

Out[62]:

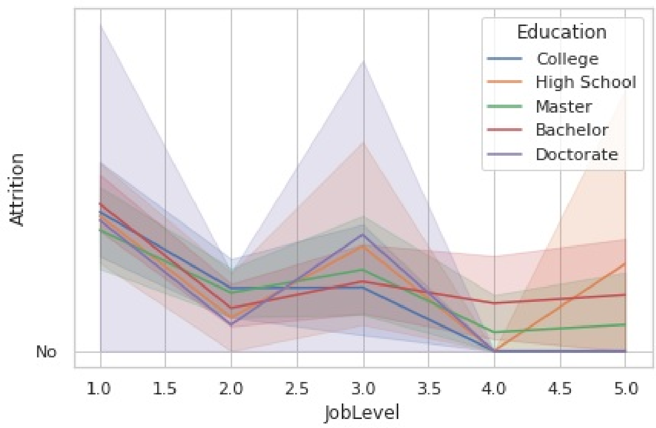
	Education	JobSatisfaction
2	Doctorate	2.333333
4	Master	2.655172
0	Bachelor	2.663265
1	College	2.787879
3	High School	2.800000

In [63]:

```
#need to change the value to the column to get a better understanding of what the graph says
#df_employee.Education.replace({1: 'High School', 2:'Undergrad', 3:'Graduate', 4:'Post Graduate', 5:'Doctorate'},
inplace=True)
sns.lineplot(x = 'JobLevel', y = 'Attrition', data=df_employee, hue='Education')
```

Out[63]:

<AxesSubplot:xlabel='JobLevel', ylabel='Attrition'>



In [64]:

```
df_employee.groupby('StockOptionLevel')['Age'].value_counts(bins=6).sort_index()
```

Out[64]:

```
StockOptionLevel
0                 (17.956999999999997, 25.0]     74
                  (25.0, 32.0]                  169
                  (32.0, 39.0]                  187
                  (39.0, 46.0]                  102
                  (46.0, 53.0]                   66
                  (53.0, 60.0]                   33
1                 (21.961, 28.333]               84
                  (28.333, 34.667]              145
                  (34.667, 41.0]                169
                  (41.0, 47.333]                101
                  (47.333, 53.667]               53
                  (53.667, 60.0]                 44
2                 (21.962999999999997, 28.0]     25
                  (28.0, 34.0]                   47
                  (34.0, 40.0]                   44
                  (40.0, 46.0]                   21
                  (46.0, 52.0]                   15
                  (52.0, 58.0]                    6
3                 (23.964, 29.833]               20
                  (29.833, 35.667]               23
                  (35.667, 41.5]                 20
                  (41.5, 47.333]                 11
                  (47.333, 53.167]                6
                  (53.167, 59.0]                  5
Name: Age, dtype: int64
```

Looking at which age tend to have different joblevels is really clear that young people start in level 1 most, of them is in this level for 10 years. If you show some encouagement you will get to an other level faster. most of the people for level 3 are in their late 30s. from 40 years old, both level 4 and 5 are more dominant.

From the two graph above we can see that the youngest people working are single and are morlelylike to leave the company. there are also more people around 30 years old leave but the their relation ship tend to be married, but people then are seeking a new direction or a new job after working a few years at one place.

In [65]:

```
# Need to Have Numeric Values for Attrition
```

In [66]:

```
happy_job = df_Anumber.groupby('JobRole', as_index=False)[['JobSatisfaction','EnvironmentSatisfaction','JobInvolv
ement']].mean().sort_values(by=['JobInvolvement'])
happy_job.style.apply(highlight_max).apply(highlight_min)
```

Out[66]:

	JobRole	JobSatisfaction	EnvironmentSatisfaction	JobInvolvement
8	Sales Representative	2.734940	2.734940	2.650602
4	Manufacturing Director	2.682759	2.917241	2.682759
2	Laboratory Technician	2.691120	2.718147	2.694981
1	Human Resources	2.557692	2.596154	2.711538
7	Sales Executive	2.754601	2.671779	2.714724
0	Healthcare Representative	2.786260	2.770992	2.732824
3	Manager	2.705882	2.764706	2.774510
5	Research Director	2.700000	2.500000	2.775000
6	Research Scientist	2.773973	2.726027	2.797945

In [67]:

```
df_Anumber['OverTime'] = df_Anumber['OverTime'].map({'Yes': 1, 'No': 0})
```

In [68]:

```
df_employee.groupby('JobRole', as_index=False)[['Age']].mean().sort_values(by=['Age'])
```

Out[68]:

	JobRole	Age
8	Sales Representative	30.361446
2	Laboratory Technician	34.096525
6	Research Scientist	34.236301
1	Human Resources	35.500000
7	Sales Executive	36.889571
4	Manufacturing Director	38.296552
0	Healthcare Representative	39.809160
5	Research Director	44.000000
3	Manager	46.764706

In [69]:

```
df_Anumber[df_Anumber.JobLevel ==3].groupby('Education', as_index=False)[['JobSatisfaction']].mean().sort_values(
by=['JobSatisfaction'])
```

Out[69]:

	Education	JobSatisfaction
4	5	2.333333
3	4	2.655172
2	3	2.663265
1	2	2.787879
0	1	2.800000

In [70]:

```
df_employee['Attrition'] = df_employee['Attrition'].map({'Yes': 1, 'No': 0})
df_Anumber.Education.replace({'High School':1, 'Collage':2,'Bachelor':3, 'Master':4, 'Doctorate':5},inplace=True)
```

In [71]:

```
role_income = df_employee.groupby('JobRole', as_index=False)[['MonthlyIncome', 'Attrition']].mean().sort_values(
    by=['MonthlyIncome'])
role_income.style.apply(highlight_max).apply(highlight_min)
```

Out[71]:

	JobRole	MonthlyIncome	Attrition
8	Sales Representative	2626.000000	0.397590
2	Laboratory Technician	3237.169884	0.239382
6	Research Scientist	3239.972603	0.160959
1	Human Resources	4235.750000	0.230769
7	Sales Executive	6924.279141	0.174847
4	Manufacturing Director	7295.137931	0.068966
0	Healthcare Representative	7528.763359	0.068702
5	Research Director	16033.550000	0.025000
3	Manager	17181.676471	0.049020

People eith doctor degree on level 3 have lower overall job satifacation than other education levels

In [72]:

```
df_Anumber[df_Anumber.JobLevel ==4].groupby('Education', as_index=False)[['YearsSinceLastPromotion','TrainingTime
sLastYear','JobSatisfaction']].mean().sort_values(by=['YearsSinceLastPromotion'])
```

Out[72]:

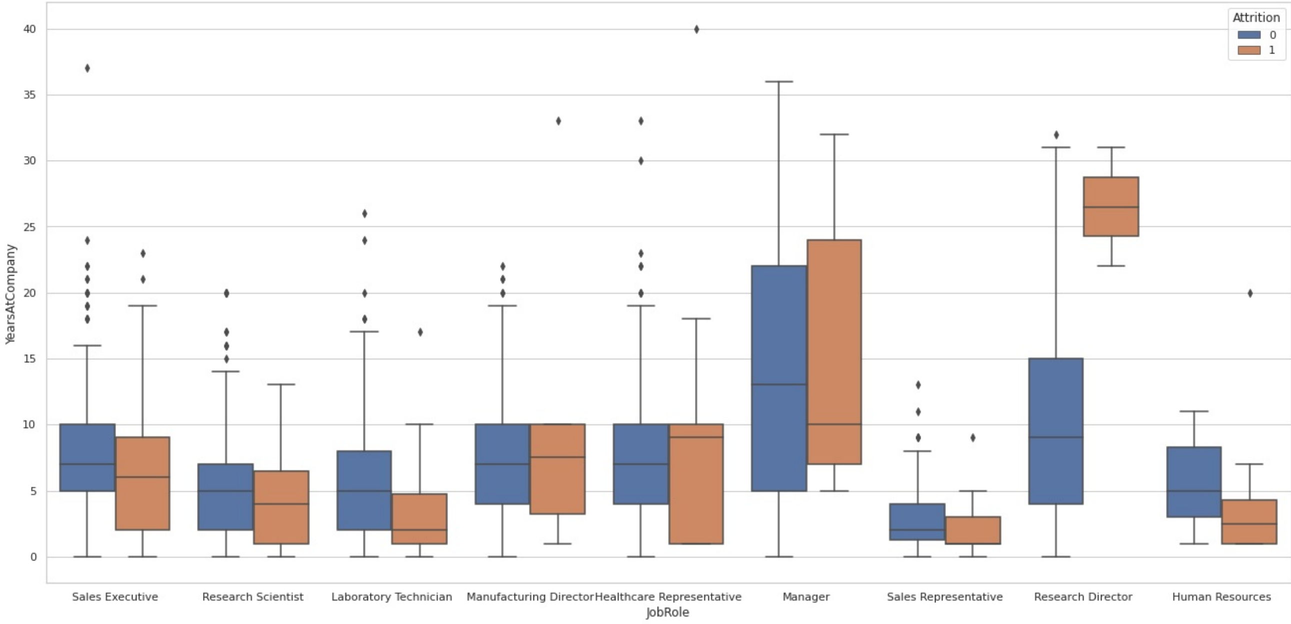
	Education	YearsSinceLastPromotion	TrainingTimesLastYear	JobSatisfaction
3	4	3.821429	2.714286	2.571429
1	2	4.823529	2.352941	3.176471
2	3	4.863636	2.431818	2.636364
4	5	5.555556	2.777778	2.888889
0	1	7.500000	2.875000	2.625000

In [73]:

```
plt.figure(figsize=(20,10))
sns.boxplot(data=df_employee, x='JobRole', y='YearsAtCompany',hue='Attrition')
```

Out[73]:

<AxesSubplot:xlabel='JobRole', ylabel='YearsAtCompany'>

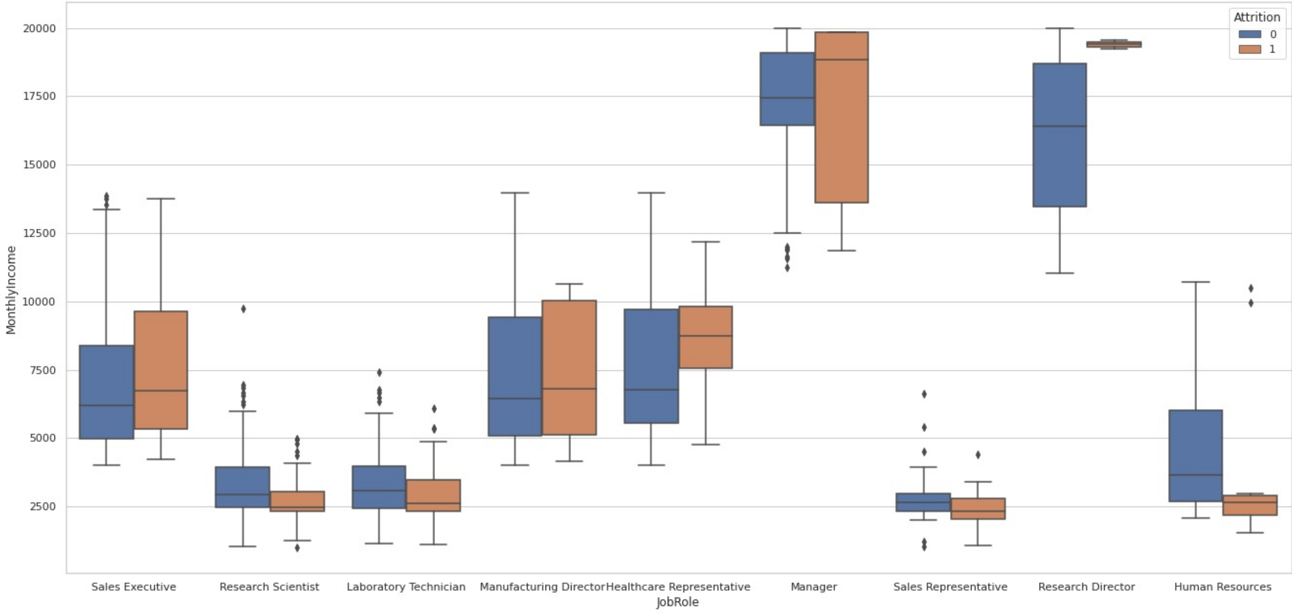


In [74]:

```
plt.figure(figsize=(20,10))
sns.boxplot(data=df_employee, x='JobRole', y='MonthlyIncome',hue='Attrition')
```

Out[74]:

```
<AxesSubplot:xlabel='JobRole', ylabel='MonthlyIncome'>
```



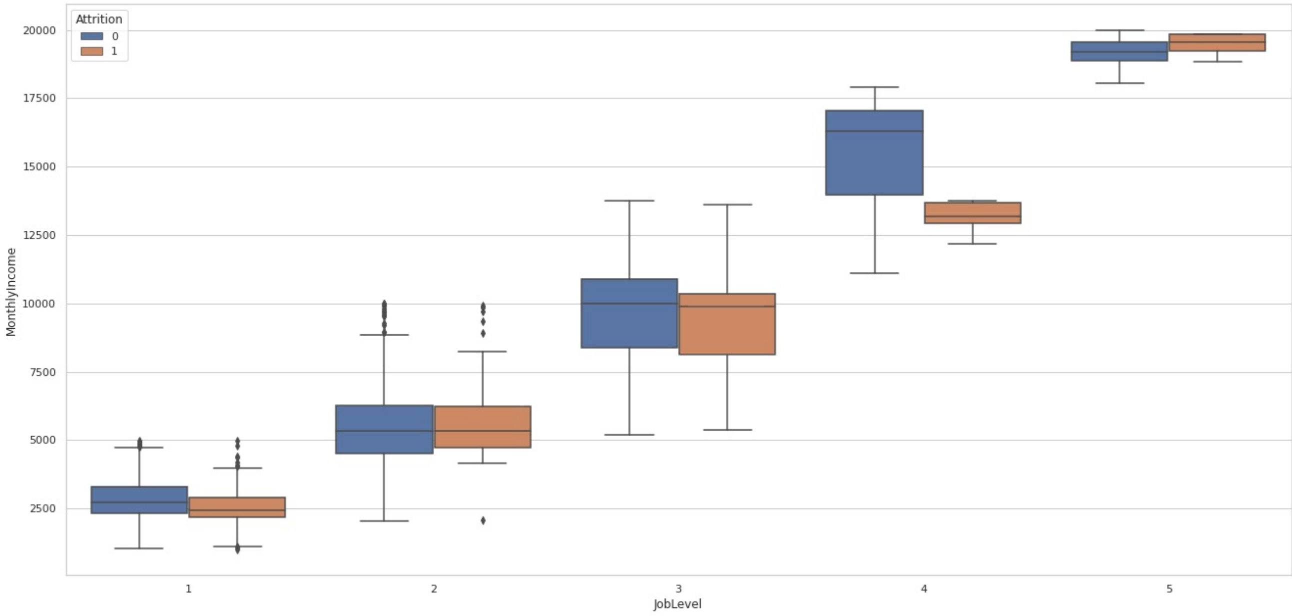
It takes time for people with doctor degree to reach level 4 among people with higher education

In [75]:

```
plt.figure(figsize=(20,10))
sns.boxplot(data=df_employee, x='JobLevel', y='MonthlyIncome',hue='Attrition')
```

Out[75]:

```
<AxesSubplot:xlabel='JobLevel', ylabel='MonthlyIncome'>
```



People in level 4 that leave has much less income than those who stay

3.7 Surprisng/Unique Findings

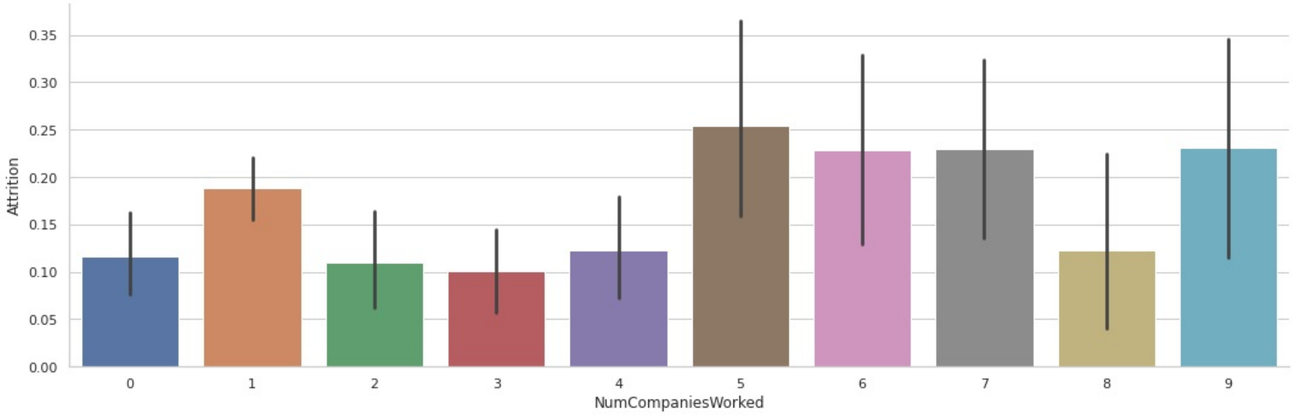
These plots need Attrition as a numeric feature.

In [76]:

```
sns.catplot(x = 'NumCompaniesWorked', y = 'Attrition', data=df_employee, aspect= 3, kind = 'bar')
```

Out[76]:

```
<seaborn.axisgrid.FacetGrid at 0x7f1918b51310>
```



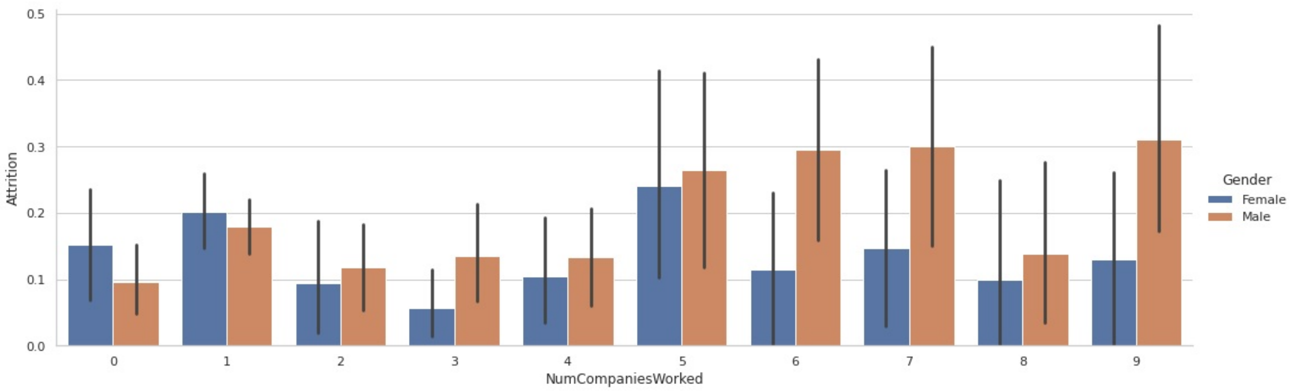
Have you worked for 2-4 companiesyou are less likely to leave.

In [77]:

```
sns.factorplot(x = 'NumCompaniesWorked', y = 'Attrition', hue = 'Gender', data=df_employee, aspect= 3, kind = 'ba
r')
```

Out[77]:

```
<seaborn.axisgrid.FacetGrid at 0x7f1918bfa610>
```



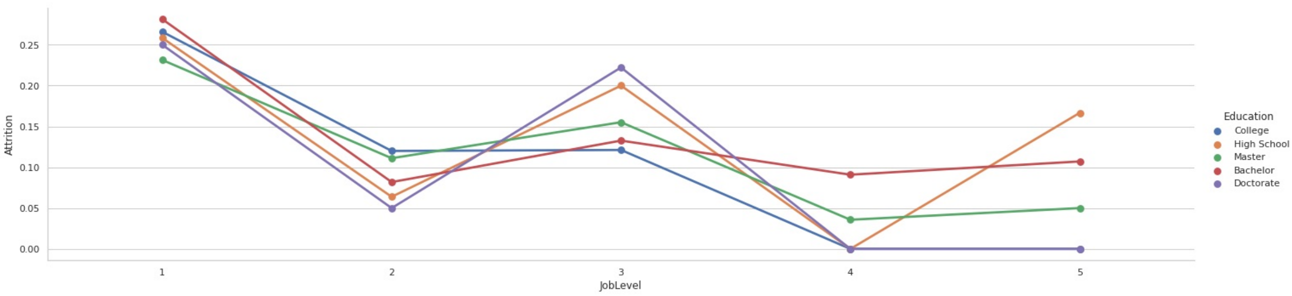
Splitting on gender, we can clearly see that the attrtion rate stays up for male working for many companies, but woman are lower

In [78]:

```
sns.factorplot(x = 'JobLevel', y = 'Attrition', hue = 'Education', data=df_employee, aspect= 4, ci=None)
```

Out[78]:

```
<seaborn.axisgrid.FacetGrid at 0x7f1914b60550>
```



A high education at job level 3 is increasing the attrition rate.

In [79]:

```
fig, ax1 = plt.subplots(figsize=(20,6))

sns.lineplot(data = role_income, x='JobRole',y='Attrition', sort = False, ax=ax1)
ax2 = ax1.twinx()

sns.barplot(data = role_income, x='JobRole', y='MonthlyIncome', alpha=0.7, ax=ax2)
```

Out[79]:

```
<AxesSubplot:xlabel='JobRole', ylabel='MonthlyIncome'>
```

Sales have hghest attrition and lower income, attrition increase for HR, due to higher income and good jobsatifaction, least attrition for those with most income

From the EDA we obtained some interesting findings.

From the EDA we can some interesting things

1. SAles representatives tend to be promoted pretty fast, but most of them have low income and are more likely to quite due to to their relative high job satisfation. and worklife balance, but their education siuation says they are undergradeuated so that why they leave.
2. Many doctors are left on the level 3, for them to reach 4 it take a really long time compared to other people with higher education
3. have you worked for some companies, you are less likely to leave

---

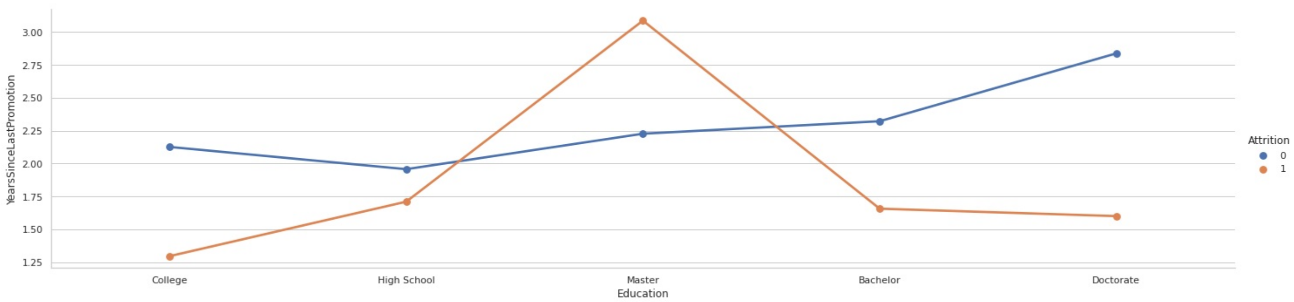
END Part 1

In [80]:

```
sns.factorplot(x = 'Education', y = 'YearsSinceLastPromotion', hue = 'Attrition', data=df_employee, aspect= 4, ci
=None)
```

Out[80]:

```
<seaborn.axisgrid.FacetGrid at 0x7f19148f8d90>
```

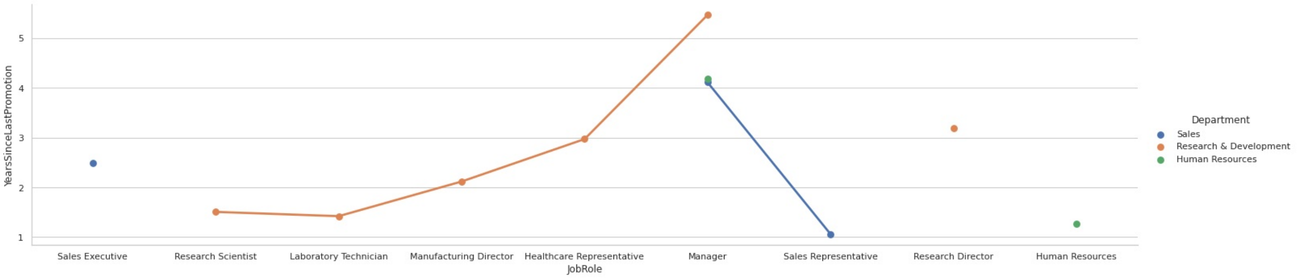


In [81]:

```
sns.factorplot(x = 'JobRole', y = 'YearsSinceLastPromotion', hue = 'Department', data=df_employee, aspect= 4, ci=None)
```

Out[81]:

<seaborn.axisgrid.FacetGrid at 0x7f1914982c10>

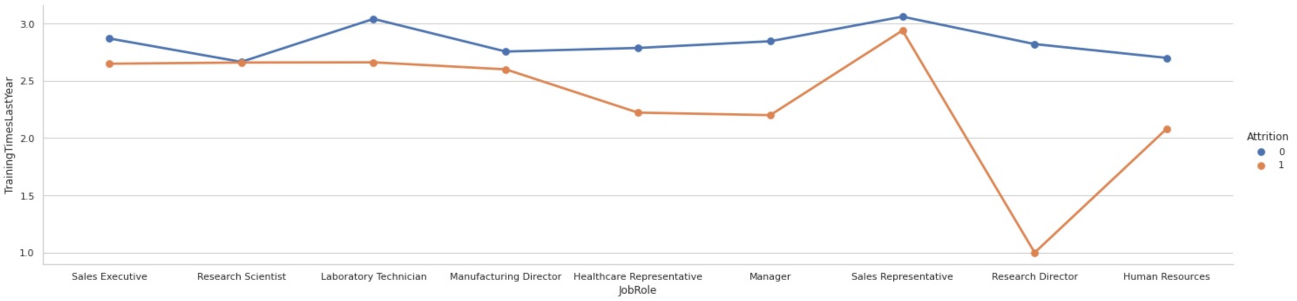


In [82]:

```
sns.factorplot(x = 'JobRole', y = 'TrainingTimesLastYear', hue = 'Attrition', data=df_employee, aspect= 4, ci=None)
```

Out[82]:

<seaborn.axisgrid.FacetGrid at 0x7f191480c350>

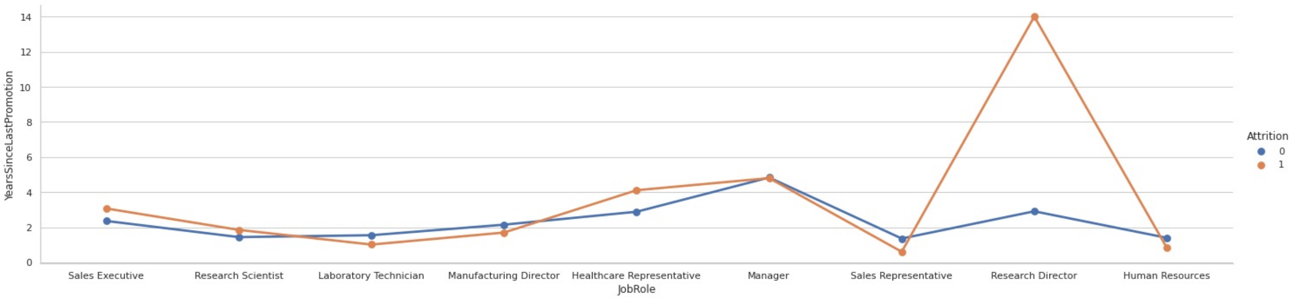


In [83]:

```
sns.factorplot(x = 'JobRole', y = 'YearsSinceLastPromotion', hue = 'Attrition', data=df_employee, aspect= 4, ci=None)
```

Out[83]:

<seaborn.axisgrid.FacetGrid at 0x7f1914756f50>



In [83]:

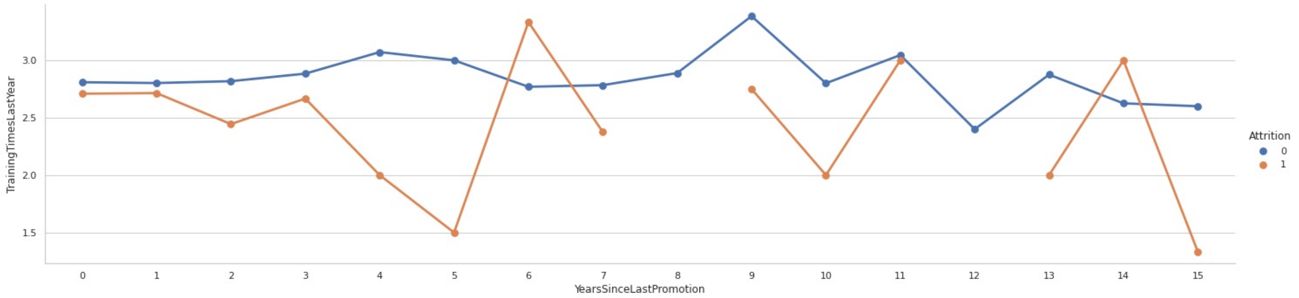
```

```

In [84]:

```
sns.factorplot(x = 'YearsSinceLastPromotion', y = 'TrainingTimesLastYear', hue = 'Attrition', data=df_employee, a
spect= 4, ci=None)
```

Out[84]:

<seaborn.axisgrid.FacetGrid at 0x7f19147efa10>

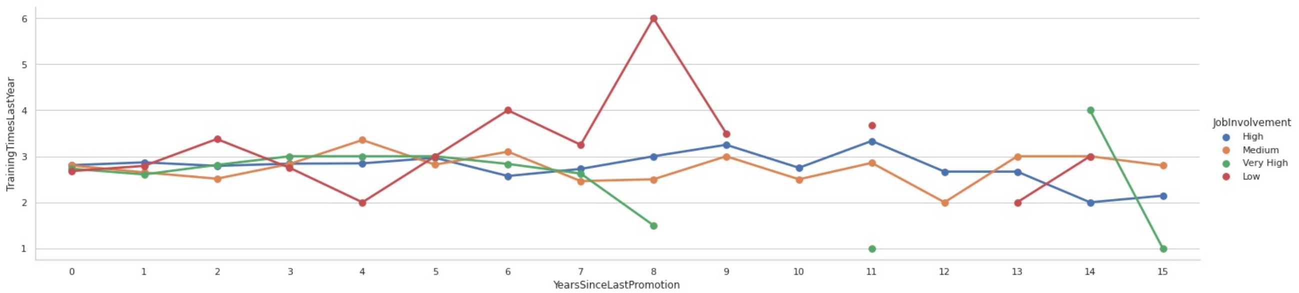


In [85]:

```
sns.factorplot(x = 'YearsSinceLastPromotion', y = 'TrainingTimesLastYear', hue = 'JobInvolvement', data=df_employ
ee, aspect= 4, ci=None)
```

Out[85]:

<seaborn.axisgrid.FacetGrid at 0x7f1914663450>

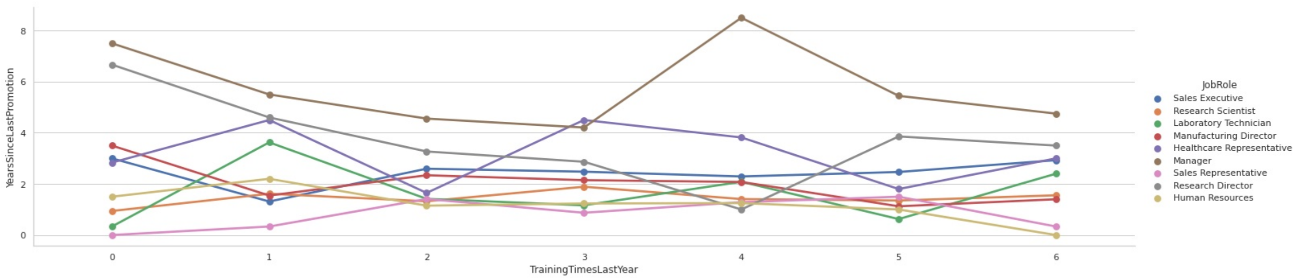


In [86]:

```
sns.factorplot(x = 'TrainingTimesLastYear', y = 'YearsSinceLastPromotion', hue = 'JobRole', data=df_employee, asp
ect= 4, ci=None)
#sns.factorplot(x = 'TrainingTimesLastYear', y = 'YearsSinceLastPromotion', hue = 'Education', data=df_employee,
aspect= 4, ci=None)
```

Out[86]:

<seaborn.axisgrid.FacetGrid at 0x7f191450df10>

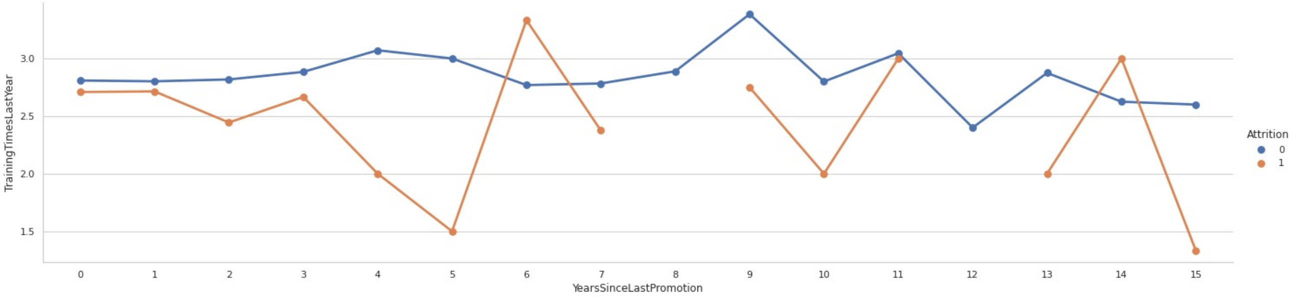


In [87]:

```
sns.factorplot(x = 'YearsSinceLastPromotion', y = 'TrainingTimesLastYear', hue = 'Attrition', data=df_Anumber, as
pect= 4, ci=None)
```

Out[87]:

<seaborn.axisgrid.FacetGrid at 0x7f191460f950>



In [88]:

```
sns.factorplot(x = 'TrainingTimesLastYear', y = 'JobSatisfaction', hue = 'JobRole', data=df_Anumber, aspect= 4, c
i=None)
```

Out[88]:

<seaborn.axisgrid.FacetGrid at 0x7f1914449c90>

M4 Part 2

In [ ]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

import seaborn as sns

from IPython.display import display
from scipy import stats

import warnings
%matplotlib inline
np.random.seed(42)
warnings.filterwarnings('always')
warnings.filterwarnings("ignore")

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import auc, roc_curve, roc_auc_score, classification_report, confusion_matrix, precision_rec
all_fscore_support

! pip install xgboost -qq
#Baseline algoritms
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from xgboost import XGBClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.svm import LinearSVC
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import KFold
from sklearn import feature_selection
from sklearn import model_selection
from sklearn import metrics

from sklearn.neural_network import MLPClassifier
from sklearn.metrics import precision_score, recall_score

#Lifeline survival imports
!pip install lifelines -qq
from lifelines import KaplanMeierFitter
```

In [ ]:

```
hr_df = pd.read_csv('/work/WA_Fn-UseC_-HR-Employee-Attrition.csv')
```

In [ ]:

```
#do not need this
hr_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 35 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   Age                       1470 non-null   int64 
 1   Attrition                 1470 non-null   object
 2   BusinessTravel            1470 non-null   object
 3   DailyRate                 1470 non-null   int64 
 4   Department                1470 non-null   object
 5   DistanceFromHome          1470 non-null   int64 
 6   Education                 1470 non-null   int64 
 7   EducationField            1470 non-null   object
 8   EmployeeCount             1470 non-null   int64 
 9   EmployeeNumber            1470 non-null   int64 
 10  EnvironmentSatisfaction   1470 non-null   int64 
 11  Gender                    1470 non-null   object
 12  HourlyRate                1470 non-null   int64 
 13  JobInvolvement            1470 non-null   int64 
 14  JobLevel                  1470 non-null   int64 
 15  JobRole                   1470 non-null   object
 16  JobSatisfaction           1470 non-null   int64 
 17  MaritalStatus             1470 non-null   object
 18  MonthlyIncome             1470 non-null   int64 
 19  MonthlyRate               1470 non-null   int64 
 20  NumCompaniesWorked        1470 non-null   int64 
 21  Over18                    1470 non-null   object
 22  OverTime                  1470 non-null   object
 23  PercentSalaryHike         1470 non-null   int64 
 24  PerformanceRating         1470 non-null   int64 
 25  RelationshipSatisfaction  1470 non-null   int64 
 26  StandardHours             1470 non-null   int64 
 27  StockOptionLevel          1470 non-null   int64 
 28  TotalWorkingYears         1470 non-null   int64 
 29  TrainingTimesLastYear     1470 non-null   int64 
 30  WorkLifeBalance           1470 non-null   int64 
 31  YearsAtCompany            1470 non-null   int64 
 32  YearsInCurrentRole        1470 non-null   int64 
 33  YearsSinceLastPromotion   1470 non-null   int64 
 34  YearsWithCurrManager      1470 non-null   int64 
dtypes: int64(26), object(9)
memory usage: 402.1+ KB
```

In [ ]:

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

In [ ]:

```
hr_df['Attrition'] = hr_df['Attrition'].map({'Yes': 1, 'No': 0})
```

In [ ]:

```
hr_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 31 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   Age                       1470 non-null   int64 
 1   Attrition                 1470 non-null   int64 
 2   BusinessTravel            1470 non-null   object
 3   DailyRate                 1470 non-null   int64 
 4   Department                1470 non-null   object
 5   DistanceFromHome          1470 non-null   int64 
 6   Education                 1470 non-null   int64 
 7   EducationField            1470 non-null   object
 8   EnvironmentSatisfaction   1470 non-null   int64 
 9   Gender                    1470 non-null   object
 10  HourlyRate                1470 non-null   int64 
 11  JobInvolvement            1470 non-null   int64 
 12  JobLevel                  1470 non-null   int64 
 13  JobRole                   1470 non-null   object
 14  JobSatisfaction           1470 non-null   int64 
 15  MaritalStatus             1470 non-null   object
 16  MonthlyIncome             1470 non-null   int64 
 17  MonthlyRate               1470 non-null   int64 
 18  NumCompaniesWorked        1470 non-null   int64 
 19  OverTime                  1470 non-null   object
 20  PercentSalaryHike         1470 non-null   int64 
 21  PerformanceRating         1470 non-null   int64 
 22  RelationshipSatisfaction  1470 non-null   int64 
 23  StockOptionLevel          1470 non-null   int64 
 24  TotalWorkingYears         1470 non-null   int64 
 25  TrainingTimesLastYear     1470 non-null   int64 
 26  WorkLifeBalance           1470 non-null   int64 
 27  YearsAtCompany            1470 non-null   int64 
 28  YearsInCurrentRole        1470 non-null   int64 
 29  YearsSinceLastPromotion   1470 non-null   int64 
 30  YearsWithCurrManager      1470 non-null   int64 
dtypes: int64(24), object(7)
memory usage: 356.1+ KB
```

In [ ]:

```
hr_survi = hr_df.copy()
```

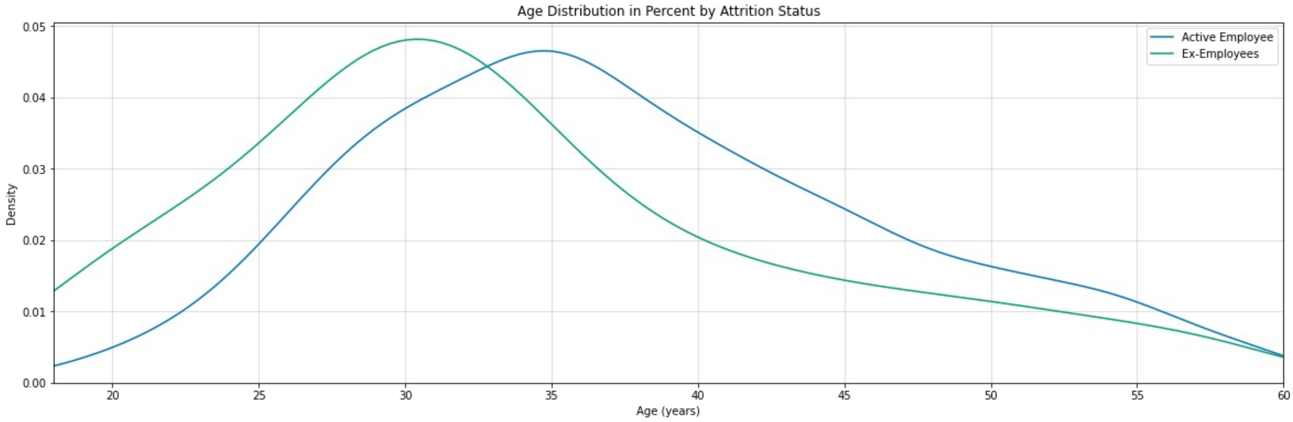
4. KDE plots between Active and Ex-employees

By creating some Kernal Density Estimation plot, we can color it by value of the target. To get a better understanding of our data

4.0.1 Age

In [ ]:

```
plt.figure(figsize=(20,6))
plt.style.use('seaborn-colorblind')
plt.grid(True, alpha=0.5)
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 0, 'Age'], label = 'Active Employee')
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 1, 'Age'], label = 'Ex-Employees')
plt.xlim(left=18, right=60)
plt.xlabel('Age (years)')
plt.ylabel('Density')
plt.title('Age Distribution in Percent by Attrition Status')
plt.legend();
```



Lower avrage age on people who left vs people who stayed

4.0.2 Distance from home

In [ ]:

```
# Distance from Home
print("Distance from home for employees to get to work is from {} to {} miles.".format(hr_df['DistanceFromHome'].
min(),
                                                                                          hr_df['DistanceFromHome'].
max()))
```

```
Distance from home for employees to get to work is from 1 to 29 miles.
```

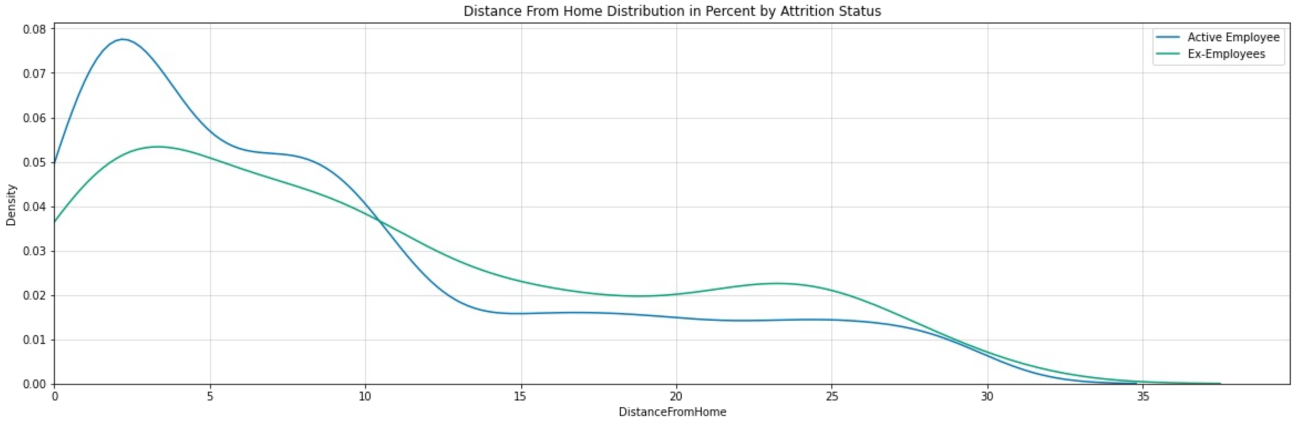
In [ ]:

```
#difference between left or stayed
print('Average distance from home for currently active employees: {:.2f} miles and ex-employees: {:.2f} miles'.fo
rmat(
    hr_df[hr_df['Attrition'] == 0]['DistanceFromHome'].mean(), hr_df[hr_df['Attrition'] == 1]['DistanceFromHome']
.mean()))
```

```
Average distance from home for currently active employees: 8.92 miles and ex-employees: 10.63 miles
```

In [ ]:

```
plt.figure(figsize=(20,6))
plt.style.use('seaborn-colorblind')
plt.grid(True, alpha=0.5)
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 0, 'DistanceFromHome'], label = 'Active Employee')
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 1, 'DistanceFromHome'], label = 'Ex-Employees')
plt.xlabel('DistanceFromHome')
plt.xlim(left=0)
plt.ylabel('Density')
plt.title('Distance From Home Distribution in Percent by Attrition Status')
plt.legend();
```



Persons that are still working at the company have more people living closer to work.

4.0.3 Years at company

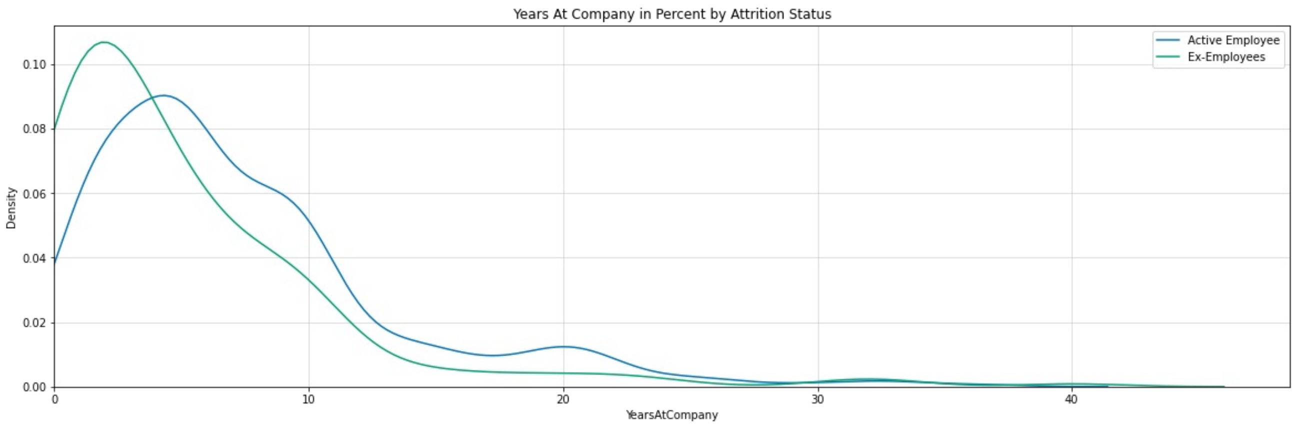
In [ ]:

```
print("Number of Years at the company varies from {} to {} years.".format(
    hr_df['YearsAtCompany'].min(), hr_df['YearsAtCompany'].max()))
```

```
Number of Years at the company varies from 0 to 40 years.
```

In [ ]:

```
plt.figure(figsize=(20,6))
plt.style.use('seaborn-colorblind')
plt.grid(True, alpha=0.5)
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 0, 'YearsAtCompany'], label = 'Active Employee')
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 1, 'YearsAtCompany'], label = 'Ex-Employees')
plt.xlabel('YearsAtCompany')
plt.xlim(left=0)
plt.ylabel('Density')
plt.title('Years At Company in Percent by Attrition Status')
plt.legend();
```



many of the people leaving tend to be there for a year or two, where its biggest density. most of the persons staying many have been there for 5 years.

there a few more people still working that have been there for 20 years than people leaving.

4.0.4 Years in current Role

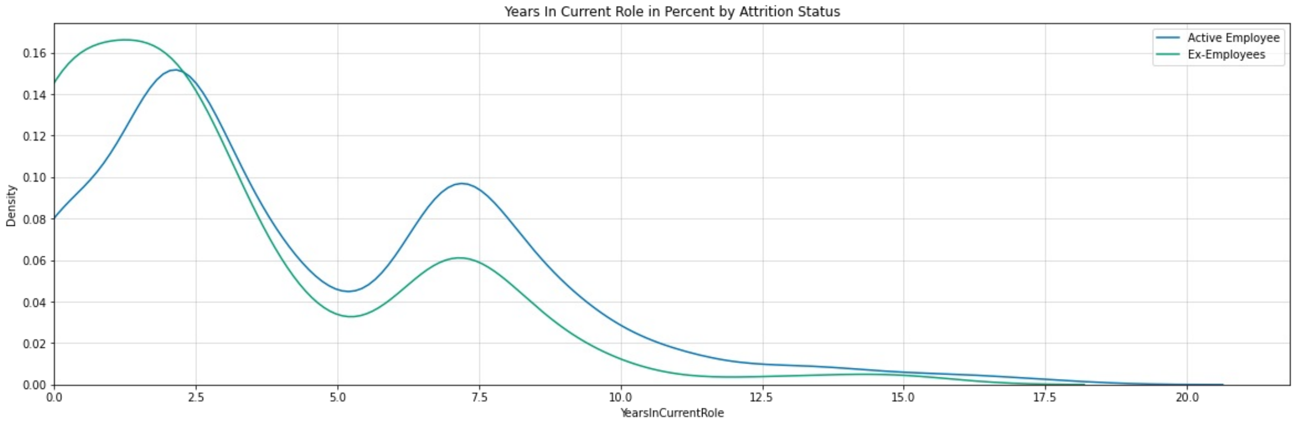
In [ ]:

```
print("Number of Years in the current role varies from {} to {} years.".format(
    hr_df['YearsInCurrentRole'].min(), hr_df['YearsInCurrentRole'].max()))
```

```
Number of Years in the current role varies from 0 to 18 years.
```

In [ ]:

```
plt.figure(figsize=(20,6))
plt.style.use('seaborn-colorblind')
plt.grid(True, alpha=0.5)
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 0, 'YearsInCurrentRole'], label = 'Active Employee')
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 1, 'YearsInCurrentRole'], label = 'Ex-Employees')
plt.xlabel('YearsInCurrentRole')
plt.xlim(left=0)
plt.ylabel('Density')
plt.title('Years In Current Role in Percent by Attrition Status')
plt.legend();
```



active employee is in there role for longer, there is also a lot of workers that have there same role for around 7 years.

4.0.5 Years since promotion

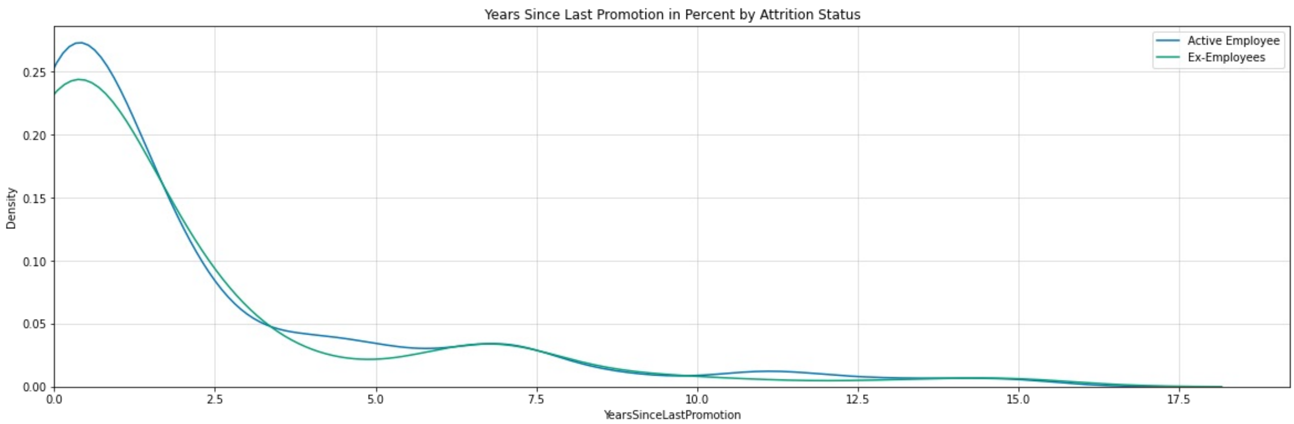
In [ ]:

```
print("Number of Years since last promotion varies from {} to {} years.".format(
    hr_df['YearsSinceLastPromotion'].min(), hr_df['YearsSinceLastPromotion'].max()))
```

```
Number of Years since last promotion varies from 0 to 15 years.
```

In [ ]:

```
plt.figure(figsize=(20,6))
plt.style.use('seaborn-colorblind')
plt.grid(True, alpha=0.5)
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 0, 'YearsSinceLastPromotion'], label = 'Active Employee')
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 1, 'YearsSinceLastPromotion'], label = 'Ex-Employees')
plt.xlabel('YearsSinceLastPromotion')
plt.xlim(left=0)
plt.ylabel('Density')
plt.title('Years Since Last Promotion in Percent by Attrition Status')
plt.legend();
```



there is almost no difference in promotion between the 2 groups. most of the people get a promotion after a year, also after 6-7 years.

4.0.6 Total working years

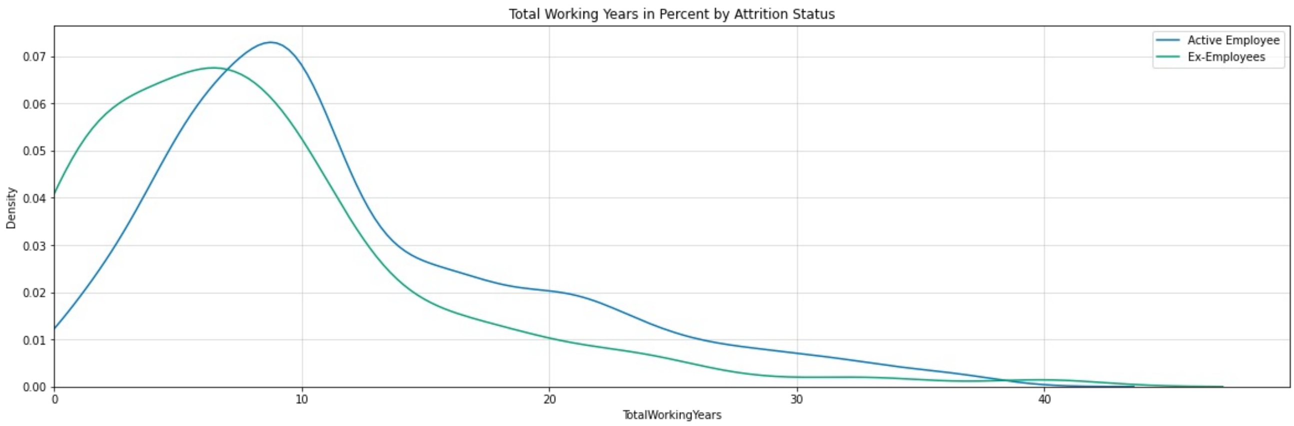
In [ ]:

```
print("Total working years varies from {} to {} years.".format(
    hr_df['TotalWorkingYears'].min(), hr_df['TotalWorkingYears'].max()))
```

```
Total working years varies from 0 to 40 years.
```

In [ ]:

```
plt.figure(figsize=(20,6))
plt.style.use('seaborn-colorblind')
plt.grid(True, alpha=0.5)
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 0, 'TotalWorkingYears'], label = 'Active Employee')
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 1, 'TotalWorkingYears'], label = 'Ex-Employees')
plt.xlabel('TotalWorkingYears')
plt.xlim(left=0)
plt.ylabel('Density')
plt.title('Total Working Years in Percent by Attrition Status')
plt.legend();
```



People that are leaving have the majority of the people working less than 10 years and few work longer 20 years.

4.0.7 Years with current manager

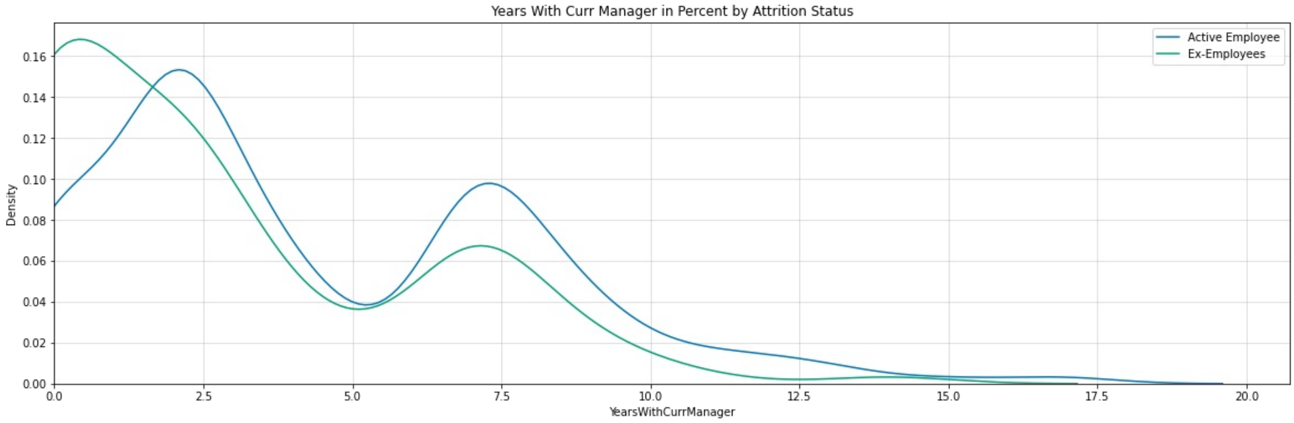
In [ ]:

```
print("Number of Years wit current manager varies from {} to {} years.".format(
    hr_df['YearsWithCurrManager'].min(), hr_df['YearsWithCurrManager'].max()))
```

```
Number of Years wit current manager varies from 0 to 17 years.
```

In [ ]:

```
plt.figure(figsize=(20,6))
plt.style.use('seaborn-colorblind')
plt.grid(True, alpha=0.5)
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 0, 'YearsWithCurrManager'], label = 'Active Employee')
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 1, 'YearsWithCurrManager'], label = 'Ex-Employees')
plt.xlabel('YearsWithCurrManager')
plt.xlim(left=0)
plt.ylabel('Density')
plt.title('Years With Curr Manager in Percent by Attrition Status')
plt.legend();
```



4.0.8 Monthly Income

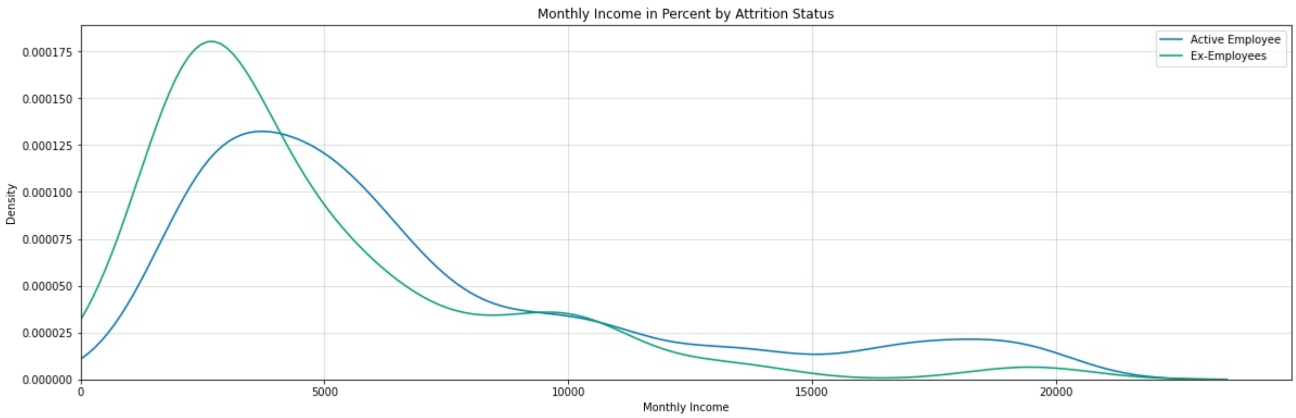
In [ ]:

```
print("Employee Monthly Income varies from ${} to ${}.".format(
    hr_df['MonthlyIncome'].min(), hr_df['MonthlyIncome'].max()))
```

```
Employee Monthly Income varies from $1009 to $19999.
```

In [ ]:

```
plt.figure(figsize=(20,6))
plt.style.use('seaborn-colorblind')
plt.grid(True, alpha=0.5)
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 0, 'MonthlyIncome'], label = 'Active Employee')
sns.kdeplot(hr_df.loc[hr_df['Attrition'] == 1, 'MonthlyIncome'], label = 'Ex-Employees')
plt.xlabel('Monthly Income')
plt.xlim(left=0)
plt.ylabel('Density')
plt.title('Monthly Income in Percent by Attrition Status')
plt.legend();
```



Many of the leavers, have a salary around 2500$

5. Machine Learning models

5.1 What to predict

We want to predict our target variable **Attrition**

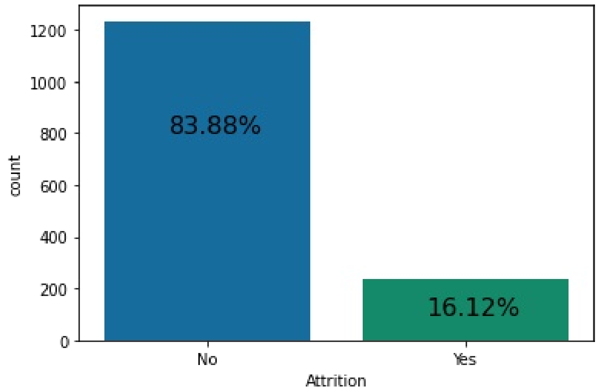
Our Hypotehsis for the most important features that influence this is *OverTime*, *MonthlyIncome*, *Age*, *MaritalStatus* and *JobSatisfaction*.
Metrics to consider : Accuracy and False Negative Rate

The False Negative Rate tells us if we predict no attrition when there is one, and this is most costly for the company. Benchmark Accuracy score: Accuracy = 83,88% (predicting every employee as leaving) Benchmark Score plotted under.

- Get dummy variables
- Feature and target Varible
- Scaling the data
- Train test split the data
- Classfication models

In [ ]:

```
sns.countplot(hr_df['Attrition'])
plt.text(x = -.15, y = 800, s = str(np.round(1233/1470.0, 4) * 100) + '%', fontsize = 16)
plt.text(x = .85, y = 100, s = str(np.round(237/1470.0, 4) * 100) + '%', fontsize = 16)
plt.xticks(np.arange(2),('No', 'Yes'))
plt.show()
hf_df['Attrition'].value_counts()
```



In [ ]:

```
print(hr_df.shape)
hr_df.head()
```

(1470, 31)

Out[ ]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EnvironmentSatisfaction	Ge
0	41	1	Travel_Rarely	1102	Sales	1	2	Life Sciences	2	Fe
1	49	0	Travel_Frequently	279	Research & Development	8	1	Life Sciences	3	
2	37	1	Travel_Rarely	1373	Research & Development	2	2	Other	4	
3	33	0	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	4	Fe
4	27	0	Travel_Rarely	591	Research & Development	2	1	Medical	1	

5 rows × 31 columns

In [ ]:

```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
# Create a label encoder object
le = LabelEncoder()
```

In [ ]:

```
# Label Encoding will be used for columns with 2 or less unique values
le_count = 0
for col in hr_df.columns[1:]:
    if hr_df[col].dtype == 'object':
        if len(list(hr_df[col].unique())) <= 2:
            le.fit(hr_df[col])
            hr_df[col] = le.transform(hr_df[col])
            le_count += 1
print('{} columns were label encoded.'.format(le_count))
```

```
2 columns were label encoded.
```

In [ ]:

```
hr_df = pd.get_dummies(hr_df, drop_first = True) #to avoid multicolinearity
```

Multicollinearity will occur when features are highly correlated with other features in the dataset. this can affect performance of regession and classfication models

In [ ]:

```
print(hr_df.shape)
```

```
(1470, 45)
```

In [ ]:

```
X = hr_df.drop('Attrition', axis = 1)
y = hr_df['Attrition']
```

In [ ]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 42)

#scale the data
scaler = StandardScaler()
# Fit_transform
X_train_scaled = scaler.fit_transform(X_train)
# transform
X_test_scaled = scaler.transform(X_test)
```

In [ ]:

```
print(X_train.shape)
print(y_train.shape)
print(X_test.shape)
print(y_test.shape)
```

```
(1102, 44)
(1102,)
(368, 44)
(368,)
```

5.2 Building Models

First just using a range of baseline algoritms using with a set of parmeters, and then fine tuning them afterwards Algoritms used

- Logistic Regression
- Random Forest Classifier
- SVM
- KNN
- Decision Tree Classifier
- Gaussian NB
- XGB
- MLPClassifier

In [ ]:

```
models = []
models.append(('Logistic Regression', LogisticRegression(solver='liblinear', random_state=42,
                                                         class_weight='balanced')))
models.append(('Random Forest', RandomForestClassifier(
    n_estimators=100, random_state=42)))
models.append(('SVM', SVC(gamma='auto', random_state=42)))
models.append(('KNN', KNeighborsClassifier()))
models.append(('Decision Tree Classifier',
               DecisionTreeClassifier(random_state=42)))
models.append(('Gaussian NB', GaussianNB()))
models.append((' Extreme Gradient Booster',XGBClassifier()))
models.append(('MLPClassifier',MLPClassifier()))
```

In [ ]:

```
acc_results = []
auc_results = []
names = []
# set table to table to populate with performance results
col = ['Algorithm', 'ROC AUC Mean', 'ROC AUC STD',
       'Accuracy Mean', 'Accuracy STD']
df_results = pd.DataFrame(columns=col)
i = 0
# evaluate each model using cross-validation
for name, model in models:
    kfold = model_selection.KFold(
        n_splits=10, random_state=42, shuffle=True)  # 10-fold cross-validation

    cv_acc_results = model_selection.cross_val_score(  # accuracy scoring
        model, X_train, y_train, cv=kfold, scoring='accuracy')

    cv_auc_results = model_selection.cross_val_score(  # roc_auc scoring
        model, X_train, y_train, cv=kfold, scoring='roc_auc')

    acc_results.append(cv_acc_results)
    auc_results.append(cv_auc_results)
    names.append(name)
    df_results.loc[i] = [name,
                         round(cv_auc_results.mean()*100, 2),
                         round(cv_auc_results.std()*100, 2),
                         round(cv_acc_results.mean()*100, 2),
                         round(cv_acc_results.std()*100, 2)
                         ]
    i += 1
df_results.sort_values(by=['ROC AUC Mean'], ascending=False)
```

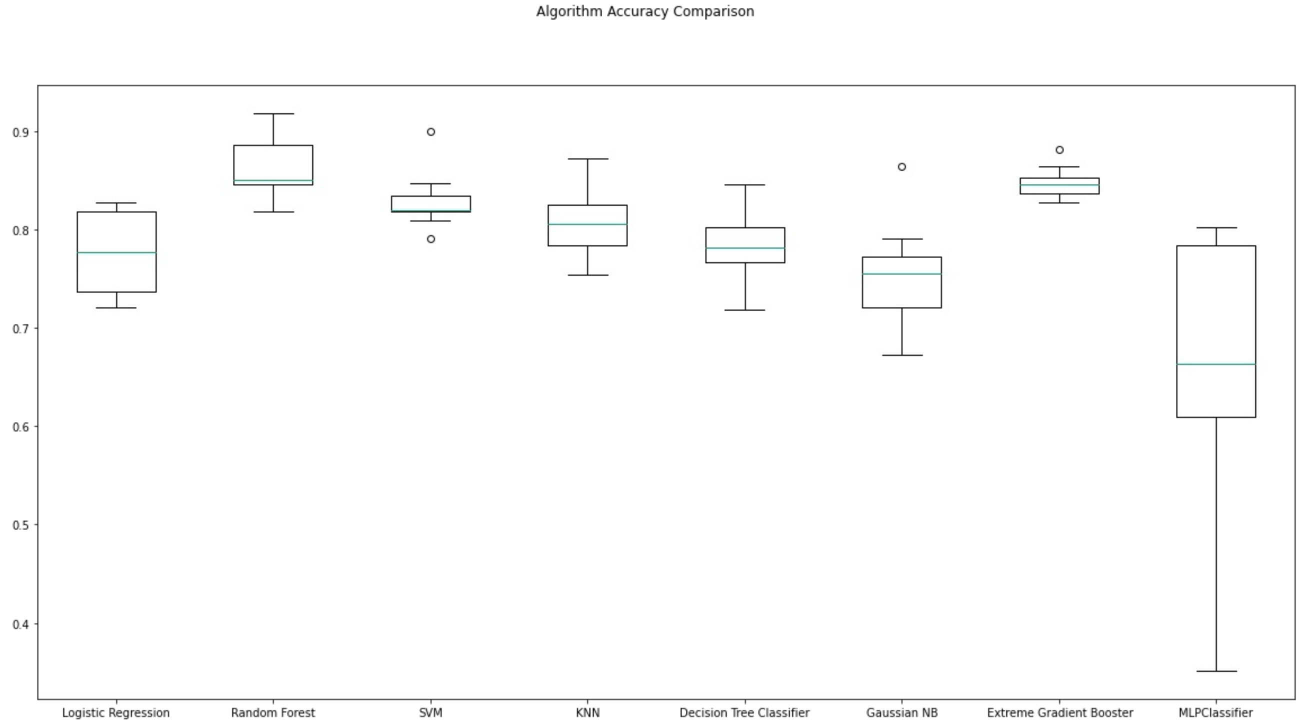
```
[23:36:51] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:36:57] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:37:01] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:37:06] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:37:11] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:37:16] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:37:20] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:37:25] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:37:30] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:37:35] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:37:40] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:37:45] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:37:50] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:37:55] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:37:59] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:38:04] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:38:09] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:38:13] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:38:18] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[23:38:21] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
```

Out[ ]:

	Algorithm	ROC AUC Mean	ROC AUC STD	Accuracy Mean	Accuracy STD
0	Logistic Regression	83.56	4.82	77.59	4.29
1	Random Forest	82.15	5.67	86.03	3.09
6	Extreme Gradient Booster	80.02	5.45	84.66	1.60
5	Gaussian NB	77.35	4.66	75.41	4.97
7	MLPClassifier	63.57	4.53	64.44	15.50
3	KNN	63.34	6.40	80.85	3.31
4	Decision Tree Classifier	62.21	3.14	78.23	3.42
2	SVM	50.00	0.00	82.85	2.77

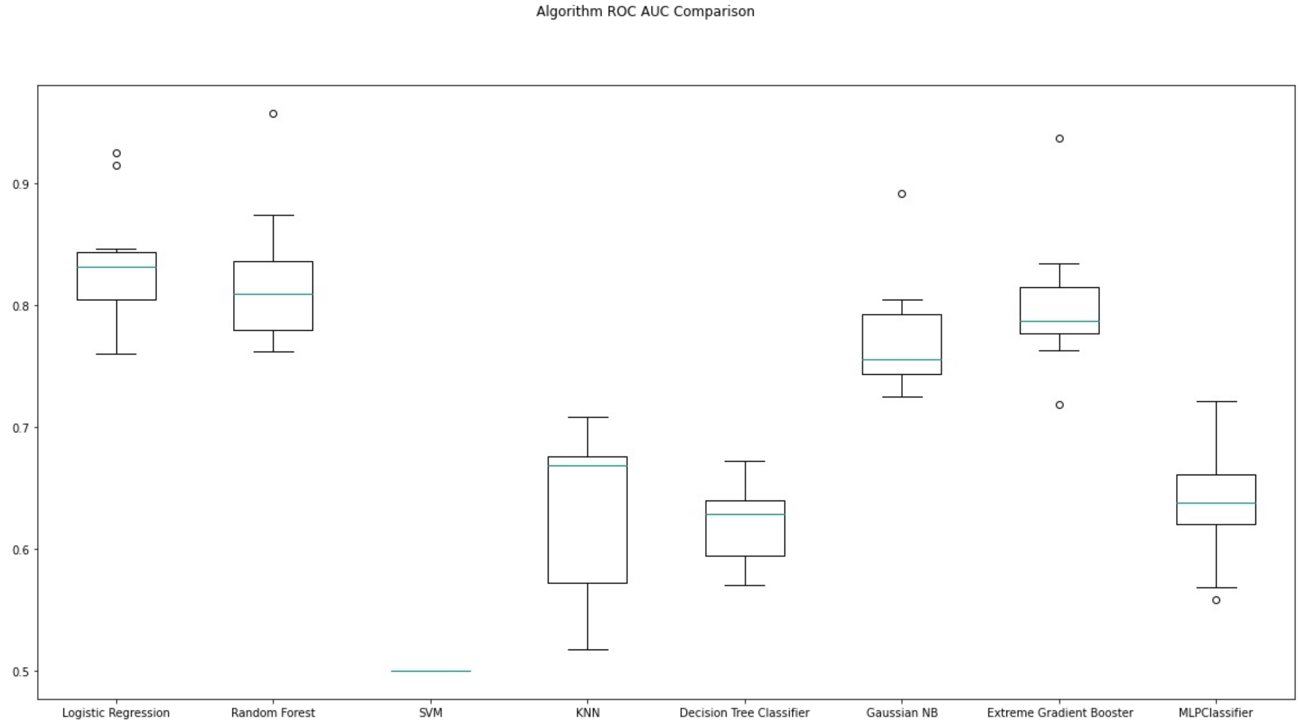
In [ ]:

```
fig = plt.figure(figsize=(20, 10))
fig.suptitle('Algorithm Accuracy Comparison')
ax = fig.add_subplot(111)
plt.boxplot(acc_results)
ax.set_xticklabels(names)
plt.show()
```



In [ ]:

```
fig = plt.figure(figsize=(20,10))
fig.suptitle('Algorithm ROC AUC Comparison')
ax = fig.add_subplot(111)
plt.boxplot(auc_results)
ax.set_xticklabels(names)
plt.show()
```



5.3 Finetuning Models

5.3.1 Logistic Regresssion

In [ ]:

```
# define evaluation
# define search space
logr_params = {
    'solver': ['newton-cg','lbfgs', 'liblinear'],
    'penalty': ['none','l1','l2', 'elasticnet'],
    'C': [1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 10, 100]
}
# define search
logr_gs = GridSearchCV(LogisticRegression(), param_grid=logr_params, scoring='accuracy',cv=5, n_jobs=-1,verbose=1
)
logr_gs.fit(X_train_scaled, y_train)
logr_gs.best_params_
```

```
Fitting 5 folds for each of 96 candidates, totalling 480 fits
```

Out[ ]:

```
{'C': 0.1, 'penalty': 'l2', 'solver': 'newton-cg'}
```

In [ ]:

```
print('Train acc =', logr_gs.score(X_train_scaled, y_train))
print('Test acc = ', logr_gs.score(X_test_scaled, y_test))
```

```
Train acc = 0.8929219600725953
Test acc =  0.907608695652174
```

In [ ]:

```
y_pred = logr_gs.predict(X_test_scaled)
cm = confusion_matrix(y_test, y_pred)
print(cm)
print(classification_report(y_test, y_pred))
```

```
[[313   7]
 [ 27  21]]
              precision    recall  f1-score   support

           0       0.92      0.98      0.95       320
           1       0.75      0.44      0.55        48

    accuracy                           0.91       368
   macro avg       0.84      0.71      0.75       368
weighted avg       0.90      0.91      0.90       368
```

5.3.2 Random Forrest Classification

In [ ]:

```
#Create the parameter grid based on the results of random search
rf_params = {
    'bootstrap': [True],
    'max_depth': [80, 90, 100],
    'max_features': [2, 3],
    'min_samples_leaf': [3, 4, 5],
    'min_samples_split': [8, 10],
    'n_estimators': [100, 200, 300]
}# Create a based model
rf_gs = GridSearchCV(RandomForestClassifier(),param_grid= rf_params, cv= 3, n_jobs=-1,verbose=1)# Instantiate the
grid search model
rf_gs.fit(X_train_scaled, y_train)
rf_gs.best_params_
```

```
Fitting 3 folds for each of 108 candidates, totalling 324 fits
```

Out[ ]:

```
{'bootstrap': True,
 'max_depth': 90,
 'max_features': 3,
 'min_samples_leaf': 3,
 'min_samples_split': 8,
 'n_estimators': 200}
```

In [ ]:

```
print('Train acc =', rf_gs.score(X_train_scaled, y_train))
print('Test acc = ', rf_gs.score(X_test_scaled, y_test))
```

```
Train acc = 0.8974591651542649
Test acc =  0.875
```

In [ ]:

```
y_pred = rf_gs.predict(X_test_scaled)
cm = confusion_matrix(y_test, y_pred)
print(cm)
print(classification_report(y_test, y_pred))
```

```
[[318   2]
 [ 44   4]]
              precision    recall  f1-score   support

           0       0.88      0.99      0.93       320
           1       0.67      0.08      0.15        48

    accuracy                           0.88       368
   macro avg       0.77      0.54      0.54       368
weighted avg       0.85      0.88      0.83       368
```

5.3.3 XGradientBooster

In [ ]:

```
xgbparams = {
    'max_depth':[1,3,5],
    'learning_rate':[.1,.5,.7,.8],
    'n_estimators':[25,50,100]
}

xgb_gs = GridSearchCV(XGBClassifier(), param_grid = xgbparams, cv=5, n_jobs=-1, verbose = 1)
xgb_gs.fit(X_train_scaled, y_train)
xgb_gs.best_params_
```

```
Fitting 5 folds for each of 36 candidates, totalling 180 fits
[23:58:02] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
```

Out[ ]:

```
{'learning_rate': 0.5, 'max_depth': 1, 'n_estimators': 100}
```

In [ ]:

```
print('Train acc =', xgb_gs.score(X_train_scaled, y_train))
print('Test acc = ', xgb_gs.score(X_test_scaled, y_test))
```

```
Train acc = 0.9165154264972777
Test acc =  0.8831521739130435
```

In [ ]:

```
y_pred = xgb_gs.predict(X_test_scaled)
cm = confusion_matrix(y_test, y_pred)
print(cm)
print(classification_report(y_test, y_pred))
```

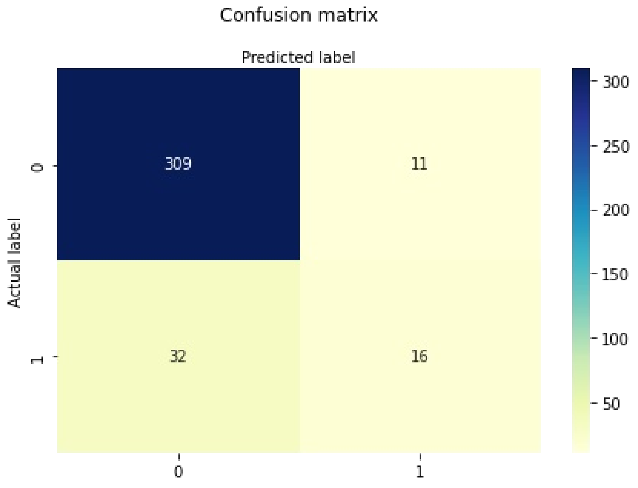
```
[[309  11]
 [ 32  16]]
              precision    recall  f1-score   support

           0       0.91      0.97      0.93       320
           1       0.59      0.33      0.43        48

    accuracy                           0.88       368
   macro avg       0.75      0.65      0.68       368
weighted avg       0.87      0.88      0.87       368
```

In [ ]:

```
fig, ax = plt.subplots()
sns.heatmap(pd.DataFrame(cm), annot=True, cmap="YlGnBu" ,fmt='g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
```

Out[ ]:

```
Text(0.5, 257.44, 'Predicted label')
```



5.3.4 MLPClassifier

In [ ]:

```
#from sklearn.neural_network import MLPClassifier
```

In [ ]:

```
mlpparams = {
            'learning_rate': ["constant", "invscaling", "adaptive"],
            'hidden_layer_sizes': [(30,), (60,), (50,), (40,)],
            'alpha': [.1],
            'activation': ["logistic", "relu", "tanh"]
            }

mlp_gs = GridSearchCV(MLPClassifier(), param_grid = mlpparams, cv = 5, verbose = 1)
mlp_gs.fit(X_train_scaled, y_train)
mlp_gs.best_params_
```

```
Fitting 5 folds for each of 36 candidates, totalling 180 fits
```

Out[ ]:

```
{'activation': 'logistic',
 'alpha': 0.1,
 'hidden_layer_sizes': (30,),
 'learning_rate': 'adaptive'}
```

In [ ]:

```
print('Train acc =', mlp_gs.score(X_train_scaled, y_train))
print('Test acc =', mlp_gs.score(X_test_scaled, y_test))
```

```
Train acc = 0.8929219600725953
Test acc = 0.9021739130434783
```

In [ ]:

```
y_pred = mlp_gs.predict(X_test_scaled)
cm = confusion_matrix(y_test, y_pred)
print(cm)
print(classification_report(y_test, y_pred))
```

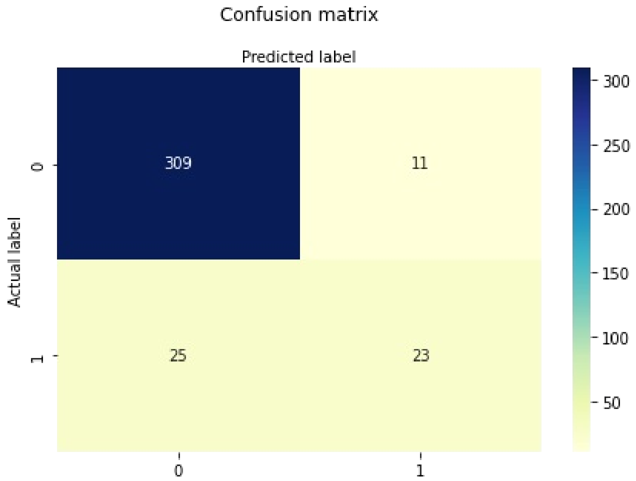
```
[[309  11]
 [ 25  23]]
              precision    recall  f1-score   support

           0       0.93      0.97      0.94       320
           1       0.68      0.48      0.56        48

    accuracy                           0.90       368
   macro avg       0.80      0.72      0.75       368
weighted avg       0.89      0.90      0.89       368
```

In [ ]:

```
fig, ax = plt.subplots()
sns.heatmap(pd.DataFrame(cm), annot=True, cmap="YlGnBu" ,fmt='g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
```

Out[ ]:

```
Text(0.5, 257.44, 'Predicted label')
```



5.3.5 Making Tuned models

Making models

In [ ]:

```
logreg = LogisticRegression()
logreg.fit(X_train_scaled, y_train)
#'C': 0.1, 'penalty': 'l2', 'solver': 'liblinear'}
```

Out[ ]:

```
LogisticRegression()
```

In [ ]:

```
xgb = XGBClassifier(max_depth = 1, learning_rate = .8, n_estimators = 50)
xgb.fit(X_train_scaled, y_train)
```

```
[00:07:50] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
```

Out[ ]:

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
              gamma=0, gpu_id=-1, importance_type=None,
              interaction_constraints='', learning_rate=0.8, max_delta_step=0,
              max_depth=1, min_child_weight=1, missing=nan,
              monotone_constraints='()', n_estimators=50, n_jobs=2,
              num_parallel_tree=1, predictor='auto', random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=None)
```

In [ ]:

```
mlp = MLPClassifier(activation = 'logistic', hidden_layer_sizes = (60,), alpha = .1, \
                    learning_rate = 'adaptive')
mlp.fit(X_train_scaled, y_train)
```

Out[ ]:

```
MLPClassifier(activation='logistic', alpha=0.1, hidden_layer_sizes=(60,),
              learning_rate='adaptive')
```

In [ ]:

```
rf = RandomForestClassifier(min_samples_leaf =3, min_samples_split =8, max_features =3, max_depth = 90,random_sta
te =42)
rf.fit(X_train_scaled, y_train)
```

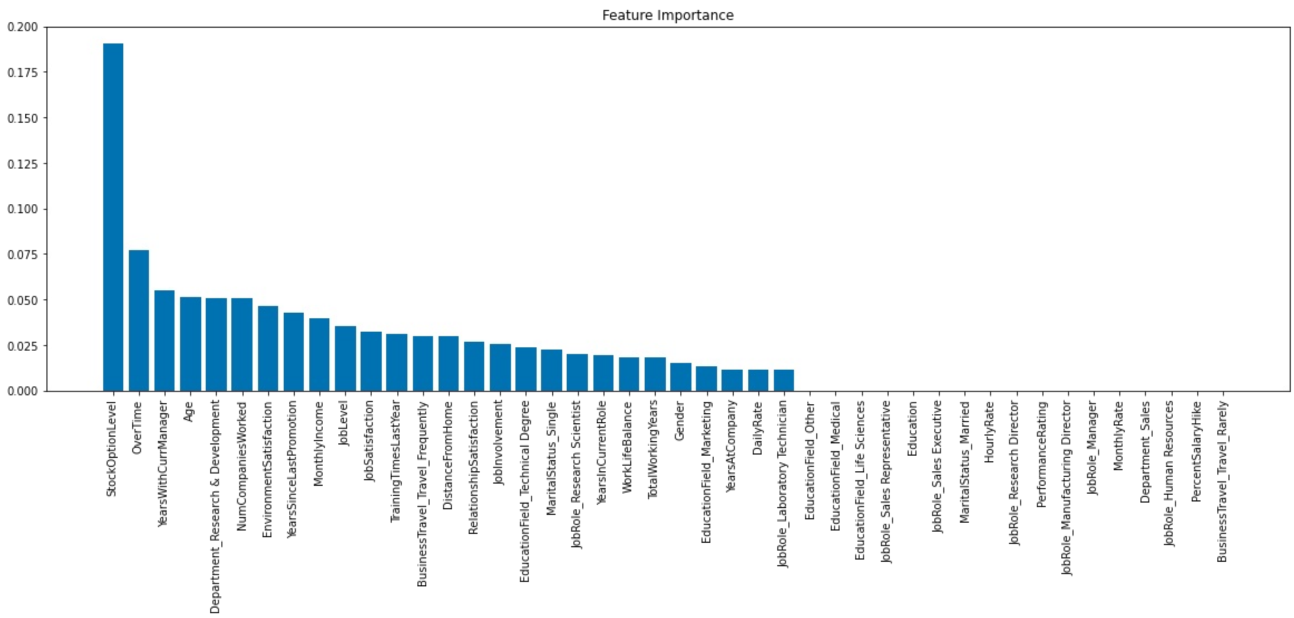
Out[ ]:

```
RandomForestClassifier(max_depth=90, max_features=3, min_samples_leaf=3,
                       min_samples_split=8, random_state=42)
```

5.4 Most Important Features

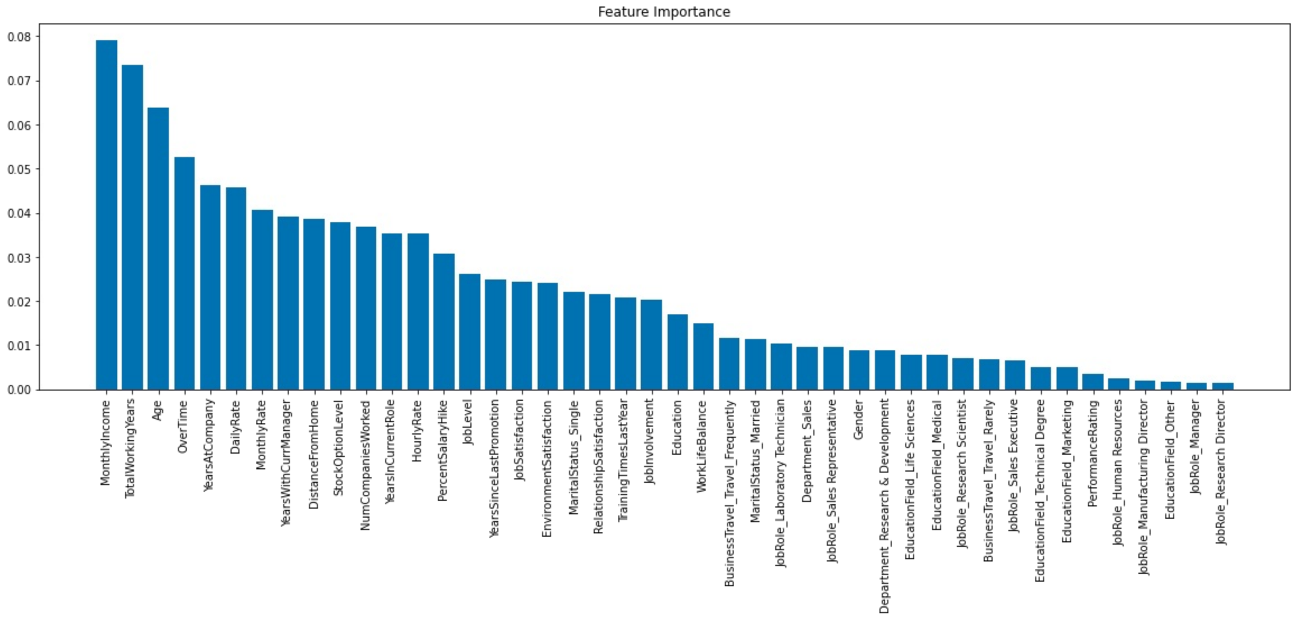
In [ ]:

```
importances = xgb.feature_importances_
indices = np.argsort(importances)[::-1] # Sort feature importances in descending order
names = [X_train.columns[i] for i in indices] # Rearrange feature names so they match the sorted feature importan
ces
plt.figure(figsize=(20, 6)) # Create plot
plt.title("Feature Importance") # Create plot title
plt.bar(range(X_train_scaled.shape[1]), importances[indices]) # Add bars
plt.xticks(range(X_train_scaled.shape[1]), names, rotation=90) # Add feature names as x-axis labels
plt.show() # Show plot
```



In [ ]:

```
importances = rf.feature_importances_
indices = np.argsort(importances)[::-1] # Sort feature importances in descending order
names = [X_train.columns[i] for i in indices] # Rearrange feature names so they match the sorted feature importances
plt.figure(figsize=(20, 6)) # Create plot
plt.title("Feature Importance") # Create plot title
plt.bar(range(X_train_scaled.shape[1]), importances[indices]) # Add bars
plt.xticks(range(X_train_scaled.shape[1]), names, rotation=90) # Add feature names as x-axis labels
plt.show() # Show plot
```



5.5 ROC and AUC Scores

In [ ]:

```
mlp_gs.best_params_
probs = mlp_gs.predict_proba(X_test_scaled) # predict probabilities
probs = probs[:, 1] # we will only keep probabilities associated with the employee leaving
mlp_roc_auc = roc_auc_score(y_test, probs) # calculate AUC score using test dataset
print('AUC score: %.3f' % mlp_roc_auc)
```

```
AUC score: 0.819
```

In [ ]:

```
xgb_gs.best_params_ # fit optimised model to the training data
probs = xgb_gs.predict_proba(X_test_scaled) # predict probabilities
probs = probs[:, 1] # we will only keep probabilities associated with the employee leaving
xgb_roc_auc = roc_auc_score(y_test, probs) # calculate AUC score using test dataset
print('AUC score: %.3f' % xgb_roc_auc)
```

```
AUC score: 0.814
```

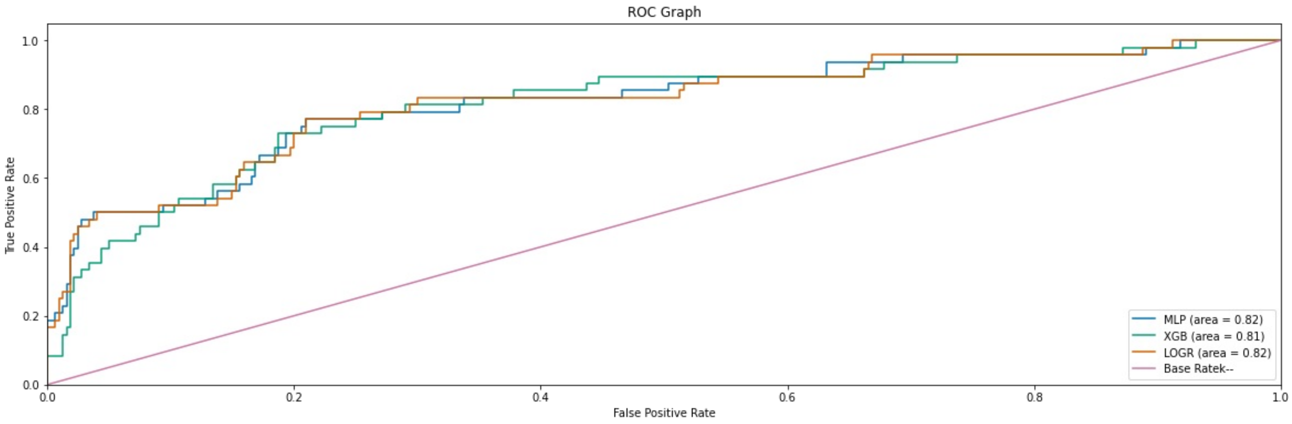
In [ ]:

```
logr_gs.best_params_ # fit optimised model to the training data
probs = logr_gs.predict_proba(X_test_scaled) # predict probabilities
probs = probs[:, 1] # we will only keep probabilities associated with the employee leaving
logr_roc_auc = roc_auc_score(y_test, probs) # calculate AUC score using test dataset
print('AUC score: %.3f' % logr_roc_auc)
```

```
AUC score: 0.818
```

In [ ]:

```
# Create ROC Graph
from sklearn.metrics import roc_curve
fpr, tpr, thresholds = roc_curve(y_test, mlp_gs.predict_proba(X_test_scaled)[:,1])
xgb_fpr, xgb_tpr, xgb_thresholds = roc_curve(y_test, xgb_gs.predict_proba(X_test_scaled)[:,1])
logr_fpr, logr_tpr, logr_thresholds = roc_curve(y_test, logr_gs.predict_proba(X_test_scaled)[:,1])
plt.figure(figsize=(20, 6))

# Plot MLP ROC
plt.plot(fpr, tpr, label='MLP (area = %0.2f)' % mlp_roc_auc)
# Plot XGB ROC
plt.plot(xgb_fpr, xgb_tpr, label='XGB (area = %0.2f)' % xgb_roc_auc)
# Plot LogR ROC
plt.plot(logr_fpr, logr_tpr, label='LOGR (area = %0.2f)' % logr_roc_auc)
# Plot Base Rate ROC
plt.plot([0,1], [0,1],label='Base Rate' 'k--')

plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Graph')
plt.legend(loc="lower right")
plt.show()
```



6. Cost Function

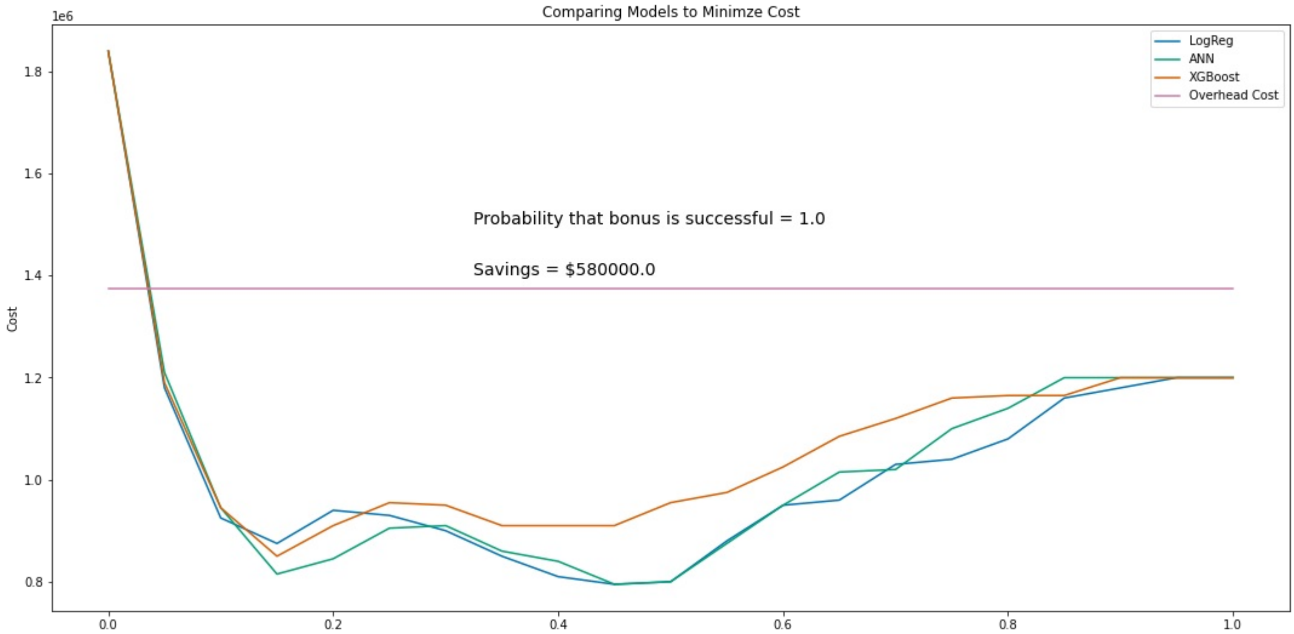
In [ ]:

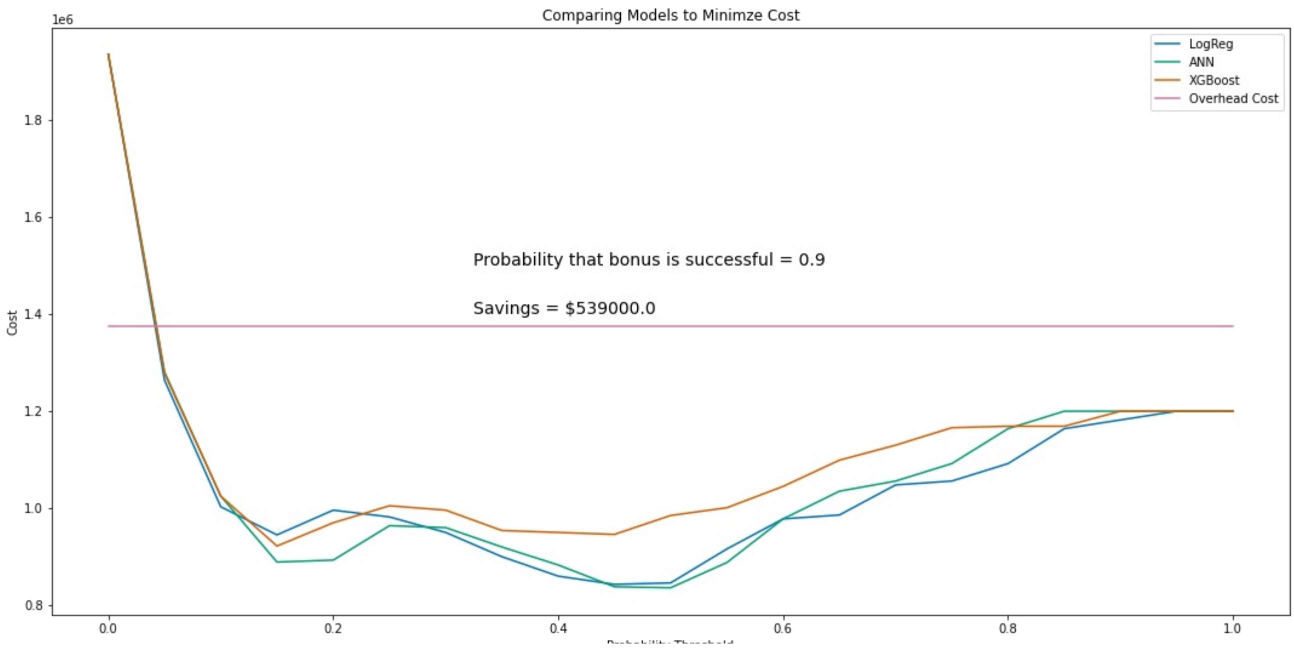
```
def best_threshold (model, steps, X, y, p):
    salary = 50000.0
    bonus = 5000.0
    TN_cost = 0
    TP_cost = p*bonus + (1-p)*.5*salary
    FP_cost = bonus
    FN_cost = .5*salary

    cost = []
    threshold = 0

    m = model
    #train test split
    X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 42)
    #scale the data
    scaler = StandardScaler()
    # Fit_transform
    X_train_scaled = scaler.fit_transform(X_train)
    # transform
    X_test_scaled = scaler.transform(X_test)

    m.fit(X_train_scaled, y_train)

    for i in range(steps + 1):
        y_pred_train = (model.predict_proba(X_train_scaled)[:,1] > threshold)
        y_pred_test = (model.predict_proba(X_test_scaled)[:,1] > threshold)

        cm = confusion_matrix(y_test, y_pred_test)
        TN = cm[0,0]
        TP = cm[1,1]
        FP = cm[0,1]
        FN = cm[1,0]

        total_cost = TN_cost*TN + TP_cost*TP + FP_cost*FP + FN_cost*FN
        results_dict = {
                'threshold' : threshold,
                'cost' : total_cost,
                'precision_score_test': precision_score(y_test, y_pred_test),
                'recall_score_test': recall_score(y_test, y_pred_test),
                'TN': TN,
                'FP': FP,
                'FN': FN,
                'TP': TP,
                        }
        cost.append(results_dict)
        threshold += (1.0/steps)

    thresh_results = pd.DataFrame(cost, columns=['cost', 'threshold', 'precision_score_test',\
                        'recall_score_test','FN', 'FP','TN','TP'])
    return thresh_results
```

In [ ]:

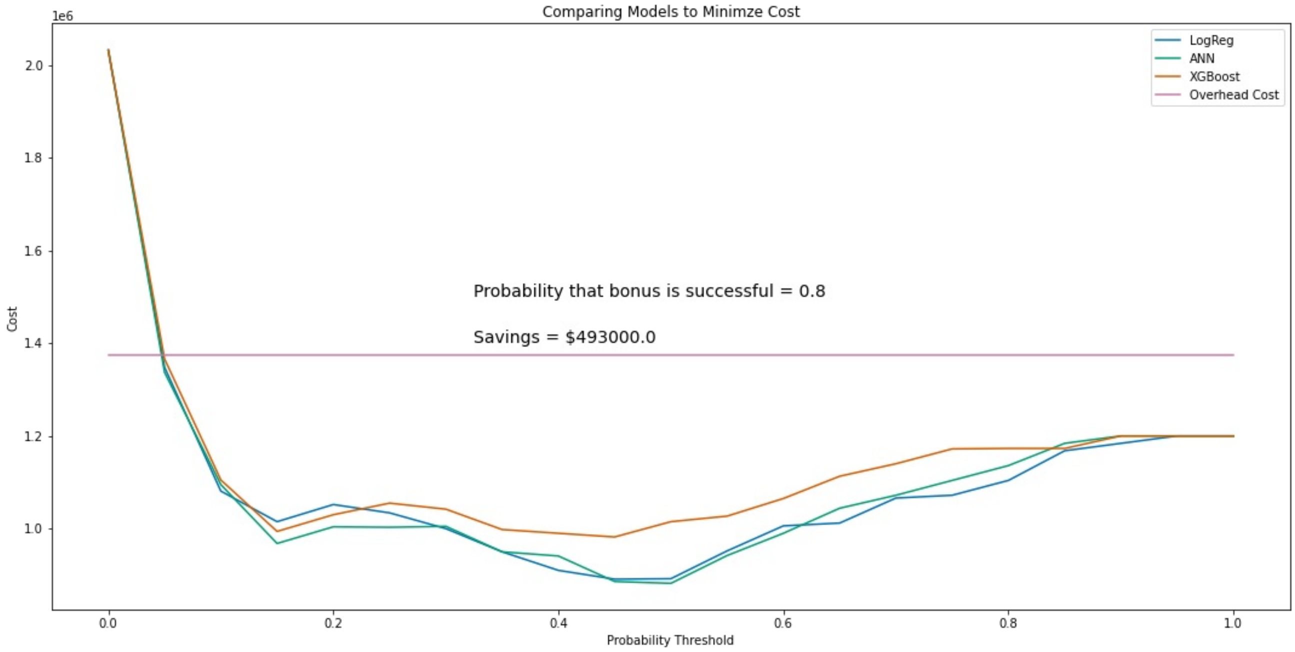
```
fig = plt.figure(figsize = (15,90))
j = 0
probabilities = -(np.sort(-np.linspace(0,1,11)))

for i in probabilities:
    df1 = best_threshold(xgb,20,X,y, i)
    df2 = best_threshold(mlp,20,X,y, i)
    df3 = best_threshold(logreg,20,X,y, i)

    j += 1
    fig.add_subplot(12,1,j)
    plt.plot(df3['threshold'], df3['cost'], label = 'LogReg')
    plt.plot(df2['threshold'], df2['cost'], label = 'ANN')
    plt.plot(df1['threshold'], df1['cost'], label = 'XGBoost')
    plt.plot(np.linspace(0,1,9), 1375000*np.ones(9), label = 'Overhead Cost')
    plt.ylabel('Cost')
    plt.xlabel('Probability Threshold')
    plt.title('Comparing Models to Minimze Cost')
    plt.text(x = .325, y = 1500000, s = 'Probability that bonus is successful = ' + str(i), fontsize = 14)

    m = df2['cost'].min()
    profit = 1375000 - m
    plt.text(x = .325, y = 1400000, s = 'Savings = $' + str(profit), fontsize = 14)
    plt.legend()
plt.tight_layout()
```

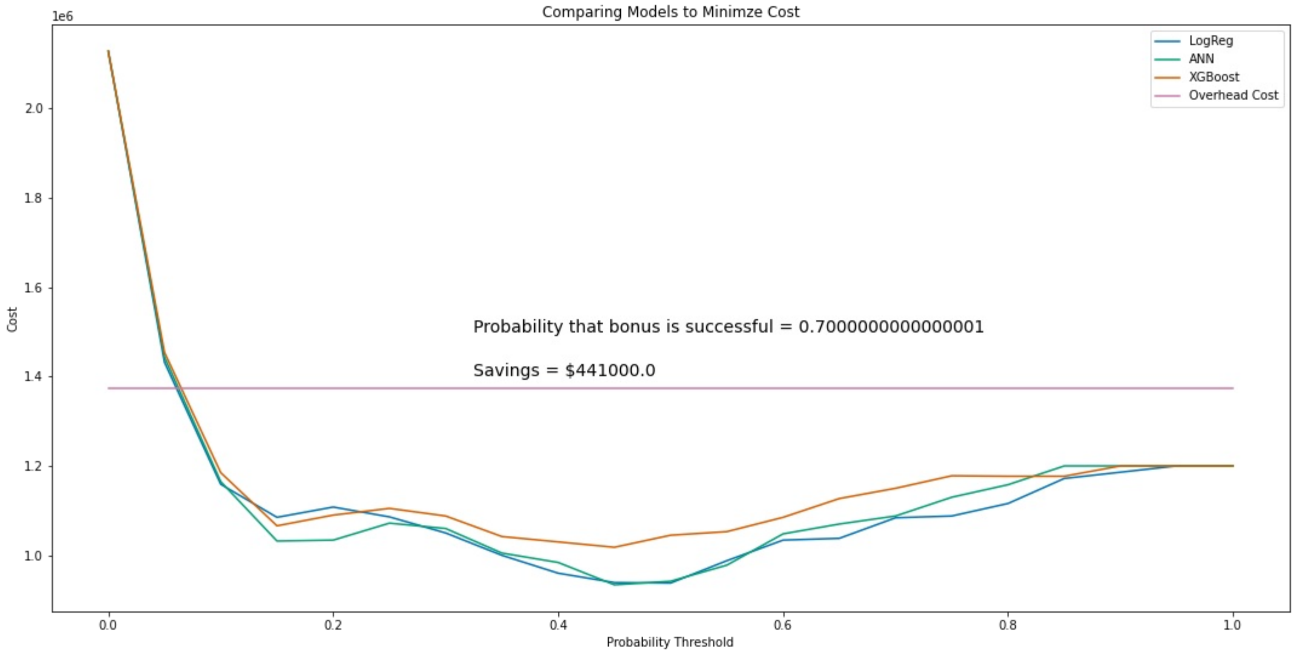
```
[00:08:00] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[00:08:07] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[00:08:15] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[00:08:22] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[00:08:28] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[00:08:37] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[00:08:46] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[00:08:54] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[00:09:02] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[00:09:11] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
[00:09:19] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluation metric
used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval
_metric if you'd like to restore the old behavior.
```

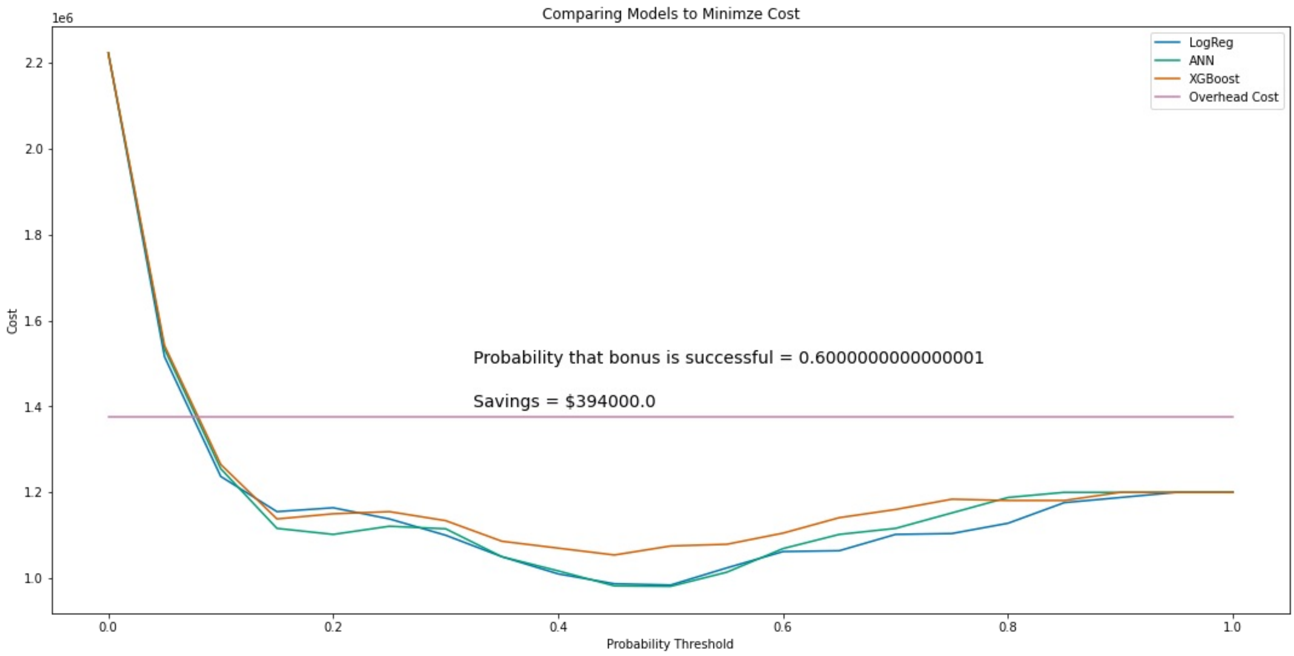


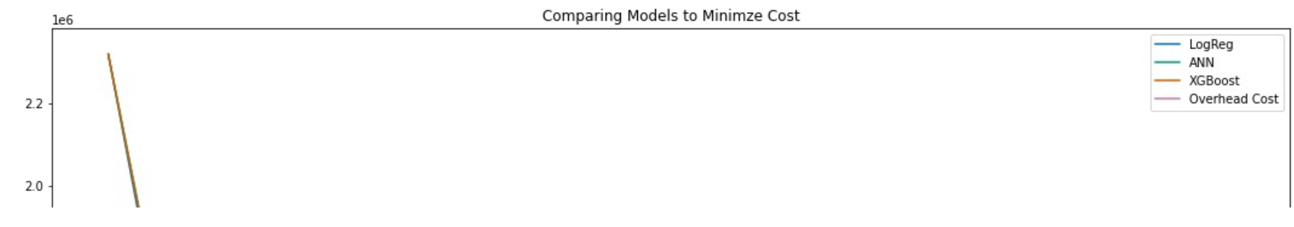


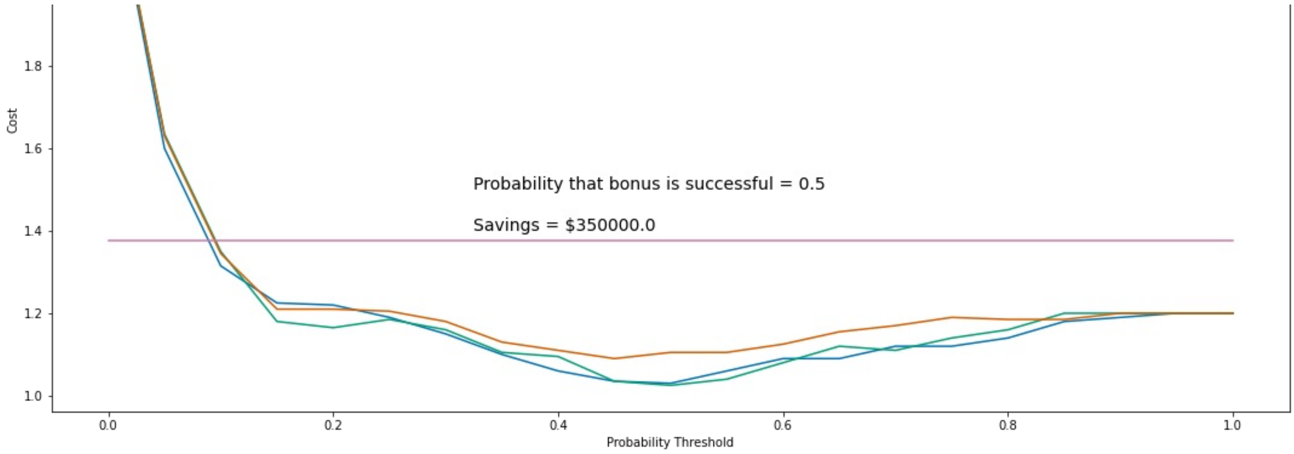
Probability Threshold

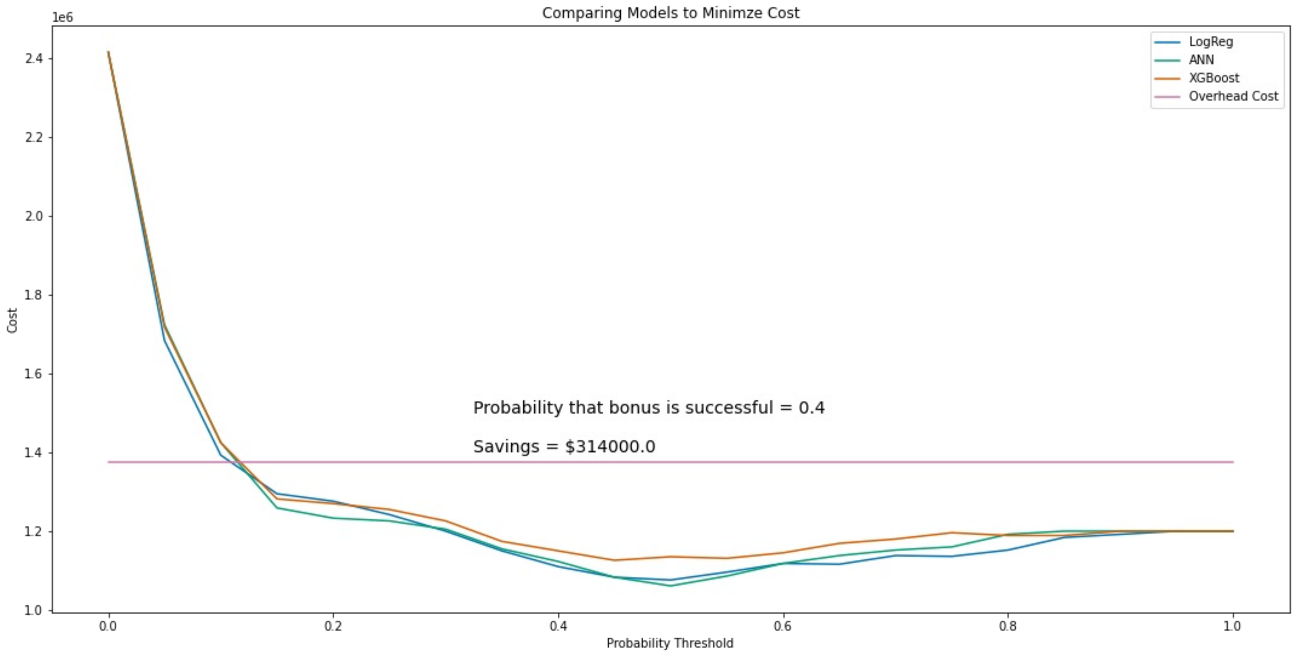


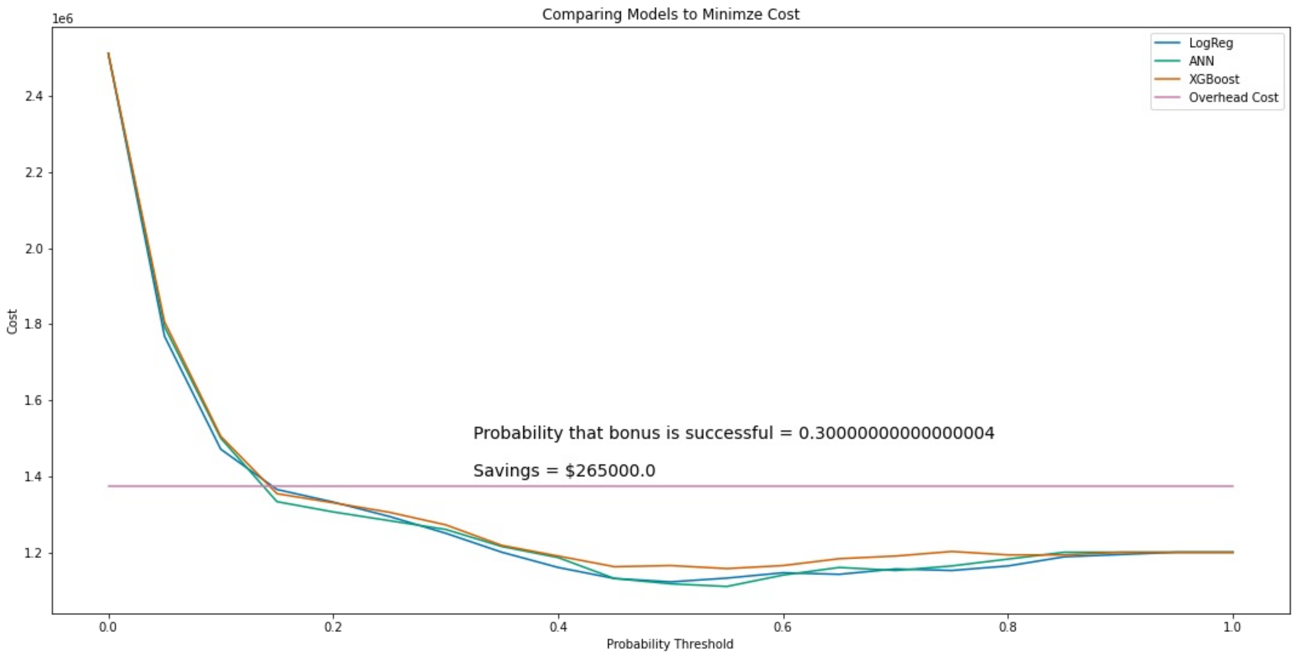


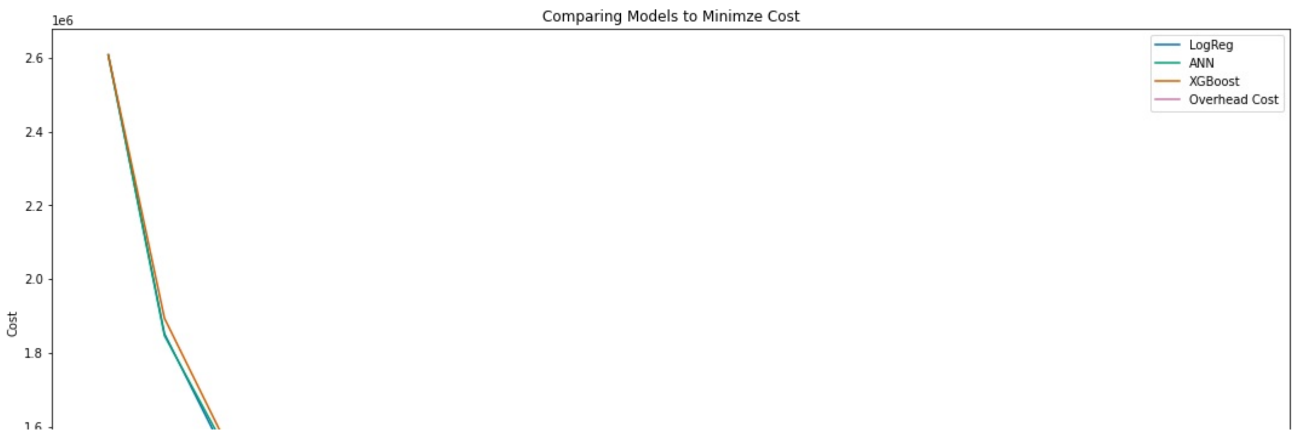


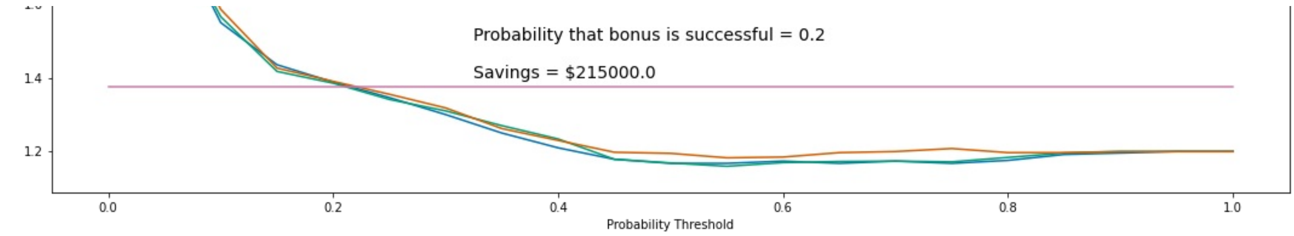


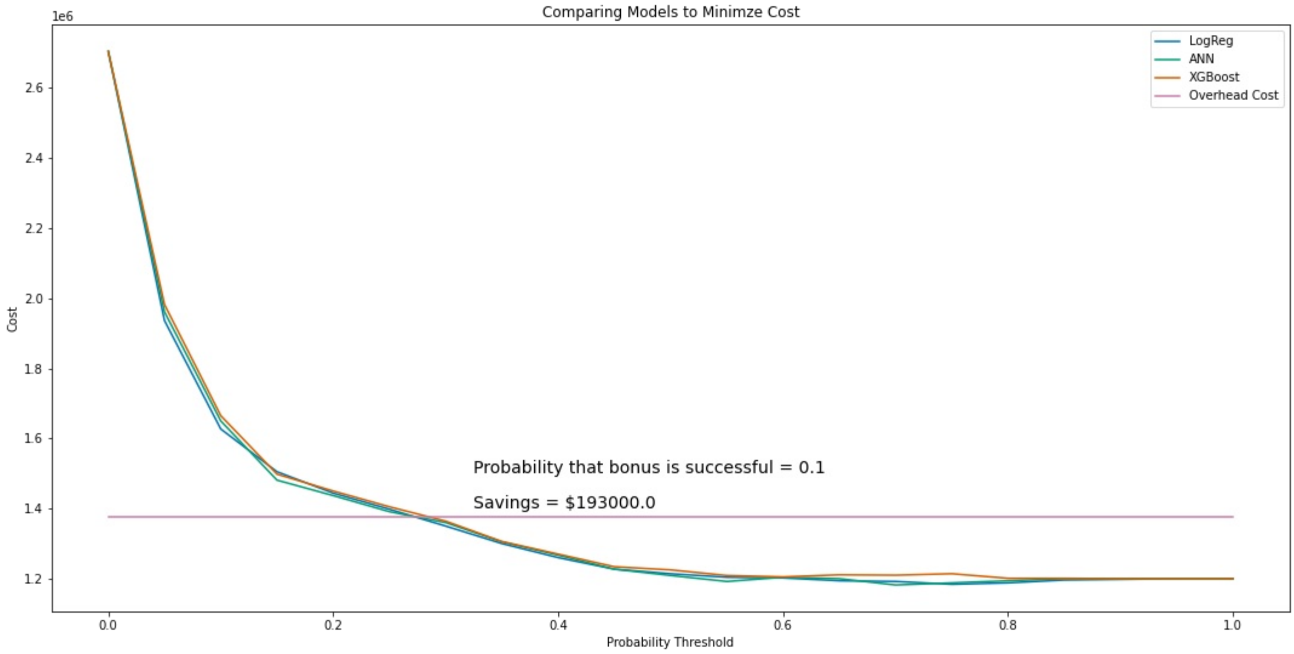

Cost
Probability that bonus is successful = 0.5
Savings = $350000.0
Probability Threshold

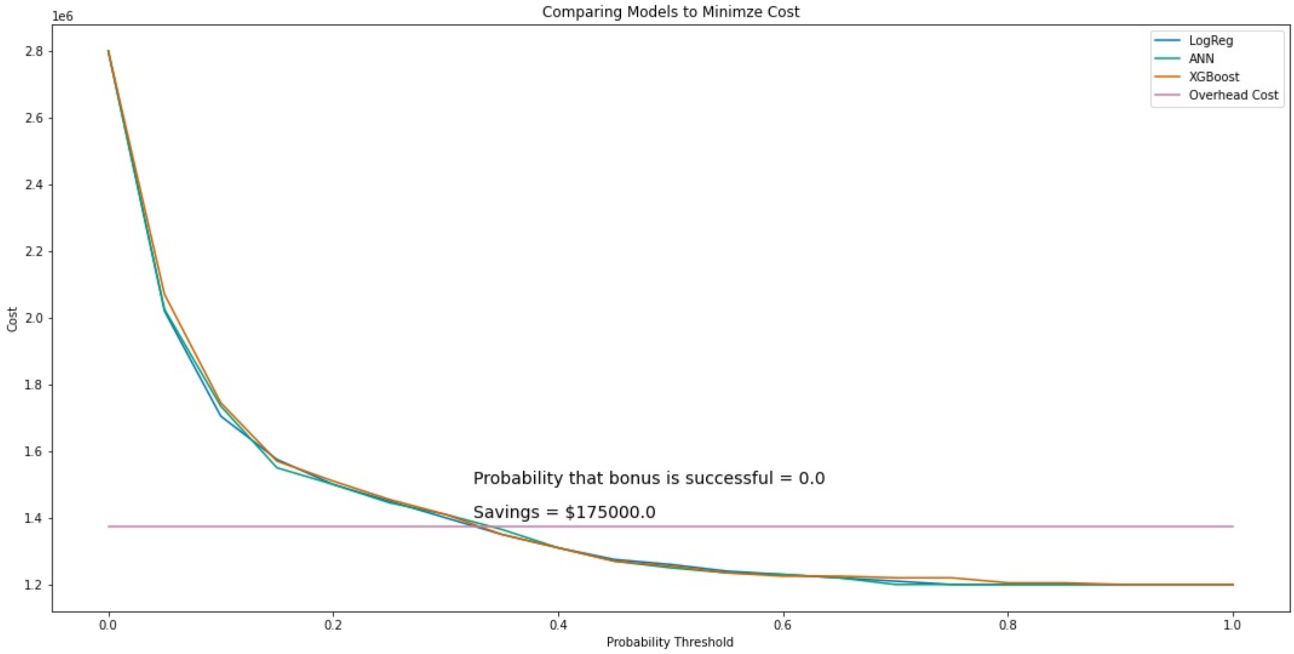

Comparing Models to Minimze Cost
1e6
LogReg
ANN
XGBoost
Overhead Cost
Cost
Probability that bonus is successful = 0.4
Savings = $314000.0
Probability Threshold


Comparing Models to Minimze Cost
1e6
LogReg
ANN
XGBoost
Overhead Cost
Cost
Probability that bonus is successful = 0.30000000000000004
Savings = $265000.0
Probability Threshold


Comparing Models to Minimze Cost
1e6
LogReg
ANN
XGBoost
Overhead Cost
Cost







7. Survival Analysis

The Kaplan-Meier survival curve is defined as the probability of surviving in a given lenght of time while considering time in many small intervals. in this case using *YearsAtCompany* as our time frame and *Attrition* as the survival condition. By then estimating conditional probabilities at each time point, we can get an estimate survival rate at each point in time.

The YearsAtCompany coulmn have 25 values above 3 STD above MEAN This values are 1.7% of the data and will be dropped for the survival analysis

In [ ]:

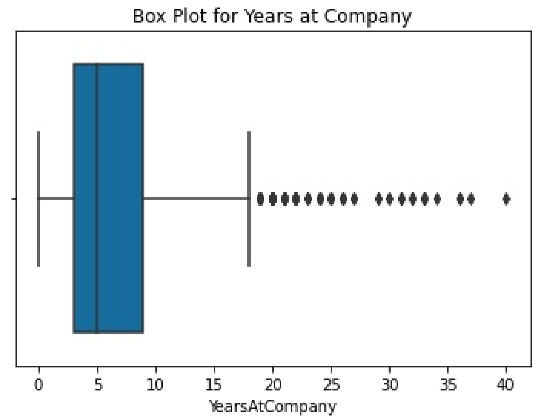
```
hr_survi['YearsAtCompany'].describe() #mean = 7 years, std = 6 years
```

Out[ ]:

```
count    1470.000000
mean        7.008163
std         6.126525
min         0.000000
25%         3.000000
50%         5.000000
75%         9.000000
max        40.000000
Name: YearsAtCompany, dtype: float64
```

In [ ]:

```
sns.boxplot(hr_survi['YearsAtCompany'])
plt.title('Box Plot for Years at Company')
plt.show()
```



In [ ]:

```
threshold = np.std(hr_survi['YearsAtCompany']) * 3 # 3 std above mean
len(hr_survi[hr_survi['YearsAtCompany'] > np.mean(hr_survi['YearsAtCompany']) + threshold])
```

Out[ ]:

```
25
```

In [ ]:

```
hr_survi = hr_survi[hr_survi['YearsAtCompany'] < np.mean(hr_survi['YearsAtCompany']) + threshold]
```

In [ ]:

```
hr_survi[hr_survi['YearsAtCompany'] < 1].describe()
```

Out[ ]:

	Age	Attrition	DailyRate	DistanceFromHome	Education	EnvironmentSatisfaction	HourlyRate	JobInvolvement	JobLe
count	44.000000	44.000000	44.000000	44.000000	44.000000	44.000000	44.000000	44.000000	44.0000
mean	31.227273	0.363636	770.659091	8.477273	2.840909	2.795455	68.886364	2.704545	1.5227
std	10.924399	0.486607	437.970687	7.659888	1.140036	1.069222	20.000251	0.667503	0.8209
min	18.000000	0.000000	111.000000	1.000000	1.000000	1.000000	32.000000	1.000000	1.0000
25%	21.750000	0.000000	388.500000	2.000000	2.000000	2.000000	51.750000	2.000000	1.0000
50%	29.500000	0.000000	649.500000	6.000000	3.000000	3.000000	71.500000	3.000000	1.0000
75%	40.500000	1.000000	1190.500000	12.250000	4.000000	4.000000	84.000000	3.000000	2.0000
max	56.000000	1.000000	1495.000000	29.000000	5.000000	4.000000	100.000000	4.000000	5.0000

8 rows × 24 columns

In [ ]:

```
kmf = KaplanMeierFitter()
kmf.fit(durations = hr_survi['YearsAtCompany'],
        event_observed =hr_survi['Attrition'])
kmf.event_table
```

Out[ ]:

event_at	removed	observed	censored	entrance	at_risk
0	44	16	28	1445	1445
1	171	59	112	0	1401
2	127	27	100	0	1230
3	128	20	108	0	1103
4	110	19	91	0	975
5	196	21	175	0	865
6	76	9	67	0	669
7	90	11	79	0	593
8	80	9	71	0	503
9	82	8	74	0	423
10	120	18	102	0	341
11	32	2	30	0	221
12	14	0	14	0	189
13	24	2	22	0	175
14	18	2	16	0	151
15	20	1	19	0	133
16	12	1	11	0	113
17	9	1	8	0	101
18	13	1	12	0	92
19	11	1	10	0	79
20	27	1	26	0	68
21	14	1	13	0	41
22	15	1	14	0	27
23	2	1	1	0	12
24	6	1	5	0	10
25	4	0	4	0	4

In [ ]:

```
hr_survi.info()
```

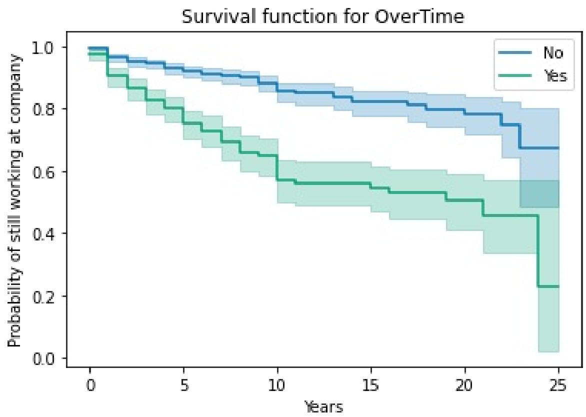
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1445 entries, 0 to 1469
Data columns (total 31 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   Age                       1445 non-null   int64 
 1   Attrition                 1445 non-null   int64 
 2   BusinessTravel            1445 non-null   object
 3   DailyRate                 1445 non-null   int64 
 4   Department                1445 non-null   object
 5   DistanceFromHome          1445 non-null   int64 
 6   Education                 1445 non-null   int64 
 7   EducationField            1445 non-null   object
 8   EnvironmentSatisfaction   1445 non-null   int64 
 9   Gender                    1445 non-null   object
 10  HourlyRate                1445 non-null   int64 
 11  JobInvolvement            1445 non-null   int64 
 12  JobLevel                  1445 non-null   int64 
 13  JobRole                   1445 non-null   object
 14  JobSatisfaction           1445 non-null   int64 
 15  MaritalStatus             1445 non-null   object
 16  MonthlyIncome             1445 non-null   int64 
 17  MonthlyRate               1445 non-null   int64 
 18  NumCompaniesWorked        1445 non-null   int64 
 19  OverTime                  1445 non-null   object
 20  PercentSalaryHike         1445 non-null   int64 
 21  PerformanceRating         1445 non-null   int64 
 22  RelationshipSatisfaction  1445 non-null   int64 
 23  StockOptionLevel          1445 non-null   int64 
 24  TotalWorkingYears         1445 non-null   int64 
 25  TrainingTimesLastYear     1445 non-null   int64 
 26  WorkLifeBalance           1445 non-null   int64 
 27  YearsAtCompany            1445 non-null   int64 
 28  YearsInCurrentRole        1445 non-null   int64 
 29  YearsSinceLastPromotion   1445 non-null   int64 
 30  YearsWithCurrManager      1445 non-null   int64 
dtypes: int64(24), object(7)
memory usage: 361.2+ KB
```

In [ ]:

```
def build_survival_plot(column, split1, split2):
    df_1 = hr_survi[hr_survi[column] == split1]
    df_2 = hr_survi[hr_survi[column] == split2]
    ax = plt.subplot(111)

    kmf.fit(durations = df_1['YearsAtCompany'],
        event_observed = df_1['Attrition'], label = split1)
    kmf.plot(ax = ax)

    kmf.fit(durations = df_2['YearsAtCompany'],
        event_observed = df_2['Attrition'], label = split2)
    kmf.plot(ax = ax)

    plt.title('Survival function for ' + column)
    plt.xlabel('Years')
    plt.ylabel('Probability of still working at company')
```

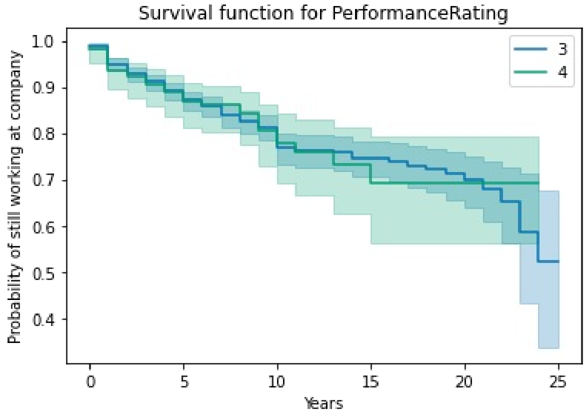
In [ ]:

```
build_survival_plot('OverTime', 'No', 'Yes')
```



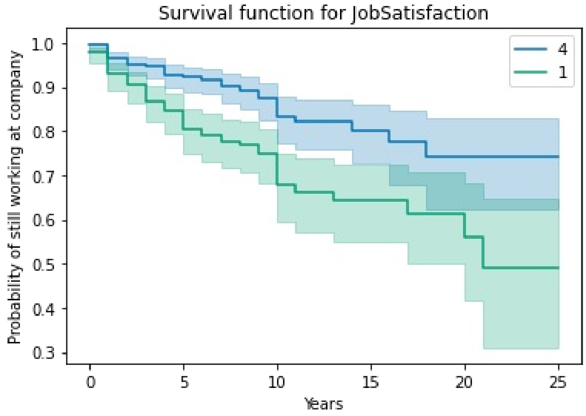
In [ ]:

```
build_survival_plot('PerformanceRating', 3, 4)
```



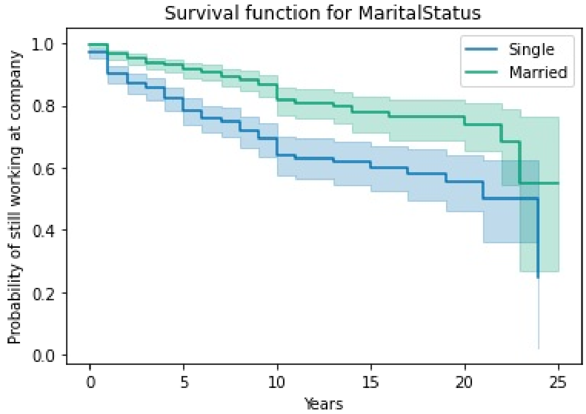
In [ ]:

```
build_survival_plot('JobSatisfaction', 4, 1)
```



In [ ]:

```
build_survival_plot('MaritalStatus', 'Single', 'Married')
```



In [ ]:

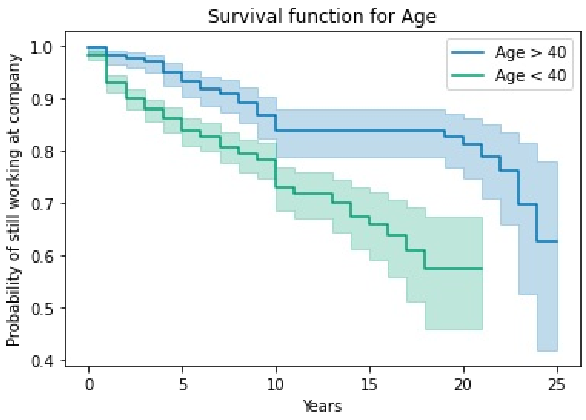
```
df_old = hr_survi[hr_survi['Age'] >= 40]
df_young = hr_survi[hr_survi['Age'] < 40]
```

In [ ]:

```
ax = plt.subplot(111)

kmf.fit(durations = df_old['YearsAtCompany'],
        event_observed = df_old['Attrition'], label = 'Age > 40')
kmf.plot(ax = ax)

kmf.fit(durations = df_young['YearsAtCompany'],
        event_observed = df_young['Attrition'], label = 'Age < 40')
kmf.plot(ax = ax)

plt.title('Survival function for Age')
plt.xlabel('Years')
plt.ylabel('Probability of still working at company')
```

Out[ ]:

```
Text(0, 0.5, 'Probability of still working at company')
```



In [ ]:

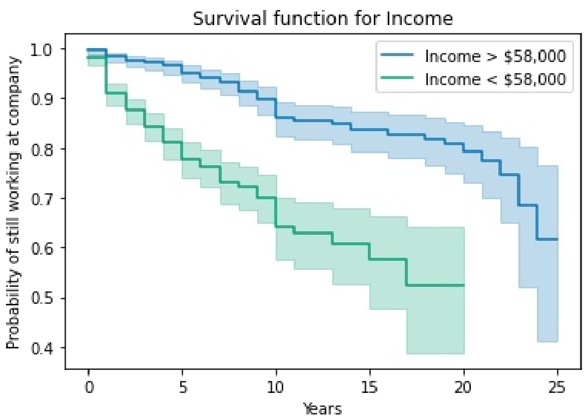
```
df_rich = hr_survi[hr_survi['MonthlyIncome'] >= 4833]
df_poor = hr_survi[hr_survi['MonthlyIncome'] < 4833]
```

In [ ]:

```
ax = plt.subplot(111)

kmf.fit(durations = df_rich['YearsAtCompany'],
        event_observed = df_rich['Attrition'], label = 'Income > $58,000')
kmf.plot(ax = ax)

kmf.fit(durations = df_poor['YearsAtCompany'],
        event_observed = df_poor['Attrition'], label = 'Income < $58,000')
kmf.plot(ax = ax)

plt.title('Survival function for Income')
plt.xlabel('Years')
plt.ylabel('Probability of still working at company')
```

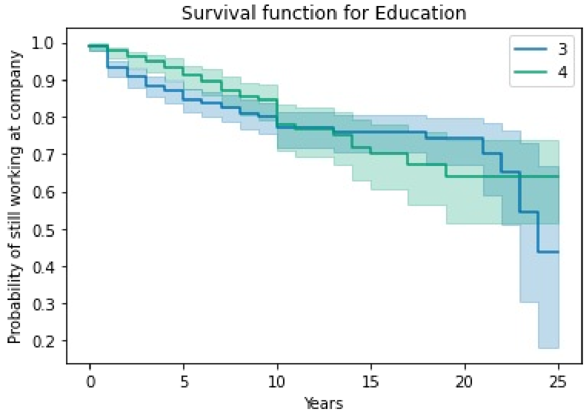
Out[ ]:

```
Text(0, 0.5, 'Probability of still working at company')
```



In [ ]:

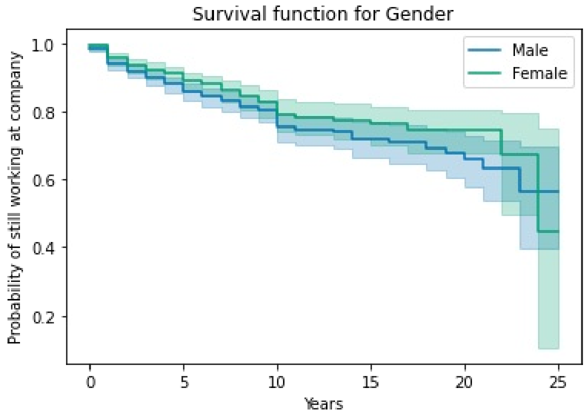
```
build_survival_plot('Education', 3, 4)
```



Gender

In [ ]:

```
build_survival_plot('Gender', 'Male', 'Female')
```



In [ ]:

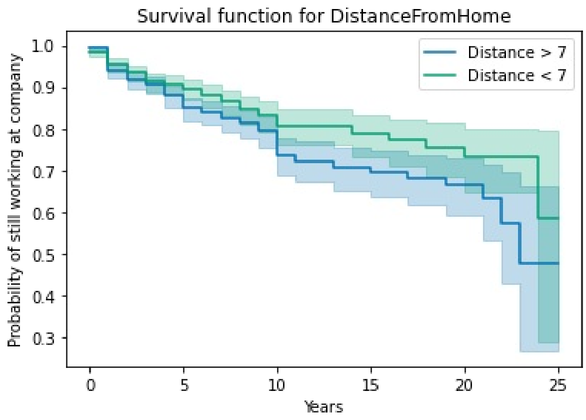
```
df_far = hr_survi[hr_survi['DistanceFromHome'] >= 7]
df_close = hr_survi[hr_survi['DistanceFromHome'] < 7]
```

In [ ]:

```
ax = plt.subplot(111)

kmf.fit(durations = df_far['YearsAtCompany'],
        event_observed = df_far['Attrition'], label = 'Distance > 7')
kmf.plot(ax = ax)

kmf.fit(durations = df_close['YearsAtCompany'],
        event_observed = df_close['Attrition'], label = 'Distance < 7')
kmf.plot(ax = ax)

plt.title('Survival function for DistanceFromHome')
plt.xlabel('Years')
plt.ylabel('Probability of still working at company')
```

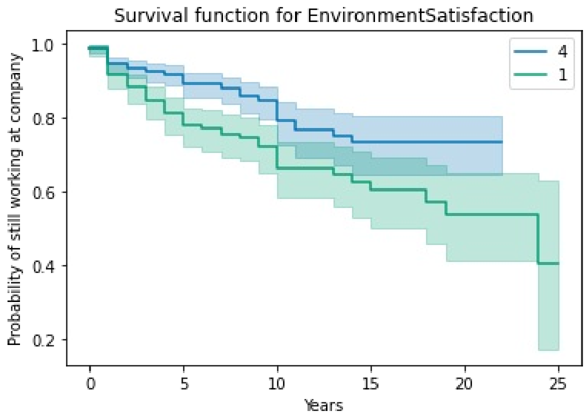
Out[ ]:

```
Text(0, 0.5, 'Probability of still working at company')
```



In [ ]:

```
build_survival_plot('EnvironmentSatisfaction', 4, 1)
```



In [ ]:

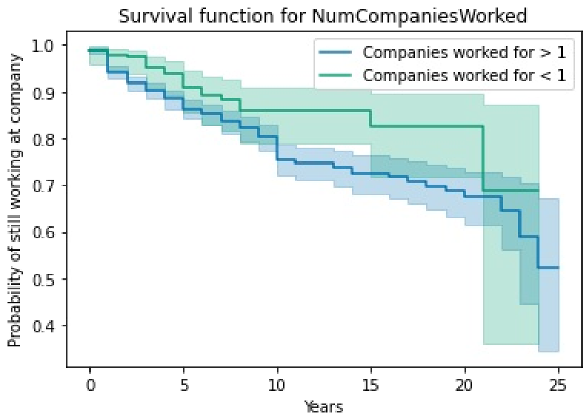
```
df_b = hr_survi[hr_survi['NumCompaniesWorked'] >= 1]
df_s = hr_survi[hr_survi['NumCompaniesWorked'] < 1]
```

In [ ]:

```
ax = plt.subplot(111)

kmf.fit(durations = df_b['YearsAtCompany'],
        event_observed = df_b['Attrition'], label = 'Companies worked for > 1')
kmf.plot(ax = ax)

kmf.fit(durations = df_s['YearsAtCompany'],
        event_observed = df_s['Attrition'], label = 'Companies worked for < 1')
kmf.plot(ax = ax)

plt.title('Survival function for NumCompaniesWorked')
plt.xlabel('Years')
plt.ylabel('Probability of still working at company')
```

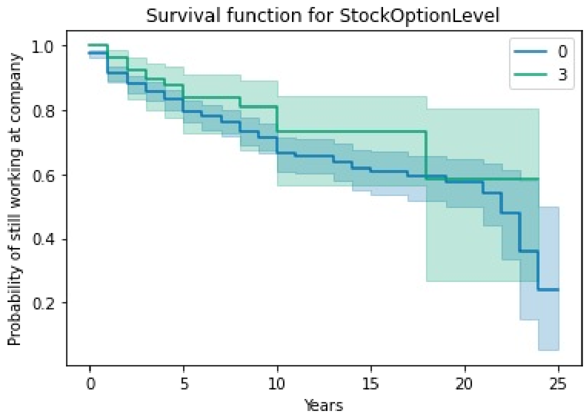
Out[ ]:

```
Text(0, 0.5, 'Probability of still working at company')
```



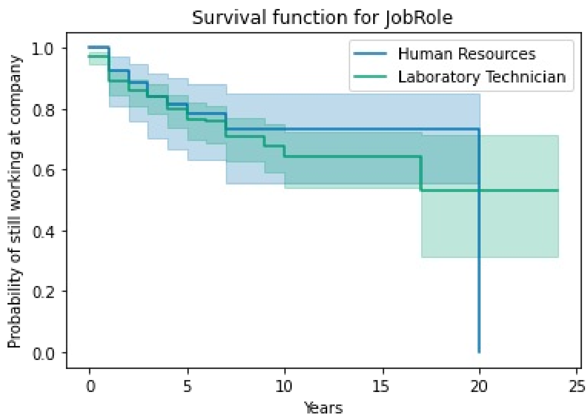
In [ ]:

```
build_survival_plot('StockOptionLevel', 0, 3)
```



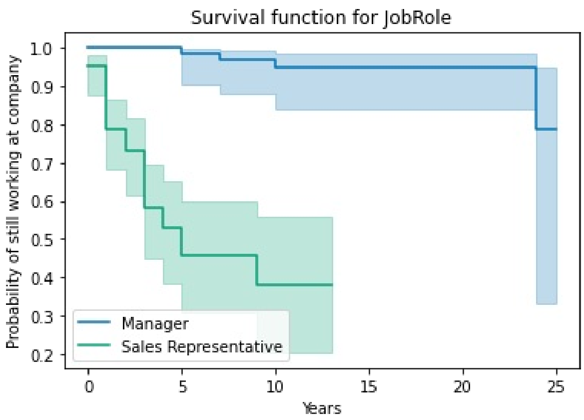
In [ ]:

```
build_survival_plot('JobRole', 'Human Resources','Laboratory Technician')
```



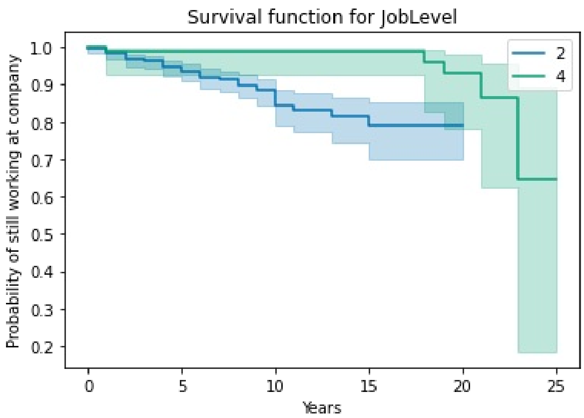
In [ ]:

```
build_survival_plot('JobRole', 'Manager','Sales Representative')
```



In [ ]:

```
build_survival_plot('JobLevel', 2, 4)
```



8. Summary

Despite this dataset being fictional, most of the metrics are record by companies and would not be to obtain this data. From our benchmark score of 83.88 we where able to predict with 90 percent accuracy.

The most important attributes influencing attrition is:

- **MonthlyIncome**: People with higher wages are less likely to leave. Maybe get some industry benchmarks to provide competitive wages.
- **OverTime**: with more overtime will more people leave. looking at project scope and ensure the is qualified manpower so the overtime is reduced.
- **Age**: Younger people between 25-35 are more likely to leave. Having a long-term strategy for the company where young employee fits in would be the best. Also a clear path for promotions.
- **DistanceFromHome**:Employees with a long distance from work are more likely to leave.
- **TotalWorkingYears**: More experienced workers will be there longer. people with 5-8 years of experience can have a higher risk of leaving.
- **YearsAtCompany**: Are employees about to hit 2 year anniversary, then there is high-risk of leaving. Being a loyal company you will keep people longer.
- **YearsWithCurrentManager**:

A way further would be to use models that deals with class inbalance better.

Cost funtion

From the cost function we can see if we should do nothing about our employee turnover or try to fix it. Using a more accurate estimate for how much it will cost to replace an employee.

A bonus program could solve some problems, but eve

Survival function

example: The survival curves show how employees without Overtime have less probability of attrition at each time point.

Retetion plan

When a company generate more data on employees, both new joins and recent leavers. It is possible to retrain to get a more accurate predictions for **high-risk employees**. they could be assigned a *risk category* based on the predicted label.

- **Low-risk** Label < 0.6
- **Medium-risk** Label 0.6 >=<= 0.8
- **High-risk** Label > 0.8

Having a plan for each group is important, discussing the work enviornment with manager to indentify steps to improve the situation.

End Project

A6 Plotting similar pattens

Added as html file A6