Table of Contents

1. Introduction & Problem

This report discusses the process of developing a predictive model that can accurately predict whether an employee is likely to leave our client's company. Beyond providing a prediction, through a full analysis of the model's components and conclusions, suggestions and points of improvement will be suggested as per what motivates employees to leave. The ultimate goal is to potentially prevent attrition by understanding the factors that contribute to it.

To achieve this, the analysis and model selection will be performed on the provided training set, while the test set will only be used for the final estimation of model performance. This allows for a realistic scenario to produce a complex model that is able to grasp some insights into what drives high levels of attrition while not overfitting on the available data.

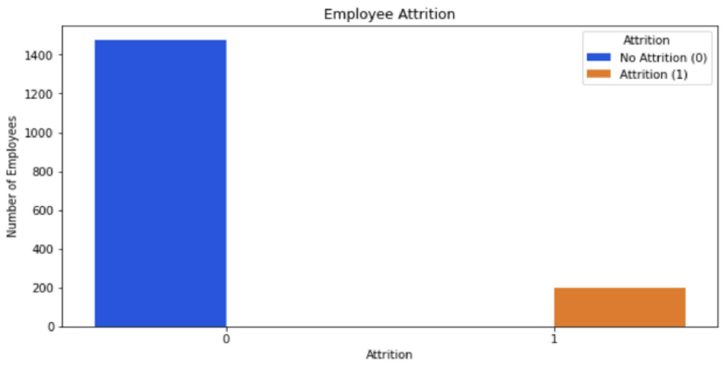
The steps taken to tackle this as well as the thought process that drove the decisions made are discussed below. The conclusion provides a summary of the final results and the final recommendations to our client.

2. Data Exploration

Data exploration is a critical step in the machine learning process as it lays the foundation for the development of the best and most effective model that can be used to make informed decisions and predictions. The main objective of data exploration is to gain a better understanding of the data and its underlying patterns, relationships, and characteristics. Likewise, data exploration helps uncover important features and trends that improve model development and decision-making.

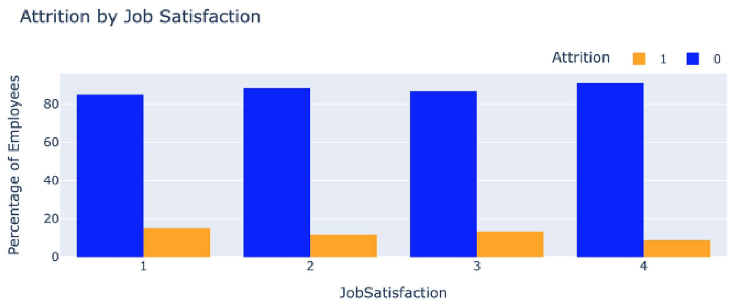
Our dataset is composed of 35 different features and 1677 rows. To start, a pair plot of the histogram of numerical features was made to be able to understand the distribution of the variables and the relationships between them **(Appendix 1)**. In this way, it's easier to identify potential issues and guide further exploration and analysis. This evidenced that there are some features that are right-skewed like *Years At Company*, *Distance from Home*, *Years in Current Role*, among others. Also, some features like *Job Satisfaction* and *Stock Option Level* are categorical features but have numerical data types. All of this suggests an initial idea of the **transformations** that need to be performed on the data for the modeling process.

The analysis of the data was done from a **business perspective**. The selected variables to do the in-depth analysis on were chosen based on an assumption of which ones could **affect attrition** (our target variable) the **most**. To begin with, a univariate analysis was made to comprehend how the attrition was distributed between the employees. This evidenced that from 1677 total employees, 1477 stayed in the company and 200 left which means that the data is **unbalanced**.



Graph 1. Attrition vs. Non-attrition

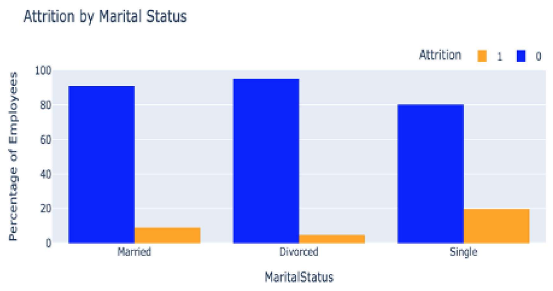
Next, a bivariate analysis was made to understand how two variables were related between each other. In this sense, the first comparison made was between *Job Satisfaction* and *Attrition*. From this, it can be concluded that the least satisfied employees are prone to turnover and those employees that have **high satisfaction** have a **lower probability** of leaving the company. These results are as expected and illustrate the relation there is between *Attrition* and *Job Satisfaction*.

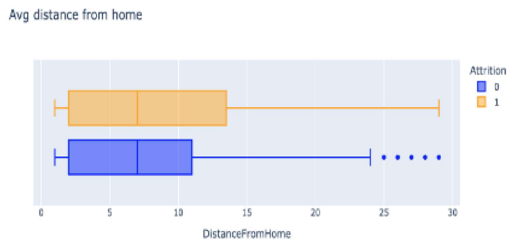


Graph 2. Attrition vs. Job Satisfaction

A comparison between *Marital Status* and *Attrition* was also made and it seems **single people** are more prone to leaving the company than those who are married or divorced **(Graph 3).** This can be related to different reasons, like married people are looking for more financial stability and singles are willing to take more risks. Still, this demonstrates that marital status has a relation to Attrition and explains how it is related.

Another variable that seemed to relate to attrition was the *average distance from home*. From analysis, it is possible to conclude that those employees that **live further from work** are more prone to leave the job **(Graph 4).** This could be because of the commuting time the employees have to undertake to get to the office which could affect their job satisfaction.

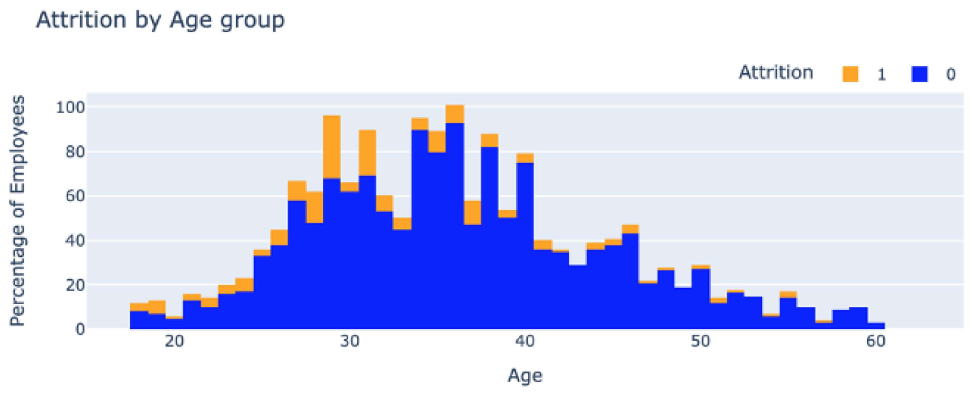




Graph 3. Attrition vs. Marital Status

Graph 4. Attrition vs. Distance from Home

The **age composition** of the company's employees can be of great value in analyzing attrition. To appreciate this, a comparison between age and attrition was made. This led to the understanding that most of the employees are between the ages of 25 and 40 years old. Along this, the ones that have higher attrition rates are **between 28 and 31 years old**, and as age increases, **attrition decreases**.

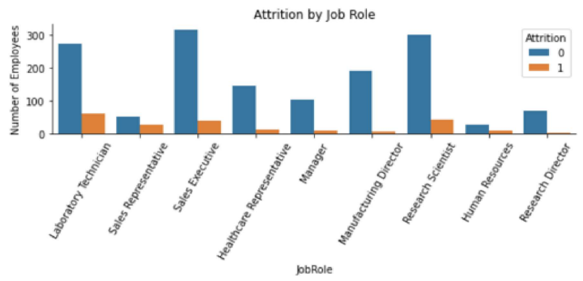


Graph 5. Attrition vs. Age

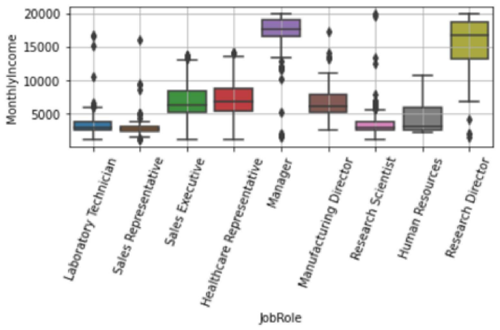
Some more bivariate analysis was made individually comparing *Attrition* against *Education*, *Total Working Years, Years At The Company, Years Since Last Promotion*, and *Years With The Current Manager* **(Appendix 2)**. Based on this analysis some conclusions were drawn in that those working in **marketing and human resources** have a **higher attrition rate**. Also, the employees that have between 1 and 10 **total working years** have a high attrition rate, similar to the employees that have been between 1 and 10 years at the company. Likewise, those employees that had their last **promotion** 13 years ago had a high attrition rate and those who had their last promotion 10 and 12 years ago had no attrition. Finally, employees that have been 0 or 1 years with their **current manager** had the highest attrition rate.

In addition to this, Attrition was also compared against *Overtime, Stock Option Level,* and *Years in Current Role* **(Appendix 3)**. The distribution of the employees that work overtime vs not overtime is balanced. As expected, it can be concluded that those that actually work overtime are more prone to leave the company. Also, considering *Stock Option Level*, the employees with fewer stocks have a higher attrition rate than those with more stocks. Finally, the employees that have been fewer years in their current role have a higher turnover rate.

On the other hand, a multivariate analysis was made comparing the relation between *Job Role, Monthly Income,* and *Attrition*. The Laboratory Technicians, Sales Representatives and Research Scientists have the highest attrition rates while Managers and Research Directors have the lowest ones **(Graph 6).** At the same time, the roles that have the higher monthly income are Managers and Research Directors while Laboratory Technicians, Sales Representatives, and Research Scientists have the lowest monthly income **(Graph 7).** With this, it is possible to draw the conclusion that *monthly income* does have a relation with attrition.



Graph 6. Attrition vs. Job Role



Graph 7. Monthly Income vs. Job Role

3. Feature Engineering

Before building the model, it is crucial to perform feature engineering since the performance of machine learning algorithms is significantly influenced by the **quality of features** used as input to the model. A comprehensive analysis of the dataset was performed, from simple analysis to encoding techniques that are tailored to the specific characteristics of each column. This was focused on identifying **relevant features** and **transforming raw data** into **meaningful representations** that can be used by each machine learning algorithm for accurate predictions.

3.1 Checking Missing Values

Checking for null values is a crucial step before going further, as it can significantly impact the accuracy and reliability of the results obtained. If these values are not handled appropriately, there can be a bias or incorrect conclusions can be made, leading to errors in the process. Fortunately, in the current dataset under investigation, **no null values were found.** The absence of missing values eliminates the need for imputation or removal of missing data.

3.2 Label Encoding for Random Forest & XGBoost

There are several columns (BusinessTravel, Department, EducationField, Gender, JobRole, MaritalStatus, Over18, Overtime) that have an object data type, meaning they are non-numeric features. In order to incorporate these variables into the model, there is a need to convert them into numerical values. Among various methods to accomplish this, **label encoding** was the one used. This decision was made based on several factors, including the simplicity of the method and the fact that it **does not increase the number of features** or the **computational complexity** of the model. When the Random Forest and XGBoost models are trained later on, label encoding provides a suitable approach for converting categorical features into numerical features.

3.3 Summary Statistics with ProfileReport

The ProfileReport library was utilized to conduct a more thorough analysis of the data in order to improve the accuracy and quality of the data engineering. It provides a comprehensive summary of the data including statistics such as mean, standard deviation, quartiles, visualizations of distributions, correlations between variables, and identification of outliers. This allowed us to quickly understand data quality issues, patterns, and trends in the data, which could help guide further data processing and modeling decisions.

3.4 Dropping Columns

After examining the ProfileReport, certain features from the dataset were eliminated. Specifically, the three constants that were removed were *StandardHour*, *EmployeeCount*, and *Over18*, as they did not provide any useful information for the analysis. Keeping constant variables in the dataset may even lead to overfitting, where the model learns to fit noise, which can lead to poor performance on the result. Additionally, certain features that were highly correlated with each other and with some of the other features were identified, such as *JobLevel,* *YearsAtCompany*, and *YearWithCurrManager*. To avoid multicollinearity issues, these features were removed as well. In a similar manner, *ID* was removed because it is typically used as a unique identifier and does not provide any relevant information for modeling. The team experimented with removing and including a mixture of the correlated features and this combination tended to yield the best results.

3.5 Clipping Outliers

Some unusual and extreme outliers in the *DailyRate* feature were identified. These outliers can have a significant impact on the data distribution and analysis, potentially leading to biased or inaccurate results. Therefore, outliers were clipped based on the interquartile range (IQR) method. In this case, there was only a single outlier, however if the model were to be put into production it is important to ensure no new outliers influence future runs of the model.

3.6 Dealing with Skewed Data

Because the *MonthlyIncome* feature is heavily skewed to the right, a log transformation was tried to reduce the skewness and improve the performance of the results. However, the log transformation of this feature was not used in the final model because it did not contribute to improving the model performance.

3.7 One Hot Encoding, PCA, and Logistic Regression Trial

In order to address potential overfitting, we planned to also apply a **simpler model** as part of the analysis. With a relatively small amount of data in this case, a linear model may be more appropriate. Additionally, some challenges were encountered in previous attempts to improve the predictions using multiple variants of the Random Forest.

For that purpose, we created a re-encoded version of the data that could be used by a linear model later on. These types of models assume a linear relationship between input features and output variables. When using label encoding, numerical values may not reflect the actual relationship between categories and the outcome. This can result in suboptimal model performance. Moreover, label encoding implies a distance between categories, which may not always be meaningful.

Instead, **one hot encoding** was used to compensate for the potential limitations of label encoding for linear models. One hot encoding creates a binary variable for each category, avoiding the issues of ordering and distance implied by label encoding. This approach allows the model to treat each category as a separate feature with a weight that can be learned independently. Therefore, all categorical features were transformed (*BusinessTravel, Department, EducationField, Gender, JobRole, MaritalStatus, Over18, Overtime*) into numerical values by using one hot encoding. In order to alleviate the issue of high dimensionality resulting from the encoding, the columns were reviewed and the original column of those that were encoded was eliminated.

Finally, when creating a Logistic Regression model, we did not achieve a better performance than with the tree-based models created. Thus, we discarded the model again and focused on further optimizing the tree-based models instead.

4. Model Building & Training

4.1 Approach Outline

Given the context of the **classification problem** and the provided dataset, the two models that immediately come to mind are **Random Forest** and **Logistic Regression**, due to their simplicity of use and understanding. On closer analysis, the sensible starting point seems to be Random Forest which not only requires **fewer transformations**, due to its robustness to dealing with outliers and correlated features as well as its insensitivity to the scale of features. Beyond these characteristics that simplify the modeling process, Random Forest's ability to deal with high-dimensional and imbalanced data makes it a promising fit for this problem.

Beyond this, **XGBoost** was experimented as well, as XGBoost tends to have lower bias and higher variance than Random Forest, meaning it can better capture complex relationships in the data. However, XGBoost is more prone to overfitting, and that is why experimentation with a version where a **mix of XGBoost and Random Forest** models were averaged to yield the best result. The results of these two models also provided very different feature importances, which can be interpreted as a positive sign, as they value and understand **different relationships** that can help predict the target data.

To add, we also tried testing out a Logistic Regression supported by PCA in the feature engineering to see if it had improved results. The results, while relatively solid, were **not strong enough to warrant the loss in the explainability** of the model. Explainability in a case like this one where a company would want to understand why an employee is attriting is extremely important. Losing this factor meant that the Logistic Regression and PCA could not be part of our final model and why a **decision tree** of some sort would be the better option.

4.2 Checking Correlations

Random Forest is robust to deal with correlation, as it selects a subset of features randomly for each decision tree, reducing the impact of correlated features. It is, however, still important to check for highly correlated features, as their inclusion in the model might lead to overfitting, where the model is too complex and fits the noise in the data rather than the underlying pattern.

Moreover, highly correlated features can also affect the interpretability of the model, as the feature importance might be split between them, and not reflect the real importance distribution.

In our dataset, there are three features that immediately stand out, due to them being constant among all observations (*EmployeeCount, StandardHours, Over18*). The first two seem to be general company information that should not be associated with any specific employee, while the third one does not promote any new information as it is just a boolean analysis of the age parameter of each employee. And in this case, it quite obviously shows all employees as being above 18.

The *JobLevel* feature is very highly correlated with *MonthlyIncome* as well as significantly correlated with *TotalWorkingYears*. The features: *YearsAtCompany*, *YearsInCurrentRole*, *YearsSinceLastPromotion* and *YearsWithCurrentManager* are not only moderately correlated with *JobLevel* but also significantly correlated with each other.

Given the amount of data provided regarding an employee's stay at the company, as well as due to the high correlation with *MonthlyIncome*, *JobLevel* doesn't seem to be providing any particularly valuable information and should be dropped from further analysis.

The first attempt at building a model can be run with all the other features being included, however, there is room for experimentation and possible improvements when choosing which features to remove based on correlation analysis.

4.3 Dealing with Imbalance

The dataset provided is highly imbalanced when it comes to the target variable. Only about 12% of employees have been recorded as positive values for attrition. It is important to consider this during the modeling process, as even a model that never predicted any positive values for attrition would still be correct 88% of the time.

4.4 Bagging

Bagging is a machine-learning technique that involves creating **multiple independent subs**ets of the dataset by random sampling. These subsets are then used to train multiple models, and the results are aggregated to produce a final prediction. Bagging is used when there is an **imbalanced dataset** because it helps to balance the **distribution** of the classes, making it easier for the model to learn the patterns in both the majority and minority classes.

The first step was to determine the size of each training set, which in this case is set to three times the size of the minority class, allowing for the new class distribution to be 2:1 instead of the 25:3 ratio in the original dataset. Then, multiple independent subsets of the majority class are sampled and concatenated with the minority class to create multiple training sets.

These training sets are then used to train multiple models, whose ROC-AUC scores are averaged, producing a final score.

4.5 Scoring with ROC-AUC Metric

ROC curve is a graphical representation of the performance of a binary classification algorithm, which plots the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. This metric is commonly used in **imbalanced datasets**, as it is less sensitive to changes in the class distribution than other metrics such as accuracy or precision, focusing on the **percentual tradeoff** between TP and FP instead of raw number of correct or incorrect predictions. Overall, the ROC-AUC metric provides a more comprehensive view of the model's ability to distinguish between positive and negative instances.

4.6 Hyperparameter Tuning

Hyperparameter tuning is the process of finding the best **combination of hyperparameters** for a machine learning algorithm to optimize its performance on a given dataset. There are two methods for finding this optimal combination, *GridSearch* and *RandomSearch*. Grid search explores all possible combinations of hyperparameters while random search randomly samples hyperparameters within a defined range. Grid search guarantees finding the optimal combination but is computationally expensive, while random search is less expensive but does not guarantee finding the optimal combination.

The approach taken in this scenario was a Grid Search over a small set of parameter values so as to not increase the computational complexity of finding a satisfactory combination of parameters. The values were based around the **default values** provided, and adjusted to ensure that both **upper and lower hyperparameter values** were explored.

The parameters where the grid search was performed were the following:

bootstrap: [True, False]

This parameter controls whether bootstrap samples are used to build the trees. When bootstrap is set to True, the algorithm will sample the data at random with replacement when building each tree. Because each tree is built with a different subset of the data, this can help avoid overfitting. In **small** datasets, however, setting bootstrap to False may produce better results because using the entire dataset to build each tree may help improve the model's accuracy.

max_depth: [10, 12, 16, 18, 21]

The maximum depth of each tree in the forest is determined by this parameter. A deeper tree can capture more complex variable interactions, but it can also overfit the data. A lower value for max depth can help prevent **overfitting**, but it can also lead to **underfitting** if the model is too simple. The optimal value of max depth is determined by the data's complexity and the number of features.

n_estimators: [100, 200, 500]

The number of trees in the forest is determined by this parameter. Increasing the number of trees can improve model accuracy while also increasing computational costs. In general, it's best to start with fewer trees and **gradually increase** the number until the model's performance plateaus.

min_samples_leaf: [2, 5]

This parameter specifies the minimum number of samples that must be present at a leaf node. A lower value makes the model more adaptable and sensitive to noise, but it can also overfit the data. A higher value can make the model more robust to noise, but it can also result in underfitting if the model is overly simple.

max_features: ['sqrt']

This parameter specifies the maximum number of features to take into account when splitting a node. When max features is set to sqrt, the algorithm will choose the square root of the total number of features at each split at random. This can help to reduce the model's variance and improve its **generalization performance**.

5. Final Model & Results

Overall, the final model selected was one that combined several elements from all those mentioned above. Firstly, it uses a **random forest model** that is augmented by a **bagging** process to better balance the data. The way this works is we split our training data into 6 smaller models, where each model is made up of the same employees with attrition = 1, coupled with a random subset (2 times in size) of employees where attrition = 0. We ensure that for the subset where attrition = 0, it is **only used once** and **cannot be present** in any other bag. This strategy now ensures each bag is more evenly balanced to make sure the model is not overly predicting 0s.

A **grid search** is then conducted on each **individual bag** to tune each individual model. This process is then looped six times and yields 6 different predictions that are tuned and trained on a different piece of the rebalanced data. We then take the mean result of all 6 predictions, in order to obtain a single probability per individual.

This same exact process is then **repeated and performed using an XGBoost** model. Each individual bag is also tuned with a grid search run with XGB parameters and fitted to each model.

We then combine the results of both models by taking the mean of both results in order to obtain a **single probability** for each row (in this case per person).

Lastly, since we are predicting binary results, we developed a loop to help in converting our combined probabilities into 1s and 0s. The loop selects the optimal threshold of probabilities by finding the threshold that maximizes our AUC-ROC score. In the internal testing on our validation dataset we obtained an AUC-ROC score of **just over 78%** which is the highest of all of our models attempted.

With that testing result, we felt confident to test the model on the official test data we had set aside. If our model predicted well on this data set, we felt ready to deploy the model for the business. Thankfully we scored our highest result on this dataset with a final score of just **over 79%**.

6. Conclusion & Recommendation

We were tasked to create an employee attrition model in order to predict which employees might be at risk of leaving the company and in order to understand which factors contribute to employee attrition. By creating a classification model that combines two machine learning algorithms, we were able to create a model that predicts the **probability of resignment** with **79% accuracy** (as measured by ROC-AUC score). Our model predicted that out of the 1120 employees in the final dataset provided, **312 would resign,** given the **ideal threshold of 0.52** found during the training and testing phase.

Next, in order to understand the most important factors influencing whether these employees were predicted to resign, we first looked at the **feature importance** provided by the model. We then **combined** these insights with the data on the **312 people** that were predicted to resign in order to understand how these features were related to attrition. Finally, we deducted the following key insights that will form the basis for our final recommendation:

1. **Payment Structure:** Employees with **monthly income** between **$2.5k and $5.0k** and **no / low stock options** in their contract are **more likely to resign**
2. **Junior Attrition:** **Single, young** employees between **ages 25-30** with **fewer than 5 total working years** and **less than 2.5 years** in their **current role** are **more likely to resign**
3. **Middle Management Attrition:** Senior employees with around **10 years of experience** and around **7.5 years** in their **current role** are **more likely to resign**

Our analysis has shown that there could be a general dissatisfaction with the current payment structure, both regarding salaries and stock options. This is most relevant for junior profiles that have been at the company for fewer than 2.5 years, but also for middle managers. Both profiles probably feel like they are stagnating in their roles either because of unclear responsibilities or missing advancement opportunities. For junior positions, it should probably be a general review of the current conditions whereas the management level should be addressed in a more personal and case-by-case basis. In general, the 312 people identified as being highly likely to retire should be prioritized in the review process. However, the following recommendations are applicable to all employees in order to ensure that attrition risk will be reduced in the long run.

For junior roles, we recommend the next steps:

1. Review **payment structure** for junior positions
2. Potentially raise **salaries**, improve **stock option conditions** or **overtime payment**
3. Critically review **roles** and **promotion** possibilities for **high-performers**

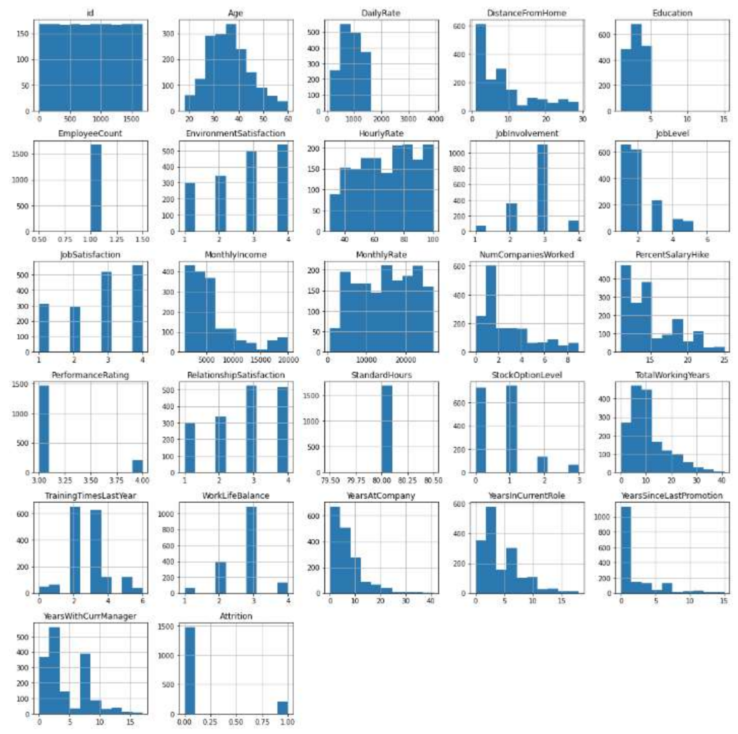
For middle manager roles, we recommend the following:

1. Review **role profiles** of middle managers on a **case-by-case basis**
2. Conduct **dedicated interviews** to understand job **satisfaction** and **involvement** levels
3. Discuss **potential role changes** with dissatisfied managers (vertically and horizontally)

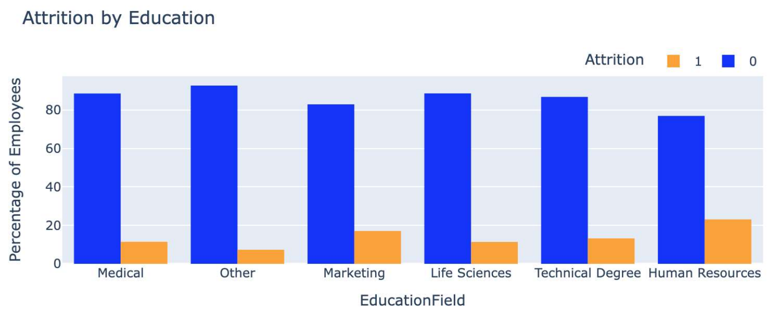
By applying these recommendations in the short term and also incorporating our insights into what is important to employees into the long-term HR and organizational strategy of the company, we are confident that our client can sustainably reduce its attrition rate in the future. By using the model **regularly** to predict attrition, the company can spot potential resignations **early on** and start understanding the needs and expectations of these employees more dedicated and quickly. With new data, the company will also be able to observe changes in **employee expectations** overall as different features might become more important for attrition in the future.

Appendix

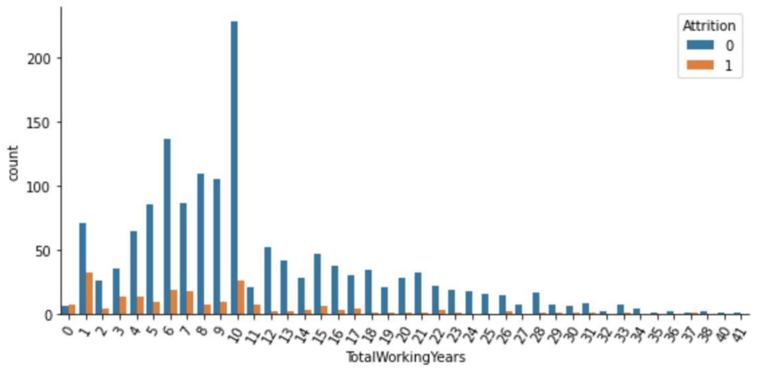
Appendix 1 - Pairplots



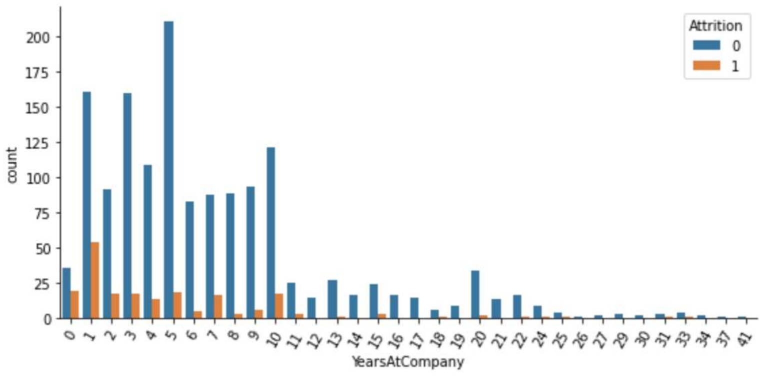
Appendix 2 - Attrition x Features



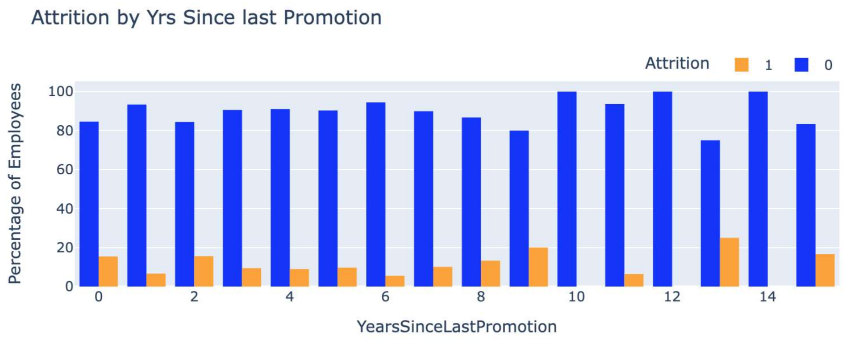
Attrition vs. Education



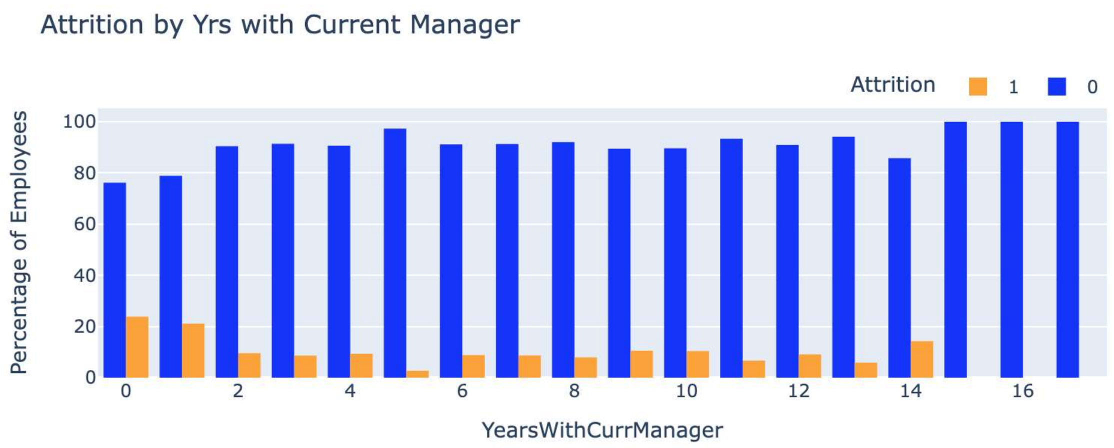
Attrition vs. Total Working Years



Attrition vs. Years At The Company

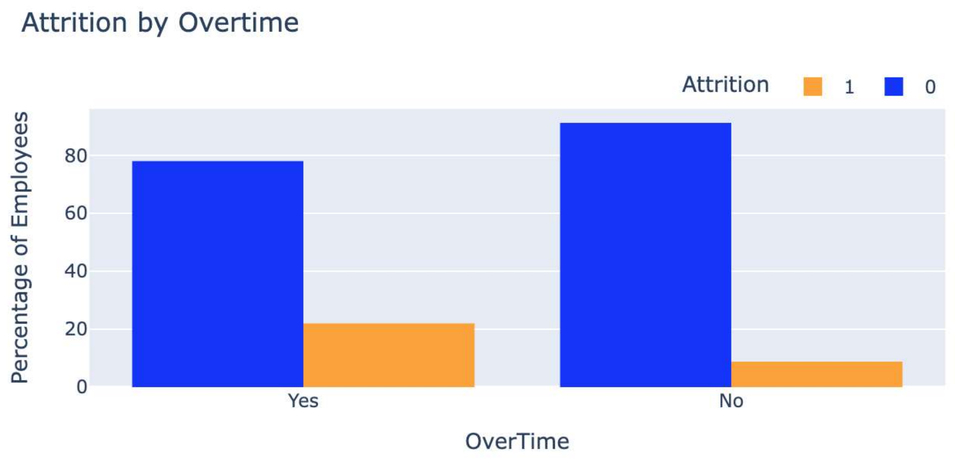


Attrition vs. Years Since Last promotion

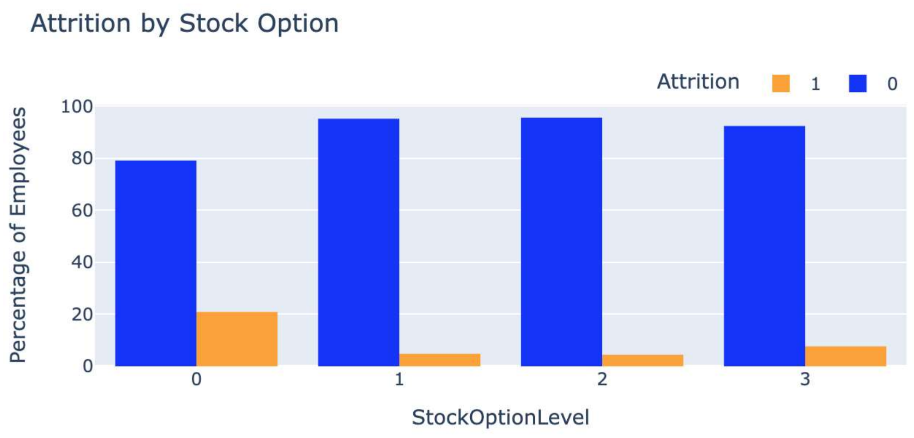


Attrition vs. Years With Current Manager

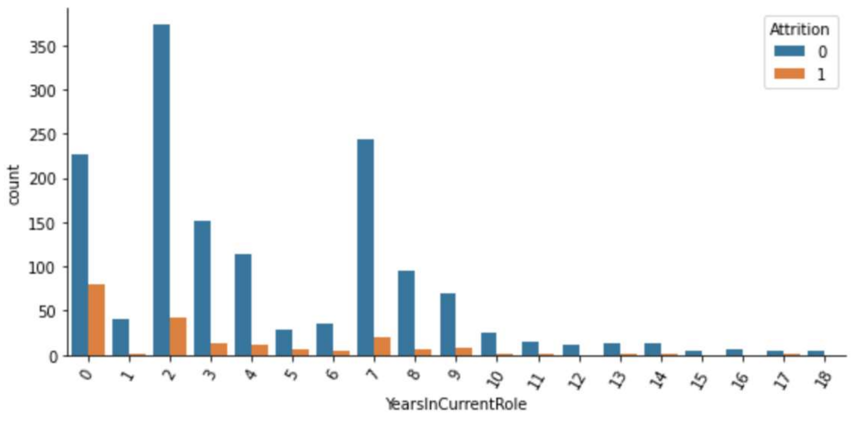
Appendix 3



Attrition vs. Overtime



Attrition vs. Stock Option Level



Attrition vs. Years in Current Role