


















 Julez89 / ds\_phase\_5



 Code  Issues  Pull requests  Actions  Projects  Wiki  Security  Insights 



☆ 0 stars  0 forks  1 watching  Activity

 Public repository

 main ▾

...

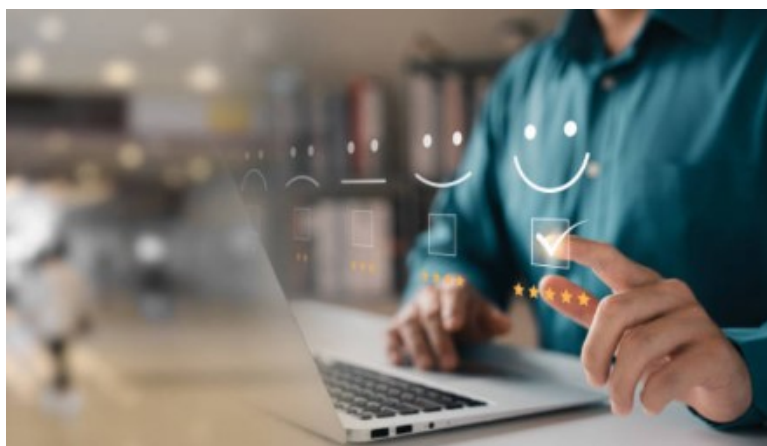
 Branches  Tags

Julez89 final changes ...

1 minute ago ⌚ 14

[View code](#) README.md 

# Predicting employee attrition



## Project Overview

For this project, I tried different classification methods to predict attrition in employees based on a data set from Kaggle. My final model uses a random forest classifier after addressing the class imbalance by applying SMOTE (Synthetic Minority Oversampling TEchnique)

## Business problem & stakeholder

IBM is struggling with a tough recruiting market where it takes a long time to fill vacancies. For some positions, they don't find any applicants at all. Therefore, it is even more important than ever to retain the internal talent to avoid higher workload for the other employees and high costs. To HR and the line managers, resignations often come as a surprise. Therefore, we can help by building an attrition classification model that can answer the following questions:

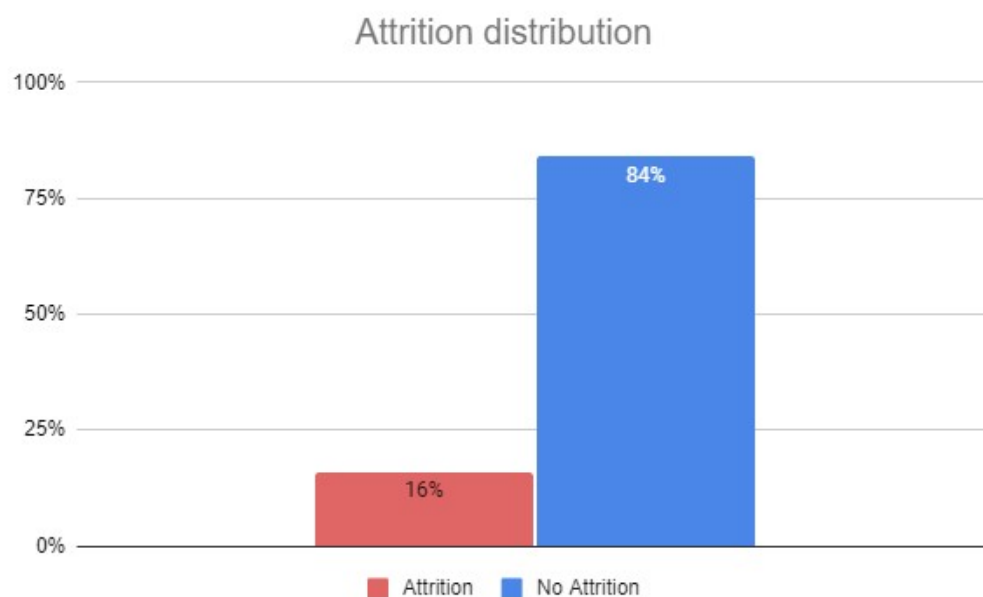
- Who of my employees is likely to leave the company?
- What are the most important features of previous resignations?
- What can we offer employees to stay?

## The data [↗](#)

This project uses a data set with around 1400 employees from Kaggle. The dataset includes 35 columns including information about:

- Demographics of the employee
- History within the company
- Performance
- Satisfaction score
- Position
- Attrition (yes vs. no) as the target variable

It's a binary classification problem (attrition yes vs. no) and we have a strong class imbalance. 16% of the employees are leaving and the other 84% are staying. This will be addressed by applying SMOTE during the process.



I've included instructions in the notebook on how to access the data from Kaggle. You will have to insert your Kaggle username and API Key to run the code.

## Approach [↗](#)

This task is a classification problem because my target variable is a categorical variable. It is either positive or negative for attrition. I used a machine learning approach to solve the classification task. Machine learning algorithms are designed to automatically identify patterns and relationships within data, allowing them to detect complex relationships that may not be immediately apparent through manual analysis. Also, machine learning algorithms can improve over time as they are trained on more data, providing an ongoing opportunity to refine and improve the accuracy of the classification model. Finally, they are able to handle large amounts of data efficiently and quickly, making them an ideal choice for tasks involving large datasets. I am using an iterative approach to try to finding the best model for my predictions. In summary, I took the following steps:

- Loading and inspecting the data
- Exploratory data analysis (looking at outliers, dropping features, visual data inspection, class imbalance, bivariate analysis and correlation analysis)
- Prepare data for modelling (train - test split, standardization, applying SMOTE)
- Start with a vanilla, simple logistic regression model
- Try different classification models (random forest, XGBoost)
- Try a more complex model with random forest classification and tune it
- Evaluate the final model (random forest with reduced threshold).

## Modeling [↗](#)

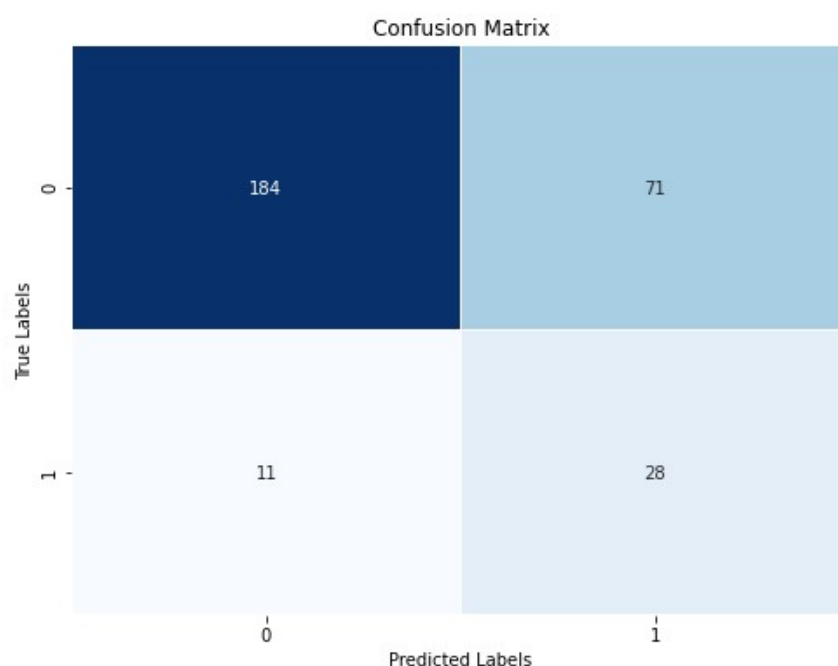
For my customer, the main priority is to be able to identify actual potential leavers, so that they can still change something about their decision. Therefore, I'm prioritizing recall and accept that I will have some false positives.

My preparation steps included dropping some columns, standardization of data, one-hot encoding and applying SMOTE. To follow an iterative approach, I am starting with a very simple, vanilla logistic regression model which already performed ok (weighted avg recall = 0.70) but it had a lower recall on class 1 (0.67).

As next model, I am trying a Random Forest model with the standard hyperparameters. This model was much worse (recall class 1 = 0.23). I tried the same model with lowered threshold (prioritizing recall and sacrificing precision) and improved the class 1 recall to 0.56.

Before looking at hyperparameter tuning, I tried a XGBoost Classifier which performed similar as the random forest model.

Next, I'm using GridSearchCV for both the logistic regression and the random forest model to decide which model works best with tuned hyperparameters. Even though there it is not perfect, I'm choosing the random forest model as it has the highest class 1 recall of 0.72. I am aware that the precision of class 1 (0.28) is not good but I am ok with this compromise.



## Evaluation

---

My final model is a tuned random forest model after standardization and SMOTE with specified parameters such as max depth, minimum sample split and the number of estimators as well as a lowered threshold. My final model has a weighted average recall of .71. My model successfully identifies about 72% of the employees who actually leave and does not miss too many people who leave. It identifies still too many people as potential leavers who don't have an intention to leave. We can improve the model by collecting more data.

## Conclusion

---

In times of a very difficult recruiting market where there are more vacancies than candidates, it is really important to retain your employees and focus on developing and reskilling them. Sometimes, it comes very unexpected when an employee resigns and it's especially painful when they are one of your top performers or if they resign after only a short time with the company. Therefore, we developed a classification model based on a dataset from Kaggle that can help in flagging if an employee is likely to leave or not.

The model focuses on correctly identifying those who really want to leave by accepting that it will classify some employees as potential leavers who don't really want to leave. It's recommended to have an HR employee or line managers evaluate the actual risk of leaving by talking to the employees.

As next steps, the model would be deployed in the HR department after collecting more data on leavers. Since the model is dealing with high class imbalance, the more information we gather about leavers, the better the model will perform. When deployed, we receive a probability for each employee that indicates their likelihood on leaving the company. Based on this information, a process needs to be defined on how to use this information.

In any case, we already know the most important factors (overtime, income, job satisfaction, work-life balance and stock options) so the company should review its policies on remote working opportunities, salary progression and other benefits.

## This repository

---

My technical code is stored in this [jupyter notebook](#)

My presentation can be found [here](#)

My github repository is [here](#)

A list of libraries and versions can be found [here](#)

## Releases

No releases published

[Create a new release](#)

## Packages

No packages published  
[Publish your first package](#)

---

## Languages

- Jupyter Notebook 100.0%