أكاديمية سدايا
SDAIA Academy

PREDICTING EMPLOYEES ATTRITION

Classification project report

A PROPOSAL BY
MUNEERA ALSHUNAIFI

DATE
28 OCTOBER 2021

Abstract

Employee attrition is one of the problems that faces some companies for long times. The Employee attrition occurs when an employee departs a company for a variety of reasons, however, Several companies lost their employees without an obvious reason, in which it affect negatively to the company's productivity and performance. In this project we will try to predict the appropriate reasons that led to the attrition by using machine learning classification algorithms:

- **Logistic regression:** Is a statistical model that in its basic form uses a logistic function to model a binary dependent variable
- **Decision Tree:** Is a non-parametric supervised model, its goal to create a model that predicts the value of target variable by learning simple decision rules inferred from data features
- **SVM:** support vectors are simply the coordinates of individual observation. It is a frontier that best segregates the two classes (hyperplane/line)
- **AdaBoost**: Is an ensemble learning model which was initially created to increase efficiency of binary classifier, it uses an iterative approach to learn from mistakes of weak classifiers and turn them to strong one

To evaluate the models we will use performance measures as Recall, Precision, F1-score and ROC AUC. The objective in this case is that all the employees predicted to be attritted are truly having the attrition decision, thus, We want to be as precise as possible so we will focus the most on the precision measure because a model that has high precision will make few mistakes predicting the positive label which we are focusing on.

Design

This project originates from the Data Science Bootcamp (T5) to predict an employee attrition by using multiple classification algorithms. The model developed will facilitate any company or organization in knowing what is the reasons that led to attrition decision.

Data

The data used is downloaded from Kaggle website. It is a fictional data set having an csv format. The dataset created by IBM data scientists and consist of 1470 rows and 35 columns.

Column Type	Description	Columns names
Numeric columns	Related to personal information	age, distance_from_home, employee_number (id variable)
	Related to income	hourly_rate, daily_rate, monthly_rate, monthly_income, percent_salary_hike
	Related to time in company	years_at_company, years_in_current_role, years_since_last_promotion, years_with_curr_manager, total_working_years
	Others	num_companies_worked, standard_hours(to delete), training_times_last_year, employee_count (to delete)
Categorical columns	Binary	Attrition (Target variable), gender, over18 (to delete), over_time
	Nominal	department, education_field, job_role, marital_status
	Ordinal	environment_satisfaction, job_satisfaction, relationship_satisfaction, work_life_balance,job_involvement,performance_rating business_travel, education, job_level, stock_option_level

Algorithms

Data Pre-Processing

1. Exploratory Data analysis

We explored and visualized the data by following the EDA process, we focused on exploring the relations between Attrition and other features in order to have future look to the modeling process. We used matplotlib, seaborn and Pandas profiling libraries to analyze and visualize also we used Power BI tool. However, figures below shows some of the analysis and visuals that we did.

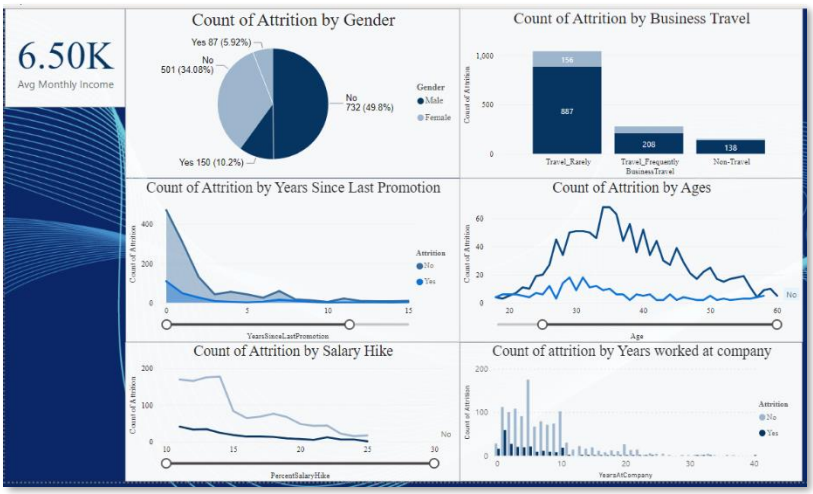

Figure 1 visualizing using Power BI

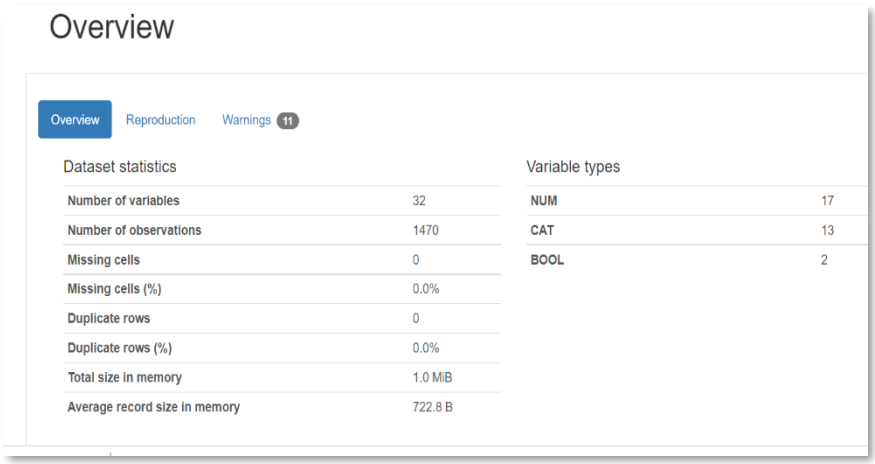

Figure 2 part of pandas profiling report

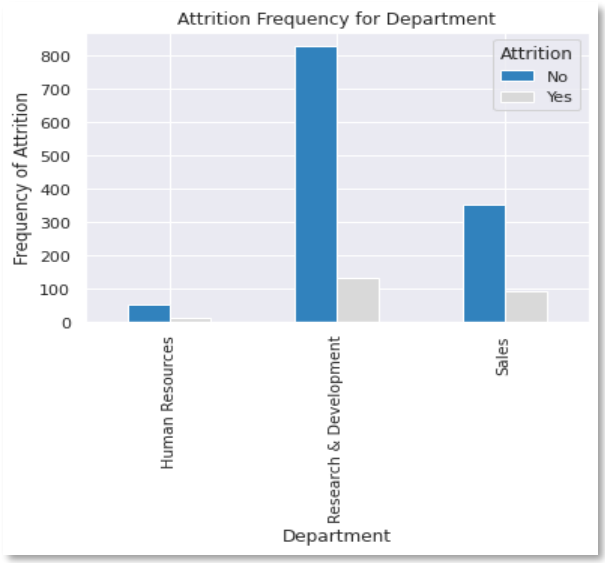

Figure 3 count of attrition for each department

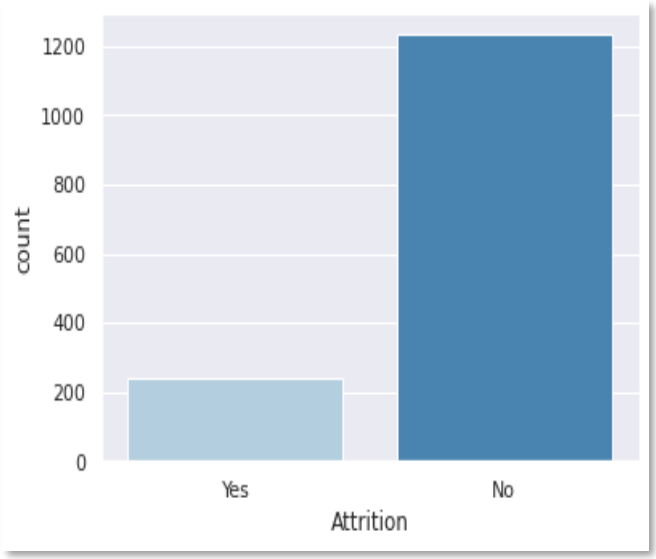

Figure 4 class imbalance for Attrition column

2. Feature Engineering

Firstly, we analyze the data and found that there is unnecessary features which are (EmployeeCount, Over18,StandardHours) because they have the same value for all observations. To continue, Our main target is the Attrition feature which is a binary column with the values Yes or No, so we convert this column to dummy variables in order to feed the model. Then we split the data into 70% train set and 30% test set and we defined the dependent and independent features. As for the dependent it was Attrition and for the independent it was whole dataset except the target.

3. Scaling

To standardize our train and test sets we using a standardization technique called MinMax scaler in which it subtracts the minimum value in the feature and then divides by the range. The range is the difference between the original maximum and original minimum. Also this technique preserves the shape of the original distribution.

4. Sampling

While we perform the EDA on the dataset we noticed that there is class imbalance in our target feature: Attrition, to handle the class imbalance we used a hybrid sampling technique: *SMOTE+ENN*. An ENN refers to Extended Nearest Neighbor and it is an algorithm for pattern recognition that predicts the pattern of an unknown test sample hinged on the highest gain of intraclass coherence. Moreover, Integrating this technique with oversampled data done by SMOTE helps in doing extensive data cleaning. So the misclassification by NN's samples from both the classes are removed.

Modeling and Evaluation

We perform 4 models: Logistic regression, SVM, Decision tree and AdaBoost. The inputs to all models was the whole dataset except the target feature. Furthermore, The goal in in this project is to ensure that all employees who are predicted to be attritioned are actually making the attrition decision. As a result, we will place the greatest emphasis on the precision metric because a model with high precision will make fewer mistakes when predicting the positive label that we are focusing on. In this stage we fitted the models with the data before and after sampling

1. Logistic regression confusion matrix

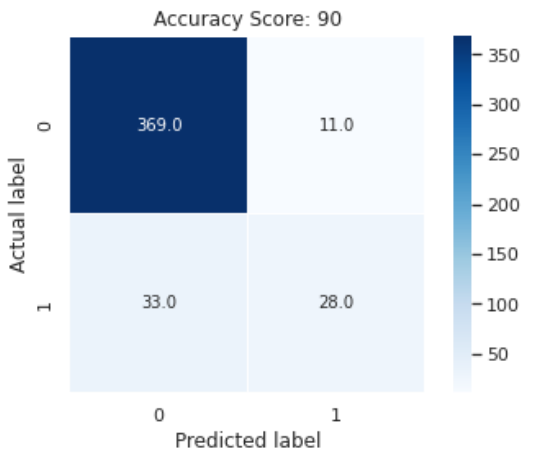

Figure 6 confusion matrix before sampling

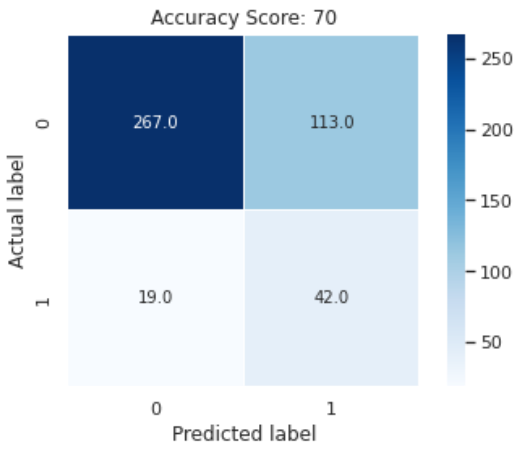

Figure 5 confusion matrix after sampling

2. Decision Tree confusion matrix

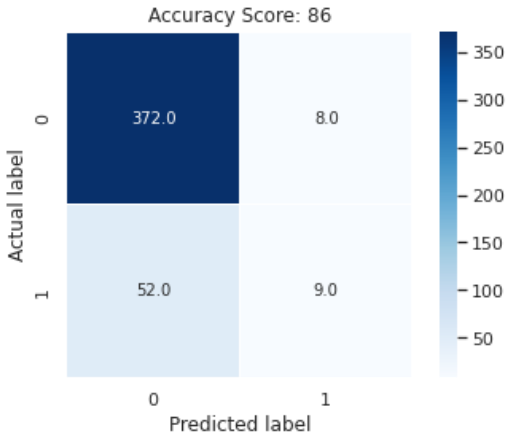

Figure 7 DT before sampling

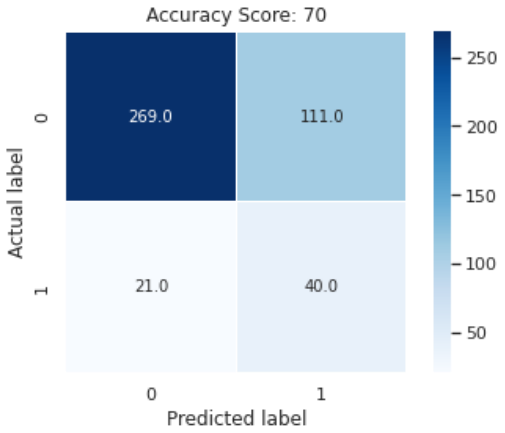

Figure 8 DT after sampling

3. SVM confusion matrix

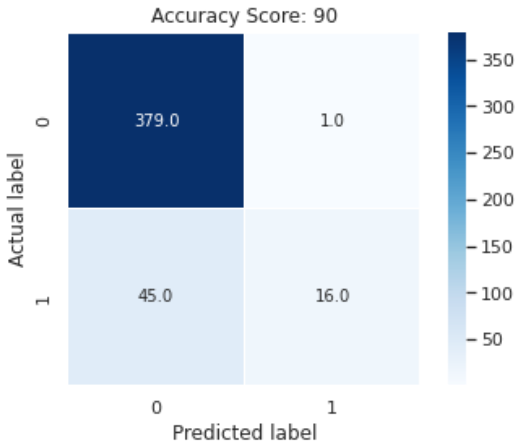

Figure 10 before sampling

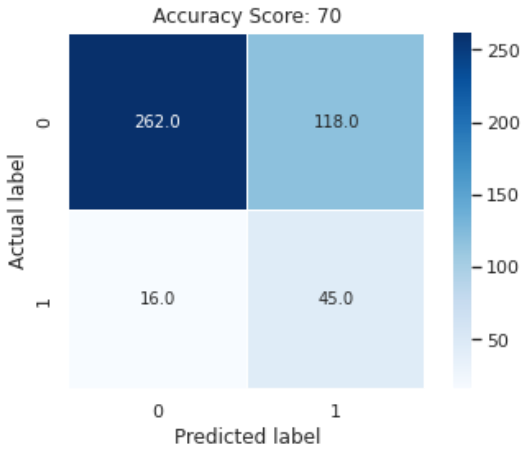

Figure 9 After sampling

4. AdaBoost classifier confusion matrix

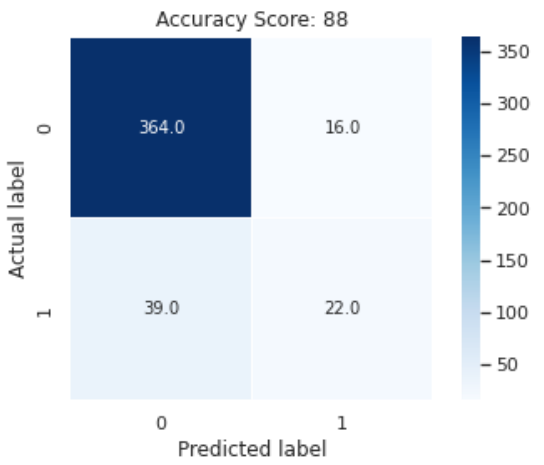

Figure 12 before sampling

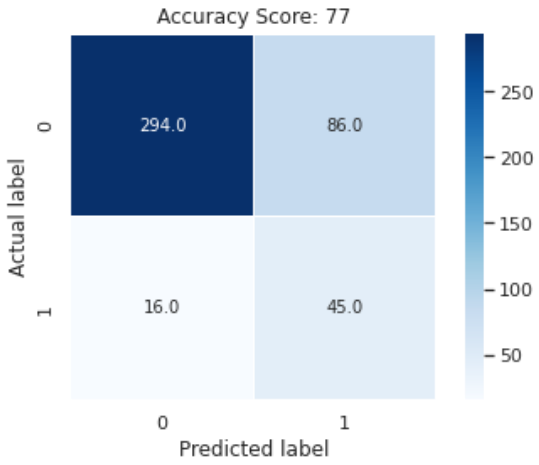

Figure 11 after sampling

Comparing the models

Model	Resample	Precision	Recall	F1-score
Logistic Regression	actual	0.717949	0.459016	0.560000
Logistic Regression	smote+enn	0.270968	0.688525	0.388889
Decision Tree	actual	0.529412	0.147541	0.230769
Decision Tree	smote+enn	0.264901	0.655738	0.377358
SVM	actual	0.941176	0.262295	0.410256
SVM	smote+enn	0.276074	0.737705	0.401786
AdaBoost	actual	0.578947	0.360656	0.444444
AdaBoost	smote+enn	0.343511	0.737705	0.468750

Figure 13 scores comparison

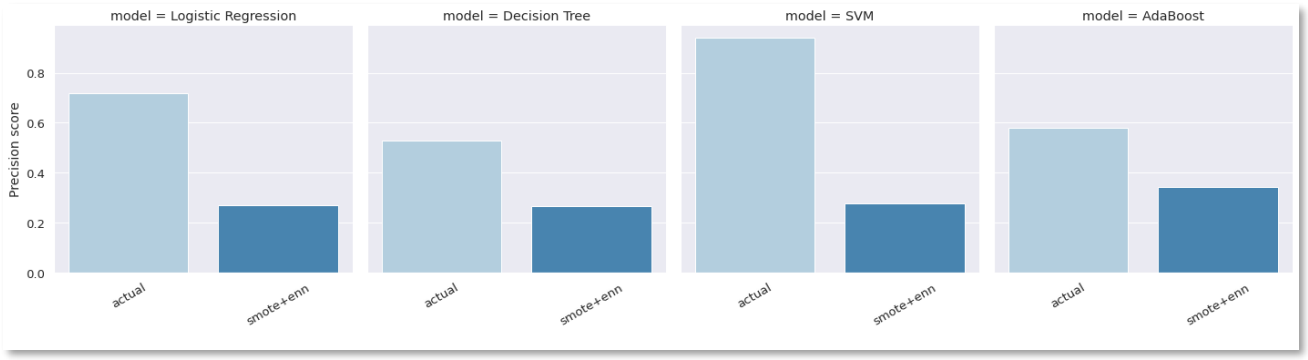

Figure 14 precision score for all models before and after sampling

We can notice from above figures that SVM before sampling has best precision score , however we cannot decide yet if it's the best model because comparing to other metrics scores we can notice a huge gap which may seems illogical. So we decided to optimize the SVM.

Optimization

We will use GridSearchCV to find the optimal parameter for the SVM model

GridSearchCV is a process of performing hyperparameter tuning in order to determine the optim al values for a given model. We define the parameter range as follows:

'C': [0.1, 1, 10, 100, 1000],

'gamma': [1, 0.1, 0.01, 0.001, 0.0001]

'kernel': ['rbf']

Then we inspect the best parameters found by GridSearchCV and it was:

'C': 1

'gamma': 1

'kernel': 'rbf'

The figures below shows the results after optimizing the SVM model and it seems more logical for the sampled data than before optimizing even though the precision score decreased for the data before sampling.

Model	Resample	Precision	Recall	F1-score
Logistic Regression	actual	0.717949	0.459016	0.560000
Logistic Regression	smote+enn	0.270968	0.688525	0.388889
Decision Tree	actual	0.529412	0.147541	0.230769
Decision Tree	smote+enn	0.264901	0.655738	0.377358
SVM	actual	0.941176	0.262295	0.410256
SVM	smote+enn	0.276074	0.737705	0.401786
AdaBoost	actual	0.578947	0.360656	0.444444
AdaBoost	smote+enn	0.343511	0.737705	0.468750
SVM-Optimized	actual	0.692308	0.442623	0.540000
SVM-Optimized	smote+enn	0.500000	0.311475	0.383838

Figure 15 scores comparison after optimizing SVM

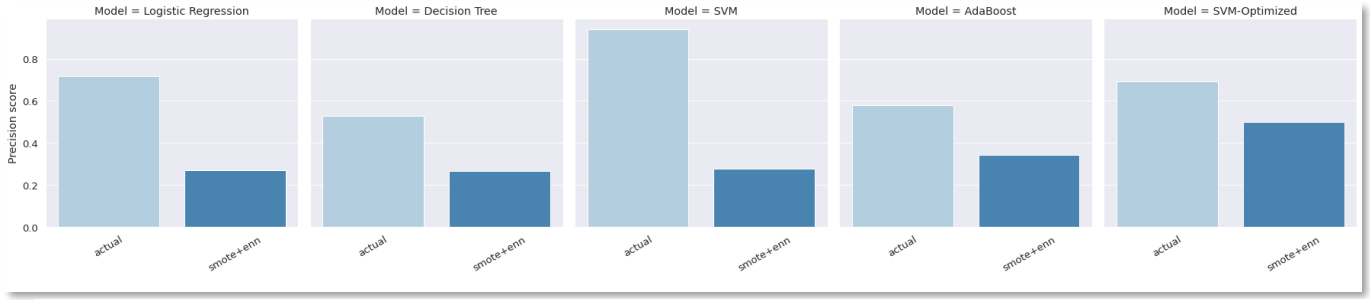

Figure 16 precision score for all models before and after sampling and after adding optimized SVM

To complete the models comparison we plotted the ROC-AUC

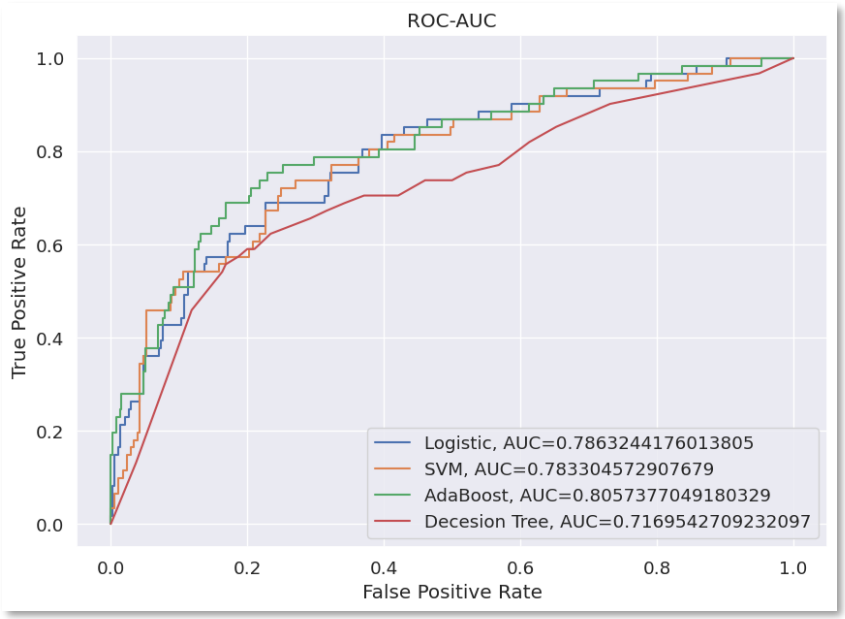

Figure 17 roc-auc plot

Result:

We can conclude from the evaluation that SVM after optimization and Logistic regression may be good models for our case also, since the AdaBoost classifier has the largest area under the curve among all models we can say that its also a good classifier.

Tools:

- Main language used for development is Python Language.
- Google colab is used for python code execution.

Python libraries, such as:

- Pandas and NumPy packages for data manipulation.
- Matplotlib, seaborn library for data visualization.
- Imblearn library for sampling methods
- Sklearn library for applying the classification algorithms and preprocessing steps.

Visualization and statistics:

- Power BI tool
- Pandas profiling library to generate report of the dataset.

Communication

- Presentation.
- GitHub
- Power BI Dashboard