Employee Attrition Analysis

Prateksha U
prateksha.u@iiitb.org
IMT2017517

Bishal Pandia
bishal.pandia@iiitb.org
IMT2017010

Lakshay Agarwal
lakshay.agarwal@iiitb.org
IMT2017026

***Abstract*—Every year around 15% of the employees of a company leave the company and need to be replaced. The management of the company believes that this level of attrition (employees leaving, either on their own or because they got fired) is bad for the company. Therefore to understand what factors they should focus on, we build a machine learning model in order to curb attrition.**

***Index Terms*—Machine Learning, Data pre-processing, Feature engineering, One-hot Encoding, Label Encoding, XGBoost classifier, SVM**

I. Introduction

Employees are the most valuable assets of an organization. It is they who add value to the organization in terms of quantity and quality as well. To find and retain the right talent is a major part of management. Therefore, it is indispensable to maintain a permanent and promising workforce; which over the years has become a tough task for employers and thereby increased attrition in the organizations. Attrition refers to the gradual loss of employees over time.

Attrition is a major problem which highlights in all the organizations because of the following reasons -

1) The former employees' projects get delayed, which makes it difficult to meet timelines, resulting in a reputation loss among consumers and partners.
2) A sizeable department has to be maintained, for the purposes of recruiting new talent.
3) More often than not, the new employees have to be trained for the job and/or given time to acclimatise themselves to the company.

We have implemented certain Machine Learning techniques to address the above issue.

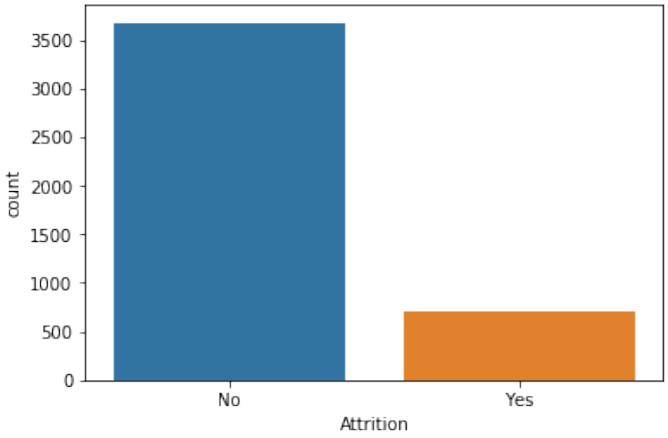
II. Data

The dataset used in the project is taken from one of the datasets provided on kaggle: Link to dataset. It consists of five csv files namely: employee_survey_data, general_data, in_time, manager_survey_data, out_time, and a data dictionary file.

1) The **employee_survey** file contains information about an employee's work environment satisfaction, job satisfaction and work life balance.
2) The **general_data** file contains information about an employee's work profile(work experience, background, etc).
3) The **in_time** file contains information about the time at which the employees enter the office on a particular day.
4) The **manager_survey** file contains information about the manager's job involvement and performance rating as given by his/her respective employees.
5) The **out_time** file contains information about the time at which the employees leave the office on each day.
6) The **data dictionary** contains the meta data (details of all the attributes which are present in the csv files).

The dataset contains the records of **4410** employees.

The Target variable (Attrition) is labeled as Yes/No. Yes represents that the employee left, either on his own or because he got fired and a No represents that he is still working with the company.



The above plot denotes that the dataset is highly unbalanced with respect to the target variable. The number of employees with attrition value 'Yes', 705 is very less compared to the number of employees with attrition value 'No', 3677.

III. Data Preprocessing

Data preprocessing is an important step to prepare the data for feeding it to any model. There are many important steps in data preprocessing, such as data cleaning, data transformation, and feature selection. Data cleaning and transformation are methods used to remove outliers and standardize the data so that they take a form that can be easily used to create a model. Feature selection on the other hand includes selecting various features (say using a correlation matrix) which can result in better predictions.

As a part of preprocessing, we have done the following:

1) **Handing missing data**

Fig. 1. Null values in the attributes of file general_data

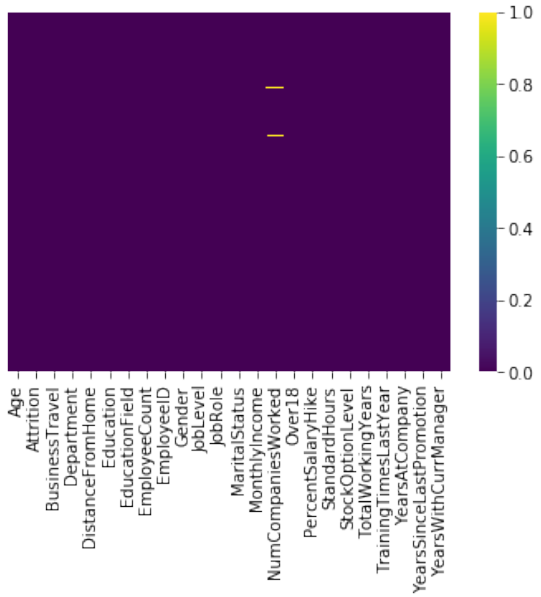


Fig 1. shows the attributes in the file general_data that have null values. we see that the columns NumCompaniesWorked and TotalWorkingYears contain null values in it. As the number of null values in each of these columns is only 19 and 9 each which accounts for 0.43% and 0.20% of the total dataset respectively, we decided to drop the records with these null values.

Fig. 2. Null values in the attributes of file employee_survey

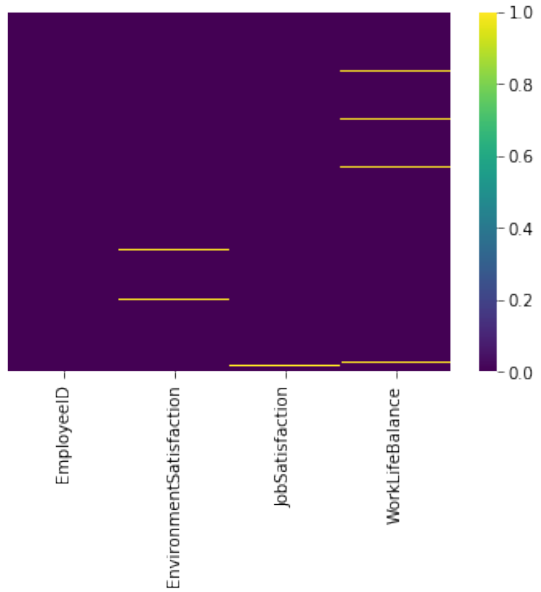


Fig 2. shows the attributes in the file employee_survey that have null values. we see that the columns EnvironmentSatisfaction, JobSatisfaction, WorkLifeBalance, all have null values present in them.

Fig. 3. Frequency of values occurring in EnvironmentSatisfaction

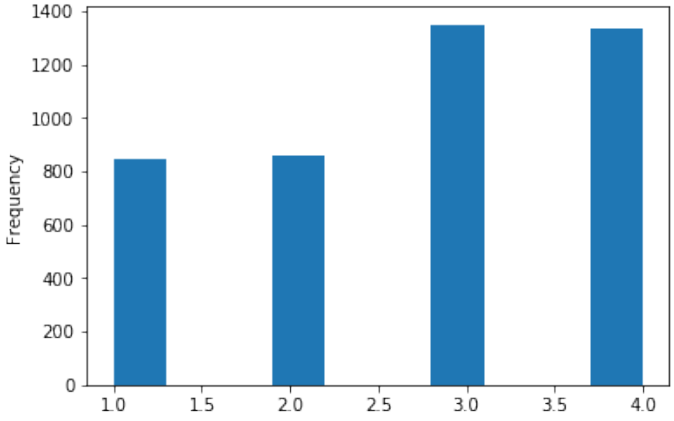


Fig 3. shows the frequency of the values occurring in the column EnvironmentSatisfaction. We see that the value 3 occurs the maximum number of times and is therefore the mode of this column. This will be a good representation for the records with null values because company's environment is common to all the employees. Therefore we fill the null values of this column with mode.

Fig. 4. Frequency of values occurring in JobSatisfaction

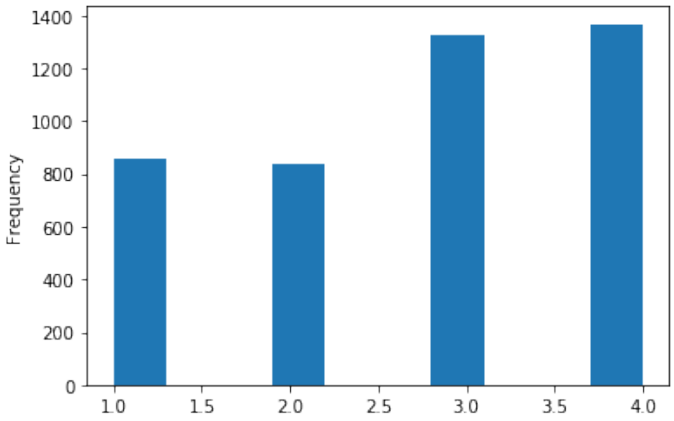


Fig 4. shows the frequency of the values occurring in the column JobSatisfaction. We see that the value 4 occurs the maximum number of times and is therefore the mode of this column. We consider this to be a good representation for all the employees of the company. Therefore we fill the null values of this column with the same.

Fig. 5. Frequency of values occurring in WorkLifeBalance

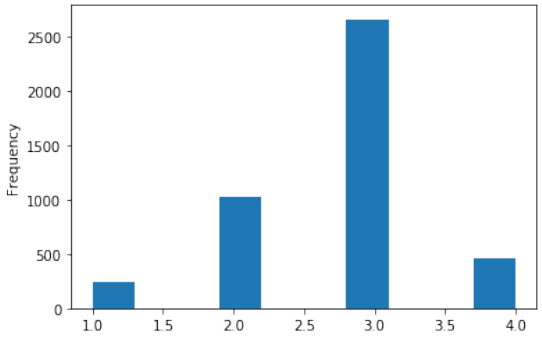


Fig 5. shows the frequency of the values occurring in the column WorkLifeBalance. We see that 75% of the count has the value 3 and is therefore the mode of this column. Therefore we fill the null values of this column with the same.

Fig. 6. Null values in the attributes of file manager_survey

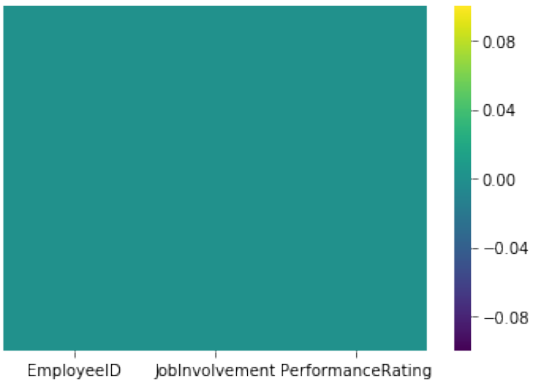


Fig 6. shows the attributes in the file manager_survey that have null values. We see that none of the columns have any null values present in them. Therefore this file doesn't require any missing data processing.

2) **Dropping columns**

Fig 7. shows the number of unique values in the columns of the combined dataset. We see that the number of unique values in the columns - EmployeeCount, Over18, StandardHours is 1. So, these columns are not necessary in the analysis as they do not convey any extra information about the employee. Therefore, we are drop these columns.

3) **Merging all the dataframes together**
After handing all the missing values and dropping the necessary columns of each of the seperate datafiles, we merge all the dataframes - employee_survey, general_data, manager_survey, processed in_time and out_time- into a single dataframe. Processed in_time and out_time dataframe is a dataframe which consists of some extracted information from the original in_time and out_time dataframe, which may help us in feature engineering.
After merging all the dataframes to a single dataframe based on the 'EmployeeId' field, we remove this field as it is just an index field and doesn't provide any information regarding attrition analysis.

Fig. 7. Number of unique values in each column

Number of unique values:

	Atrribute_Name	Num_Unique_Vals
0	Age	43
1	Attrition	2
2	BusinessTravel	3
3	Department	3
4	DistanceFromHome	29
5	Education	5
6	EducationField	6
7	EmployeeCount	1
8	EmployeeID	4410
9	Gender	2
10	JobLevel	5
11	JobRole	9
12	MaritalStatus	3
13	MonthlyIncome	1349
14	NumCompaniesWorked	10
15	Over18	1
16	PercentSalaryHike	15
17	StandardHours	1
18	StockOptionLevel	4
19	TotalWorkingYears	40
20	TrainingTimesLastYear	7
21	YearsAtCompany	37
22	YearsSinceLastPromotion	16
23	YearsWithCurrManager	18
24	EnvironmentSatisfaction	4
25	JobSatisfaction	4
26	WorkLifeBalance	4
27	JobInvolvement	4
28	PerformanceRating	2

4) **One-hot Encoding for all the categorical features**
The merged dataframe has many attributes which consists of categorical data, namely- 'Department','EducationField','Gender','BusinessTravel', 'JobRole', 'MaritalStatus'.
Many machine learning algorithms cannot work with categorical data directly. The categories must be converted into numbers. We chose One-hot encoding as it allows the representation of categorical data to be more expressive.

5) **Label Encoding for Attrition column**
The Attrition column consists of Yes/No. To convert this to numerical data we use a label encoder which changes Yes/No to 1/0 respectively.

6) **Normalization**

We have normalised the dataset as we have features which contain data values in different ranges. For eg - Age ranges from 18 to 60 and monthly income ranges from 10,090 to 1,99,990.
We have used MinMaxScaler to normalize the dataset.

IV. FEATURE ENGINEERING

Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work. The new attributes we introduced are -

1) **No_of_leaves :** We calculated the number of leaves taken by each employee over an year using the in_time and out_time dataframes. We have interpreted that if there is no information about both the in_time and out_time of an employee on a particular day, then the employee has taken a leave on that day.

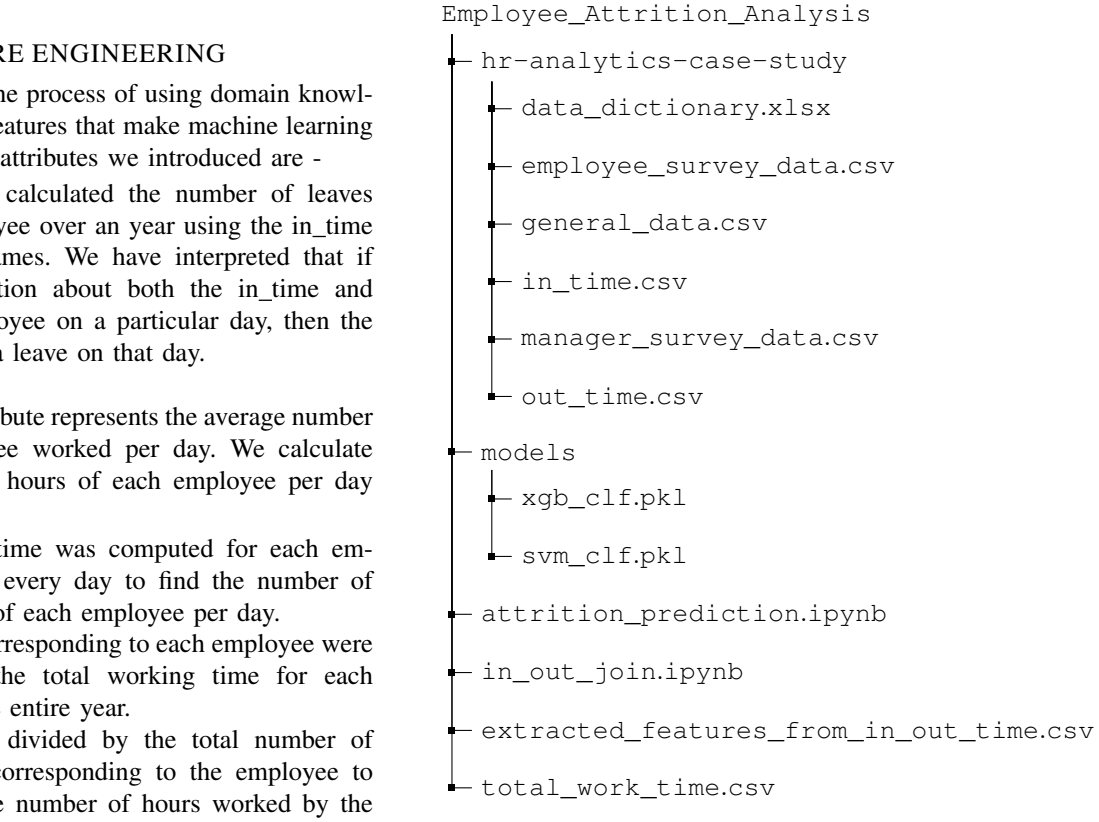
2) **AvgHours :** This attribute represents the average number of hours an employee worked per day. We calculate the average working hours of each employee per day as follows.
   a) Out_time - in_time was computed for each employee and for every day to find the number of working hours of each employee per day.
   b) These values corresponding to each employee were added to get the total working time for each employee in the entire year.
   c) This time was divided by the total number of working days corresponding to the employee to find the average number of hours worked by the employee when he goes to the company.

We used *explained_variance_ratio_* of the PCA class to analyse the variances caused by each of the principal components. Based on the results we observed that there is no significant principal component which contributes heavily to the variance. Since our dataset contains a mix of categorical and continuous variables applying PCA directly on this dataset doesn't yield good results. Hence we didn't use PCA.

V. CODE STRUCTURE

We have used Jupyter Notebook for the project. The file structure and code structure is shown below.

1) The *hr-analytics-case-study* directory consists of the 5 CSV files (dataset) and one data dictionary file describing the dataset.
2) The *models* directory contains two pickle files. - XGBoost Model and SVM model.
3) The *attrition_prediction.ipynb* is the jupyter notebook that contains the code for merging all the dataframes, processing the data, building models and predicting output.
4) *in_out_join.ipynb* is the jupyter notebook that contains the code for merging and extracting features (avg_hours and no_of_leaves) from *in_time.csv* and *out_time.csv*.
5) The *extracted_features_from_in_out_time.csv* contains the two extracted features (avg_hours and no_of_leaves) along with the EmployeeID.
6) The *total_work_time.csv* contains the work time of each employee for each day.



VI. MODEL BUILDING

train_test_split imported from *sklearn.model_selection* has been used to split the dataset into training and testing data. The ratio of split for training:testing data is 80:20.
Since the given dataset is highly imbalanced, the data has been split in a stratified fashion using the target variable (attrition). This ensures that the ratio of class 0 : class 1 remains the same in both the training(2941:564) as well the testing(736:141) data.
General classification models and techniques do not perform well on skewed or imbalanced data. Therefore we tried to implement a few techniques that can handle such data. We tried implementing standard classifier algorithms like logistic regression. Although an accuracy of 84% was obtained, the precision and recall values were extremely poor for class 1. This is because logistic regression tends to have a bias towards classes which have more number of instances. This implies that they tend to only predict the majority class data. Therefore this model is not appropriate.

The following two models were implemented to handle this issue -

1) **XGBoost (Extreme Gradient Boosting)**
   XGBoost algorithm addresses the issue of imbalanced

data by increasing the penalization for misclassifying certain classes. This technique of Boosting makes XGBoost classifier an appropriate model for the given problem statement.

2) **SVM (Support Vector Machine)**
We also tried implementing SVM model with a linear kernel. We tune the value for the parameter *class_weights* to address the issue of imbalanced data.

The *scale_pos_weight* hyperparameter of the XGBoost algorithm was set to an appropriate value to handle imbalanced data. The value of *scale_pos_weight* hyperparameter was calculated by finding the ratio of class 0 : class 1 samples, which is equal to $2941/564 = 5.21$.

VII. Results

1) **XGBoost :** AUC : 0.87

	precision	recall	f1-score	support
0	0.97	0.86	0.91	736
1	0.54	0.87	0.67	141
accuracy			0.86	877
macro avg	0.76	0.87	0.79	877
weighted avg	0.90	0.86	0.87	877

Fig 8. shows the Confusion matrix :

Fig. 8.

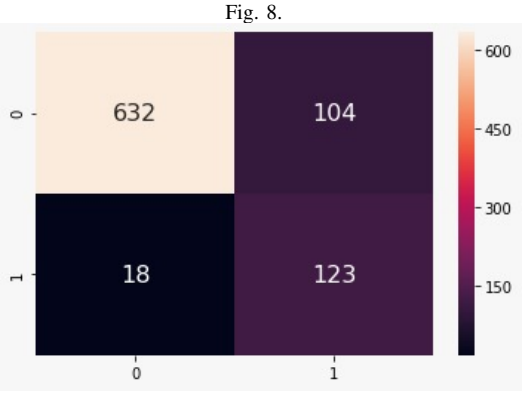
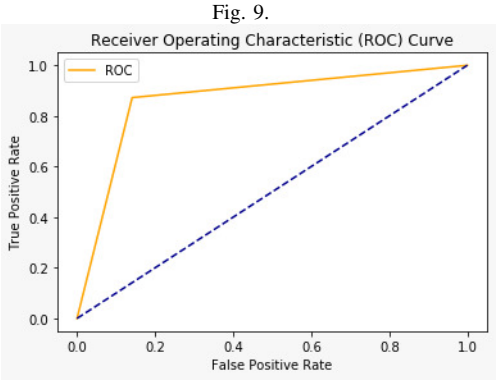


Fig 9. shows the ROC curve :

Fig. 9.



2) **SVM :** AUC : 0.62

	precision	recall	f1-score	support
0	0.91	0.48	0.63	736
1	0.22	0.76	0.34	141
accuracy			0.53	877
macro avg	0.57	0.62	0.49	877
weighted avg	0.80	0.53	0.58	877

Fig 10. shows the Confusion matrix :

Fig. 10.

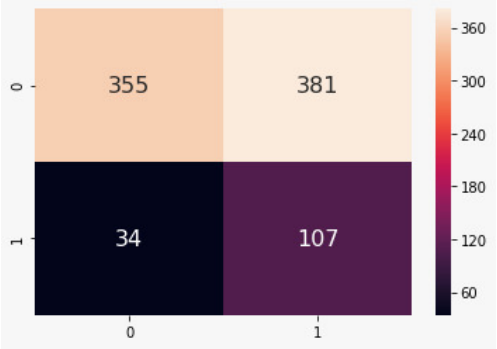
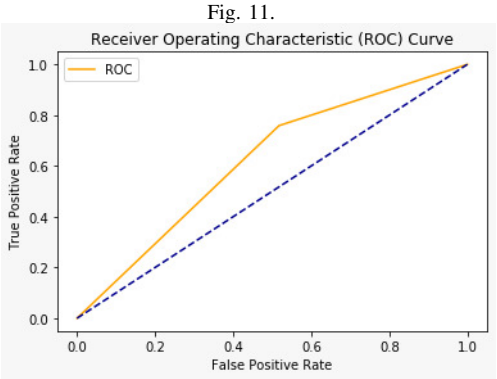


Fig 11. shows the ROC curve :

Fig. 11.



We have inferred that XGBoost performs well on classifying imbalanced dataset when tuned appropriately.

References

[1] Imbalance-XGBoost: Leveraging Weighted and Focal Losses for Binary Label-Imbalanced Classification with XGBoost
[2] Dataset from kaggle.
[3] Scikit-learn documentation
[4] Pandas documentation
[5] Numpy documentation
[6] Google drive link