Capstone Project

Virender Kumar

Machine Learning Engineer Nanodegree

June 9, 2019

Definition

Project Overview

Every year a lot of companies hire a number of employees. The companies invest time and money in training those employees, not just this but there are training programs within the companies for their existing employees as well. The aim of these programs is to increase the effectiveness of their employees. If an employee leaves a company in this scenario, it is a waste of investment in terms of both time and money for that company.

In this project, I created a python program implementing the machine learning algorithms to predict the attrition rate of employees. The program also tries to predict the topmost factors used by the machine learning model to predict the outcome. The program uses a classifier trained on the sample dataset provided by IBM on Kaggle.

https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset

Problem Statement

The goal is to create a model that can predict if a certain employee will leave the company or not. The tasks involved are the following:

- Download and preprocess the IBM HR Analytics dataset.
- Separate the dataset into training and test sets.
- Train a classifier on the training dataset that can predict how likely it is for an employee to quit the company.
- Evaluate the classifier on the test data using the set of predefined metrics.
- Extract the topmost relevant features in a model which are contributing towards the final outcome.

The final program is expected to predict the attrition rate of employees with a reasonable accuracy.

Metrics

ROC-AUC i.e. "Area under the ROC (**receiver operating characteristic**) Curve" measures the entire two-dimensional area underneath the entire ROC curve from (0,0) to (1,1).

A **ROC curve** is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

- True Positive Rate
- False Positive Rate

True Positive Rate (**TPR**) is a synonym for recall and is therefore defined as follows:

$TPR=TP / TP+FN$

False Positive Rate (**FPR**) is defined as follows:

$FPR=FP / FP+TN$

I have chosen this metric for its advantages listed below:

- AUC is **scale-invariant**. It measures how well predictions are ranked, rather than their absolute values.
- AUC is **classification-threshold-invariant**. It measures the quality of the model's predictions irrespective of what classification threshold is chosen.

Since my input dataset is biased towards active employees and I am unsure of the threshold to assign for the quit cases, this metric seems ideal for me to evaluate the model.

Sensitivity/Recall/True Positive Rate is another metric which could help identify the accuracy of my true positives/quit cases.

Sensitivity = TP / TP + FN

It makes sense to use this metric as our final objective is to prevent the attrition by predicting the employees who are more likely to quit and using the sensitivity rate, we can determine how accurate our quit case prediction is.

Accuracy is a common metric for binary classifiers; it takes into account both true positives and true negatives with equal weight.

Accuracy = true positives + true negatives / dataset size

This was used as the last metric to predict the general performance of our model both in terms of true positives and true negatives.

Analysis

Data Exploration

IBM has gathered information on employee satisfaction, income, seniority and some demographics. It includes the data of 1470 employees. To use a matrix structure, we changed the model to reflect the following data:

Name	Description
AGE	Numerical Value
ATTRITION	Employee leaving the company (0=no, 1=yes)
BUSINESS TRAVEL	(1=No Travel, 2=Travel Frequently, 3=Tavel Rarely)
DAILY RATE	Numerical Value - Salary Level
DEPARTMENT	(1=HR, 2=R&D, 3=Sales)
DISTANCE FROM HOME	Numerical Value - THE DISTANCE FROM WORK TO HOME
EDUCATION	Numerical Value
EDUCATION FIELD	(1=HR, 2=LIFE SCIENCES, 3=MARKETING, 4=MEDICAL SCIENCES, 5=OTHERS, 6= TEHCNICAL)
EMPLOYEE COUNT	Numerical Value
EMPLOYEE NUMBER	Numerical Value - EMPLOYEE ID
ENVIROMENT SATISFACTION	Numerical Value - SATISFACTION WITH THE ENVIROMENT
GENDER	(1=FEMALE, 2=MALE)
HOURLY RATE	Numerical Value - HOURLY SALARY
JOB INVOLVEMENT	Numerical Value - JOB INVOLVEMENT
JOB LEVEL	Numerical Value - LEVEL OF JOB
JOB ROLE	(1=HC REP, 2=HR, 3=LAB TECHNICIAN, 4=MANAGER, 5= MANAGING DIRECTOR, 6= REASEARCH DIRECTOR, 7= RESEARCH SCIENTIST, 8=SALES EXECUTIEVE, 9= SALES REPRESENTATIVE)
JOB SATISFACTION	Numerical Value - SATISFACTION WITH THE JOB
MARITAL STATUS	(1=DIVORCED, 2=MARRIED, 3=SINGLE)
MONTHLY INCOME	Numerical Value - MONTHLY SALARY
MONTHY RATE	Numerical Value - MONTHY RATE
NUMCOMPANIES WORKED	Numerical Value - NO. OF COMPANIES WORKED AT
OVER 18	(1=YES, 2=NO)
OVERTIME	(1=NO, 2=YES)
PERCENT SALARY HIKE	Numerical Value - PERCENTAGE INCREASE IN SALARY
PERFORMANCE RATING	Numerical Value - ERFORMANCE RATING
RELATIONS SATISFACTION	Numerical Value - RELATIONS SATISFACTION
STANDARD HOURS	Numerical Value - STANDARD HOURS
STOCK OPTIONS LEVEL	Numerical Value - STOCK OPTIONS
TOTAL WORKING YEARS	Numerical Value - TOTAL YEARS WORKED
TRAINING TIMES LAST YEAR	Numerical Value - HOURS SPENT TRAINING
WORK LIFE BALANCE	Numerical Value - TIME SPENT BEWTWEEN WORK AND OUTSIDE
YEARS AT COMPANY	Numerical Value - TOTAL NUMBER OF YEARS AT THE COMPNAY
YEARS IN CURRENT ROLE	Numerical Value -YEARS IN CURRENT ROLE
YEARS SINCE LAST PROMOTION	Numerical Value - LAST PROMOTION
YEARS WITH CURRENT MANAGER	Numerical Value - YEARS SPENT WITH CURRENT MANAGER

From the above table, we can see that the data has a total of 26 numerical fields and 9 categorical fields.

Out of those 26 numerical fields we have:

- 5 continuous variables: Age, DailyRate, HourlyRate, MonthlyIncome, MonthlyRate
- 1 Id variable: EmployeeNumber (it is a label for each of the employees)
- 18 discrete variables: Rest all of them

Please find below a subset of the data with each image containing the subset of given fields:

Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber
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	4
33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5
27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7

EnvironmentSatisfaction	Gender	HourlyRate	JobInvolvement	JobLevel	JobRole	JobSatisfaction	MaritalStatus	MonthlyIncome	MonthlyRate
2	Female	94	3	2	Sales Executive	4	Single	5993	19479
3	Male	61	2	2	Research Scientist	2	Married	5130	24907
4	Male	92	2	1	Laboratory Technician	3	Single	2090	2396
4	Female	56	3	1	Research Scientist	3	Married	2909	23159
1	Male	40	3	1	Laboratory Technician	2	Married	3468	16632

NumCompaniesWorked	Over18	OverTime	PercentSalaryHike	PerformanceRating	RelationshipSatisfaction	StandardHours	StockOptionLevel	TotalWorkingYears
8	Y	Yes	11	3	1	80	0	8
1	Y	No	23	4	4	80	1	10
6	Y	Yes	15	3	2	80	0	7
1	Y	Yes	11	3	3	80	0	8
9	Y	No	12	3	4	80	1	6

TrainingTimesLastYear	WorkLifeBalance	YearsAtCompany	YearsInCurrentRole	YearsSinceLastPromotion	YearsWithCurrManager
0	1	6	4	0	5
3	3	10	7	1	7
3	3	0	0	0	0
3	3	8	7	3	0
3	3	2	2	2	2

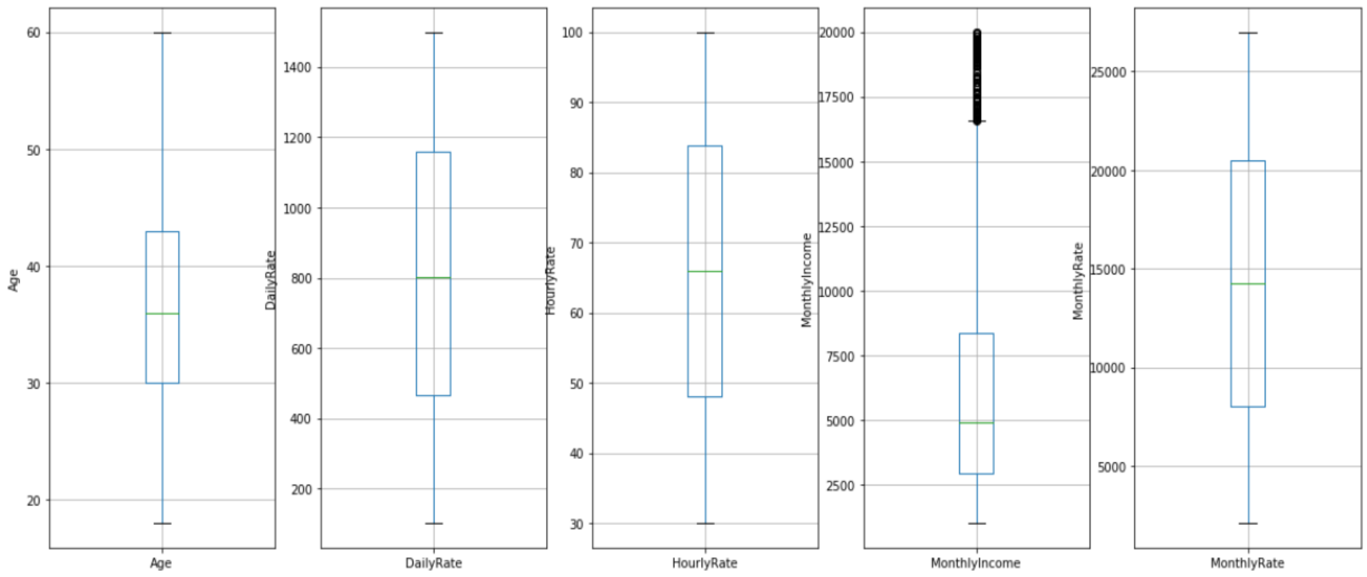
Abnormalities or types of problems within the variables:

- There are no missing values in any of the fields.

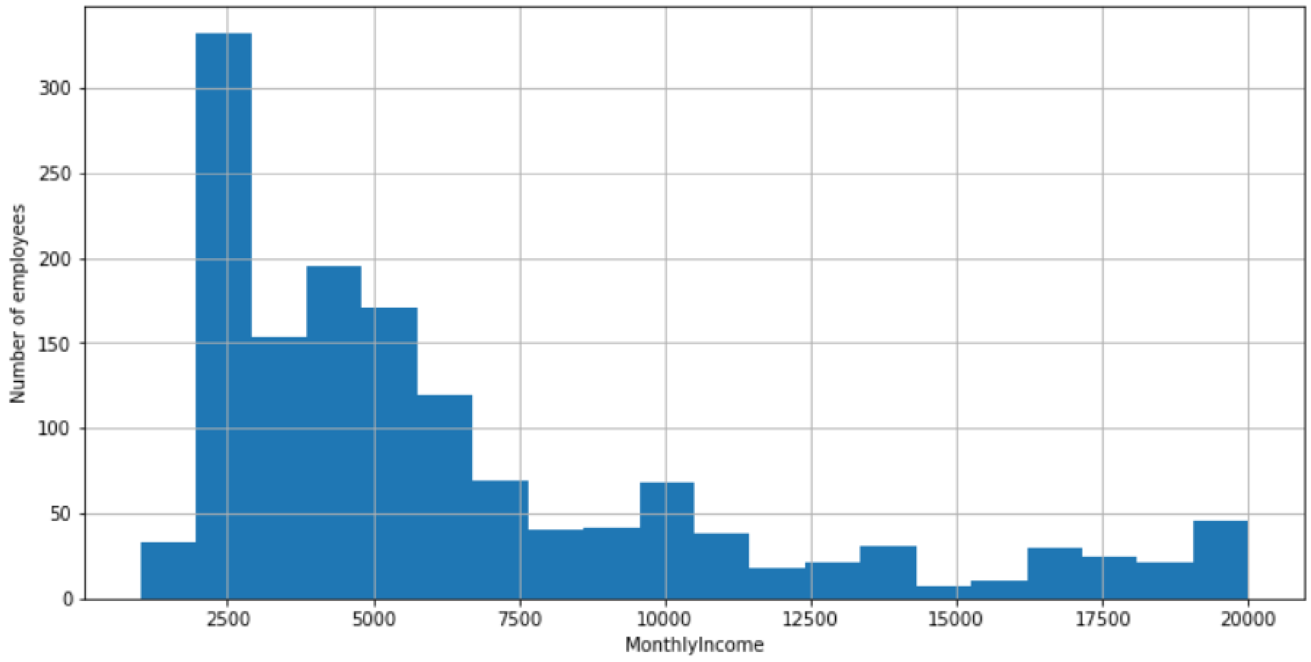
- Some of the fields like ‘EmployeeCount’, ‘Over18’ and ‘StandardHours’ have only a single label or value for all of employees, so they can be dropped.
- Some of the fields like 'JobLevel', 'TotalWorkingYears', 'YearsInCurrentRole' and 'YearsWithCurrManager' are highly correlated with the other features and can be dropped.
- Some of the numerical fields like ‘MonthlyIncome’, ‘DistanceFromHome’, ‘YearsAtCompany’ and ‘YearsSinceLastPromotion’ contains outliers which needs to be handled.

Exploratory Visualization

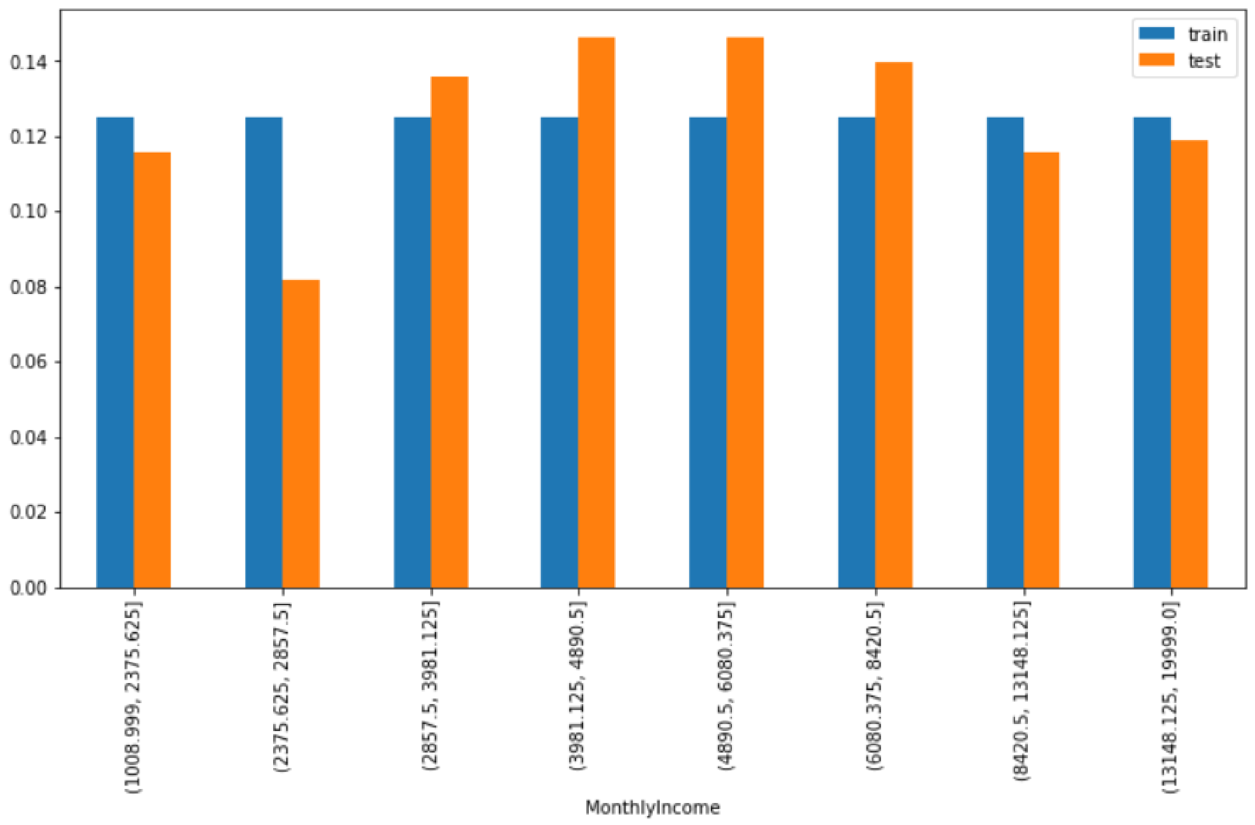
The box plots below show the outliers for the continuous numerical variables discussed above. These are helpful for predicting if a variable contains outliers or not and if it does, then what is the approximate range and the number of outliers.



From the figure above, it is clearly visible that only ‘MonthlyIncome’ contains the outliers. Now we can plot a histogram to see the distribution of ‘MonthlyIncome’ with respect to number of employees. The below graph helps us to see if the distribution of a variable is gaussian or skewed. In this case, it seems to be skewed towards the lower values.



After finding out the skewedness of the ‘MonthlyIncome’ feature, I applied the technique of discretization to make it more uniform. The below graph helps us to visualize the discrete set of quantiles and final distribution of training and test values of an input feature distributed among those quantiles.



Algorithms and Techniques

Initially, I have opted for a set of algorithms to evaluate, which are mentioned below:

- **XGBoost**: XGBoost stands for eXtreme Gradient Boosting. It is an implementation of gradient boosted decision trees designed for speed and performance. Gradient boosting is an approach where new models are created that predict the residuals or errors of prior models and then added together to make the final prediction. It is called gradient boosting because it uses a gradient descent algorithm to minimize the loss when adding new models.
- **Random Forests:** Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.
- **AdaBoost:** AdaBoost, short for “Adaptive Boosting”, is the first practical boosting algorithm proposed by Freund and Schapire in 1996. It focuses on classification problems and aims to convert a set of weak classifiers into a strong one. For any

classifier (decision tree in our case) with accuracy higher than 50%, the weight is positive. The more accurate the classifier, the larger the weight. While for the classifier with less than 50% accuracy, the weight is negative. It means that we combine its prediction by flipping the sign. For example, we can turn a classifier with 40% accuracy into 60% accuracy by flipping the sign of the prediction. Thus, even the classifier performs worse than random guessing, it still contributes to the final prediction. We only don't want any classifier with exact 50% accuracy, which doesn't add any information and thus contributes nothing to the final prediction.

- **Logistic Regression:** Logistic regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous i.e. binary. Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables. It tries to establishes a linear relationship between the independent and dependent variables.

I have used 'random_state' as an input parameter for all of the above models to replicate the results I achieved.
The input dataset is divided into 80% training data which is fed to all of the above models for training and 20% of testing data to evaluate their outputs. Each of the above model will fit the training data and do the prediction based on what they learned using that training data as described in the definitions above.

Benchmark

Since my project proposal is available as an online competition on Kaggle, I have used one of the publicly available implementations as a benchmark. Please find below the link to that particular implementation:

https://www.kaggle.com/nitya123/ibm-hr-analytics-employee-attrition-performance#Business-Problem

The user ran three classification models on the given problem and below are the stats for each of them:

Logistic AUC: 0.58

Decision Tree AUC: 0.57

Random Forest AUC: 0.57

The user has done significant analysis of the input features and some data preprocessing to achieve the above results. The detailed implementation is available on the Kaggle link above.

In addition to the above stats, user has also created a feature importance map with the below top five most important features:

- JobLevel > JobInvolvement > EnvironmentSatisfaction > JobSatisfaction > RelationshipSatisfaction

My goal was to improve on the above figures and create my own set of topmost important features.

Methodology

Data Preprocessing

The preprocessing done in the solution notebook consists of the following steps:

- There was a high correlation among some of the fields, so I had to drop out some of these fields listed below:
    - 'JobLevel', 'TotalWorkingYears', 'YearsInCurrentRole', 'YearsWithCurrManager'
- Some of the fields like 'EmployeeCount', 'Over18', 'StandardHours' contained only a single value or a label, so I had to drop them out as well.
- Handling outliers in types of numerical values:
    - The following discrete variables contains outliers and I have used a technique called top coding to handle them:
        - DistanceFromHome: top-coding (26)
        - YearsAtCompany: top-coding (20)
        - YearsSinceLastPromotion: top-coding (11)
    - Top coding caps the outlier values to the specified upper limit, e.g. in case of DistanceFromHome variable above, all the values above 26 will be capped to 26.
    - I have calculated the value distribution across employees to find the upper range and the outliers of the above discrete variables. Any value with a distribution of less than 1% was considered an outlier for these variables.
    - The continuous variable ‘MonthlyIncome’ contains outliers and its distribution is left-skewed as was discussed in the Analysis section above.
    - In order to handle the outliers in this skewed distribution, I have discretized the field into a set of quantiles to make the distribution more uniform.
- Encoding of categorical variables:
    - I first calculated the cardinality (number of labels) for all the categorical variables, which is given as below:
        - BusinessTravel contains 3 labels
        - Department contains 3 labels
        - EducationField contains 6 labels

        - Gender contains 2 labels
        - JobRole contains 9 labels
        - MaritalStatus contains 3 labels
        - OverTime contains 2 labels
    - Based on above figures, I decide to use:
        - Gender and OverTime: one hot encoding
        - Remaining variables: replace by risk probability
    - One hot encoding technique creates a new variable for each label present inside the applied variable and assigns values of 0 and 1 across all the variables based on whether a particular label was present or not.
    - Risk probability technique helps calculate the distribution percentage of a label in an input variable with respect to the output variable and replace that label with the distribution percentage.
        - I will use the same encoding technique for the 'MonthlyIncome' variable which was discretized before to handle its outliers.
- Last but not least, I have used the MinMax scaler to scale all the feature values within the range of 0 and 1, so that they could be easily handled by one of the linear models called 'Logistic Regression', although the rest of the non-linear models in my case did not require this scaling step.

Implementation

The implementation process can be split into three main stages:

1. The classifier training and validation stage.
2. Creation of feature importance graphs.
3. Choosing the best model and the final evaluation.

Training and Validation:

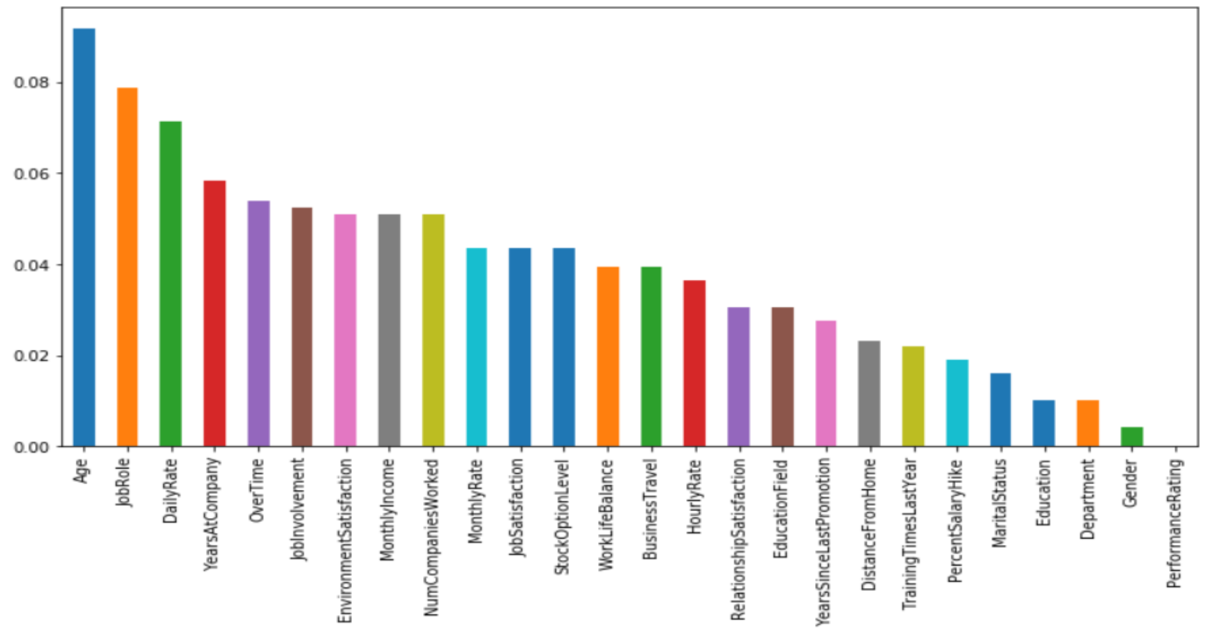
- I have used four classifiers to train my dataset on, including XGBoost, Random Forest, AdaBoost and Logistic Regression.
- The training and validation datasets are first converted into a matrix form. This was done to make a better compatibility with the XGBoost classifier which does not work well with the dataframes.
- I started out with the XGBoost model definition giving it a random_state of 18 which was used throughout for all the models to replicate my results and to compare the model performances.
- I then created an evaluation set with the testing data and output labels and chose AUC (area under the curve) as my evaluation metric to train the model.
- I then fit the model on the training data and labels to train the model.
- Finally, I used the predict_proba method to get the prediction on testing data set and used those predictions to get the following results for XGB.

  - **XGB train roc-auc**: 0.9752305525287692
  - **XGB test roc-auc**: 0.7796283309957925
  - **XGB test accuracy**: 0.8843537414965986
- I followed the same steps of training and validation for the rest of the classifiers and below are the results:
  - **RF train roc-auc:** 0.9996412150849124
  - **RF test roc-auc:** 0.6530066619915849
  - **RF test accuracy:** 0.8537414965986394
  - **AdaBoost train roc-auc:** 0.9227150716240997
  - **AdaBoost test roc-auc:** 0.7855890603085554
  - **AdaBoost test accuracy:** 0.8809523809523809
  - **Logit train roc-auc:** 0.7539160709065298
  - **Logit test roc-auc:** 0.7223001402524544
  - **Logit test accuracy:** 0.8503401360544217

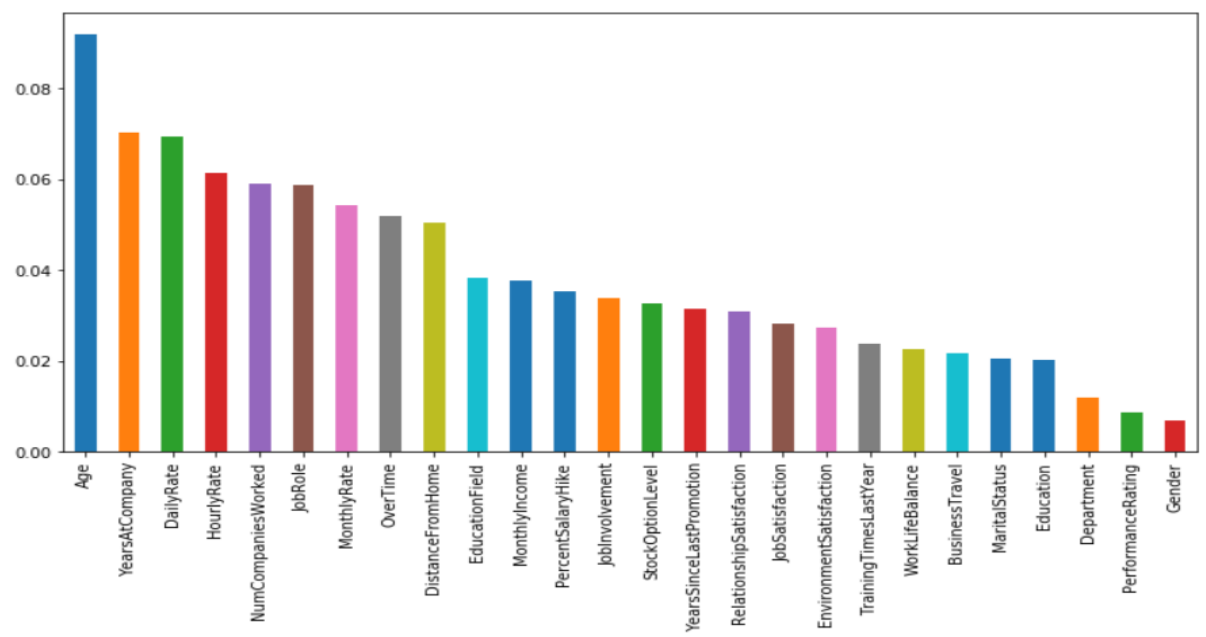
Feature Importance Graphs:

- Some of the above classifiers give out an option to get the top features used by the classifier to get the results. I've used this capability to get those features and draw the feature importance graph for these classifiers.

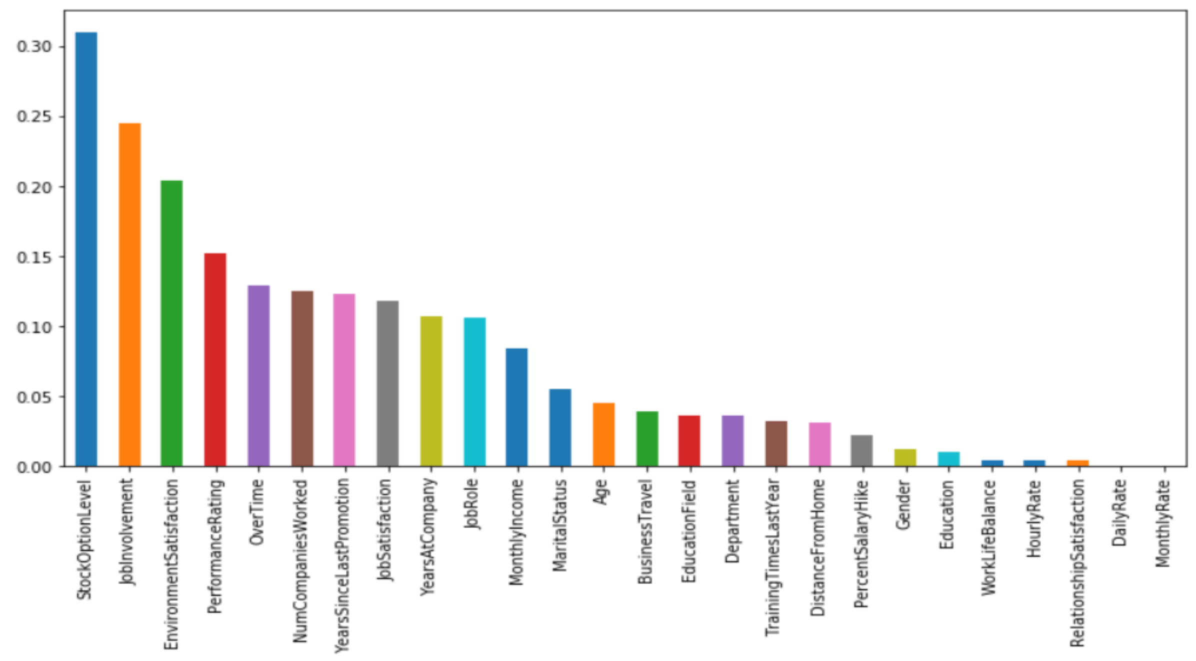
XGB:



Random Forest:

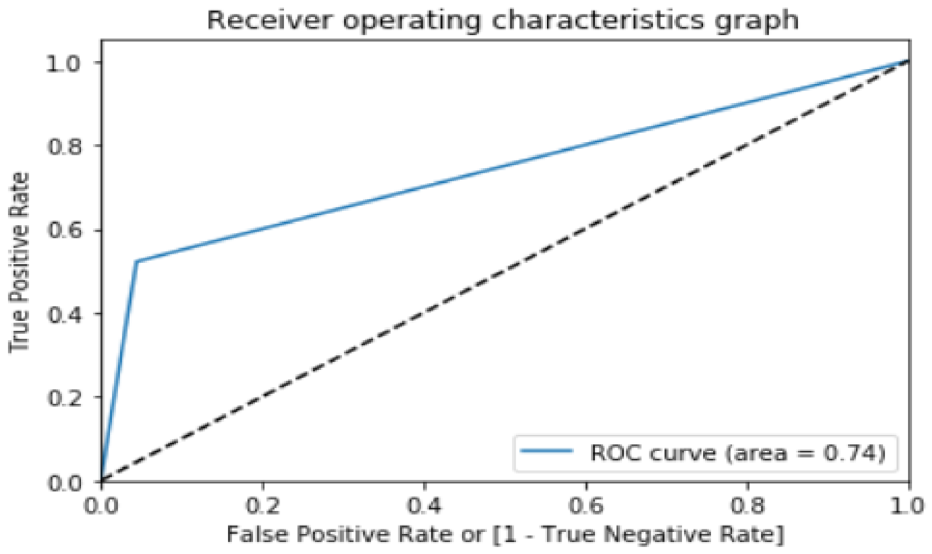


Logistic Regression:



Choosing the best model and final evaluation:
Looking at the initial stats and the above feature importance graphs, I decided to use XGBoost as the final model for prediction.
One of the challenges now was choosing the optimal threshold value to decide the quits and after trying out with some of the values, I finally decided to go for the value of 0.40 as the final threshold number.
Some of the thresholds which were not successful were 0.50, 0.30, 0.35.

Using the decided threshold value, I did the predictions on the test dataset again, marking anything above or equal to 0.40 probability as Quit and anything below this figure as Active. Using the now 0 and 1 predictions, I draw the ROC curve to get a feel of the model performance.



From the above figure, we can deduce our final AUC to be 0.74.

I created a confusion matrix to get the final figures for True Positives and True Negatives.

```
[[237,  11],
 [ 22,  24]]
```

TN = 237 # true negative/actives
FP = 11 # false positive
FN = 22 # false negative
TP = 24 # true positive/ quits
And the sensitivity = TP / (TP + FN) = 0.5217391304347826, which seems to be good enough to be used in the live environment.

Refinement

Initially, I started out taking all the features into consideration for the model training and validation part, which was not giving me acceptable results. After a bit of thinking, I decided to check for the correlation among the input parameters and decided to drop out some of the fields based on them being highly correlated.

XGB Stats before correlation dropout:
AUC: 0.69
Sensitivity: 0.45

XGB Stats after correlation dropout:

AUC: 0.74
Sensitivity: 0.52

The second improvement was to choose a sensible threshold point to get the optimal metric figures, after a few misses, I was finally able to find a good threshold figure. Please find below the changes in the metrics for various thresholds I tried out:

Threshold 0.50
AUC: 0.67, Sensitivity: 0.35

Threshold 0.30
AUC: 0.71, Sensitivity: 0.52

Threshold 0.35
AUC: 0.73, Sensitivity: 0.52

Threshold 0.30
AUC: 0.74, Sensitivity: 0.52

Results

Model Evaluation and Validation

During development, a validation set was used to evaluate the model.
The final model and parameters were chosen because they performed the best among the tried combinations.
For the complete description of the final model, please refer to the following list:

- A set of 4 algorithms were evaluated on the input dataset and XGBoost was chosen as the final model to do the predictions.
- AUC (area under the ROC curve) was chosen as the evaluation metric for the XGBoost model.
- A threshold range of 0.4, to predict anything equal or above it as a quit and everything else as an active.
- A feature importance graph was drawn to get the topmost input features responsible for the model outcome.
- A sensitivity calculation was done to get the Quit Prediction Accuracy of the model.

Looking at the above points, we can safely say that the final outcome aligns with our original goal of predicting employees who are more like to quit the company and predicting the features contributing towards an employee exit from the company.’

To verify the robustness of the model, several tests were conducted making changes in the training and validation datasets by modifying the random state parameter to different values while splitting the data and while creating the model instance.
The following observations are based on the results of these tests:

- The classifier can reliably detect the quit cases which is always better than that of a random guess.
- The classifier in almost all cases performs better than the benchmark model.
- The number of false negatives and false positives are under control for almost all the cases.
- The topmost features are more or less coming out same for different combinations of training and testing datasets.

Justification

Using the final XGBoost model, I got the following results:

- AUC score: 0.74
- Topmost Features:
    - Age, JobRole, DailyRate, YearsAtCompany, OverTime

In comparison to the benchmark model, whose results are as follow:

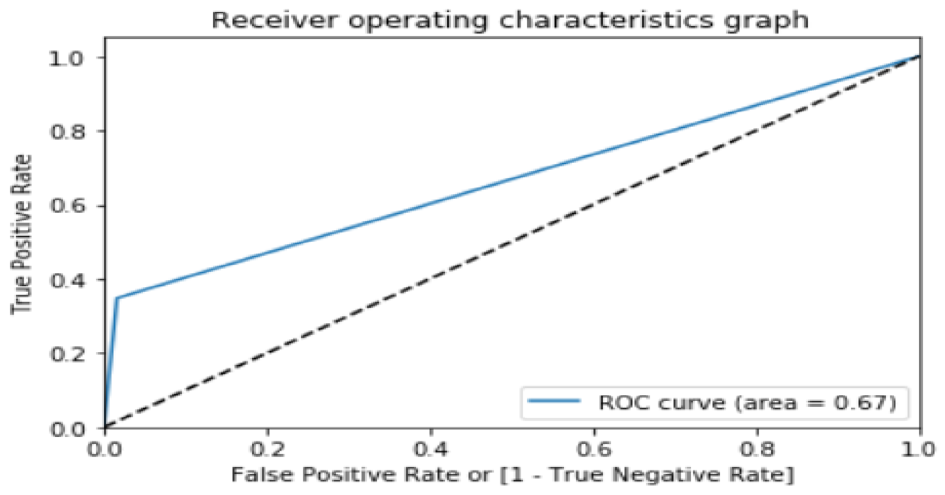
- AUC score: 0.57
- Topmost Features:
    - JobLevel, JobInvolvement, EnvironmentSatisfaction, JobSatisfaction, RelationshipSatisfaction

We can clearly say that the one on the top is clearly better in terms of AUC score. Also, if you look at the list of topmost features, one can intuitively say that the one on the top seems more relevant to an employee decision of leaving or staying in a company.
Based on what we've seen so far, I can say with confidence that the model that I created seems significant enough to act as a starting point for HRD Prediction Analysis. It has the potential of improving further as we introduce more data or more use cases of employees quitting/staying in the company.

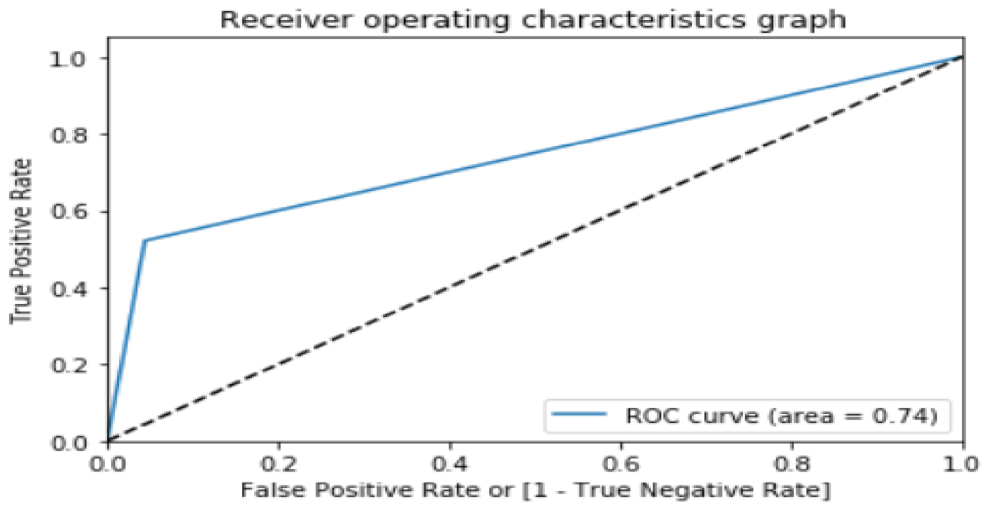
Conclusion

Free-Form Visualization

I would like to discuss one very significant parameter, i.e. threshold, a slight change in whose value was impacting the outcome. I'll show this with the help of a ROC-Curve:



ROC-Curve with a threshold of 0.5



ROC-Curve with a threshold of 0.4

It's clearly visible from the graphs above how a slight change in threshold selection has improved our AUC score.

Reflection

The process used for this project can be summarized using the following steps:

- An initial problem and relevant, public datasets were found.
- The data was downloaded and preprocessed.
- A benchmark was created for the classifier.
- A set of classifiers were trained using the data (multiple times, until a good set of parameters were found).
- The feature importance graphs were plotted for the classifiers.
- The best classifier was chosen and evaluated on the set of predefined metrics.

The most interesting part of the project was to evaluate the dataset on multiple classifiers, seeing how they behave on the dataset and choosing the best one among them.
Similarly, the most difficult part was to decide on the set of final variables (Feature Selection) to be fed into the classifier for training/testing.
Overall, I think the developed solution definitely fit my expectations for the proposed problem and is definitely an option to be used in a general setting for this kind of problem.

Improvement

One case of improvement is to use a better dataset for training/testing in terms of both quantity and quality. The dataset used for training seems to be very small and our model cannot generalize itself with such small amount of data. With the increase in data quantity, we might be able to see some more type of quit cases, which will further help in the live environment setting where the input data would be more flexible and can vary a lot.

Additional Citations

https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc
https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/

https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd

https://towardsdatascience.com/boosting-algorithm-adaboost-b6737a9ee60c