PREDICTING EMPLOYEE ATTRITION

BANA7043 Final Project

Group #3

Lavanya Tharanipathy
Pradnya Patil
Prateek Kumar
Neelabh Sengar

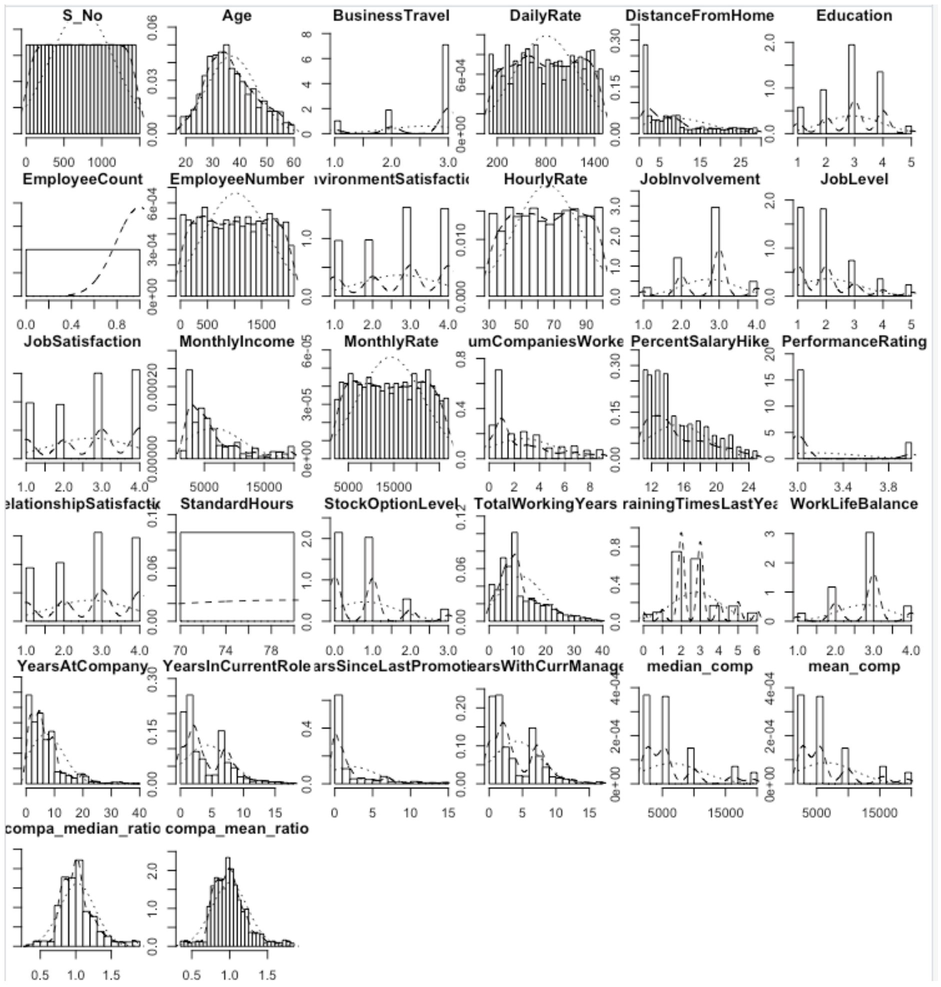
ABSTRACT

Employee attrition is a significant cost outgo for businesses. Through this dataset (source), we aim to predict whether an employee is likely to leave their employer or not. We check the data for integrity, conduct exploratory data analysis and finally, build classification models to predict the attrition for each employer. We train three different classifiers Boosted Logistic Regression, Decision Trees and Random Forest and evaluate their performance. We used the ‘CARET’ (Appendix 4) and ‘pROC’ libraries in R for modelling and evaluation.

Exploratory Data Analysis

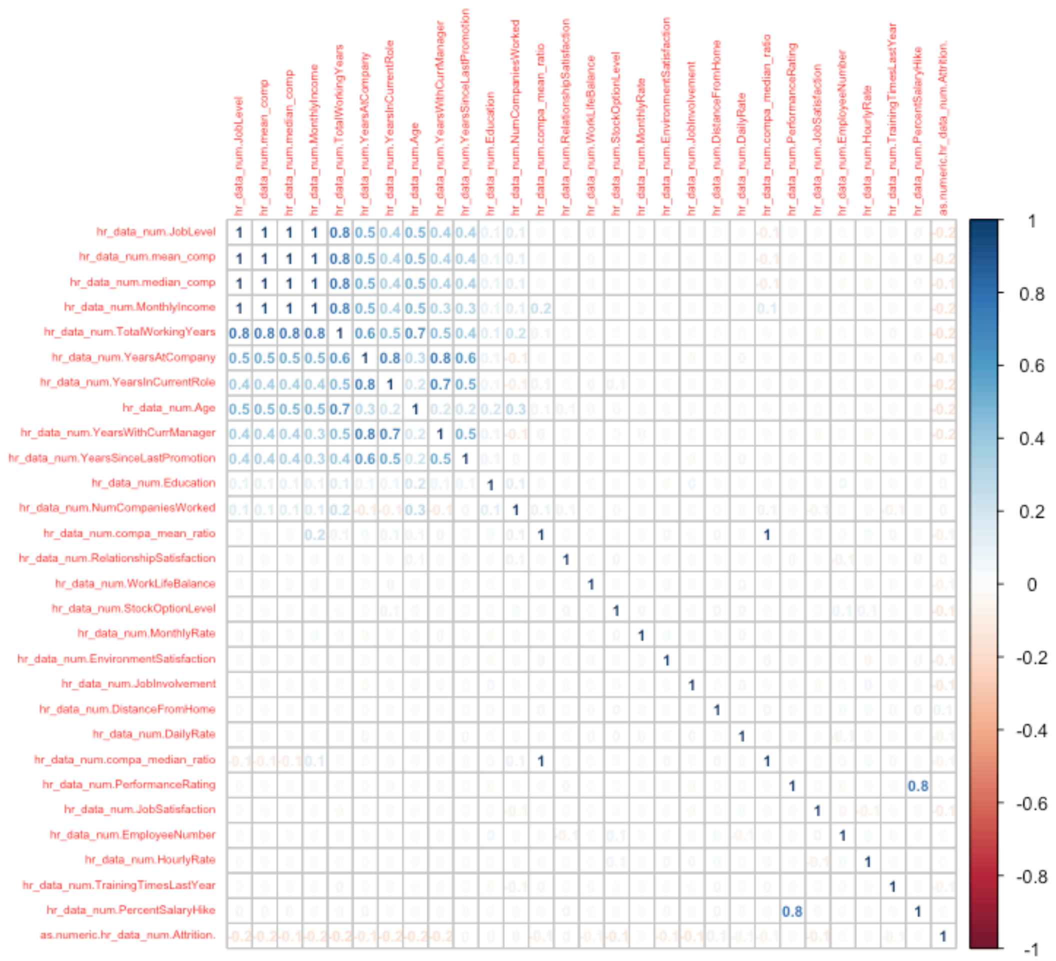
Basics checks performed:

- *Database size* – [1470, 35]
- *Variable types* – The variables are either Categorical or Continuous variables with Factor and Integer types respectively, with the variable "EmployeeNumber" being an ID column
- *Duplicates* – No duplicates were found during EDA
- *Presence of NA values* – No NA values were found
- *Basic variable stats and distributions* -



- Correlation among continuous variables
- Imbalanced Dataset

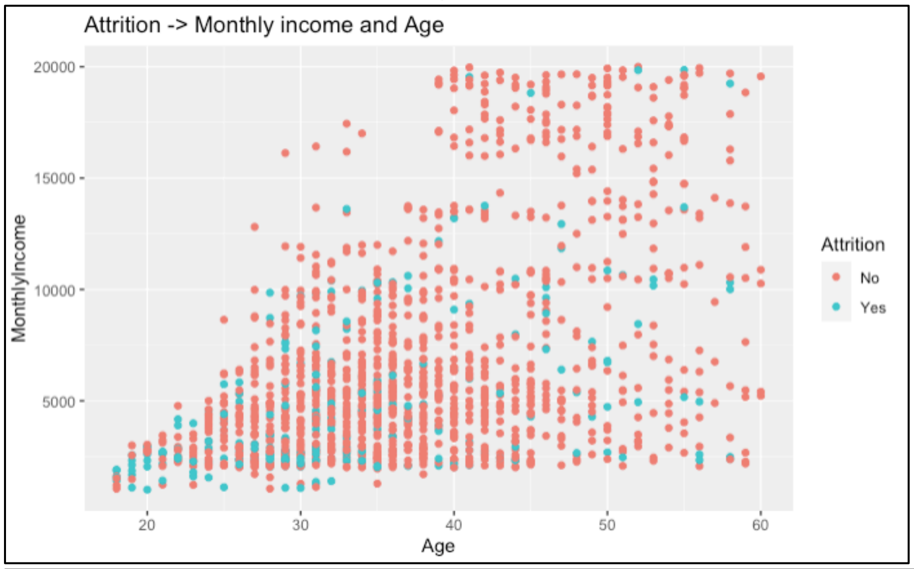
  The HR Dataset is imbalanced. Out of total 1470 data points, only 16% datapoints are of employees who switched their current company.

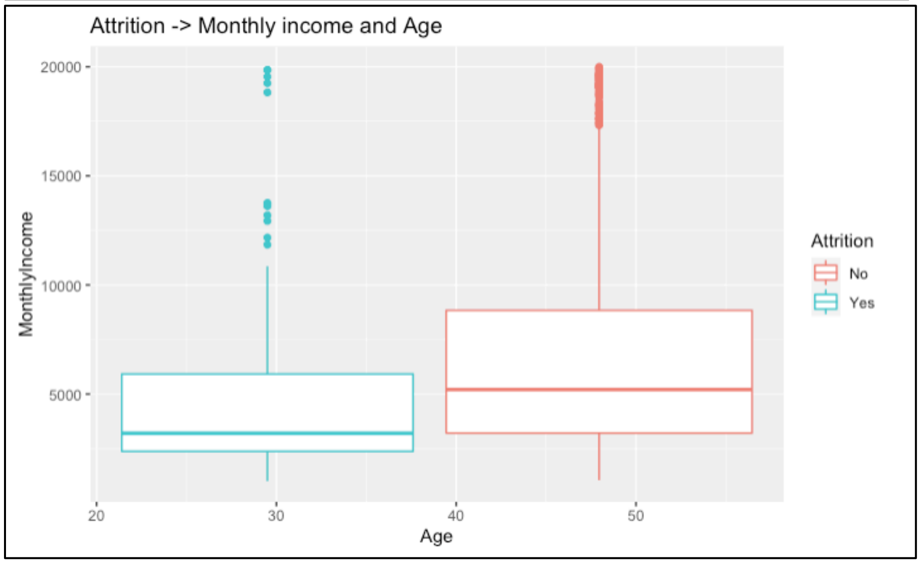

hr_data_num.JobLevel
hr_data_num.mean_comp
hr_data_num.median_comp
hr_data_num.MonthlyIncome
hr_data_num.TotalWorkingYears
hr_data_num.YearsAtCompany
hr_data_num.YearsInCurrentRole
hr_data_num.Age
hr_data_num.YearsWithCurrManager
hr_data_num.YearsSinceLastPromotion
hr_data_num.Education
hr_data_num.NumCompaniesWorked
hr_data_num.compa_mean_ratio
hr_data_num.RelationshipSatisfaction
hr_data_num.WorkLifeBalance
hr_data_num.StockOptionLevel
hr_data_num.MonthlyRate
hr_data_num.EnvironmentSatisfaction
hr_data_num.JobInvolvement
hr_data_num.DistanceFromHome
hr_data_num.DailyRate
hr_data_num.compa_median_ratio
hr_data_num.PerformanceRating
hr_data_num.JobSatisfaction
hr_data_num.EmployeeNumber
hr_data_num.HourlyRate
hr_data_num.TrainingTimesLastYear
hr_data_num.PercentSalaryHike
as.numeric.hr_data_num.Attrition.
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1

EDA Overview

The below checks were performed to explore variables that can possibly help explain attrition:

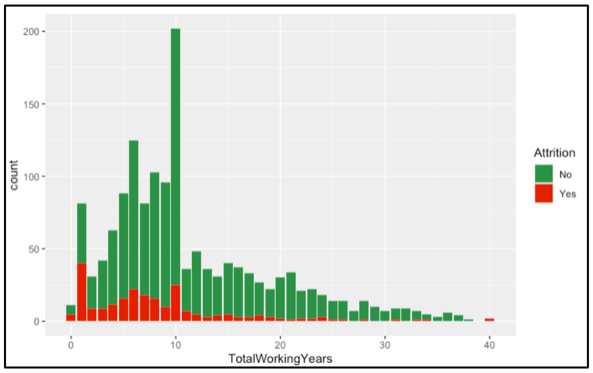
1. Attrition - Monthly-Income and Age



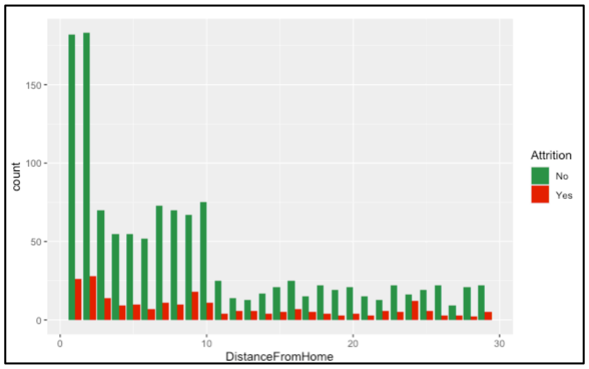


The above plots indicate '**lower income**' is related to higher attrition. Also, mostly the younger generation fall in the lower income range. Indicating both age and monthly income can be a good indicator for attrition.

2.Total working years

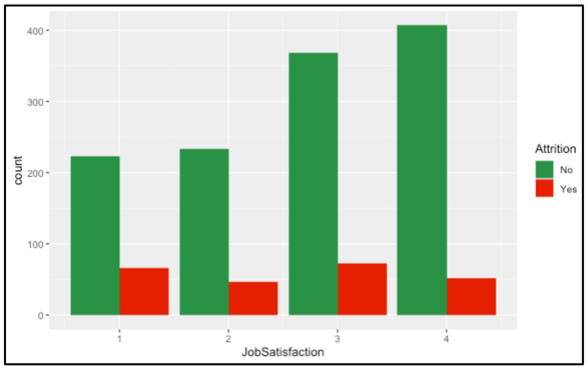


3.Distance from Home

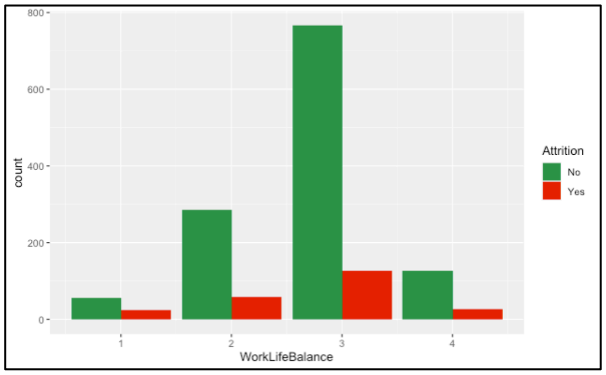


We observe relatively high attrition in population with Total Working Years between 0-10 years, with significantly high attrition ratio with **Total Work Years = 1**
Distance from Home can play a role in Attrition as the **Ratio of Attrition** is fairly pronounced in people living at **greater distances from work**.

4. Work life Balance

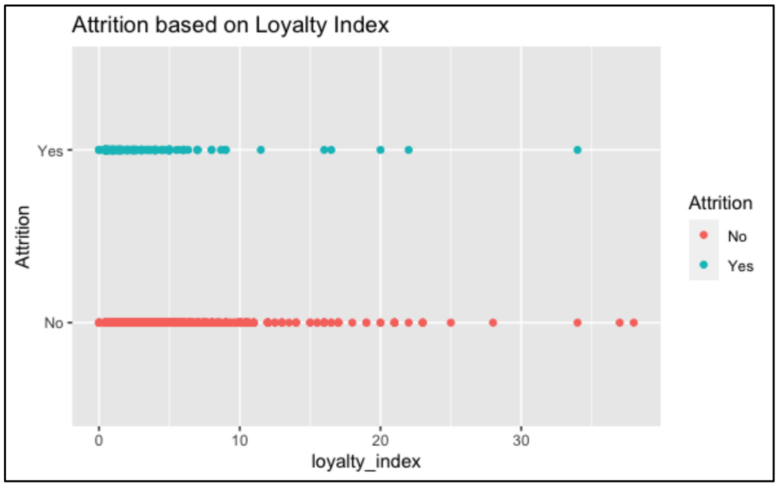


5.Job Satisfaction



Job Satisfaction Low (1) -> High (4)
Clearly Ratio of Attrition is highest for population with **Low Job Satisfaction** and **Low Work life Balance**.

5. Loyalty-Index Attrition



We observe that employees with higher loyalty index are less likely to leave the company.

Other Statistics

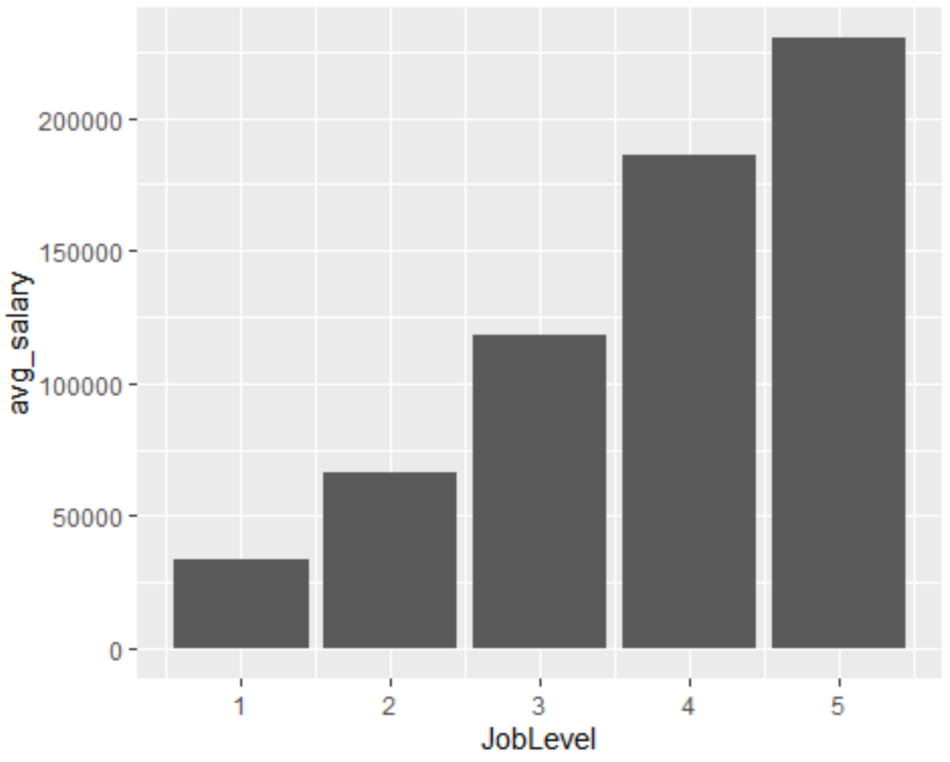
-> It is also observed factors like **Performance Rating** and **Number of companies worked** before have positive correlations to Attrition, whereas **Job Level** & **Years in Current Role** being negatively correlated.

-> Factors like **Gender, Education, Department** do not show significant correlation to explain Attrition.

Feature Engineering

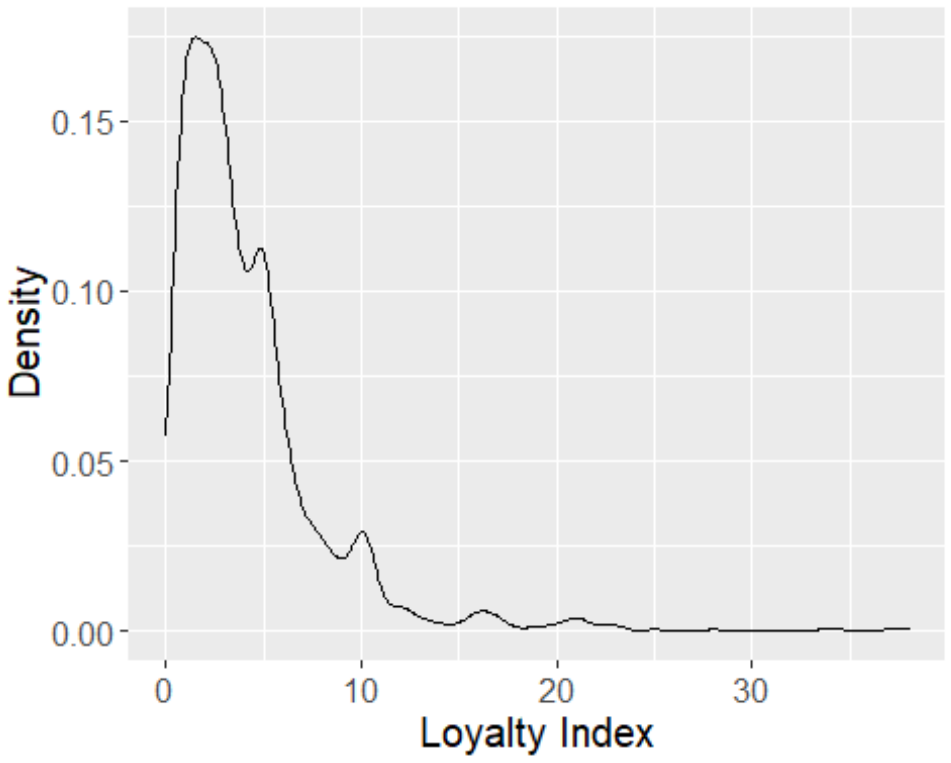
As part of feature engineering we begin with the calculation of a new variable - Annual Salary, where we multiply the variable monthly salary with 12. We then integrate this variable to our dataframe.
Plotting the variable JobLevel with Annual Salary, we see that there are higher salaries with increase in Job level.

```
##Feature 1: Annual Salary
df1 <- df1 %>%mutate(AnnualSalary = MonthlyIncome * 12)
```

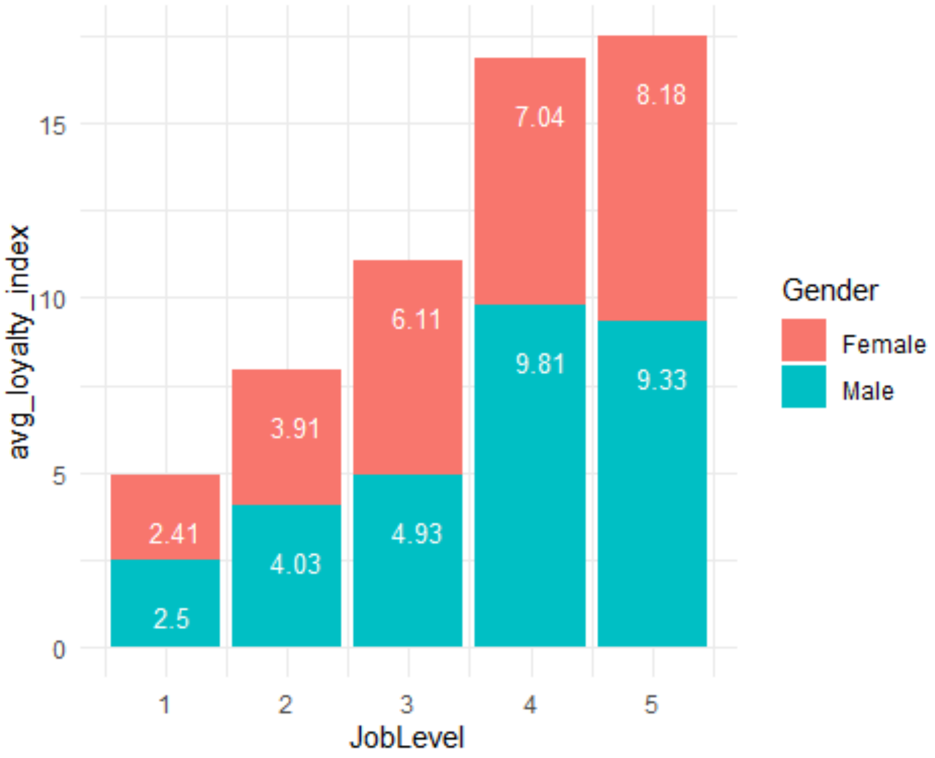


The second feature we introduce is loyalty index, which is the number of organizational jumps an employee has taken in their total work experience. We calculate this as the total working years divided by the Number of companies worked + 1. On observing the distribution of loyalty index on a density plot, we see that most employees have a loyalty index between 0-5, followed by 5-10, and decaying density with increase in the loyalty index value.

```
##Feature 2: Loyalty Index
df1<- df1 %>%
      mutate(loyalty_index = (TotalWorkingYears/ (NumCompaniesWorked + 1)))
```



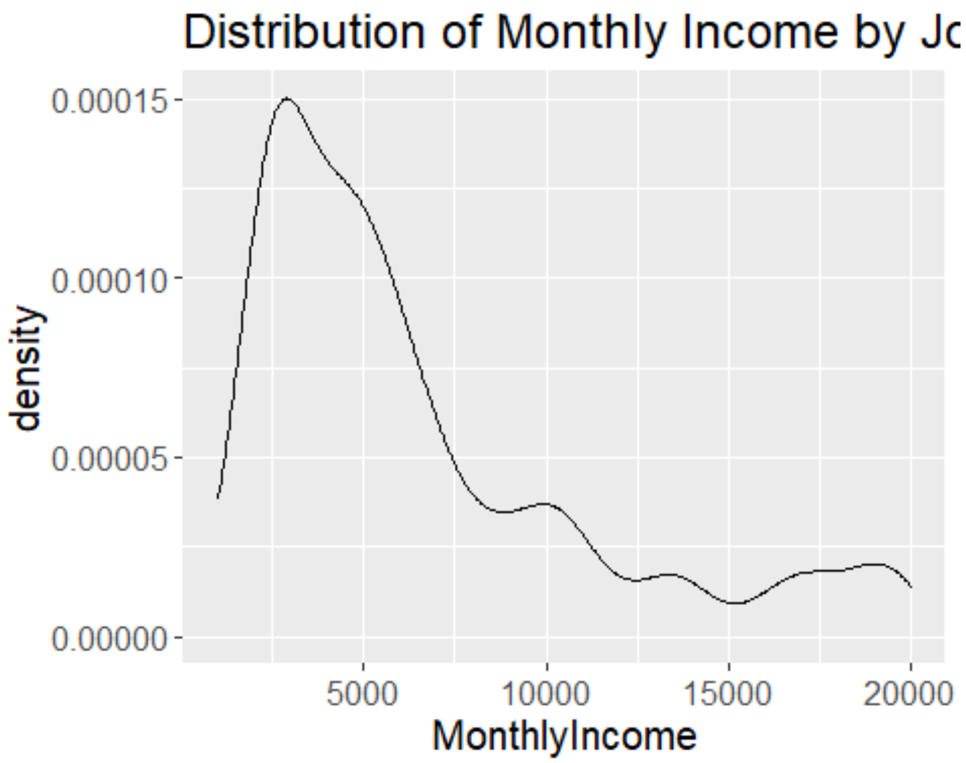
Plotting JobLevel vs Loyalty gender-wise, we observe that with the exception of level 3, Females have a slightly lower loyalty index compared to males.



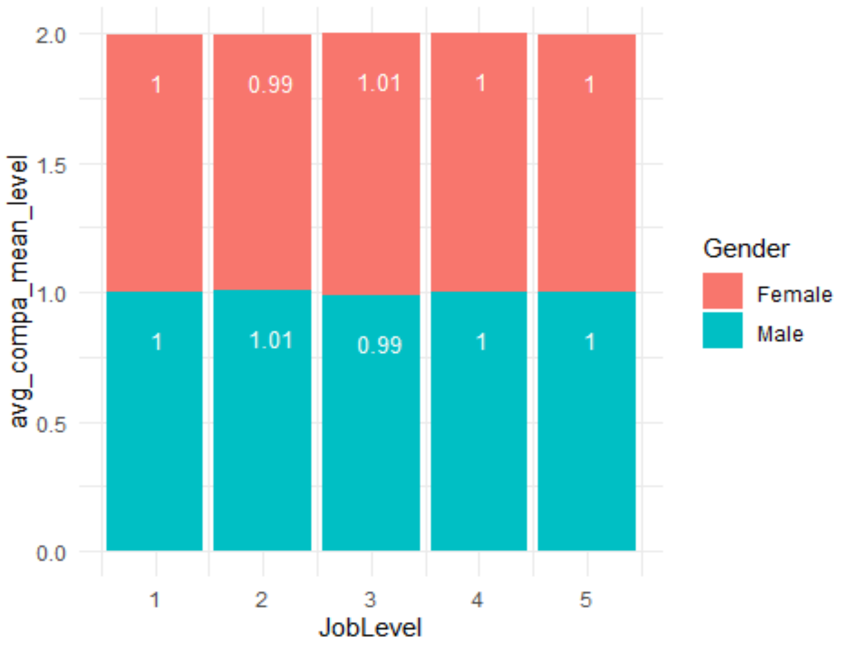
Compensation Ratios – Comparison to mean and median pay at Job Level
We compute the mean and median pays at job level and compute the ratio of employee pay against the mean and median pays as two separate variables. We then introduce the variables compa_mean_level and compa_median_level, wherein we flag whether the employee falls above the mean/median pay.

```r
##Feature 3: Compensation Ratio - Comparison to Mean and Median Pay
df1 <-  df1 %>%
       group_by(JobLevel) %>%
       mutate(median_comp = median(MonthlyIncome),
       mean_comp = mean(MonthlyIncome),
       compa_median_ratio = (MonthlyIncome/ median_comp),
       compa_mean_ratio = (MonthlyIncome/ mean_comp))
```

On plotting the density chart of monthly income, we see that a higher number of employees have low income.



We plot a stacked bar chart to observe gender-wise compensation ratio for distinct roles, and see that there is no significance imbalance with respect to gender when it comes to compensation across job levels.



All the newly calculated features are added to original dataframe.

DATA ANALYSIS

DATA DESCIPTION

The given problem statement is a classification (supervised) problem which can be solved through classification algorithms like Decision Tree Learning, Logistic Regression, SVM or ensemble methods like Random Forest, Adaboost, Gradient Boost etc. For this analysis, we will use Logistic Regression, decision trees and Random Forest to train a classifier and evaluate the results.
The given dataset has 41 features, a summary of the values can be found in the appendix 2.

The Feature ‘Attrition’ is the dependent variable. It takes two values “Yes” and “No” depending on whether the person left the company or not.

Data Transformations

1. Removed redundant (single valued) columns Over18, Standard hours and EmployeeCount
2. Encoding of categorical columns:
    - The categorical feature are converted into numerical using an ordinal encoder as ML models usually have numeric values as predictor variables.
    - Methods used for encoding:

      1. Factor: Conversion from categories to levels

      2. Unclass: Converting to numeric value from factor

      3. Rescale: Rescaling all the columns between 0 and 1

      4. One hot encoding: Creating a binary column for each category value
    - Variables with 2 unique values are factored, unclassed and rescaled to 0 to1.
    - Variables with more than 2 unique values are factored and one hot encoding is applied.
3. Feature scaling:
   Feature scaling is standardizing every numeric feature to particular range so that model does not favor particular features just because it is numerically higher. Additionally, models perform better when features fall in particular range. In our case, all features are scaled between 0 to 1. Rescale function is defined which locally.

   The Logistic Regression classifier is susceptible to difference s in the range of values taken by the features and hence to mitigate this effect we need to perform some sort of Scaling
4. Finally, one dataframe is created which contains only final numeric features, scaled between 0 and 1 for training a testing the model.
5. Created new numeric dependent variable ‘turnover’ which takes values 0 and 1 for ‘No’ and ‘Yes’ respectively.

```
##Elimination of redundant columns which has single values across the column
df1$Gender <- rescale(unclass(factor(df$Gender))) #2 unique values : factor+
unclass + rescale
df1$MaritalStatus <- one_hot(as.data.table(factor(df$MaritalStatus))) #3 uniq
ue values: onehotencoding  +factor

#Function : To scale columns between 0 and 1
rescaleDF <-  function(x) (x-min(x))/(max(x) - min(x))
```

Modelling

1. By using prop.table function, ratio of attrition values in each class is checked, which is 0.83 to 0.16, which makes it a imbalanced dataset. Models tend to perform poorly as seen below, when trained with imbalanced dataset. Due to imbalanced dataset, the algorithms become accuracy driven and tend to improve accuracy than an overall error.

   Methods used to treat imbalanced datasets:

   a. Under sampling + Over sampling
      The minority class is over sampled with replacement and majority class is under sampled without replacement. Ovun.sample function from ROSE library is used implement this method.

   b. Synthetic Data Generation (Rose method)
      Using synthetic minority oversampling technique (SMOTE) data is sampled. This algorithm creates artificial data based on feature space similarities from minority samples.

   Accuracy is a common evaluation metric for classification problems. It is often misused as it is most useful when equal number of observations in each class and all predictions and prediction errors are equally important. This is not the case with this project hence *we will use a different scoring metric may be more suitable like AUC or ROC curve.*

```
##Handling Class Imbalance
prop.table(table(df5$Attrition))
##         0         1
## 0.8387755 0.1612245
```

2. Splitting data into training and testing sets:
   Training and testing datasets are created from decoupling feature dataframe in ration 70:30.

```
set.seed(123)
#Ceating train and testing dataset
n <- nrow(df5)
rnd <- sample(n, n * .70)
train <- df5[rnd,]
test <- df5[-rnd,]
```

3. DECISION TREE:

   A single decision tree is ought to perform poorly but with improvement in dataset using balanced data techniques mentioned above performance of decision tree is improved. In addition, DT can help in visualizing most important patterns.

   Three variants of decision tree models are as following:

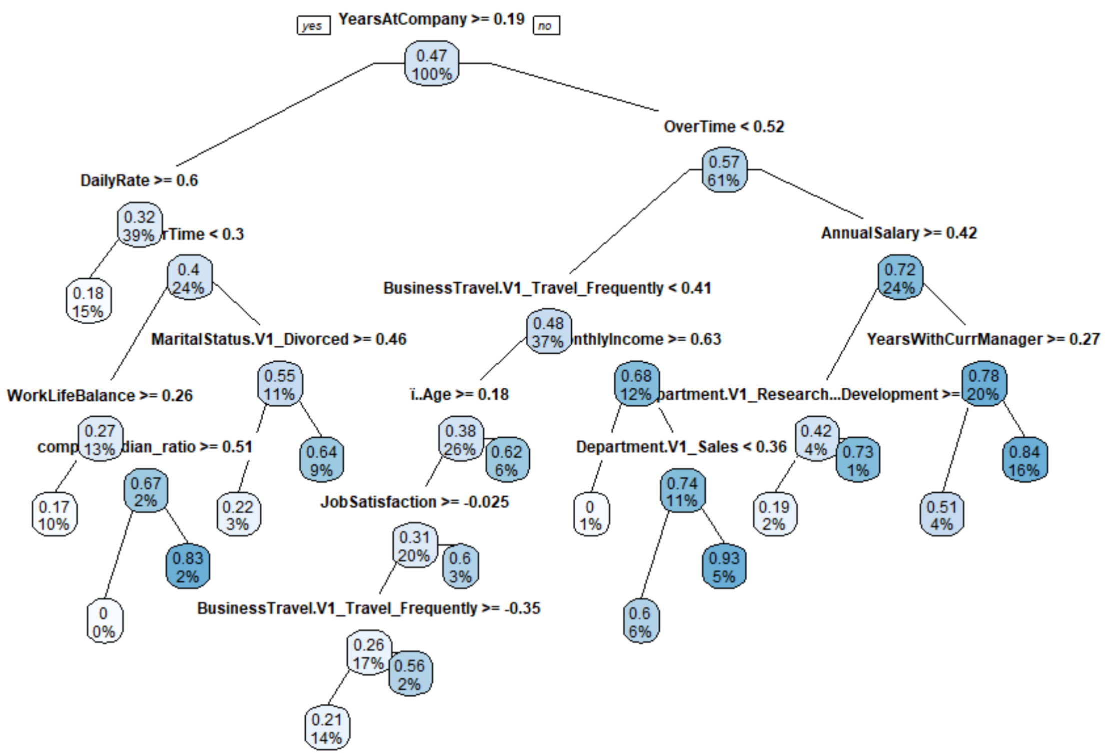
1. M1: Decision tree + Unbalanced data | **AUC: `0.7139`**
2. M2: Decision tree + Oversample-Undersample method | **AUC :0.64**
3. M3: Decision tree + Rose method for class imbalance | **AUC: 0.7378**

Hence Rose sampling method is providing highest auc, hence in further models ROSE method will be used to handle data imbalance.

```
##Model 3 : Decision tree + Rose method for class imbalance

#Dataframe: Input dataframe using ROSE method
data_balanced_rose <- ROSE(Attrition ~ ., data = df5, seed = 1)$data

#Modelling and accuracy of model 3
dtreepr_ROSE <- rpart(Attrition ~., data = data_balanced_rose)
preds_rose <- predict(dtreepr_ROSE,  newdata=test)
rocv_rose <- roc(as.numeric(test$Attrition), as.numeric(preds_rose))

## Area under the curve: 0.7378
```

Output of M3:

As per the diagram, years at the company, overtime, annual salary are the most important factors that may help in predicting weather employees is going to leave the company.



4. RANDOM FOREST:

RF outperforms single decision tree even with unbalanced data set. By using unbalanced data techniques further improvement in AUC is observed.

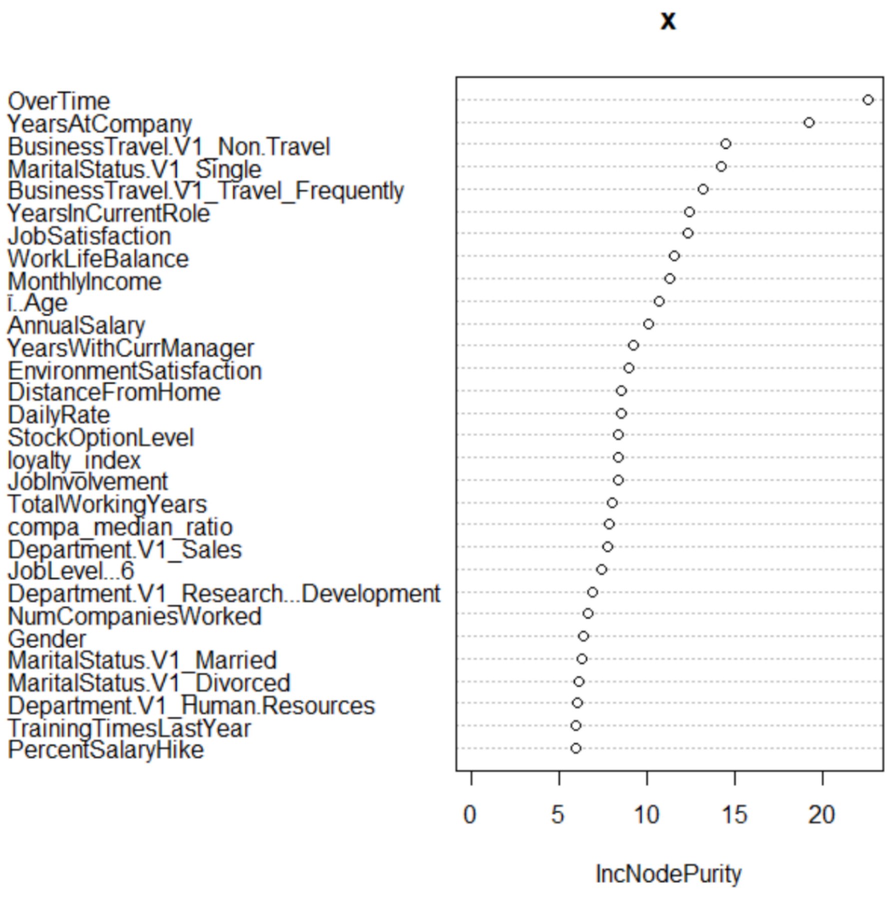
Two variants of decision tree models are as following:
1. M4: Random forest + Unbalanced data | **AUC: 0.801**

2. M5: Random forest + Rose method for class imbalance | **AUC: 0.8492**

```
##Random Forest

##Model 5 : Random forest + Rose method for class imbalance
rfpreds_rose <- predict(fit.forest_rose, test, type = "class")
rocrf_rose <- roc(as.numeric(test$Attrition), as.numeric(rfpreds_rose))rocrf_
rose$auc

## Area under the curve: 0.8492
```

Feature importance by M5:

RF helps in ranking features causing attrition.

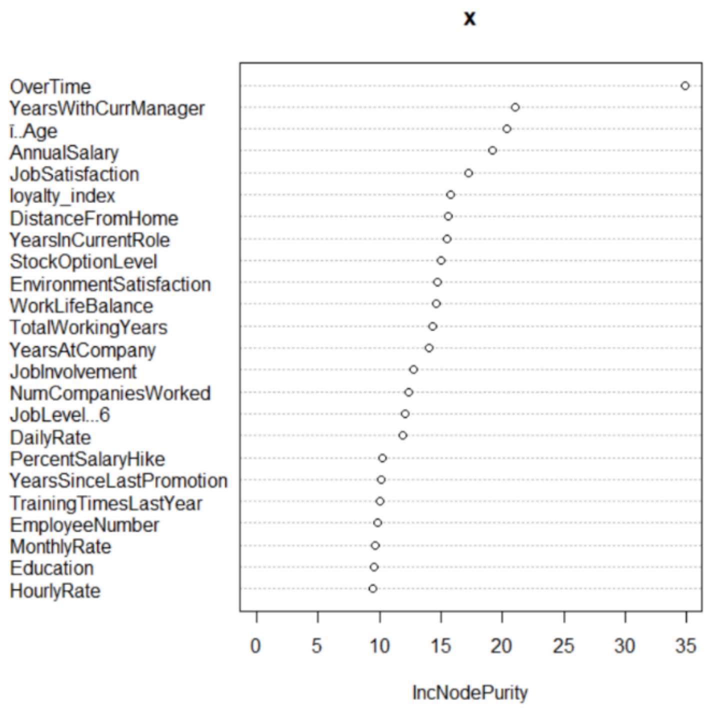


5. RANDOM FOREST WITH SELECTED FEATURES:
   Based on feature importance given above, few features were selected to implement in RF. Two variants were implemented, one included engineered features and other did not.
   Two variants of RF models are as following:
   1. M6: Random forest + Rose method for class imbalance + Selected features from raw data set | **AUC : 0.7733**
   2. M7: Random forest + Rose method for class imbalance + Selected features from raw data set + engineered feature | **AUC : 0.8569**

   M7 helped in increasing AUC which highlight importance of engineered features in increasing the model performance.

```
##Modelling using selective features from raw data set + engineered features
df7 <-df5 %>%
      select(Attrition,loyalty_index,AnnualSalary,ï..Age,DailyRate,
             OverTime,TotalWorkingYears,MonthlyRate,EmployeeNumber,
             HourlyRate,DistanceFromHome,YearsAtCompany,WorkLifeBalance,
             StockOptionLevel,PercentSalaryHike,NumCompaniesWorked,JobSatisfa
ction,
             EnvironmentSatisfaction,YearsWithCurrManager,JobInvolvement,
             TrainingTimesLastYear,YearsSinceLastPromotion,JobLevel...6,
             YearsInCurrentRole,Education)
```

Feature importance:



This indicates that engineered features like annual salary and loyalty index have very high weightage in modelling.

6. BOOSTED LOGISTIC REGRESSION

Logistic regression is a classification technique that evaluates probability values from a linear regression model. The model classifies out of sample data based on a threshold, usually set to p=0.5. Therefore, any probability value over 0.5 is classified as class 1 ('Yes' in this case) and class 0 ('No') otherwise. Boosted refers to the use of Adaboost algorithm for improving the model performance.

After preprocessing, we are left with a dataset comprising 41 features (from One Hot Encoding) and 1030 samples for training and 440 samples for testing.

We build a logistic regression model using the 'CARET' library which allows us to define the model, its hyperparameters and at the same time encodes the categorical columns.

Since there is a significant class imbalance in the dataset (83% of the values are from class 0 or 'No'), we perform stratified sampling accompanied with cross-validation to equalize the samples of each class in the training data. We obtain 77.7% accuracy on the test data (76.7% on the training data) from the model through this approach. We then calculate metrics like Accuracy, Sensitivity, and specificity to plot the ROC/AOC curve and evaluate the model performance.

Model parameters:

```
ctrl <- trainControl(method = "repeatedcv", number = 10,repeats = 10,
verboseIter = FALSE, sampling = "up")
```

Model definition:

```
model_ls_under <- caret::train(turnover ~ ., data = train_data, method
=   "LogitBoost",preProcess   =   c("scale",   center"),   metric   =
"Accuracy",trControl = ctrl)
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 299  28
         1  70  43

               Accuracy : 0.7773
                 95% CI : (0.7355, 0.8153)
    No Information Rate : 0.8386
    P-Value [Acc > NIR] : 0.9997

                  Kappa : 0.3357

 Mcnemar's Test P-Value : 3.449e-05

            Sensitivity : 0.8103
            Specificity : 0.6056
         Pos Pred Value : 0.9144
         Neg Pred Value : 0.3805
             Prevalence : 0.8386
         Detection Rate : 0.6795
   Detection Prevalence : 0.7432
      Balanced Accuracy : 0.7080
```

MODEL COMPARISON

ROC/AUC curve:

We can use the ROC (Receiver Operating Characteristic) curve and the AUC score (Area under curve) as well to evaluate the performance of the trained models (Logistic Regression and Random Forest). ROC curve plots the sensitivity of the model against the specificity of the model for different classification thresholds and AUC is the area under the ROC curve. A perfect classifier would have the AUC score of 1 i.e., the true positive rate and the true negative rate are both 1. In practice it is however more common to encounter ROC curves as shown below. A higher AOC curve is indicative of a better classifier.

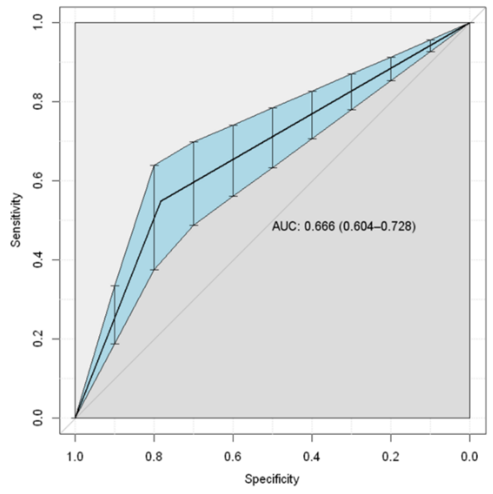


Figure: ROC curve for Logistic regression (AUC score: 0.666)

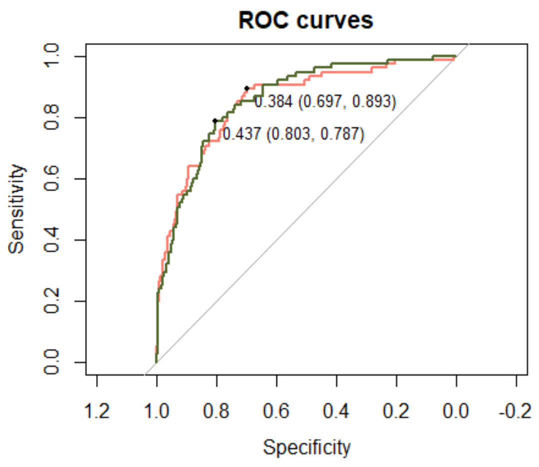


Figure: ROC curve for Random Forest (AUC score: 0.84)

AUC scores of the two models are 0.666 and 0.84, respectively. We can therefore conclude that Random Forest model is a better classifier than Logistic Regression.

CONCLUSION

The problem at hand was to classify an employee as a potential candidate for attrition. The dataset comprised 1470 employee records and career information of the employees recorded in 35 features. We explored the dataset and found that a few features were highly correlated with the turnover rate. For example, attrition increased for employees with lower incomes. Similar trend was observed between total work experience and attrition. We created new features using the correlated features to better predict the turnover risk. The R library used for the purpose automatically identified the features with highest prediction power. We have appended the model feature importance in Appendix 3.

The Random Forest model outperformed the Logistic Regression model by near 10%. The better performance can possibly be attributed to the greedy approach which Random Forests use to maximize the information gain which reduces the influence of multicollinearity among the features. Logistic regression performance however suffers from redundancy among the features.

The models however agree on certain features being important to employee retention. Features like *loyalty_index, AnnualSalary, OverTime, Age, TotalWorkingYears, DistanceFromHome, YearsAtCompany, PercentSalaryHike, WorkLifeBalance, JobSatisfaction, EnvironmentSatisfaction, JobInvolvement, YearsWithCurrManager, TrainingTimesLastYear, MaritalStatus* were among the most important features for both the models. The above variables provide an important insight for organizations to aid in developing a good employee retention policy.

REFERENCES

- **Dataset**: "IBM HR Analytics Employee Attrition & Performance"
(link: https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset)
- "Attrition in an Organization || Why Workers Quit?", (link: https://www.kaggle.com/janiobachmann/attrition-in-an-organization-why-workers-quit)
- "The caret Package", (link: https://topepo.github.io/caret/)
- "CRAN: The R Manuals"
- "Practical Guide to deal with Imbalanced Classification Problems in R", (link: https://www.analyticsvidhya.com/blog/2016/03/practical-guide-deal-imbalanced-classification-problems/)

APPENDICIES

Appendix 1 [EDA]

Appendix 2

Data Description

	Feature Name	Datatype	[Number of samples] Sample Values
1	Age	num	[1470] 41 49 37 33 27 32 59 30 38 36 ...
2	Attrition(**Dependent Variable)**	Factor w/ 2 levels "No","Yes"	[1470] 2 1 2 1 1 1 1 1 1 1 ...
3	BusinessTravel	Factor w/ 3 levels "Non-Travel","Travel_Frequently",..	[1470] 3 2 3 2 3 2 3 3 2 3 ...
4	DailyRate	num	[1470] 1102 279 1373 1392 591 ...
5	Department	chr	[1470] "Sales" "Research & Development" "Research & Development" "Research & Development" ...
6	DistanceFromHome	num	[1470] 1 8 2 3 2 2 3 24 23 27 ...
7	Education	Factor w/ 5 levels "1","2","3","4",..	[1470] 2 1 2 4 1 2 3 1 3 3 ...
8	EducationField	chr	[1470] "Life Sciences" "Life Sciences" "Other" "Life Sciences" ...
9	EmployeeCount	num	[1470] 1 1 1 1 1 1 1 1 1 1 ...
10	EmployeeNumber	num	[1470] 1 2 4 5 7 8 10 11 12 13 ...
11	EnvironmentSatisfaction	num	[1470] 2 3 4 4 1 4 3 4 4 3 ...
12	Gender	Factor w/ 2 levels "Female","Male"	[1470] 1 2 2 1 2 2 1 2 2 2 ...
13	HourlyRate	num	[1470] 94 61 92 56 40 79 81 67 44 94 ...
14	JobInvolvement	num	[1470] 3 2 2 3 3 3 4 3 2 3 ...
15	JobLevel	Factor w/ 5 levels "1","2","3","4",..	[1470] 2 2 1 1 1 1 1 1 3 2 ...
16	JobRole	chr	[1470] "Sales Executive" "Research Scientist" "Laboratory Technician" "Research Scientist" ...
17	JobSatisfaction	num	[1470] 4 2 3 3 2 4 1 3 3 3 ...
18	MaritalStatus	chr	[1470] "Single" "Married" "Single" "Married" ...
19	MonthlyIncome	num	[1470] 5993 5130 2090 2909 3468 ...
20	MonthlyRate	num	[1470] 19479 24907 2396 23159 16632 ...
21	NumCompaniesWorked	num	[1470] 8 1 6 1 9 0 4 1 0 6 ...
22	Over18	chr	[1470] "Y" "Y" "Y" "Y" ...

23	OverTime	chr	[1470] "Yes" "No" "Yes" "Yes" ...
24	PercentSalaryHike	num	[1470] 11 23 15 11 12 13 20 22 21 13 ...
25	PerformanceRating	num	[1470] 3 4 3 3 3 3 4 4 4 3 ...
26	RelationshipSatisfaction	num	[1470] 1 4 2 3 4 3 1 2 2 2 ...
27	StandardHours	num	[1470] 80 80 80 80 80 80 80 80 80 80 ...
28	StockOptionLevel	num	[1470] 0 1 0 0 1 0 3 1 0 2 ...
29	TotalWorkingYears	num	[1470] 8 10 7 8 6 8 12 1 10 17 ...
30	TrainingTimesLastYear	num	[1470] 0 3 3 3 3 2 3 2 2 3 ...
31	WorkLifeBalance	num	[1470] 1 3 3 3 3 2 2 3 3 2 ...
32	YearsAtCompany	num	[1470] 6 10 0 8 2 7 1 1 9 7 ...
33	YearsInCurrentRole	num	[1470] 4 7 0 7 2 7 0 0 7 7 ...
34	YearsSinceLastPromotion	num	[1470] 0 1 0 3 2 3 0 0 1 7 ...
35	YearsWithCurrManager	num	[1470] 5 7 0 0 2 6 0 0 8 7 ...
36	median_comp	num	[1470] 5340 5340 2670 2670 2670 2670 2670 2670 9980 5340 ...
37	mean_comp	num	[1470] 5502 5502 2787 2787 2787 ...
38	compa_median_ratio	num	[1470] 1.122 0.961 0.783 1.09 1.299 ...
39	compa_mean_ratio	num	[1470] 1.089 0.932 0.75 1.044 1.244 ...
40	compa_median_level	chr	[1470] "Above" "Below" "Below" "Above" ...
41	compa_mean_level	chr	[1470] "Above" "Below" "Below" "Above" ...

Appendix 3

Model feature importance:

Logistic regression	**Random Forest**
MonthlyIncome	loyalty_index
StockOptionLevel	AnnualSalary
OverTime	OverTime
NumCompaniesWorked	Age
RelationshipSatisfaction	TotalWorkingYears
JobLevel	DistanceFromHome
DistanceFromHome	YearsAtCompany
YearsWithCurrManager	PercentSalaryHike
BusinessTravelTravel	WorkLifeBalance
JobSatisfaction	JobSatisfaction
Age	EnvironmentSatisfaction
compa_median_ratio	JobInvolvement
compa_mean_ratio	YearsWithCurrManager
JobInvolvement	TrainingTimesLastYear
EnvironmentSatisfaction	MaritalStatus.V1_Single
YearsSinceLastPromotion	YearsWithCurrManager

Appendix 4 : CARET (https://topepo.github.io/caret/index.html)

“The **caret** package (short for Classification And REgression Training) contains several functions that streamline the model training process for complex regression and classification problems.“

The package automatically takes care of encoding categorical variables, normalization of features, imputing missing values etc. It flags predictors which have multicollinearity for removal. It also allows for conducting repeated k-fold cross-validation in conjunction with stratified sampling to handle class imbalances in classification problems.

Other functions include:

- Data Splitting
- Feature Selection
- Model Hyperparameter tuning
- Variable importance estimation

The package supports multiple Classification and Regression models which are called by specifying the ‘method’ parameter in the ‘train’ function. The hyper-parameters of the model can be tuned by using the ‘trainControl’ function.

We used the package to automate pre-processing and hyper-parameter tuning, consequently build several models and evaluate the results. This allowed us to cover a wide spectrum of parameters and helped in choosing the best ones for the model.

Appendix 4: ROSE (https://cran.r-project.org/web/packages/ROSE/ROSE.pdf)

Title ROSE: Random Over-Sampling Examples

Description The package provides functions to deal with binary classification problems in the presence of imbalanced classes. Synthetic balanced samples are generated according to ROSE (Menardi and Torelli, 2013). Functions that implement more traditional remedies to the class imbalance are also provided, as well as different metrics to evaluate a learner accuracy. These are estimated by holdout, bootstrap or cross-validation methods.

Appendix 5: Package ‘randomForest’ (https://cran.r-project.org/web/packages/randomForest/randomForest.pdf)
Description: Classification and regression based on a forest of trees using random inputs, based on Breiman (2001)

Appendix 5: Package ‘randomForest’
(https://www.rdocumentation.org/packages/rpart/versions/4.1-15/topics/rpart)
Function rpart: Recursive Partitioning and Regression Trees