Predictive Analytics for Employee Attrition & Performance

By

Akhil Reddy Dereddy

MIS, 2022

A creative component submitted to the graduate facility in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE

Major: Information Systems

The student author, whose presentation was approved by the program of the study committee, is solely responsible for the content of the report. The Graduate College will ensure this report is globally accessible and will not permit alteration after a degree is conferred.

Iowa State University, Ames, Iowa 2022

Table of Contents

List of Figures

Abstract

Today's world, every company is interested in predicting employee performance and attrition based on employee performance evaluations conducted quarterly or semi-annually. The company's growth is determined by how talented their current employees are. However, predictive analysis of employee performance and attrition gives the company an advantage in analyzing and making business decisions based on the results. Predictive analytics is a machine learning technique that uses techniques such as classification to predict future performance or a data point we request to be predicted using 80 percent train data and 20% test data. Furthermore, classification is a data mining technique that aligns attributes in a dataset to target specific groups or categories. Binary classification techniques such as k-nearest neighbors, support vector machines, and decision trees are used to predict employee attrition. In this paper, we would like to apply a random forest classification technique that employs the ensemble classification method to predict from the results of multiple decision trees.

Random forest classification, in particular, emphasizes high predictive accuracy and low computational cost. Finally, random forest is recommended alongside binary classification techniques such as decision tree, KNN, and Naive Bayes because it outperforms due to high predictive accuracy.

Acknowledgement

First and foremost, I would like to express my gratitude and heartfelt appreciation to my advisor, Dr. Antony Townsend, for his assistance in writing the project and approving my proposal. Furthermore, I would like to thank my faculty and staff members at Iowa State University for their assistance throughout my degree program. Last but not least, I would like to thank my parents, Mr. Srinivas Reddy, Mrs. Uma Devi, and cousin brother (Nani), who have been my role models throughout my life with their love, care, financial support, and mental health support towards my master's program.

Introduction

Companies all over the world have been grappling with the challenges of retaining talent and managing loss through attrition, such as retirement or existing employees resigning from their current positions. Employee attrition can result in a potential loss of projects or workflow of a specific project at a company, causing an imbalance in human resource planning and a loss of team harmony and social purposes. The research work's goal is to investigate various factors such as salary, growth opportunities, facilities, policies, procedures, recognition, appreciation, and suggestions that will aid in determining the attrition rate in organizations and factors related to their retention. This study shows how to predict which employees are most likely to leave the company.

In this research project we will look at the prediction of attrition for a test dataset and train the existing HR IBM dataset. However, we have leveraged machine learning models to find the precision of the prediction and accuracy of the prediction. Moreover, some other important metrics to measure the machine learning model. We have encountered related work by

Data Sources

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

The dataset has been acquired from Kaggle, which is provided by IBM HR department.

Figure 1 below provides a description of the data.

```
In [72]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 35 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Age                       1470 non-null   int64
 1   Attrition                 1470 non-null   object
 2   BusinessTravel            1470 non-null   object
 3   DailyRate                 1470 non-null   int64
 4   Department                1470 non-null   object
 5   DistanceFromHome          1470 non-null   int64
 6   Education                 1470 non-null   int64
 7   EducationField            1470 non-null   object
 8   EmployeeCount             1470 non-null   int64
 9   EmployeeNumber            1470 non-null   int64
 10  EnvironmentSatisfaction   1470 non-null   int64
 11  Gender                    1470 non-null   object
 12  HourlyRate                1470 non-null   int64
 13  JobInvolvement            1470 non-null   int64
 14  JobLevel                  1470 non-null   int64
 15  JobRole                   1470 non-null   object
 16  JobSatisfaction           1470 non-null   int64
 17  MaritalStatus             1470 non-null   object
 18  MonthlyIncome             1470 non-null   int64
 19  MonthlyRate               1470 non-null   int64
 20  NumCompaniesWorked        1470 non-null   int64
 21  Over18                    1470 non-null   object
 22  OverTime                  1470 non-null   object
 23  PercentSalaryHike         1470 non-null   int64
 24  PerformanceRating         1470 non-null   int64
 25  RelationshipSatisfaction  1470 non-null   int64
 26  StandardHours             1470 non-null   int64
 27  StockOptionLevel          1470 non-null   int64
 28  TotalWorkingYears         1470 non-null   int64
 29  TrainingTimesLastYear     1470 non-null   int64
 30  WorkLifeBalance           1470 non-null   int64
 31  YearsAtCompany            1470 non-null   int64
 32  YearsInCurrentRole        1470 non-null   int64
 33  YearsSinceLastPromotion   1470 non-null   int64
 34  YearsWithCurrManager      1470 non-null   int64
dtypes: int64(26), object(9)
memory usage: 402.1+ KB
```

Fig: 1 Detailed Description of IBM HR Dataset

The dataset is composed of 35 columns and 1470 rows. Figure 2 clearly shows that employee attrition is a "0," it is a "Yes," otherwise it is a "1" it is a "No." There are 1233 "No" and 237 "Yes" responses among 1470 observations. The attrition rate is 237/1470, or

16.1%, indicating that the dataset is unbalanced, with a significantly greater number of observations belonging to class 1 (No) than class 0 (Yes). Because machine learning algorithms are typically designed to improve accuracy by reducing errors, the traditional accuracy performance metric is misleading. As a result, we will not take it into account for class distribution or class balance. Finally, other metrics for assessing the performance of the project's machine learning models will be considered.

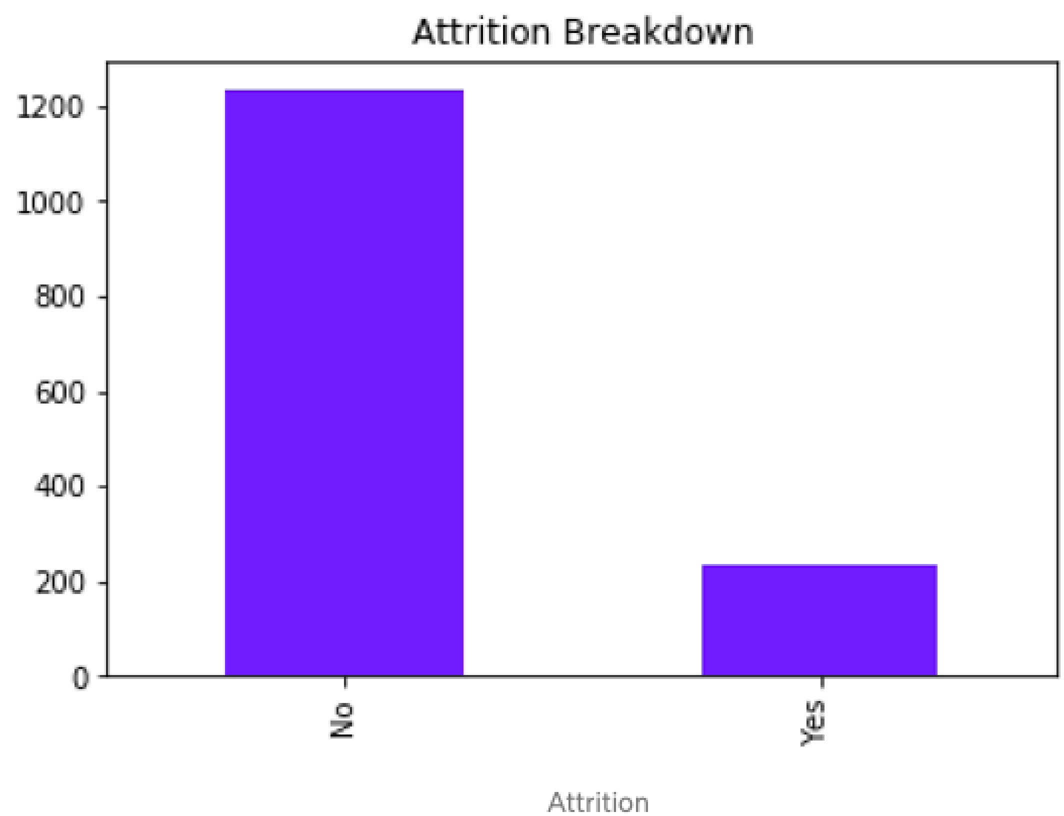

Fig: 2 Attrition Breakdown of the dataset

The columns include Age, Attrition, BusinessTravel, DailyRate, Department, DistanceFromHome, Education, EducationField, EmployeeCount, EmployeeNumber, EnvironmentSatisfaction, Gender, HourlyRate, JobInvolvement, JobLevel, JobRole, JobSatisfaction, MaritalStatus, MonthlyIncome, MonthlyRate, NumCompaniesWorked, Over18, OverTime, PercentSalaryHike, PerformanceRating, RelationshipSatisfaction, StandardHours, StockOptionLevel, TotalWorkingYears, TiningTimesLastYear, WorkLifeBalance, YearsAtCompany, YearsInCurrentRole, YearsSinceLastPromotion, YearsWithCurrentManager. Figure 3 shows numerical values encoded with respect to a specific category available in the dataset for a specific feature.

The description of categorical and numerical columns are in Figure 3 and Figure 4, respectively :

Categorical Colum Values	
Attrition	Attrition refers to employees who either resign/retire(1) or do not resign/retire(0)
Business Travel	It categorizes an employee's need for business travel into three categories: No Travel(1), Travel Frequently(2), and Travel Rarely(3)
Department	It categorizes the company's departments as HR(1), R&D(2), and Sales(3)
Education Field	It categorizes an employee's educational background into six categories: HR(1), Life Sciences(2), Marketing(3), Medical Sciences(4), Others(5), and Technical (6)
Gender	It divides an employee's gender into two categories: Male(1) and Female(2) (2)
JobRole	It describes the job role in nine categories
MaritalStatus	It describes the marital status of an employee
Over18	It describes if an emploee if over 18 years of age
OverTime	It describes if an employee worked over time

Fig: 3 Description of Categorical Columns in Dataset

Numerical Columns Description	
Age	It describes the age of an employee
DailyRate	It describes the daily pay rate of an employee
DistanceFromHome	It describes how far is the employee home address
Education	It describes the highest level of education of an employee
EmployeeCount	It describes the count of an employee
EmployeeNumber	It is the unique number to recognise and employee
EnvironmentSatisfaction	It describes how satisfied is the employee working at the company
HourlyRate	It describes the hourly pay rate on the employee
JobInvolvement	It describes the degree how involved a employee is in current job
JobLevel	It describes the job level of an employee
JobSatisfaction	It describes how satisfied the employee is in current job
MonthlyIncome	It describes the monthly income of an employee
MonthlyRate	It describes the monthly pay rate of an employee
NumCompaniesWorked	It describes the number of companies previously worked by an employee
PercentSalaryHike	It describes the percentage of salary hike of an employee
PerformanceRating	It describes rating provided to an employee based on performance by employee manager
RelationshipSatisfaction	It describes the rate of satisfication between employee and employer
StandardHours	It describes the standard hours allocaed to an employee
StockOptionLevel	It describes if an employee holds stock option
TotalWorkingYears	It describes total years worked by an employee
TrainingTimesLastYear	It describes number of months the employee was trained
WorkLifeBalance	It describes the balane between employee work and life
YearsAtCompany	It describes the number of years worked by an employee
YearsInCurrentRole	It describes the number of years the employee worked in current role
YearsSinceLastPromotion	It describes the number of years since the employee was promoted
YearsWithCurrentManager	It describes the number of years under the current manager

Fig: 4 Description of Numerical Columns

Data Profiling

Data profiling is a technique for viewing a dataset's basic statistics using Python's pandas library. Data profiling of the IBM HR dataset, as shown in Figure 6, provides an overview of the dataset variables, the count of values for each column, and so on. Figure 4 depicts the data profiling python code that we have attached.

```
In [ ]: pip install https://github.com/ydataai/pandas-profiling/archive/master.zip
        import pandas as pd
        from pandas_profiling import ProfileReport
        import os
        os.getcwd()

In [73]: df = pd.read_csv('C:\\Users\\Akhil.Dereddy\\Desktop\\archive\\HR-Employee-Attrition.csv')

In [74]: df.info()

In [20]: profile = ProfileReport(df, title="Pandas Profiling Report", explorative=True)

In [21]: profile.to_file("your_report.html")

Summarize dataset: 100% 273/273 [01:43<00:00, 1.01it/s, Completed]
Generate report structure: 100% 1/1 [00:26<00:00, 26.32s/it]
Render HTML: 100% 1/1 [00:21<00:00, 21.56s/it]
Export report to file: 100% 1/1 [00:00<00:00, 5.88it/s]
```

Fig: 5 Python program to create data profiling report

Figure 6 of the data profiling report overview of the IBM HR dataset is shown below. I've converted the report into an HTML file. There are 15 numerical variables, 3 boolean variables, and 17 categorical variables in the dataset. The dataset statistics include 0 missing cells, 0% missing cells (percent), 0 duplicate rows, 1.1 MB of memory, and an average record size of 796.8 B in memory.

Overview

Overview | Alerts 74 | Reproduction

Dataset statistics

Number of variables	35
Number of observations	1470
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	1.1 MiB
Average record size in memory	796.8 B

Variable types

Numeric	15
Boolean	3
Categorical	17

Fig: 6 Overview of the Report generated by Data Profiling

Correlation analysis measures the strength of a relationship between two item sets, which can be dependent and independent variables, or two independent variables. His relationship is numerically determined by a decimal value known as the correlation coefficient. The correlation coefficient can be determined within a specific predefined range, as well as its strength and direction. A correlation with a positive sign means that the two variables are linked together in a positive way, while a correlation with a negative sign means that they are linked together in a negative way "(Kumar, S., Chong, I. (2018). Correlation Analysis to Identify the Effective Data in Machine Learning: Prediction of Depressive Disorder and Emotion States, 5 of 24.)". The correlation plot generated by the pandas profiling is shown below in Figure 6. Moreover, the independent coefficients are EmployeCount and StandardHours columns.

We can see from the correlation plot in Figure 7 that many columns appear to be poorly correlated with one another. In general, when developing a predictive model, it is preferable to train the model with features that are not overly correlated with one another, so that we do not have to deal with redundant features. If we have a large number of correlated features, we could use a technique like Principal Component Analysis (PCA) to reduce the feature space.

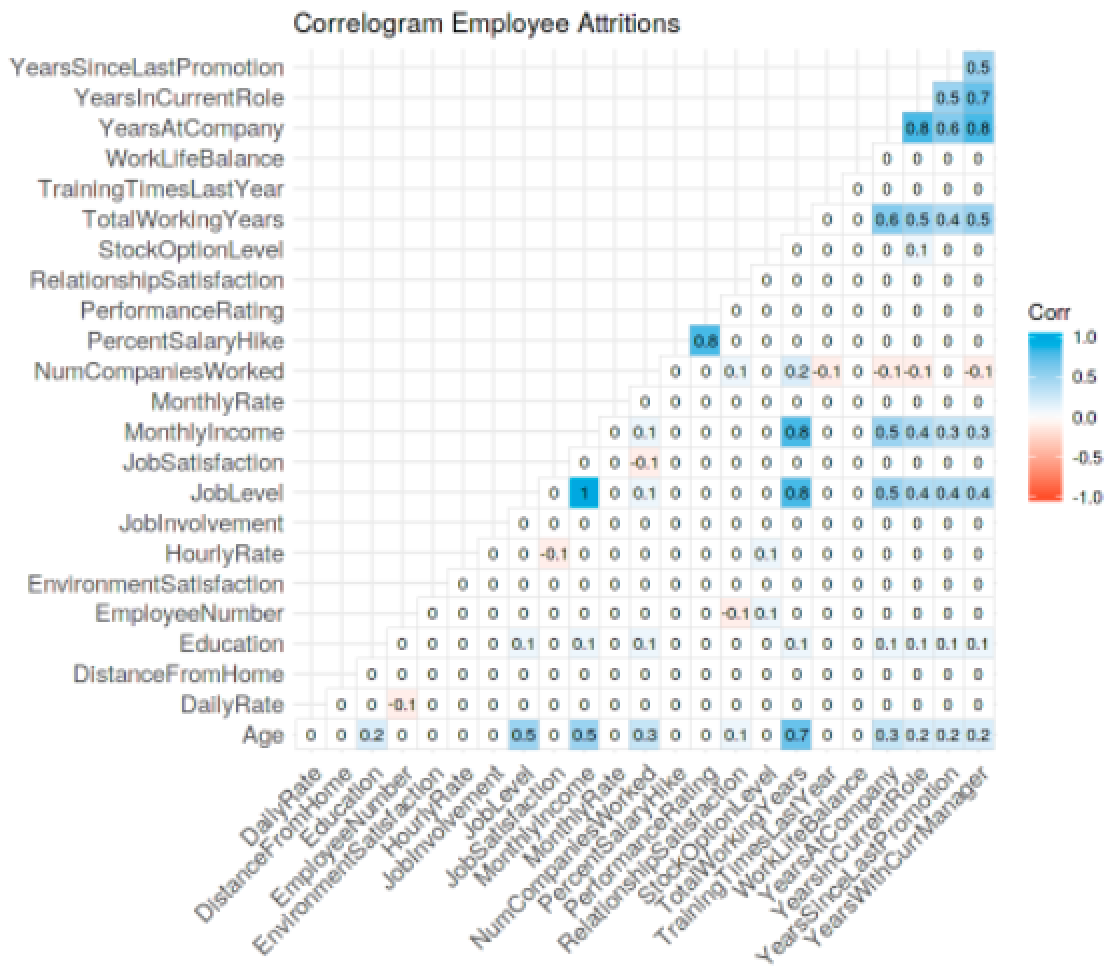

Fig: 7 Correlation Plot

Figure 8 shows that the dataset contains no missing values; if we look closely, each column is on the x-axis and the count of the rows is on the y-axis; the count of each column is at 1470 y-axis points, indicating that there are no missing values in a specific feature. Furthermore, 1470 y-axis points are touched across all columns.

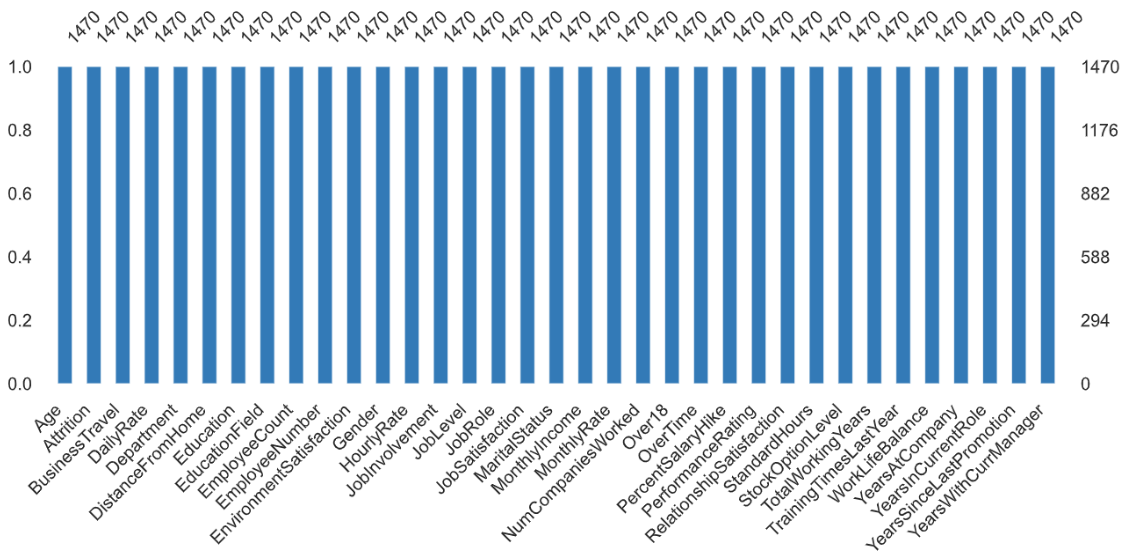

Fig: 8 Plot of count of each column values.

Exploratory Data Analysis

I ran exploratory data analysis on the IBM HR dataset to identify columns with high correlation and determine which columns could be used to build a machine learning model comparison. The chart below explains individual department attrition rates based on historical data. We can see in the below figure 9 that the "Research & Development" department has 961 attrition employees, the "Sales" department has 446 attrition employees, and the remaining employees are from the "HR" department.

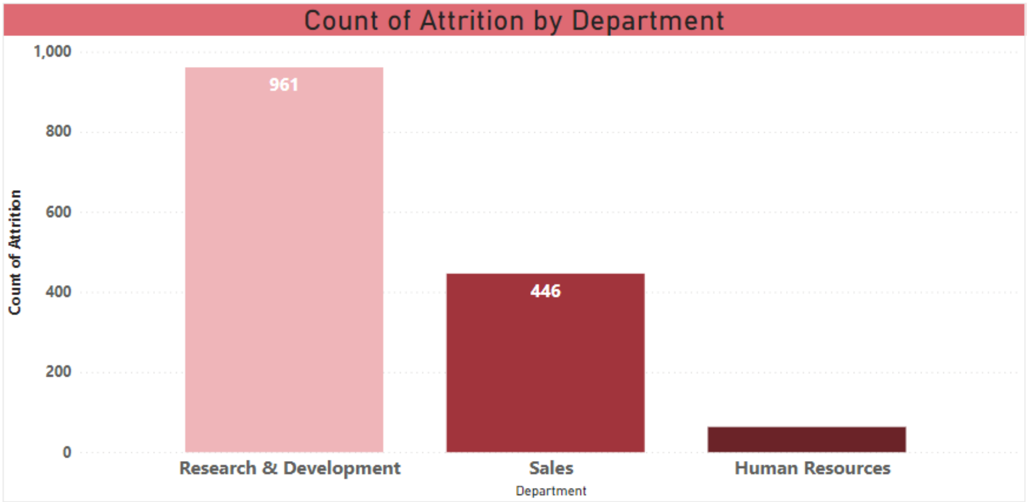

Fig: 9 Graph for Department by Attrition

As can be seen in Figure 10, the gender distribution plot by attrition count is depicted. The males have a count of 882 and the females have a count of 588.

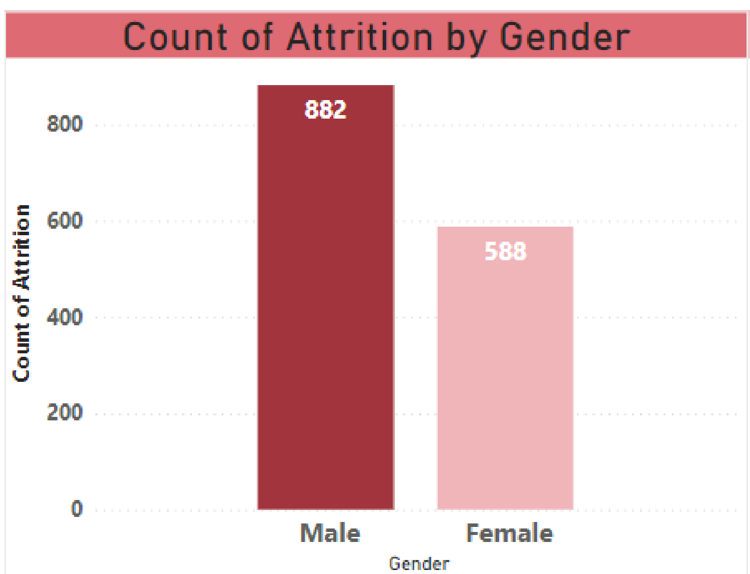

Fig: 10 Graph for Gender vs Attrition.

We can see in the figure 11 that the Married couples had 673 highest attrition count, whereas singles had 470 with second highest attrition count and divorced had 327 with lowest attrition rate .As you can see, below is the Marital Status distribution plot by attrition count.

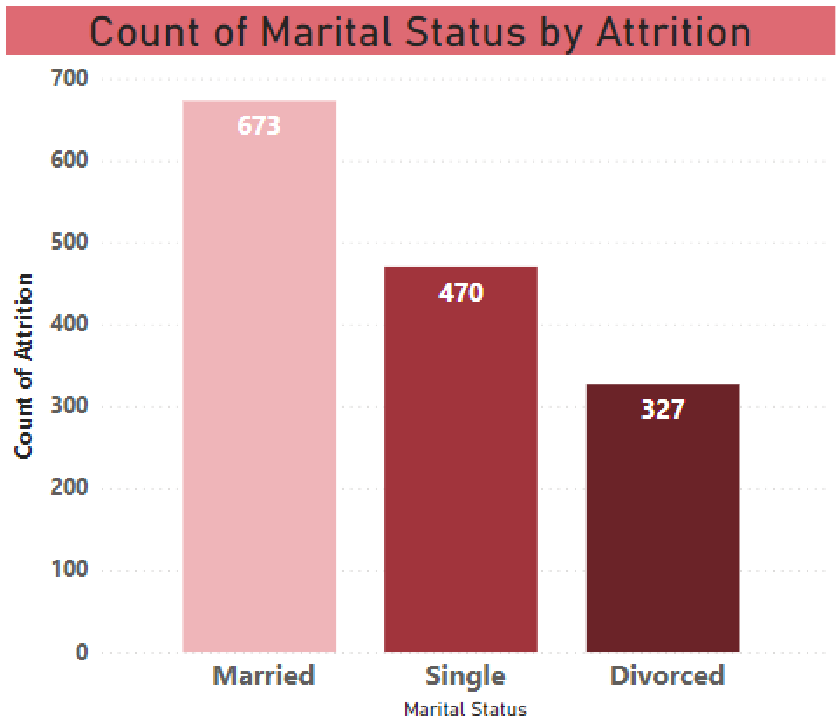

Fig: 11 Graph for Marital Status vs Attrition

As shown in the figure 12 below, attrition is highest between the ages of 28 and 32. With increasing age, the attrition rate began to fall as people sought stability in their jobs. Employees leave the company at a much younger age, between the ages of 18 and 20, because they are exploring at that age. It reaches a break even point at the age of 21.

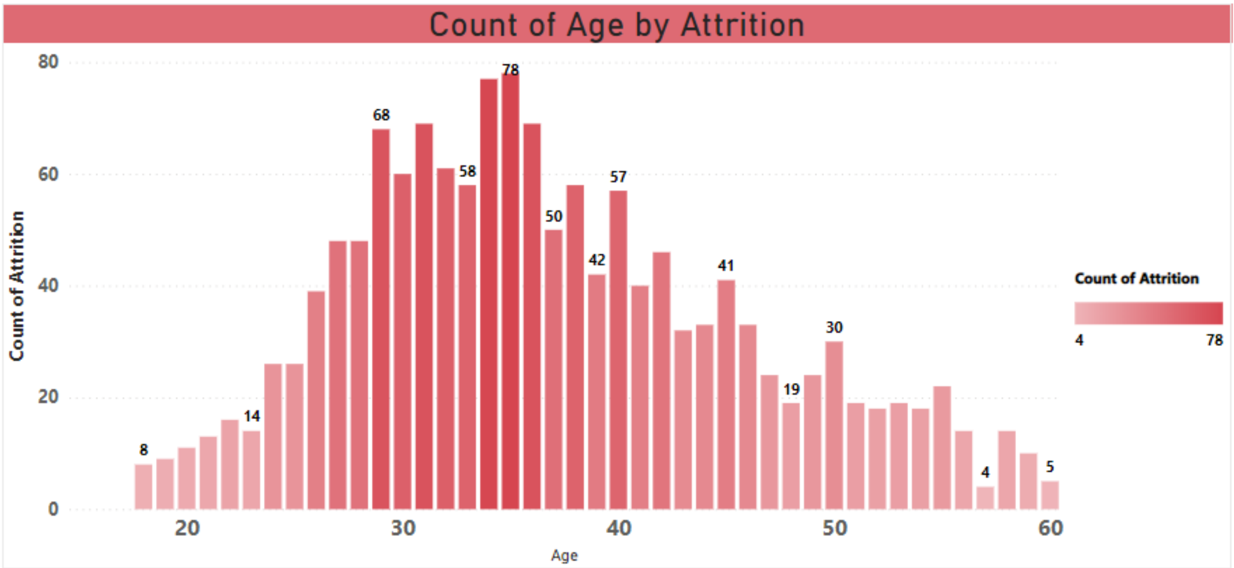

Fig: 12 Graph for Age vs Attrition

Data Preprocessing

Data preprocessing is an important step in the data mining process that involves manipulating or dropping data before it is used to ensure or improve performance. The adage "garbage in, garbage out" is especially appropriate for data mining and machine learning projects. Data preprocessing is a step in building machine learning models in which data is encoded with labels. Outlier detection, oversampling, and normalization make it simple for machine learning models to train the dataset.

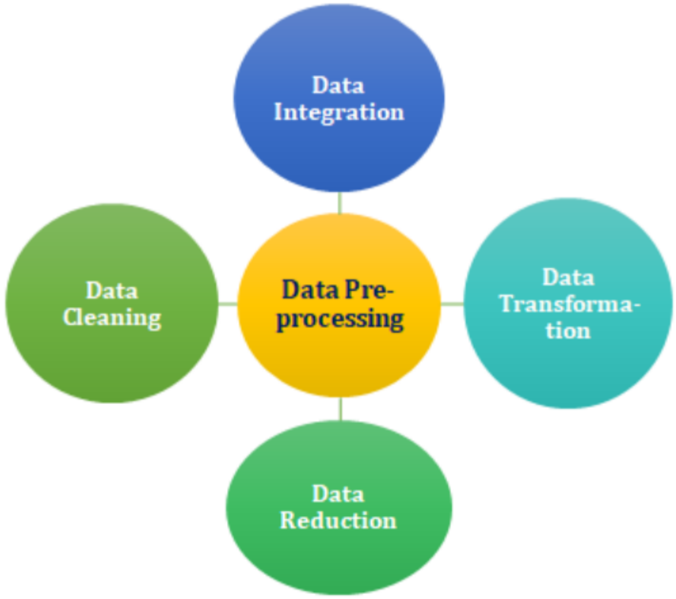

Fig : 13 Four different ways to perform data pre-processing(Keerthana. (2021, June 6). DATA PREPROCESSING TECHNIQUES. Data preprocessing is a Data Mining… | by Keerthana | AlmaBetter | Medium. Medium; medium.com. https://medium.com/almabetter/data-preprocessing-techniques-6ea145684812)

First, I'm removing some columns with low correlation while keeping others with high correlation by following the rule of thumb with 0.8 as highly correlated. Below figure shows the table with highly correlated columns used for feature selection and label encoding of the remaining features in the dataset.

Highly Correlated Features(> 0.8)		
Feature in X-Axis	Feature in Y-Axis	Correletion Coefficient
YearsWithCurrentManager	YearsAtCompany	0.8
YearsInCurrentRole	YearsAtCompany	0.8
PerformanceRating	PercentSalaryHike	0.8
TotalWorkingYears	MonthlyIncome	0.8
MonthlyIncome	JobLevel	0.8

Fig : 14 Highly correlated features greater than 0.8

Dropping unnecessary columns with low correlation, checking for missing values, and checking for duplicates are all part of data cleaning. We won't be doing any data integration because we only have one data source. Label encoding, outlier detection, normalization, and smoothing are all examples of data transformation. Data reduction is the process of reducing data dimensionality, that is, reducing features by selecting highly correlated variables, as well as reducing numerosity. Furthermore, I have done data pre-processing with Python libraries such as pandas.

In [80]:
```
new_df = df.drop(['BusinessTravel', 'DailyRate', 'DistanceFromHome', 'Education', 'EmployeeCount', 'EnvironmentSatisfaction',
        'HourlyRate', 'JobInvolvement', 'JobSatisfaction', 'MonthlyRate', 'Over18', 'StandardHours'], axis=1)
new_df.head(5)
```

Out[80]:

	Age	Attrition	Department	EducationField	EmployeeNumber	Gender	JobLevel	JobRole	MaritalStatus	MonthlyIncome	...	PerformanceRating	Relationsl
0	41	Yes	Sales	Life Sciences	1	Female	2	Sales Executive	Single	5993	...	3	
1	49	No	Research & Development	Life Sciences	2	Male	2	Research Scientist	Married	5130	...	4	
2	37	Yes	Research & Development	Other	4	Male	1	Laboratory Technician	Single	2090	...	3	
3	33	No	Research & Development	Life Sciences	5	Female	1	Research Scientist	Married	2909	...	3	
4	27	No	Research & Development	Medical	7	Male	1	Laboratory Technician	Married	3468	...	3	

5 rows × 23 columns

Fig : 15 Removing columns with less correlation

From the above figure we can say that we have dropped the columns with less correlation. Now we must perform label encoding of features like Attrition, Department, EducationField. In addition, the remaining highly correlated features listed above table have numerical values. Label encoding is to convert categorical values into numerical values for feature modal analysis using machine learning models. Finally, the data frame looks something like the figure 16 below after label encoding.

In [107]:
```
t_map2 = {'Human Resources':1, 'Life Sciences':2, 'Marketing':3, 'Medical':4, 'Other':5, 'Technical Degree':6}
new_df1['EducationField'] = new_df1["EducationField"].apply(lambda x: t_map2[x])
```

In [108]:
```
new_df1.head(5)
```

Out[108]:

	Age	Attrition	Department	EducationField	EmployeeNumber	Gender	JobLevel	JobRole	MaritalStatus	MonthlyIncome	...	PerformanceRating	Relationsh
0	41	1	3	2	1	Female	2	Sales Executive	Single	5993	...	3	
1	49	0	2	2	2	Male	2	Research Scientist	Married	5130	...	4	
2	37	1	2	5	4	Male	1	Laboratory Technician	Single	2090	...	3	
3	33	0	2	2	5	Female	1	Research Scientist	Married	2909	...	3	
4	27	0	2	4	7	Male	1	Laboratory Technician	Married	3468	...	3	

5 rows × 23 columns

Fig: 16 After encoding all the categorical column values

As we have transformed all the features, next we can start building the machine learning models. I have constructed a new dataframe which has only columns with high correlation, below figure has python code for it.

In [119]:
```
df2=new_df1[['Age', 'Attrition', 'Department', 'EducationField', 'JobLevel', 'MonthlyIncome', 'PercentSalaryHike',
             'PerformanceRating', 'TotalWorkingYears', 'YearsAtCompany', 'YearsInCurrentRole',
             'YearsSinceLastPromotion', 'YearsWithCurrManager']].copy()
df2.head(5)
```

Out[119]:

	Age	Attrition	Department	EducationField	JobLevel	MonthlyIncome	PercentSalaryHike	PerformanceRating	TotalWorkingYears	YearsAtCompany	YearsInCu
0	41	1	3	2	2	5993	11	3	8	6	
1	49	0	2	2	2	5130	23	4	10	10	
2	37	1	2	5	1	2090	15	3	7	0	
3	33	0	2	2	1	2909	11	3	8	8	
4	27	0	2	4	1	3468	12	3	6	2	

Fig: 17 New Dataframe with high correlation columns

Machine Learning Tools and Techniques

The dependent variable is whether or not the employee will leave the company. Cross validation and tuning techniques were also used during the model building process to make sure that the models built work well when they are used to predict.

Commercial classification models include the following:

- Logistic Regression
- Decision Tree
- Random Forest
- kNN

Model performance metrics are used to evaluate the performance of a machine learning model:

- To measure performance, machine learning models must be evaluated using model performance metrics. Because the dataset appears to be imbalanced, with attrition rates as low as 16%, selecting the right model performance measure is critical. As a result, model accuracy alone cannot determine the robustness of a machine learning model. Based on a confusion matrix created for training dataset predictions:

	Negetive(Predicted)	Positive(Predicted)
Negative(Observed)	True Negative(TN)	False Positive(FP)
Positive(Observed)	False Negative(FN)	True Positive(TP)

Fig: 18 Confusion matrix for training dataset predictions

- Accuracy is defined as the number of correct predictions generated by the machine learning model by the total number of datapoints. The best accuracy is 100 percent, which indicates that all predictions are correct. Given our dataset's conversion rate of 16%, accuracy is not a valid measure of model performance. Even if all of our predictions are incorrect, our model's accuracy will be 84%. As a result, additional model performance measures are included.
- Sensitivity is calculated by dividing the number of correct positive predictions by the total number of positives. It is also known as the true positive rate or the recall rate. In our dataset, the ratio of actual customers who generated revenue to the total number of customers predicted to generate revenue is provided.
- The specificity, also known as the True Negative rate, is the number of correct negative predictions divided by the total number of negatives. In our dataset, it represents the ratio of actual customers who will not generate revenue divided by the number of customers who are predicted to generate revenue.
- Precision is determined by dividing the number of correct positive predictions by the total number of positive predictions. What percentage of customers who generated revenue as customers actually generated revenue in our dataset? If the precision is low, it means the model has a high number of false positives.
- The F1 Score is a combined precision and recall measure of a model's accuracy. A high F1 Score indicates that the model produced very few false negatives, indicating that we are correctly identifying real threats and are not influenced by false alarms.

When the F1 Score is 1, the model is considered successful; when it is 0, the model is considered a failure.

- The ROC chart and Area Under the Curve (AUC) are plots of the difference between 1 and specificity on the x-axis and sensitivity on the y-axis. The AUC of a random classifier is 50%, while that of a perfect classifier is 100%. AUC greater than 70% is preferable for improved performance.

Comparison of Machine Learning Models

Firstly, I have trained and tested using a logistic regression machine learning model. I have imported multiple modules from scikit learn python library. Below figure shows the details of all the modules imported.

```
In [112]: from sklearn.model_selection import train_test_split
          from sklearn.linear_model import LogisticRegression
          from sklearn.tree import DecisionTreeClassifier
          from sklearn.svm import SVC
          from sklearn.model_selection import RandomizedSearchCV
          from sklearn.ensemble import BaggingClassifier,RandomForestClassifier
          from sklearn.metrics import classification_report
          from sklearn.metrics import f1_score, classification_report, accuracy_score, roc_auc_score, roc_curve, confusion_matrix
          from sklearn.model_selection import cross_val_score,cross_val_predict
```

Fig : 19 Python machine learning models using scikit learn module

Training and testing the logistic regression model calculates the prediction accuracy, training accuracy, f1 score, and roc auc score. In addition, precision is defined as the ratio of True Positive (TP) / True Positive(TP) + False Positive(FP), and it is the ability of the machine learning classification model to not label a sample as a positive if the sample is negative.

Recall is defined as the ratio of True Positive(TP) and True Positive(TP) + False Negative(FN,) and it is the ability of the machine learning classification model to find all the positive values. F1-score ranges from o to 1, whereas weighted mean of precision and recall weights more than precision. Support metric is the count of each class in y_test.

Below are the results of logistic regression.

```
In [132]: #Logistic Regression
logreg = LogisticRegression()
logreg.fit(x_train, y_train)
logreg_predict = logreg.predict(x_test)
logreg_predict1 = logreg.predict(x_train)
print('train accuracy ', accuracy_score(y_train, logreg_predict1))
print('f1 score ',f1_score(y_test, logreg_predict, average=None))
print('accuracy score ', accuracy_score(y_test, logreg_predict))
print('roc auc score ',roc_auc_score(y_test,logreg_predict))
print(classification_report(y_test, logreg_predict))
```

```
train accuracy  0.8445092322643343
f1 score  [0.90434783 0.        ]
accuracy score  0.8253968253968254
roc auc score  0.5
              precision    recall  f1-score   support

           0       0.83      1.00      0.90       364
           1       0.00      0.00      0.00        77

    accuracy                           0.83       441
   macro avg       0.41      0.50      0.45       441
weighted avg       0.68      0.83      0.75       441
```

Fig : 20 Results using Logistic Regression Model

Secondly, I have trained and tested the dataset using decision tree classification machine learning technique. Training and Testing the decision tree classification model calculates the prediction accuracy, training accuracy, f1 score, and roc auc score. Below are the results of decision tree classification.

```
In [139]: #DecisionTree Classifier
gm_gi = DecisionTreeClassifier(criterion='gini', random_state=100, max_depth=4, min_samples_leaf=15)
gm_gi.fit(x_train, y_train)
gm_gi_predict = gm_gi.predict(x_test)
gm_gi_predict1 = gm_gi.predict(x_train)
print('train accuracy ', accuracy_score(y_train, gm_gi_predict1))
print('f1 score ',f1_score(y_test, gm_gi_predict, average=None))
print('accuracy score ', accuracy_score(y_test, gm_gi_predict))
print('roc auc score ',roc_auc_score(y_test,gm_gi_predict))
print(classification_report(y_test, gm_gi_predict))
```

```
train accuracy  0.8542274052478134
f1 score  [0.89672544 0.06818182]
accuracy score  0.8140589569160998
roc auc score  0.5084915084915085
              precision    recall  f1-score   support

           0       0.83      0.98      0.90       364
           1       0.27      0.04      0.07        77

    accuracy                           0.81       441
   macro avg       0.55      0.51      0.48       441
weighted avg       0.73      0.81      0.75       441
```

Fig: 21 Results using gini as criterion for Decision Tree Classification

```
In [140]: gm_en = DecisionTreeClassifier(criterion='entropy', random_state=100, max_depth=4, min_samples_leaf=15)
gm_en.fit(x_train, y_train)
gm_en_predict = gm_en.predict(x_test)
gm_en_predict1 = gm_en.predict(x_train)
print('train accuracy ', accuracy_score(y_train, gm_en_predict1))
print('f1 score ',f1_score(y_test, gm_en_predict, average=None))
print('accuracy score ', accuracy_score(y_test, gm_en_predict))
print('roc auc score ',roc_auc_score(y_test,gm_en_predict))
print(classification_report(y_test, gm_en_predict))
```

```
train accuracy  0.8542274052478134
f1 score  [0.89655172 0.18181818]
accuracy score  0.8163265306122449
roc auc score  0.5405844155844156
              precision    recall  f1-score   support

           0       0.84      0.96      0.90       364
           1       0.41      0.12      0.18        77

    accuracy                           0.82       441
   macro avg       0.62      0.54      0.54       441
weighted avg       0.76      0.82      0.77       441
```

Fig: 22 Results using entropy as criterion for Decision Tree Classification

Last but not least next is the Random Forest Classification model. Here I have first fitted the actual train(x_train) and predict train(y_train) datasets into 'rmfocl'. Below is the figure with a python code snippet.

```
#Random_Forest Classification
data_final_variables = df2.columns.values.tolist()
Y=['Attrition']
X=[i for i in data_final_variables if i not in Y]
print(X)
print("")
print("List of columns with high correlation : ",len(X))
rmfocl = RandomForestClassifier()
rmfocl.fit(x_train, y_train)

Rate_of_Importances = pd.DataFrame({'Feature Name':X, 'Rate_of_Importance':np.round(rmfocl.feature_importances_,3)})
Rate_of_Importances = Rate_of_Importances.sort_values('Rate_of_Importance', ascending=False).set_index('Feature Name')

y_pr = rmfocl.predict(x_test)
```

[90] ✓ 0.2s Python

```
['Age', 'Department', 'EducationField', 'JobLevel', 'MonthlyIncome', 'PercentSalaryHike', 'PerformanceRating', 'TotalWorkingYears', 'YearsAtCompany',
'YearsInCurrentRole', 'YearsSinceLastPromotion', 'YearsWithCurrManager']

List of columns with high correlation :  12
```

Fig: 23 Random Forest Classification Python Program

Measuring the rate of importance of each feature in the dataset.

```
print(Rate_of_Importances)
```

[84] ✓ 0.5s Python

```
                         Rate_of_Importance
Feature Name
MonthlyIncome                         0.203
Age                                   0.154
TotalWorkingYears                     0.116
PercentSalaryHike                     0.105
YearsAtCompany                        0.078
YearsWithCurrManager                  0.072
EducationField                        0.068
YearsInCurrentRole                    0.060
YearsSinceLastPromotion               0.060
Department                            0.037
JobLevel                              0.033
PerformanceRating                     0.013
```

Fig: 24 Table of dataset headers and rate of importance

I have also plotted the rate of importance on a bar plot.

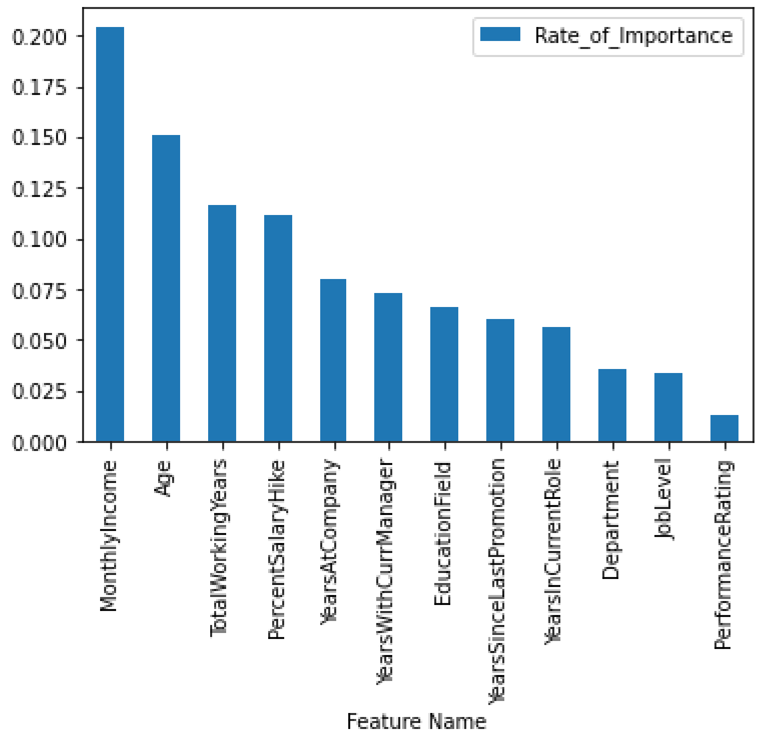

Fig: 25 Bar graph of Features/Headers vs Rate of importance

I have then measured the prediction accuracy for the train dataset, and it is 85%.

```
print('Accuracy of Random Forest classification on test dataset: {:.2f}'.format(rmfocl.score(x_test, y_test)))
✓ 0.6s                                                                    Python
Accuracy of Random Forest classification on test dataset: 0.85
```

Fig 26: Calculation of the accuracy using Random Forest Classification Model

Next, I wanted to look at the confusion matrix, i.e., a 2x2 matrix with count of true positives, true negatives, false positives, and false negatives predictions generated by the random forest classifier on the attrition feature. The matrix determines that correct predictions are 306 + 14 that is 320 and incorrect predictions are 44 + 4 that is 48. Below is the figure with the result confusion matrix.

Fig: 27 Generating the confusion matrix of the random forest classification model

Below is the plot of the generated confusion matrix using the random forest classification model.

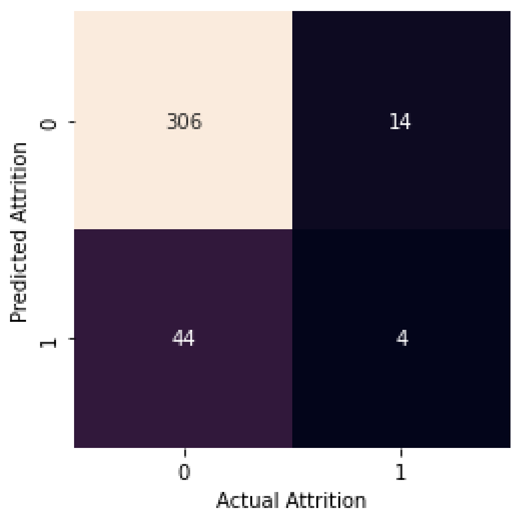

Fig: 28 Confusion Matrix from Random Classification Model

The kNN classifier's purpose is to assign unlabeled observations to the class of the most similar labeled examples. Observational characteristics are collected for both the training and test datasets. There are two important concepts to consider. The first is euclidean distance, which is used by the knn() function and can be calculated using the formula shown in Figure 29.

$$D(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2}$$

Fig: 29 Euclidean Distance equation

I discovered that the optimal k nearest neighbors are 7, which will help with misclassification error. The prediction scores for k nearest neighbors machine learning model on the given data set are in the below figure 30.

```python
from sklearn.model_selection import cross_val_predict, cross_val_score
from sklearn.metrics import accuracy_score, classification_report
from sklearn.metrics import confusion_matrix

def print_score(clf, X_train, y_train, X_test, y_test, train = True):
    if train:
        print("Classification Report: \n {}\n".format(classification_report(
                y_train, clf.predict(X_train))))
        print("Confusion Matrix: \n {}\n".format(confusion_matrix(
                y_train, clf.predict(X_train))))

        res = cross_val_score(clf, X_train, y_train,
                              cv = 10, scoring ='accuracy')
        print("Average Accuracy: \t {0:.4f}".format(np.mean(res)))
        print("Accuracy SD: \t\t {0:.4f}".format(np.std(res)))
        print("accuracy score: {0:.4f}\n".format(accuracy_score(
                y_train, clf.predict(X_train))))

knn = KNeighborsClassifier(n_neighbors = 7)
knn.fit(x_train, y_train)
print_score(knn, x_train, y_train, x_test, y_test, train = True)
print_score(knn, x_train, y_train, x_test, y_test, train = False)
```

Python

Fig: 30 Python program to measure kNN machine learning algorithm metrics

```
Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.99      0.92       922
           1       0.83      0.19      0.32       180

    accuracy                           0.86      1102
   macro avg       0.85      0.59      0.62      1102
weighted avg       0.86      0.86      0.82      1102
```

Fig: 31 Classification Report generated by kNN machine learning model

Conclusion

To summarize the results of the machine learning models produced on the IBM HR dataset, the Decision Tree - Entropy classification model predicted employee attrition with 85 percent accuracy on the test dataset, the Random Forest classification model predicted employee attrition with approximately 84 percent accuracy, and the kNN classification model generated 86 percent accuracy. Furthermore, we believe that businesses will be able to use machine learning models in the future to predict employee attrition. This has an immediate practical application in that an HR department can easily plan the efficient workforce for each department. Estimate the number of employees required for each department so that the hiring process can be completed flawlessly and efficiently. These case studies expanded on the research question by demonstrating that ensemble methods with effective feature selection are effective in predicting employee attrition, as evidenced by visualizations and accuracies by different models, and thus managers should focus on the top needs of employees by motivating entry-level employees, increasing job satisfaction, and relationship satisfaction.

This project, however, has some limitations. This study is limited to a small dataset that is insufficient to train the model well, which may result in poor results, and obtaining employee data from an organization is confidential, so this study is limited to the IBM dataset, which is the only available dataset online. The second disadvantage is that the model is limited to only supervised learning, which requires a lot of computation time, decision boundaries may be overtrained at times, and user input is required every time new features are added. This project has the potential to be expanded in the future because it has a lot of room for improvement by using deep learning techniques with a well-defined network of sufficient hidden layers on a large dataset, which can mask the project's limitations. If the data is in dae format, time series and trend analysis may be used to improve prediction performance.

References

AI-MEGHA. (2019, December 8). *IBM-HR-Analytics-Employee-Attrition-Performance/IBM HR ANALYTICS EMPLOYEE ATTRITION AND PERFORMANCE REPORT.pdf at master · AI-MEGHA/IBM-HR-Analytics-Employee-Attrition-Performance*. GitHub; github.com. https://github.com/AI-MEGHA/IBM-HR-Analytics-Employee-Attrition-Performance/blob/master/IBM%20HR%20ANALYTICS%20EMPLOYEE%20ATTRITION%20AND%20PERFORMANCE%20REPORT.pdf

Anunaya, S. (2021, August 10). *Data Preprocessing in Data Mining -A Hands On Guide - Analytics Vidhya*. Analytics Vidhya; www.analyticsvidhya.com. https://www.analyticsvidhya.com/blog/2021/08/data-preprocessing-in-data-mining-a-hands-on-guide/

BACHMANN, J. M. (2019, 0 0). *Attrition in an Organization || Why Workers Quit?* Attrition in an Organization || Why Workers Quit? | Kaggle; www.kaggle.com. https://www.kaggle.com/code/janiobachmann/attrition-in-an-organization-why-workers-quit

B. Thangaparvathi, D. Anandhavalli and S. M. Shalinie, "*A high speed decision tree classifier algorithm for huge dataset*," 2011 International Conference on Recent Trends in Information Technology (ICRTIT), 2011, pp. 695-700, doi: 10.1109/ICRTIT.2011.5972267.

Chern, C.-C., Lei, W.-U., Huang, K.-L., & Chen, S.-Y. (2021, February 2). *A decision tree classifier for credit assessment problems in big data environments - Information Systems and e-Business Management*. SpringerLink; link.springer.com. https://link.springer.com/article/10.1007/s10257-021-00511-w

Kaustubh1828 . (2021, June 25). *Generate Reports Using Pandas Profiling, Deploy Using Streamlit*. Analytics Vidhya; www.analyticsvidhya.com. https://www.analyticsvidhya.com/blog/2021/06/generate-reports-using-pandas-profiling-deploy-using-streamlit/

Keerthana. (2021, June 6). DATA PREPROCESSING TECHNIQUES. Data preprocessing is a Data Mining… | by Keerthana | AlmaBetter | Medium. Medium; medium.com. https://medium.com/almabetter/data-preprocessing-techniques-6ea145684812

Koehrsen, W. (2018, January 17). *Random Forest in Python. A Practical End-to-End Machine Learning… | by Will Koehrsen | Towards Data Science*. Medium; towardsdatascience.com. https://towardsdatascience.com/random-forest-in-python-24d0893d51c0

Kumar, S., & Chong, I. (2018, December 19). *Correlation Analysis to identify the effective data in Machine Learning: Prediction of Depressive disorder and emotion states* MDPI. Retrieved April 15, 2022, from https://www.mdpi.com/1660-4601/15/12/2907

Lafuente, A. S. (2020, February 9). *Exploratory Data Analysis with Pandas Profiling | by Albert Sanchez Lafuente | Towards Data Science*. Medium; towardsdatascience.com. https://towardsdatascience.com/exploratory-data-analysis-with-pandas-profiling-de3aae2ddff3

Milani, A. M. P., Loges, L. A., Paulovich, F. V., & Manssour, I. H. (2021, July 2). *PrAVA: Preprocessing profiling approach for visual analytics*. https://journals.sagepub.com/doi/10.1177/14738716211021591

P, A. (2019, September 14). *IBM-HR-Analytics-Employee-Attrition-Performance/IBM Attrition.ipynb at master · amrilp11/IBM-HR-Analytics-Employee-Attrition-Performance*. GitHub; github.com. https://github.com/amrilp11/IBM-HR-Analytics-Employee-Attrition-Performance/blob/master/IBM-HR-ATTRITION/IBM%20Attrition.ipynb

SENGUPTA, P. (2022, April 12). *HR Analytics Prediction: Why do People Resign?* HR Analytics Prediction: Why Do People Resign? | Kaggle; www.kaggle.com. https://www.kaggle.com/code/paramarthasengupta/hr-analytics-prediction-why-do-people-resign

SUBHASH, P. (2017, 0 0). *IBM HR Analytics Employee Attrition & Performance*. IBM HR Analytics Employee Attrition & Performance | Kaggle; www.kaggle.com. https://www.kaggle.com/datasets/pavansubhasht/ibm-hr-analytics-attrition-dataset

Ydataai. (2022, April 6). *GitHub - ydataai/pandas-profiling: Create HTML profiling reports from pandas DataFrame objects*. GitHub; github.com. https://github.com/ydataai/pandas-profiling

Yiziwinnie. (2019, March 22). *GitHub - Yiziwinnie/IBM-HR-Analytics-Employee-Attrition-Performance: Advanced Machine Learning*. GitHub; github.com. https://github.com/Yiziwinnie/IBM-HR-Analytics-Employee-Attrition-Performance

Zhang, Z. (2016, April 20). *Introduction to machine learning: k-nearest neighbors - Zhang - Annals of Translational Medicine*. Introduction to Machine Learning: K-Nearest Neighbors - Zhang - Annals of Translational Medicine; atm.amegroups.com. https://atm.amegroups.com/article/view/10170/11310