HR Analytics Project

Understanding the Attrition in HR

Submitted By

Rutuja Patil

HR Analytics Project- Understanding the Attrition in HR

Abstract. Employee attrition can become a serious issue because of the impacts on the organization's competitive advantage. It can become costly for an organization. The cost of employee attrition would be the cost related to the human resources life cycle, lost knowledge, employee morale, and organizational culture. This study aimed to analyse employee attrition using various regression model. The result obtained can be used by the management to understand what modifications they should perform to the workplace to get most of their workers to stay. Do you want to know what is that one single thing for grand success of any business Enterprise, startup in current competitive era, where razor edge like fine technological upgradation take place continuously along with continuous evolving SOP's, business model? Fundamental answer to this question trace back to core of business which is need & demand of product.

Introduction.

The key to success in an organisation is the ability to attract and retain top talents. But where Machine learning comes here? For that we need to go in background of HR analytics. HR department hire lot of employees every year. 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. HR department often play significant role in designing company compensation programs, creating ambiance in work environment with various team activities, imprinting work culture habits on mindset of peoples and training skill systems that help the organization retain smart minds & talented brain. Most of organisation run different HR analytics vertical to gather data, analyse data to improve process within organisation and to make major decision about human resources. The gradual loss of employees' overtime refers as HR attrition. Attrition can happen due to retirement, involuntary (employee is fired or terminated) or voluntary (resigns from an organization). A major problem in high employee attrition is its cost to an organization. Job postings, hiring processes, paperwork and new hire training are some of the common expenses of losing employees and replacing them. For Tech companies' loss of talented brain means loss of domain expertise, knowledge base as for these company's Technological capability exists in domain expertise, intellectual property rights of employee. On marketing or sales side customers often prefer to interact with familiar people.

Just like several factors contribute towards building a reliable team and organization, similarly, numerous factors contribute to the attrition rate: Improper work-life balance, better job opportunities, salary hike, Lack of growth or work recognition, unhealthy relations with managers. We can gather data about these factors and utilise Machine learning classification techniques to predict whether employee like to leave organisation or stay in organisation. Here I will show you what leads employee's attrition and Predication of attrition using case study on IBM HR analytics Dataset.

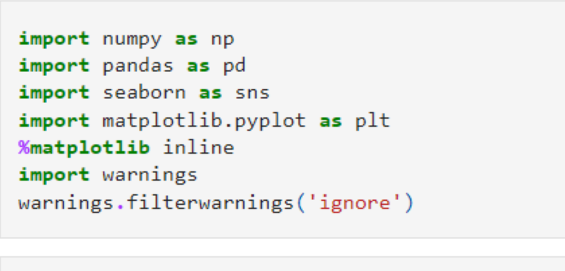
IBM HR Analytics Employee Attrition & Performance Dataset

In this case study we will use IBM HR Analytics database. This fictional dataset created by IBM employees and available to download from GitHub. This dataset consists of 1470 rows, 35 features describing each employee's background and characteristics and target variable. Attrition is target variable to be predicted. As target variable is categorial in nature, this case study falls into classification machine learning problem. We have two objectives here:

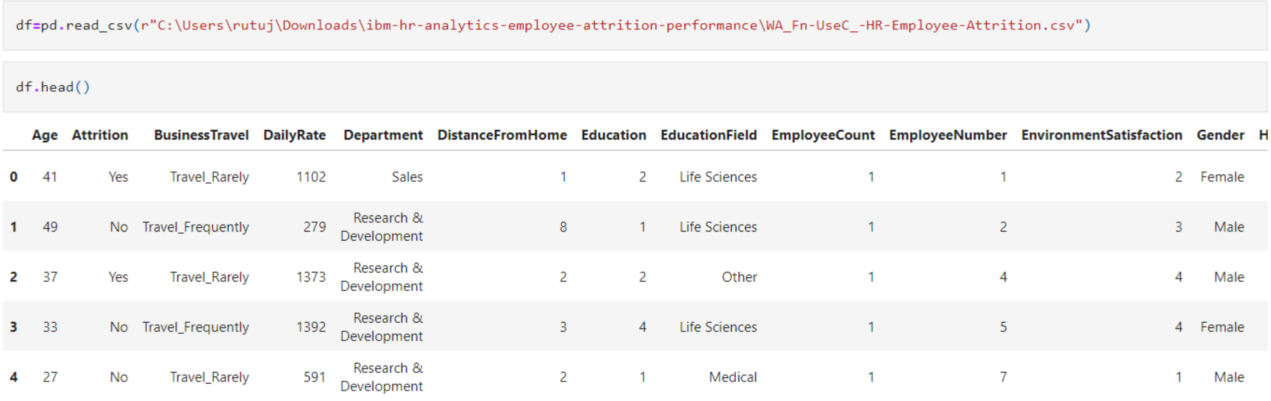
1. Which key factors result in employee attrition?
2. Building ML Model for predicting attrition.

Data Preparation: Load, Clean and Format

Let's begin with importing libraries for EDA and dataset itself.



```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

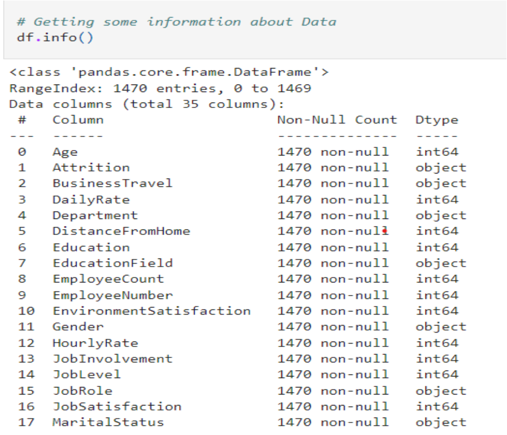


```
df=pd.read_csv(r"C:\Users\rutuj\Downloads\ibm-hr-analytics-employee-attrition-performance\WA_Fn-UseC_-HR-Employee-Attrition.csv")
```

```
df.head()
```

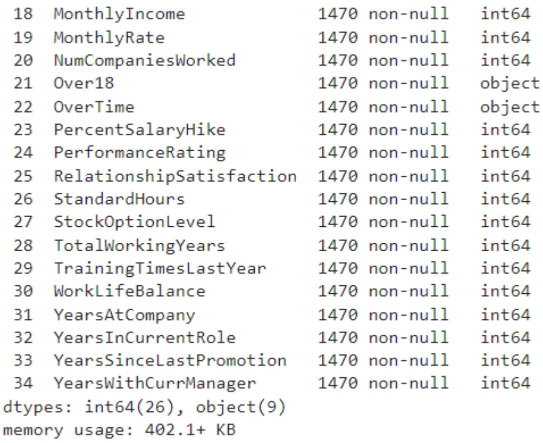
	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	Gender	H
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	2	Female	
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	3	Male	
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	4	Male	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	4	Female	
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	1	Male	

Getting some information about dataset: -



```
# Getting some information about Data
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
```

We have 9 features with object datatypes and rest are Numeric feature with int64. Out of all numeric features Education, Environment-Satisfaction, Job-Involvement, Job-Satisfaction, Relationship-Satisfaction, Performance Rating, Work Life Balance are ordinal variable. These ordinal features have unique label for each numeric value.

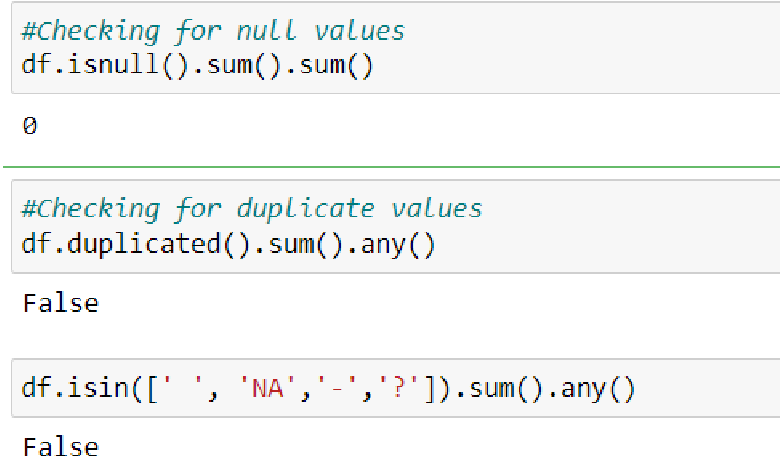
These Ordinal features come with the following label encoding:

- **Education:** 1- 'Below College’, 2 -'College', 3 -'Bachelor', 4- 'Master' ,5 -'Doctor'
- **Environment Satisfaction:** 1- 'Low', 2- 'Medium', 3 -'High', 4- 'Very High'
- **Job Involvement:** 1 -'Low', 2- 'Medium', 3- 'High', 4- 'Very High'
- **Job Satisfaction:** 1- 'Low', 2- 'Medium', 3- 'High', 4 -'Very High'
- **Performance Rating:** 1- 'Low', 2- 'Average', 3 -'Good', 4- 'Excellent', 5- 'Outstanding'
- **Relationship Satisfaction:** 1- 'Low', 2- 'Medium', 3- 'High', 4- 'Very High'
- **Work Life Balance:** 1- 'Bad', 2- 'Good', 3- 'Better', 4- 'Best'

Above nomenclature will help in better understanding of data when we perform EDA in this case study.

Data Integrity Check:

Dataset can have missing values, duplicated entries and whitespaces. Now we will perform this integrity check of dataset.



```
#Checking for null values
df.isnull().sum().sum()
```

```
0
```

```
#Checking for duplicate values
df.duplicated().sum().any()
```

```
False
```

```
df.isin([' ', 'NA','-','?']).sum().any()
```

```
False
```

Luckily for us, there is no missing data! this will make it easier to work with the dataset.

Dataset doesn’t contain Any duplicate entry, whitespace, ‘NA’, or ‘-’.

Statistical parameters like mean, median, quantile can give important details about database. Now is time to look at statistical Matrix of Dataset.

Few key observations from this statistical matrix are listed below: -

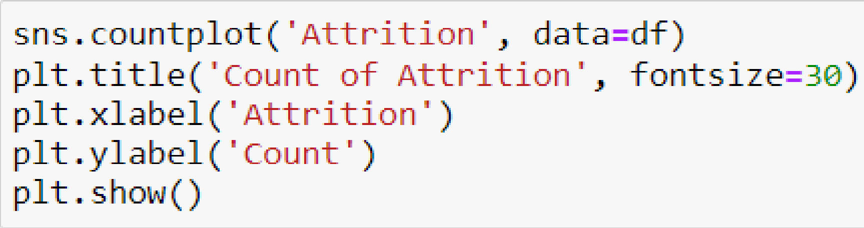
- Minimum Employee Age is 18 and Maximum age of employee 60.
- Average distance from home is 9.1 KM. It means that most of employee travel at least 18 KM in day from home to office.
- Average performance Rating of employees is 3.163 with min value 3.0. This Means that performance of most of employee is 'Good’. This implies that Attrition of Employee with 'Outstanding' or 5 rating need to investigate.

- 50% of Employees has worked at least 2 companies previously.
- For Monthly Income, Monthly Rate by looking at 50% and max column we can say outliers exist in this feature.
- By looking at Mean and Median we see that some of the features are skew in nature.
- For ordinal features statistical terminology like mean, median, std deviation are not applicable.
- Standard Hours and Employee Count contain same value for all statistical parameter. It means they contain one unique value.

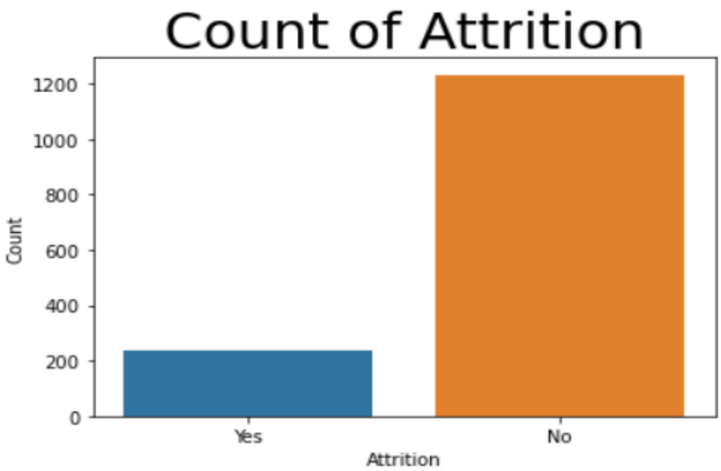
Exploratory data analysis

Exploratory data analysis refers to the critical process of performing initial investigations on data so as to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.

Let’s begin data exploration of Target variable using count plot.

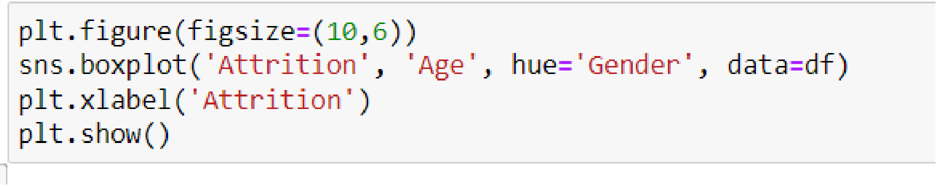


```
sns.countplot('Attrition', data=df)
plt.title('Count of Attrition', fontsize=30)
plt.xlabel('Attrition')
plt.ylabel('Count')
plt.show()
```

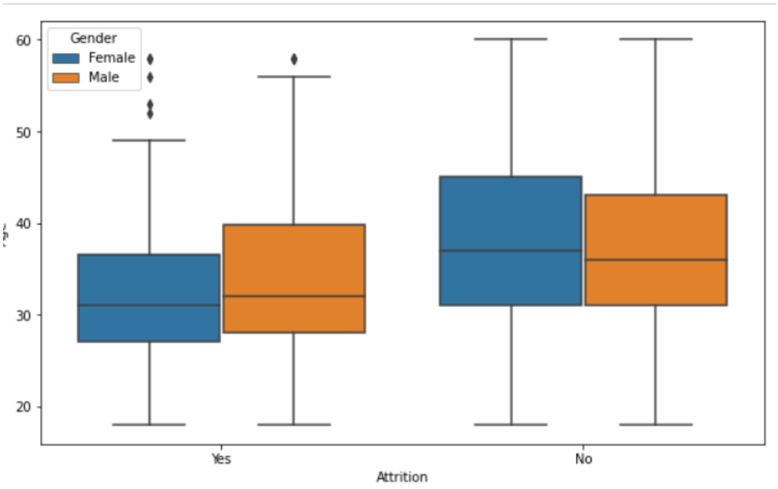


83.88% (1237 employees) Employees did not leave the organization while 16.12% (237 employees) did leave the organization **making our dataset to be consider as imbalanced** since more people stay in the organization than they actually leave. In this dataset we have features like education, department, education field, job role, job satisfaction which are inter related with each other. Job role & job position not in alignment with educational background can lead attrition. Let investigate this by visualisation of these features one by one to gain more insights.

Attrition w.r.t Age and Gender:

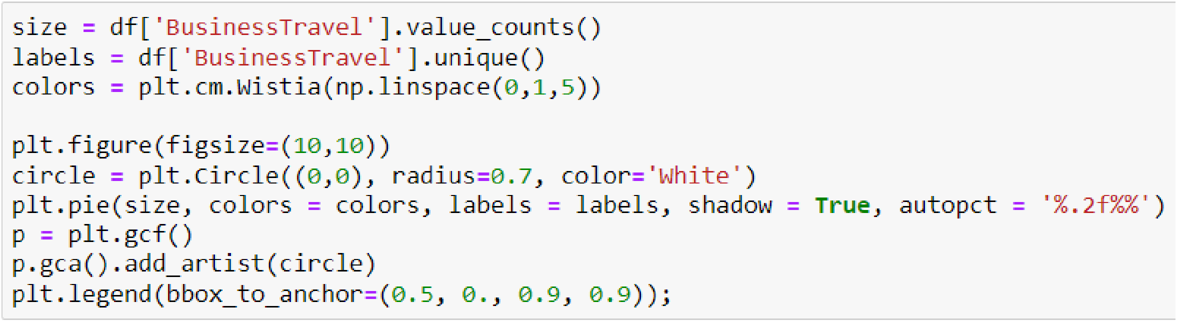


```
plt.figure(figsize=(10,6))
sns.boxplot('Attrition', 'Age', hue='Gender', data=df)
plt.xlabel('Attrition')
plt.show()
```

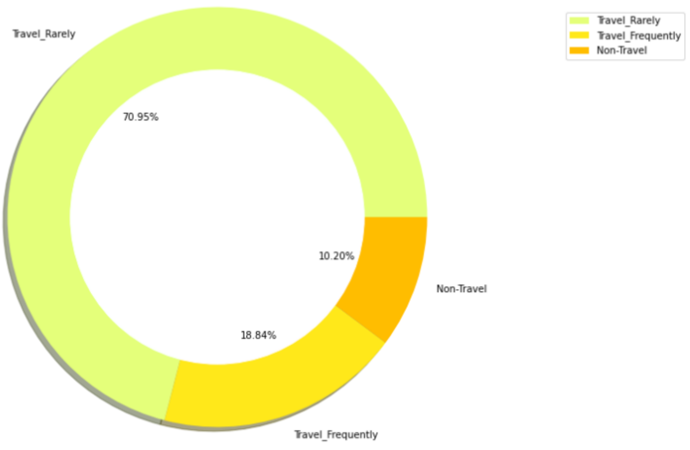


we can see that the amount of attrition is more for the Male employees ranging from age 29 to 41 as shown in the above graph. which is more than number of female employees age range 28 to 36.

Percentage of Business Travel of Employees.

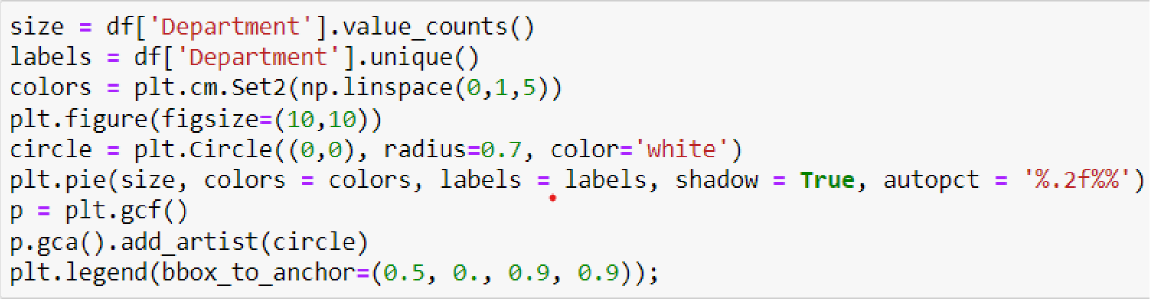


```
size = df['BusinessTravel'].value_counts()
labels = df['BusinessTravel'].unique()
colors = plt.cm.Wistia(np.linspace(0,1,5))

plt.figure(figsize=(10,10))
circle = plt.Circle((0,0), radius=0.7, color='white')
plt.pie(size, colors = colors, labels = labels, shadow = True, autopct = '%.2f%%')
p = plt.gcf()
p.gca().add_artist(circle)
plt.legend(bbox_to_anchor=(0.5, 0., 0.9, 0.9));
```

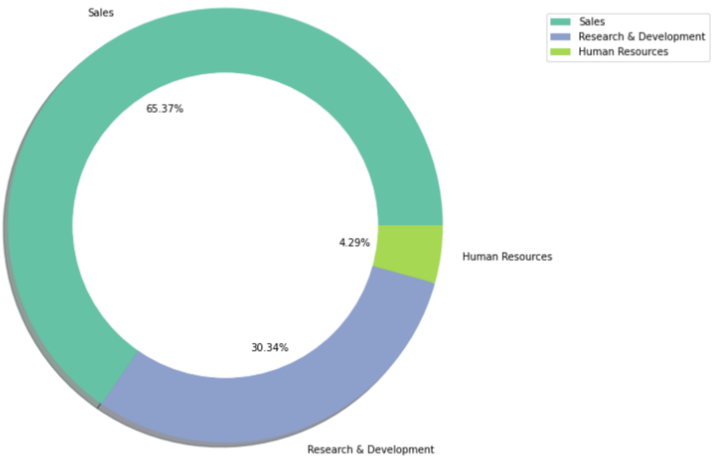


Large number of employees rarely travels which can be shown in above figure which is 70.95%. 18.84% of employees frequently traves and less which is 10.20 % employees never travels.

Percentage of Employees in various Departments.



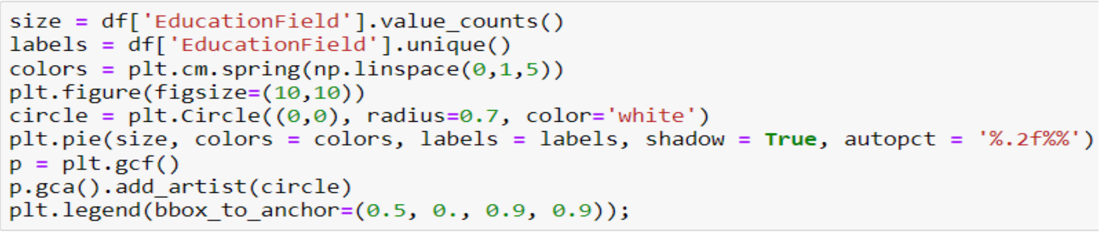
```
size = df['Department'].value_counts()
labels = df['Department'].unique()
colors = plt.cm.Set2(np.linspace(0,1,5))
plt.figure(figsize=(10,10))
circle = plt.Circle((0,0), radius=0.7, color='white')
plt.pie(size, colors = colors, labels = labels, shadow = True, autopct = '%.2f%%')
p = plt.gcf()
p.gca().add_artist(circle)
plt.legend(bbox_to_anchor=(0.5, 0., 0.9, 0.9));
```



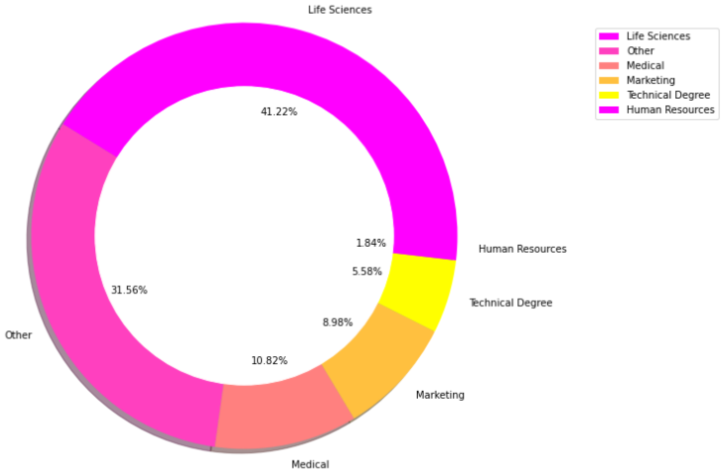
- There are 65.37% employees working in Sales Department which is very large. 30.34% employees working in R&D Department And very less that is 4.29% works in HR department.

Percentage of Education Fields.

The probability of Employees Retention is more when there working domain is in alignment with education background. Let check this with crosstab of department against education field. We will Analyse Attrition in department according to education background based on above insight further but before that explore Job role in order to include it in further attrition analyse. First build matrix of department vs job role which will give us idea about number of employees of different job role across department

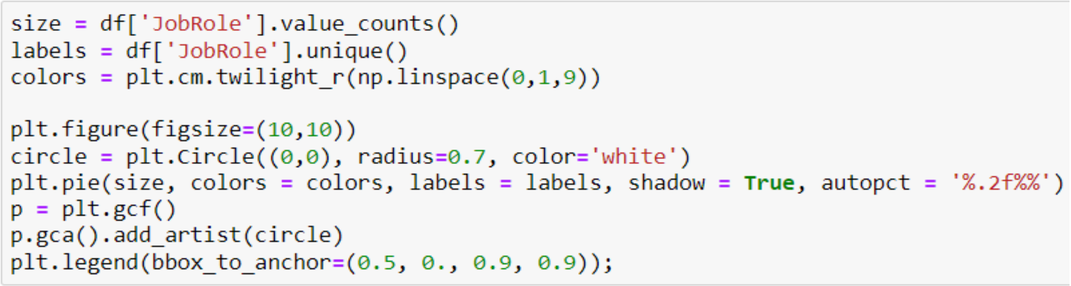


```
size = df['EducationField'].value_counts()
labels = df['EducationField'].unique()
colors = plt.cm.spring(np.linspace(0,1,5))
plt.figure(figsize=(10,10))
circle = plt.Circle((0,0), radius=0.7, color='white')
plt.pie(size, colors = colors, labels = labels, shadow = True, autopct = '%.2f%%')
p = plt.gcf()
p.gca().add_artist(circle)
plt.legend(bbox_to_anchor=(0.5, 0., 0.9, 0.9));
```

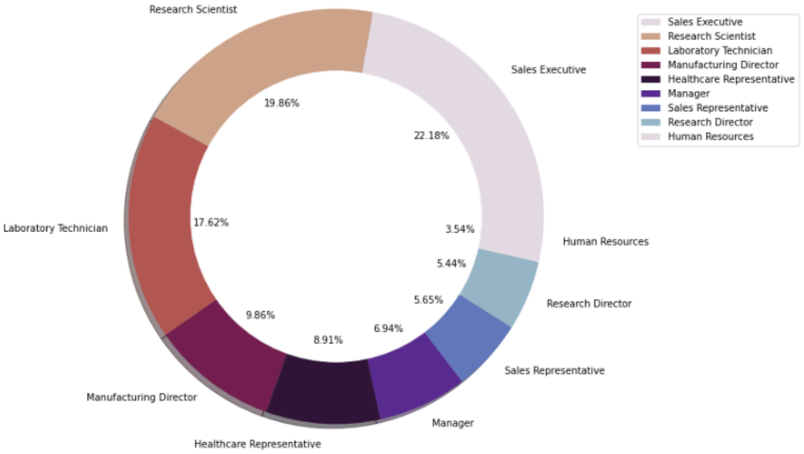


- 41.22% Employees are from Life Science field
- 31.56% Employees are from Other field.
- 10.82% Employees are from Medical field
- 8.98% Employees are from Marketing field.
- 5.58% Employees are from Technical field
- 1.84% Employees are from HR field.

Percentage of Employees in various Job Roles:

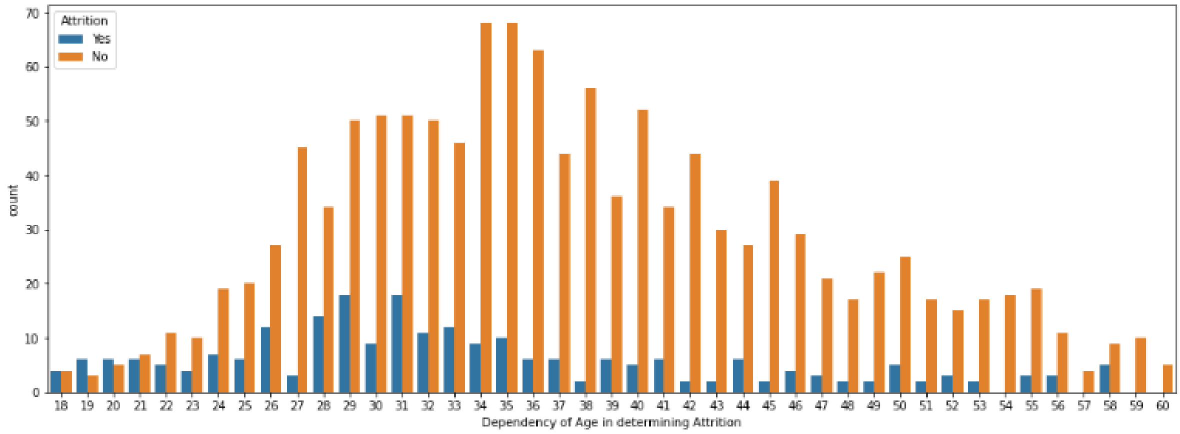


```
size = df['JobRole'].value_counts()
labels = df['JobRole'].unique()
colors = plt.cm.twilight_r(np.linspace(0,1,9))

plt.figure(figsize=(10,10))
circle = plt.Circle((0,0), radius=0.7, color='white')
plt.pie(size, colors = colors, labels = labels, shadow = True, autopct = '%.2f%%')
p = plt.gcf()
p.gca().add_artist(circle)
plt.legend(bbox_to_anchor=(0.5, 0., 0.9, 0.9));
```



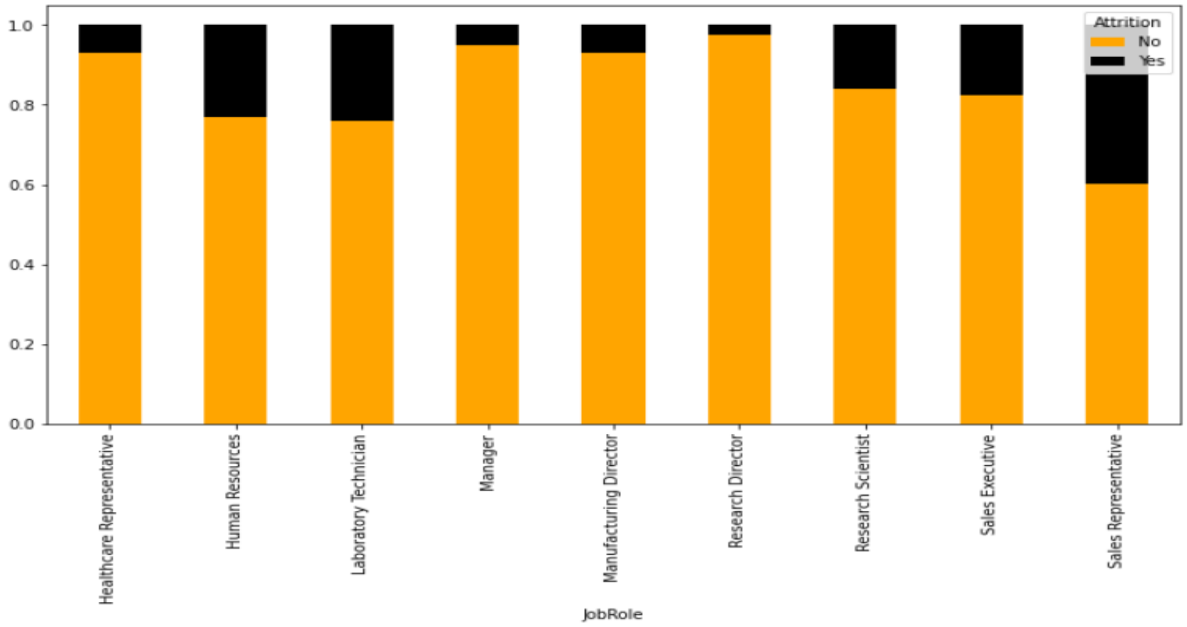
- 22.18% Employees are in Sales Executive post
- 19.86% Employees are in Research Scientist post
- 17.62% Employees are in Laboratory Technician post
- 9.86% Employees are in Manufacturing Director post
- 8.91% Employees are in HealthCare Representative post
- 6.94% Employees are in Manager post
- 5.65% Employees are in Sales Representative post
- 5.44% Employees are in Research Director post
- 3.54% Employees are in HR post.

Dependency of Age in determining Attrition.

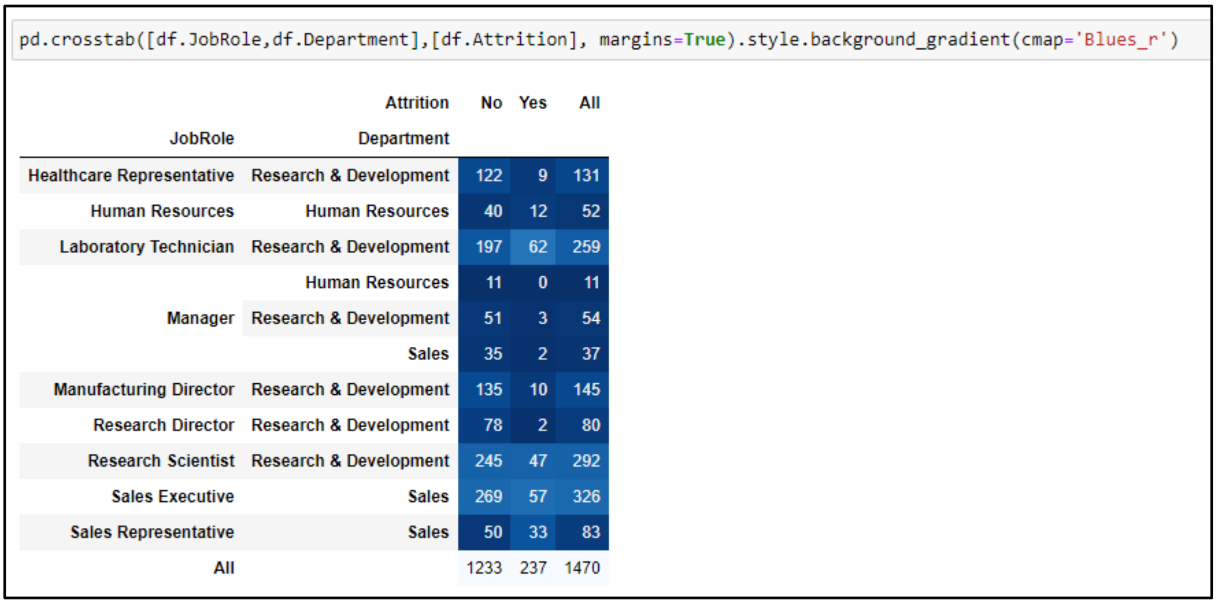


- The Attrition rate is minimum between the Age years of 34 and 45.
- The Attrition rate is maximum between the Age years of 29 and 33.

Dependency of Job Role in determining Attrition.



Let’s check absolute number matrix of attrition according job role, again this time using crosstab.



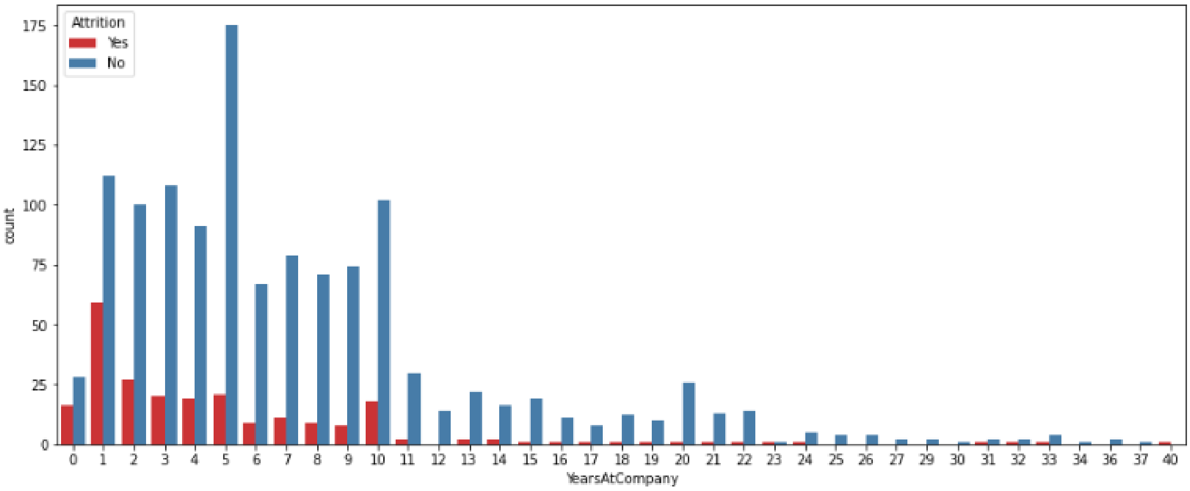
```
pd.crosstab([df.JobRole,df.Department],[df.Attrition], margins=True).style.background_gradient(cmap='Blues_r')
```

	Attrition	No	Yes	All
JobRole	Department			
Healthcare Representative	Research & Development	122	9	131
Human Resources	Human Resources	40	12	52
Laboratory Technician	Research & Development	197	62	259
Manager	Human Resources	11	0	11
	Research & Development	51	3	54
	Sales	35	2	37
Manufacturing Director	Research & Development	135	10	145
Research Director	Research & Development	78	2	80
Research Scientist	Research & Development	245	47	292
Sales Executive	Sales	269	57	326
Sales Representative	Sales	50	33	83
All		1233	237	1470

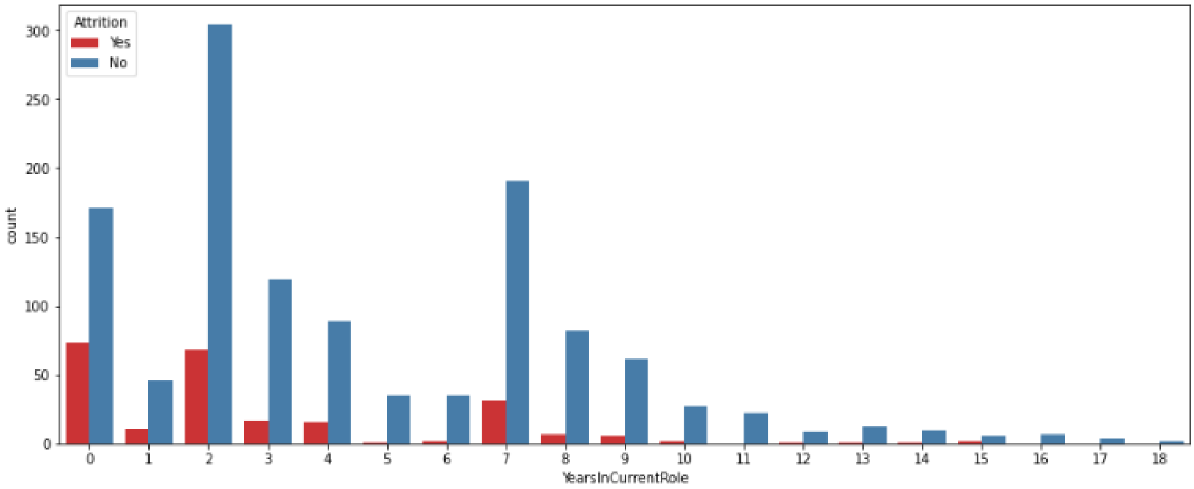
Key Insights from above Cross Tab:

- Percentage of attrition is high in Sales Representative, Laboratory Technician, Human Resources.
- At the Top chart 62 Laboratory Technician has resign from job, followed by 57 sales executive and 47 Research Scientist.
- 16 % attrition rate for Research Scientist, which involve huge investment from company. Company not only loses employee but its knowledge base, expertise & Intellectual property rights in some cases.

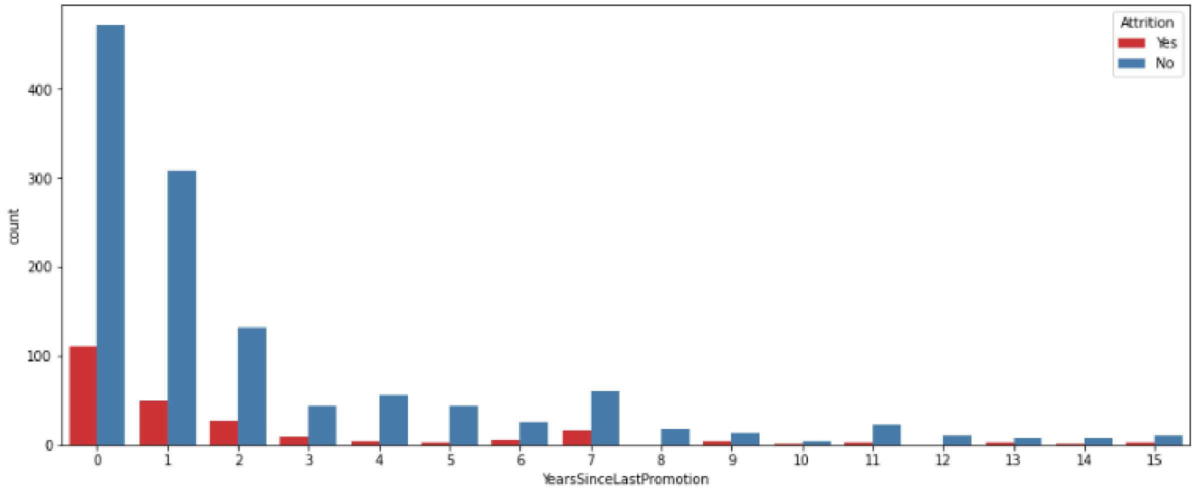
Dependency of Years At Company in determining Attrition.



Dependency of Years In Current Role in determining Attrition.



Dependency of Years Since Last Promotion in determining Attrition.



Key Insights from above Graphs:

- For Majority of people have spent 3 to 10 years at company.
- Most of people staying company up to 2 years after promotion.
- Majority of people are train 2-3 times in last year
- Majority of people stay in same role for maximum 4 yrs.

Feature Engineering: Data Pre-processing.

Feature engineering is the process of transforming raw data into features that better represent the underlying problem to the predictive models, resulting in improved model accuracy on unseen data. Feature Engineering is very important step in building Machine Learning model. Some machine learning projects succeed and some fail. What makes the difference? Easily the most important factor is the features used. In Feature engineering can be done for various reason.

Some of them are mention below:

1. **Feature Importance**: An estimate of the usefulness of a feature
2. **Feature Extraction**: The automatic construction of new features from raw data (Dimensionality reduction Technique like PCA)
3. **Feature Selection**: From many features to a few that are useful
4. **Feature Construction**: The manual construction of new features from raw data (For example, construction of new column for month out date - mm/dd/yy)

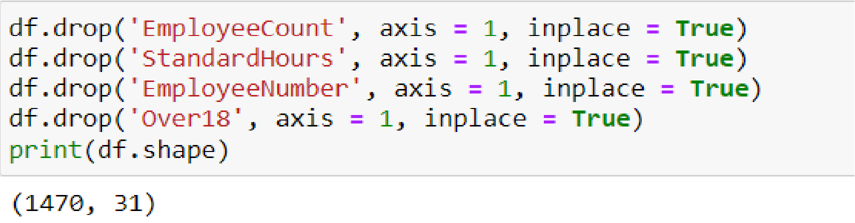
There are Varity of techniques use to achieve above mention means as per need of dataset. Some of Techniques important are as below:

- Handling missing values
- Handling imbalanced data using SMOTE
- Outliers' detection and removal using Z-score, IQR
- Scaling of data using Standard Scalar or Minmax Scalar
- Binning whenever needed
- Encoding categorical data using one hot encoding, label / ordinal encoding
- Skewness correction using Boxcox or yeo-Johnson method
- Handling Multicollinearity among feature using variance inflation factor
- Feature selection Techniques:
    - ✓ Correlation Matrix with Heatmap
    - ✓ Univariate Selection – Select K Best
    - ✓ Extra Trees Classifier method

In this case study we will use some of the mention feature engineering Techniques one by one.

1. Dropping unnecessary features

In the dataset there are 4 irrelevant columns, i.e: Employee Count, Employee Number, Over18 and Standard Hour. So, we have to remove these for more accuracy. Feature like 'Over18', 'Standard Hours' contain single unique value. Features like Employee Count, Employee Number are irrelevant from ML model building perspective. We will drop these features.

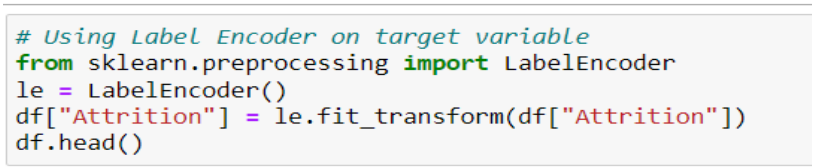


```
df.drop('EmployeeCount', axis = 1, inplace = True)
df.drop('StandardHours', axis = 1, inplace = True)
df.drop('EmployeeNumber', axis = 1, inplace = True)
df.drop('Over18', axis = 1, inplace = True)
print(df.shape)
```

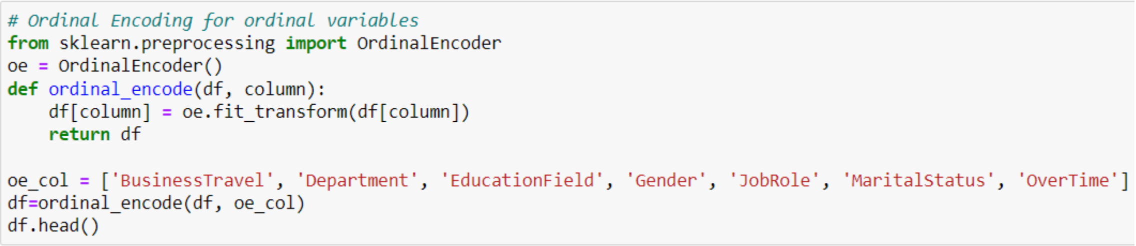
```
(1470, 31)
```

2. Encoding Categorical & Ordinal Features

Label Encoding is employed over target variable 'Attrition' while Ordinal encoding employee for rest categorical features.



```
# Using Label Encoder on target variable
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df["Attrition"] = le.fit_transform(df["Attrition"])
df.head()
```



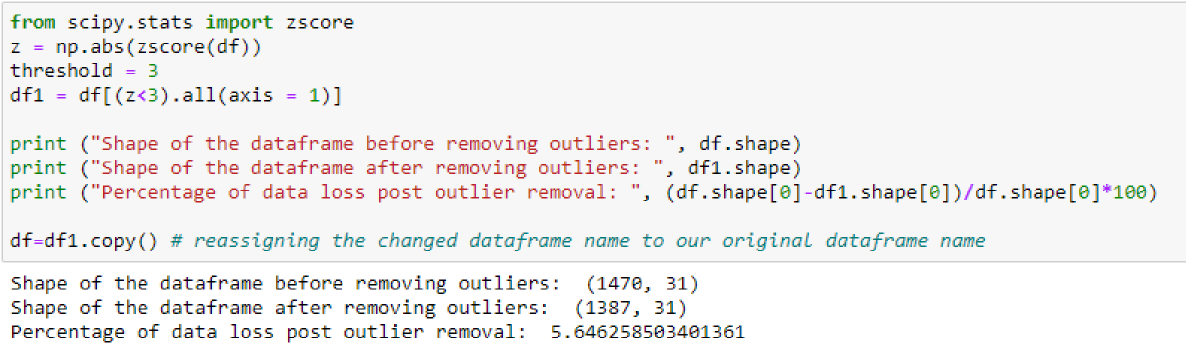
```
# Ordinal Encoding for ordinal variables
from sklearn.preprocessing import OrdinalEncoder
oe = OrdinalEncoder()
def ordinal_encode(df, column):
    df[column] = oe.fit_transform(df[column])
    return df

oe_col = ['BusinessTravel', 'Department', 'EducationField', 'Gender', 'JobRole', 'MaritalStatus', 'OverTime']
df=ordinal_encode(df, oe_col)
df.head()
```

- Since now encoding is done we will move towards outliers’ detection and removal.

3. Outliers’ detection and removal.

Identifying outliers and bad data in your dataset is probably one of the most difficult parts of data clean-up, and it takes time to get right. Even if you have a deep understanding of statistics and how outliers might affect your data, it’s always a topic to explore cautiously

Machine learning algorithms are sensitive to the range and distribution of attribute values. Data outliers can spoil and mislead the training process resulting in longer training times, less accurate models and ultimately poorer results. Outliers can be seen in boxplot of numerical feature. We did not added boxplot here as it will make this article length, I left it to reader to further investigate. Now we will use Z-score method for outliers’ detection.

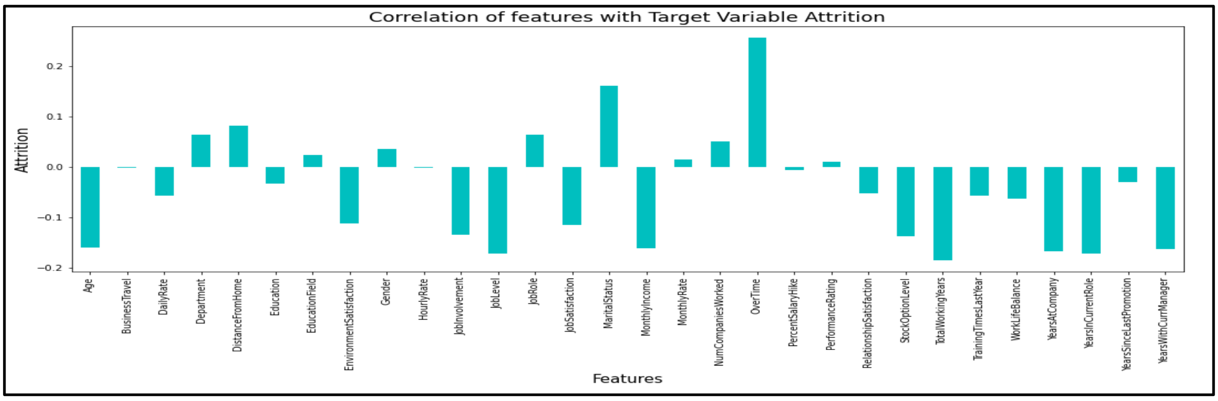


```
from scipy.stats import zscore
z = np.abs(zscore(df))
threshold = 3
df1 = df[(z<3).all(axis = 1)]

print ("Shape of the dataframe before removing outliers: ", df.shape)
print ("Shape of the dataframe after removing outliers: ", df1.shape)
print ("Percentage of data loss post outlier removal: ", (df.shape[0]-df1.shape[0])/df.shape[0]*100)

df=df1.copy() # reassigning the changed dataframe name to our original dataframe name
```

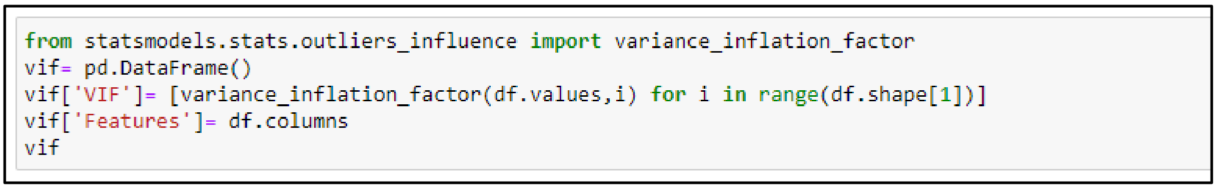
```
Shape of the dataframe before removing outliers:  (1470, 31)
Shape of the dataframe after removing outliers:  (1387, 31)
Percentage of data loss post outlier removal:  5.646258503401361
```

4. Correlation Heatmap

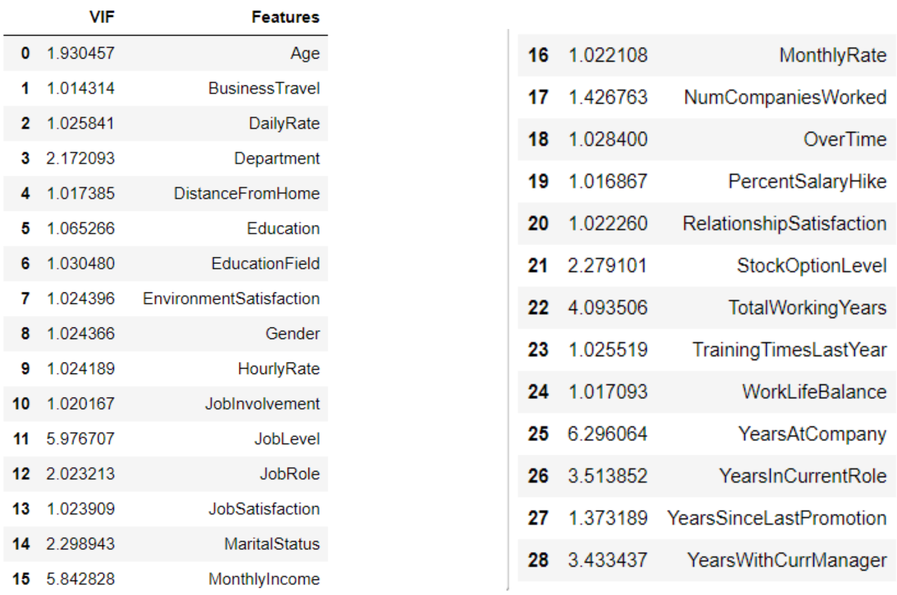
Correlation Heatmap show in a glance which variables are correlated, to what degree, in which direction, and alerts us to potential multicollinearity problems. The bar plot of correlation coefficient of target variable with independent features shown below



5. Multicollinearity between features.



```
from statsmodels.stats.outliers_influence import variance_inflation_factor
vif= pd.DataFrame()
vif['VIF']= [variance_inflation_factor(df.values,i) for i in range(df.shape[1])]
vif['Features']= df.columns
vif
```

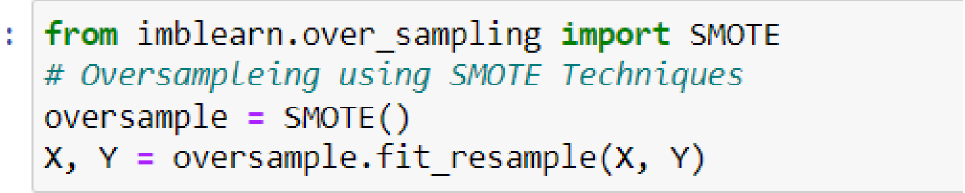


	VIF	Features
0	1.930457	Age
1	1.014314	BusinessTravel
2	1.025841	DailyRate
3	2.172093	Department
4	1.017385	DistanceFromHome
5	1.065266	Education
6	1.030480	EducationField
7	1.024396	EnvironmentSatisfaction
8	1.024366	Gender
9	1.024189	HourlyRate
10	1.020167	JobInvolvement
11	5.976707	JobLevel
12	2.023213	JobRole
13	1.023909	JobSatisfaction
14	2.298943	MaritalStatus
15	5.842828	MonthlyIncome
16	1.022108	MonthlyRate
17	1.426763	NumCompaniesWorked
18	1.028400	OverTime
19	1.016867	PercentSalaryHike
20	1.022260	RelationshipSatisfaction
21	2.279101	StockOptionLevel
22	4.093506	TotalWorkingYears
23	1.025519	TrainingTimesLastYear
24	1.017093	WorkLifeBalance
25	6.296064	YearsAtCompany
26	3.513852	YearsInCurrentRole
27	1.373189	YearsSinceLastPromotion
28	3.433437	YearsWithCurrManager

We can see that for all features Variance inflation factor in within permissible limit of 10. Multicollinearity do not pose any threat here.

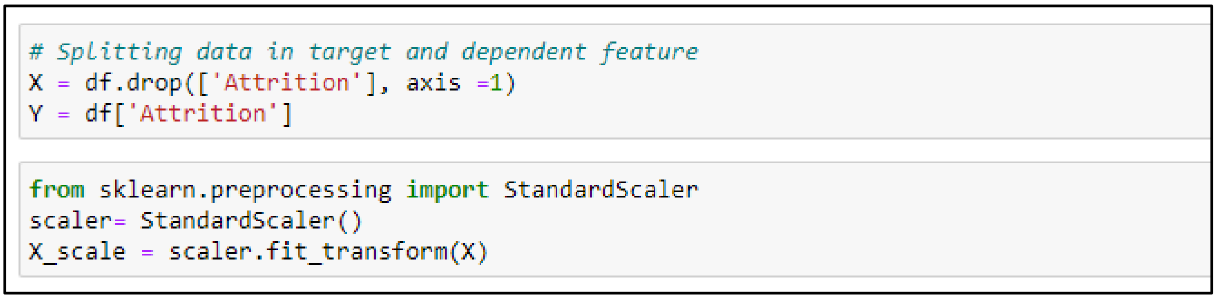
6. Handling imbalanced data using SMOTE

This two-class dataset is imbalanced (84% vs 16%). As a result, there is a possibility that the model built might be biased towards to the majority and over-represented class. We can resolve this by Synthetic Minority Oversampling Technique (SMOTE) to over-sample the minority class.



```
from imblearn.over_sampling import SMOTE
# Oversampleing using SMOTE Techniques
oversample = SMOTE()
X, Y = oversample.fit_resample(X, Y)
```

7. Scaling of data using Standard Scalar.

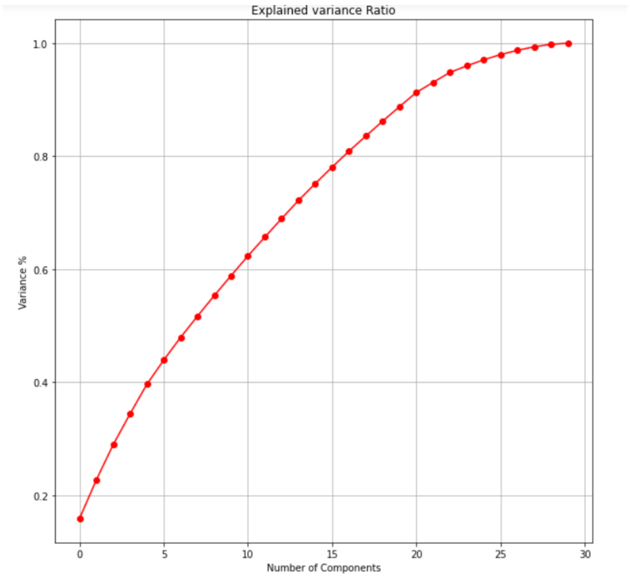


```
# Splitting data in target and dependent feature
X = df.drop(['Attrition'], axis =1)
Y = df['Attrition']
```

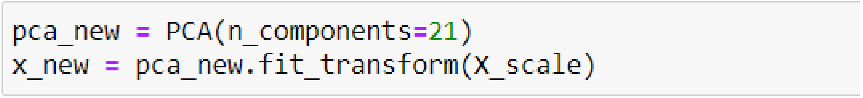
```
from sklearn.preprocessing import StandardScaler
scaler= StandardScaler()
X_scale = scaler.fit_transform(X)
```

8. Dimensionality Reduction Using PCA

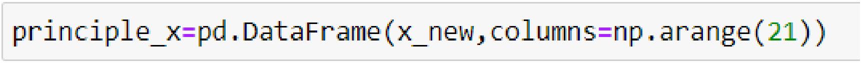
PCA used find patterns and extract the latent features from our dataset.



We can see that 21 principal components attribute for 90% of variation in the data. PCA applied for 21 components.



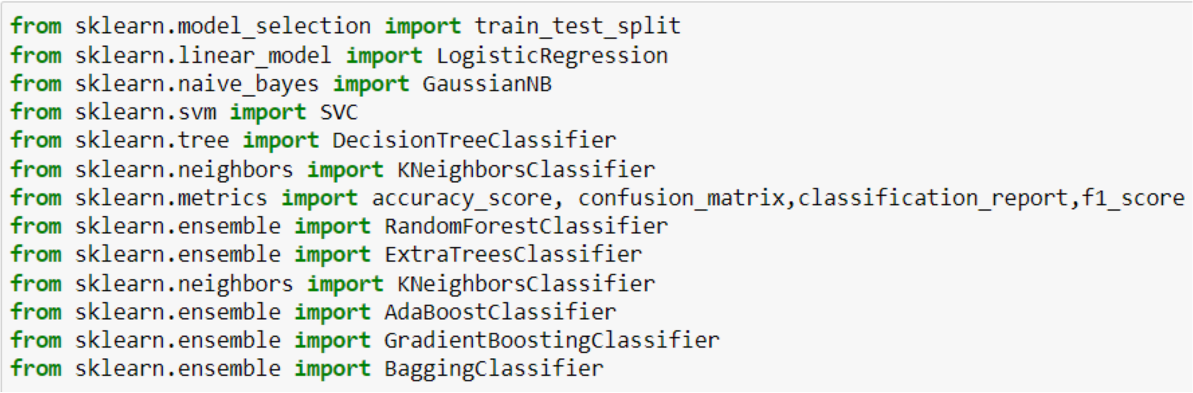
```
pca_new = PCA(n_components=21)
x_new = pca_new.fit_transform(X_scale)
```



```
principle_x=pd.DataFrame(x_new,columns=np.arange(21))
```

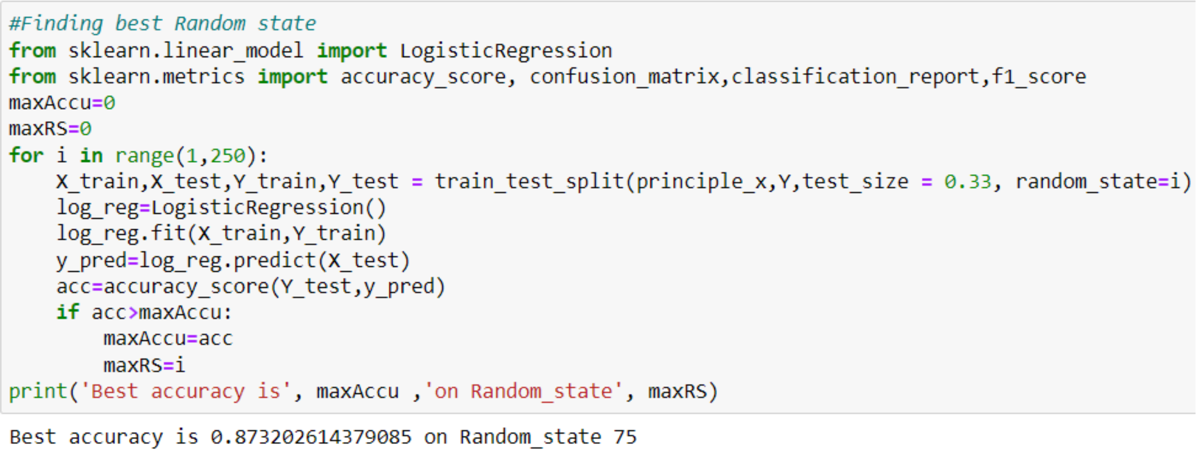
Machine Learning Model Building:

In this section we will build Supervised learning ML model-based classification algorithm. As objective is to predict attrition in 'Yes' or 'No' leads to fall problem in domain of classification algorithm. Train_test_split used to split data with size of 0.33



```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix,classification_report,f1_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import BaggingClassifier
```

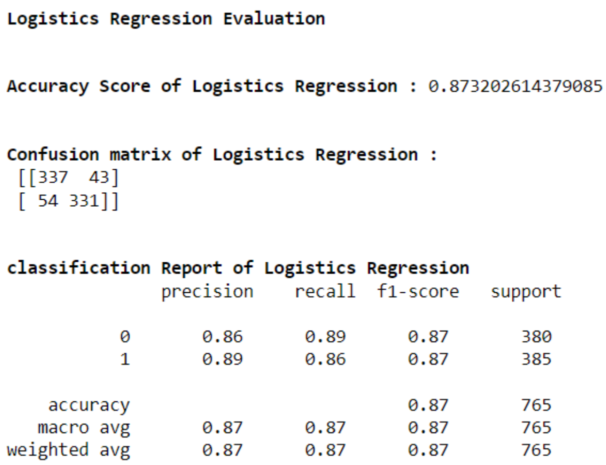
First we will build base model using logistic regression algorithim. Best random state is investigated using for loop for random state in range of (0,250).



```
#Finding best Random state
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix,classification_report,f1_score
maxAccu=0
maxRS=0
for i in range(1,250):
    X_train,X_test,Y_train,Y_test = train_test_split(principle_x,Y,test_size = 0.33, random_state=i)
    log_reg=LogisticRegression()
    log_reg.fit(X_train,Y_train)
    y_pred=log_reg.predict(X_test)
    acc=accuracy_score(Y_test,y_pred)
    if acc>maxAccu:
        maxAccu=acc
        maxRS=i
print('Best accuracy is', maxAccu ,'on Random_state', maxRS)
```

```
Best accuracy is 0.873202614379085 on Random_state 75
```

Logistics regression model is train with random state 75. The evalution matrix along with classification report is as below :



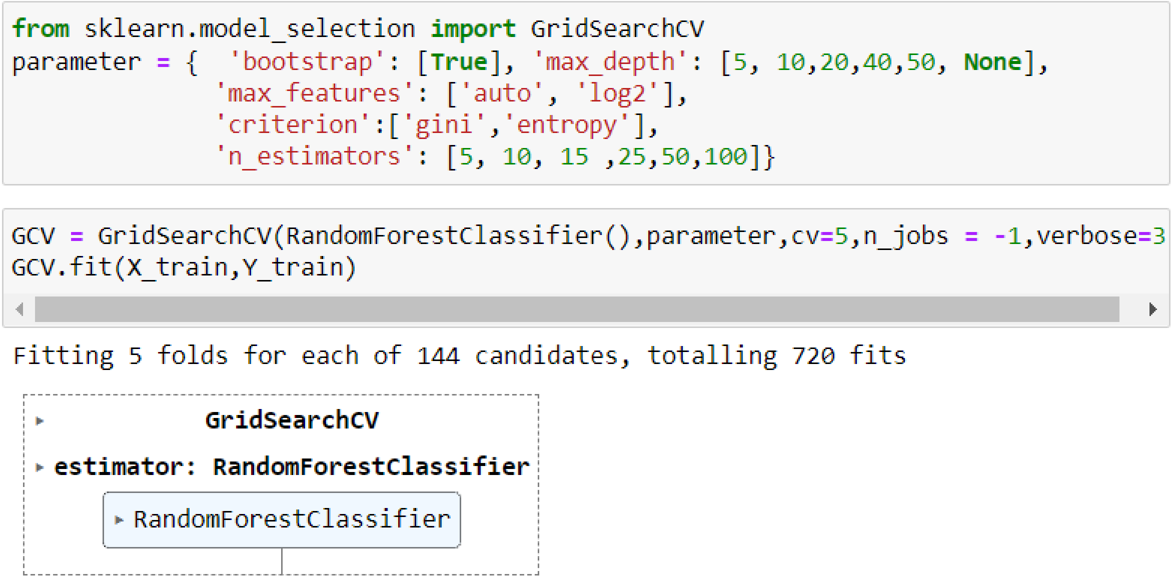
```
Logistics Regression Evaluation


Accuracy Score of Logistics Regression : 0.873202614379085


Confusion matrix of Logistics Regression :
 [[337  43]
 [ 54 331]]


classification Report of Logistics Regression
               precision    recall  f1-score   support

           0       0.86      0.89      0.87       380
           1       0.89      0.86      0.87       385

    accuracy                           0.87       765
   macro avg       0.87      0.87      0.87       765
weighted avg       0.87      0.87      0.87       765
```

As Now base model is ready with f1-score of 0.87, we will train model with different classification algorithm along with k-5 fold cross validation. The final evaluation matrix different classification algorithm is as shown table below:

ML Algorithm	Accuracy Score	CV Mean Score	f-1 Score	Recall	Precision
Logistics Regression	0.8732	0.6895	0.87	0.86	0.89
SVC	0.9137	0.6092	0.90	0.89	0.94
GaussianNB	0.8653	0.7396	0.86	0.83	0.89
DecisionTreeClassifier	0.7947	0.8390	0.80	0.80	0.79
KNeighborsClassifier	0.8653	0.7396	0.88	0.95	0.81
RandomForestClassifier	0.8980	0.9171	0.90	0.85	0.94
AdaBoostClassifier	0.8653	0.8714	0.87	0.87	0.87
GradientBoostingClassifier	0.8836	0.8844	0.88	0.88	0.89
Bagging Classifier	0.8444	0.8835	0.84	0.79	0.89

(Min Value in column -Green, Max Value in column - Pink Colour)

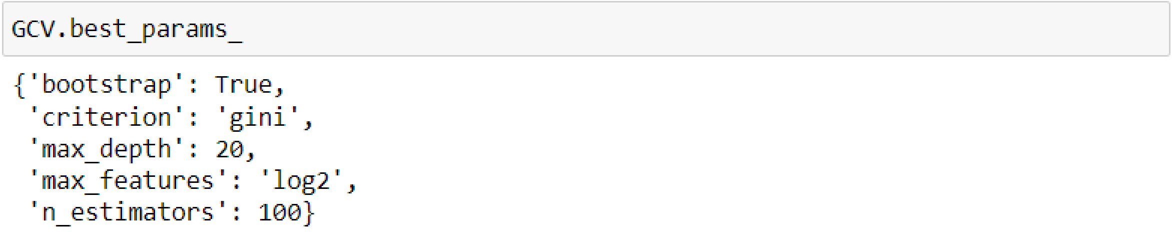
We can see that Random Forest Classifier gives us maximum f1-score & mean cross validation score. We will perform hyper parameter tuning on random forest classifier to build final ML Model.



```
from sklearn.model_selection import GridSearchCV
parameter = {  'bootstrap': [True], 'max_depth': [5, 10,20,40,50, None],
              'max_features': ['auto', 'log2'],
              'criterion':['gini','entropy'],
              'n_estimators': [5, 10, 15 ,25,50,100]}
```

```
GCV = GridSearchCV(RandomForestClassifier(),parameter,cv=5,n_jobs = -1,verbose=3
GCV.fit(X_train,Y_train)
```

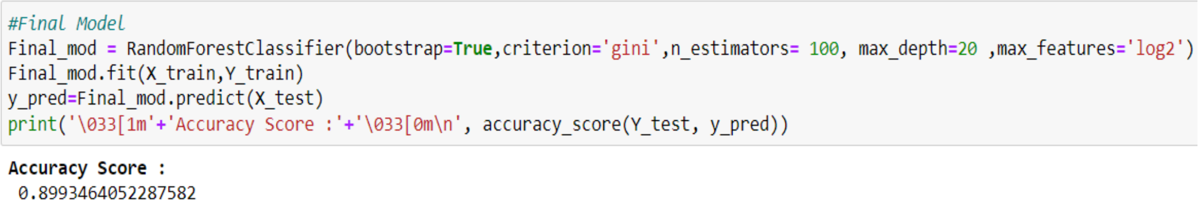
```
Fitting 5 folds for each of 144 candidates, totalling 720 fits
```



```
GCV.best_params_
```

```
{'bootstrap': True,
 'criterion': 'gini',
 'max_depth': 20,
 'max_features': 'log2',
 'n_estimators': 100}
```

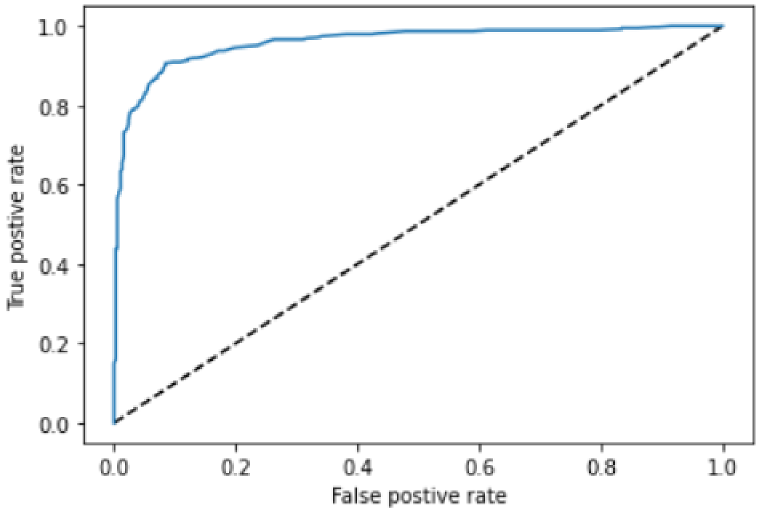
Next step is to build final machine learning model over best params in Hyper parameter tuning.

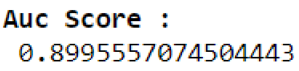


```
#Final Model
Final_mod = RandomForestClassifier(bootstrap=True,criterion='gini',n_estimators= 100, max_depth=20 ,max_features='log2')
Final_mod.fit(X_train,Y_train)
y_pred=Final_mod.predict(X_test)
print('\033[1m'+'Accuracy Score :'+'\033[0m\n', accuracy_score(Y_test, y_pred))
```

```
Accuracy Score :
 0.8993464052287582
```

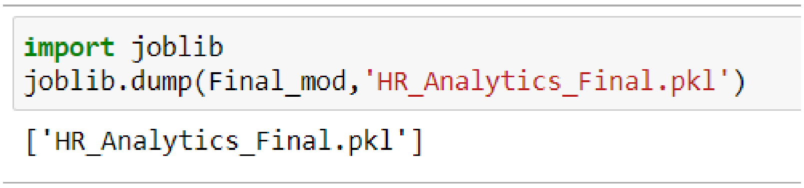
We can see that Final model with hyper parameter tuning leads to slight increase in accuracy score from 0.8980 in original model to 0.8993. This complete possible We will use model with default values as our final model. AOC-ROC score of final random forest classifier model is shown below:





```
Auc Score :
 0.8995557074504443
```

At last, we will save final model with joblib library.



```
import joblib
joblib.dump(Final_mod,'HR_Analytics_Final.pkl')
```

```
['HR_Analytics_Final.pkl']
```

Concluding Remarks on EDA and ML Model

- Bench mark of 6900$ monthly income is recommended to Prevent attrition.
- Attrition rate is high in age group of 29 to 33. HR need to keep eye over need & expectation of this age group from company.
- Percentage of attrition is high in Sales Representative, Laboratory Technician
- 16 % attrition rate among Research Scientist and no company afford to lose them.
- Almost 50% employs in sales department from different education background. There is possibility of dissatisfaction among them as attrition high among these.
- Different feature engineering techniques like balancing data, outliers' removal, label encoding, feature selection & PCA are perform on data.
- Random Forest Classifier model gives maximum Accuracy.