**HR ANALYTICS PROJECT: BLOG BY MAYUR GAUNKER**



Problem Statement:

Every year a lot of companies hire a number of employees. The companies invest time and money in training those employees, not just this but there are training programs within the companies for their existing employees as well. The aim of these programs is to increase the effectiveness of their employees. But where HR Analytics fit in this? and is it just about improving the performance of employees?

**HR Analytics**

Human resource analytics (HR analytics) is an area in the field of analytics that refers to applying analytic processes to the human resource department of an organization in the hope of improving employee performance and therefore getting a better return on investment. HR analytics does not just deal with gathering data on employee efficiency. Instead, **it aims to provide insight into each process by gathering data and then using it to make relevant decisions about how to improve these processes.**

**Attrition in HR**

Attrition in human resources refers to the gradual loss of employees overtime. In general, relatively high attrition is problematic for companies. HR professionals often assume a leadership role in designing company compensation programs, work culture, and motivation systems that help the organization retain top employees.

How does Attrition affect companies? and how does HR Analytics help in analyzing attrition? We will discuss the first question here and for the second question, we will write the code and try to understand the process step by step.

**Attrition affecting Companies**

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. Additionally, regular employee turnover prohibits your organization from increasing its collective knowledge base and experience over time. This is especially concerning if your business is customer-facing, as customers often prefer to interact with familiar people. Errors and issues are more likely if you constantly have new workers.

In this article we are going to deep dive into the employee details viz .their department, Education background, travel frequencies, gender, job satisfaction levels. Others related to salary they were withdrawing ,OT hours , their age and years spent in the company. After analyzing all the factors the aim is to predict the attrition of the employees to help retain the employees which are the most important assets by improving the processes.

As the generic rule we will follow the initial prescribed methodology for building Machine learning models

We begin our analysis by importing the 2 important libraries which will be used for computing numerical problems and creating DataFrame

Pandas & Numpy



Pandas makes importing and analyzing the dataset much easier, it provides various filtering & transformations as provided by the excel

Numpy is that Computing in python. It provides various tools for mathematical computation. Some of them are np.array; array.shape; np.append etc

Other libraries are

1.Seaborn

This library is mostly used for creating graphical visual for analysis purpose. It helps one explore and understand your data

2.Scikit-learn :

**Scikit-learn** has a wide range of supervised and unsupervised learning algorithms which are used for various data analytics and data mining. The machine learning algorithms used in this projects are linear\_model, StandardScaler , LabelEncoder , LogisticRegression



**DATA ANALYSIS PROCESS**

Importing Important Libraries

We use import function to get the python libraries. We have imported pandas & numpy for basic computation and making the dataframe, sklearn for machine learning , matplotlip for creating the graphs and warnings , which is used to show warning messages , we have filtered these messages using “ignore”

**import** pandas **as** pd

**import** numpy **as** np

**from** sklearn **import** linear\_model

**from** sklearn.preprocessing **import** StandardScaler

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** sns

**%matplotlib** inline

**import** statistics **as** stat

**import** math

**import** warnings

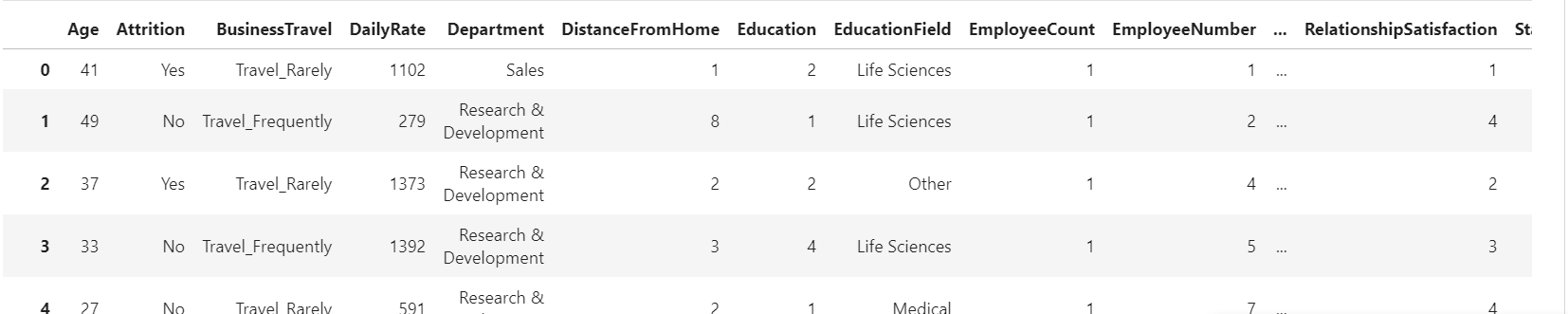
warnings**.**filterwarnings("ignore")

Importing the dataset

Pandas has been used to import the dataset. The dataset we are dealing with currently is in CSV format. It represents the data into table format with columns & rows as seen in the excel.

*# Getting the dataset*

df**=**pd**.**read\_csv('C:/Users/quccs/Downloads/ibm-hr-analytics-employee-attrition-performance (1)/WA\_Fn-UseC\_-HR-Employee-Attrition.csv')



Using df.columns code to see the column names in dataframe

*# Checking column names*

df**.**columns

*# Checking the shape*

df**.**shape

Using df.shape code to get the size of the dataframe

Data Cleaning

In real life scenarios missing data are always the main problems along with the null and some unwanted contents which required to be removed.

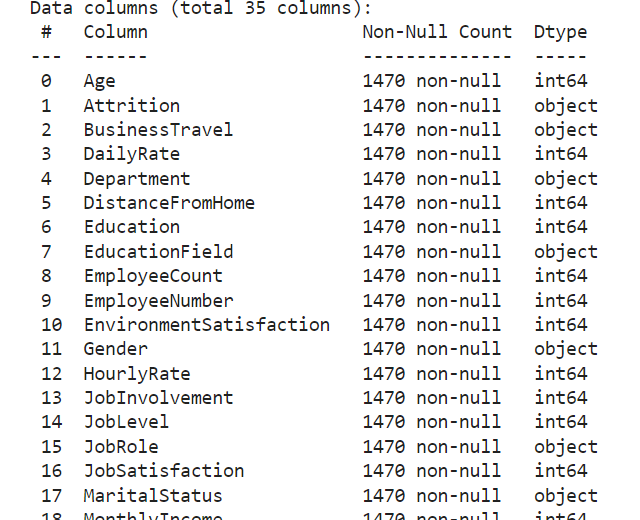
df**.**info()

Understanding each column datatype and null values if any

int64 depicts the integer value

object is for text

float for decimal numbers



Help in checking the null values in our dataset

df**.**isnull()**.**sum()

As we see there are no null values in the dataset so can proceed with EDA.

However, in most of the cases will encounter null values. To deal with this the most common function we use to fill missing values is fillna using mean, median or mode methods whichever applicable. Before using this understanding of the data distribution is mandatory



Label Encoding

The target variable here is Attrition which is present in the categorical form in object type as “YES” & “NO” , however for model building will require numerical variable.

Using Label encoding for converting categorical variable into numeric value of 0 & 1.

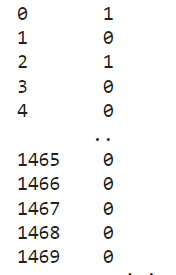
**from** sklearn.preprocessing **import** LabelEncoder

Importing label encoder from sklearn package

le**=**LabelEncoder()

df["Attrition"]**=**le**.**fit\_transform(df["Attrition"])

df["Attrition"]



“YES” will be assigned as 1 and “NO” will be 0.

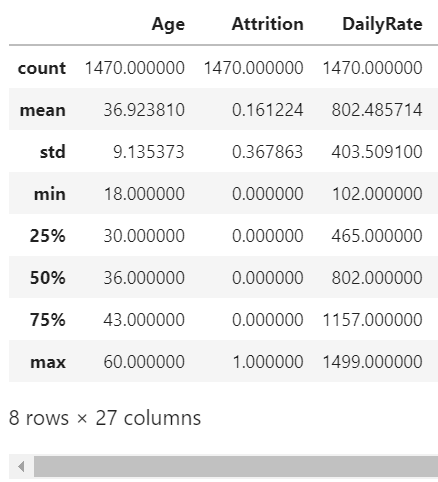
It follows the alphabetical order while assigning

Statistics

Understanding the dataset begins with analyzing each feature and digging into more details , statistics help us in understanding the data , setting the right mathematical approach.

# Checking the statistics of the dataset

df**.**describe()



Exploratory Data Analysis

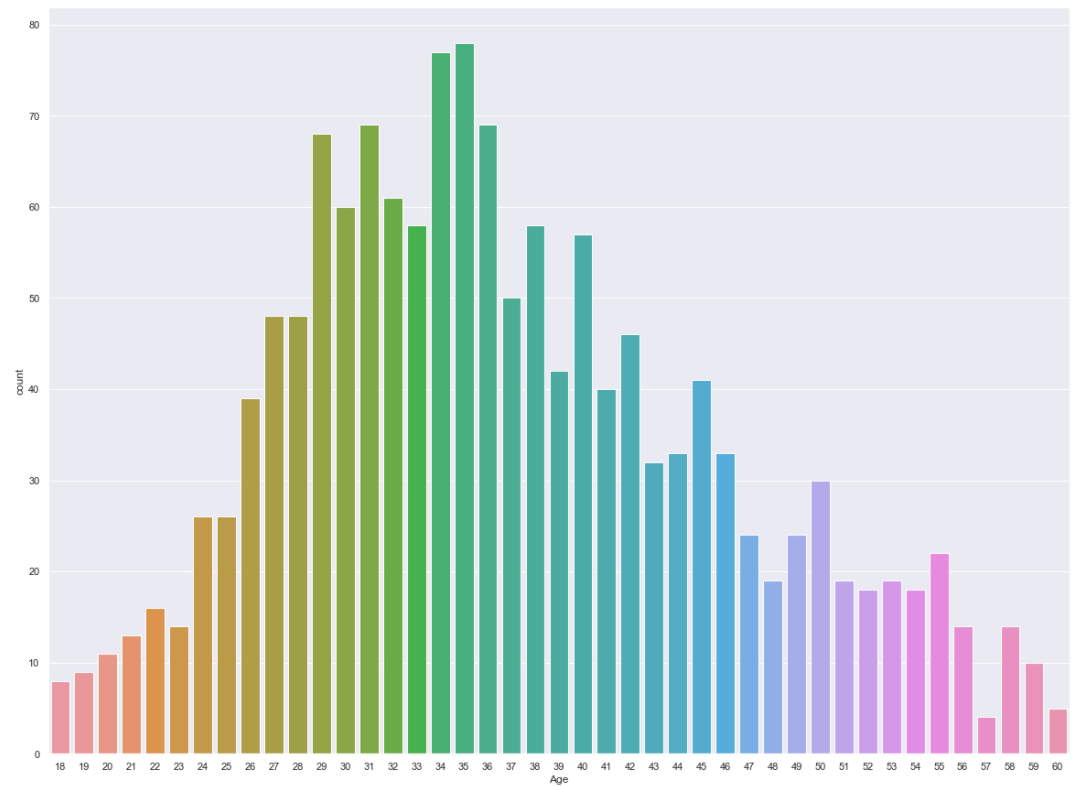
Here we use various graphs to analyze the data. This is used to communicate the information to the end user in most profound and understandable manner for decision making.

We try to get the maximum insights from the dataset and present the same using visualization tools.

1. Age distribution of the employees

sns**.**countplot(x**=**"Age",data**=**df)

sns**.**set(rc **=** {'figure.figsize':(15,15)})

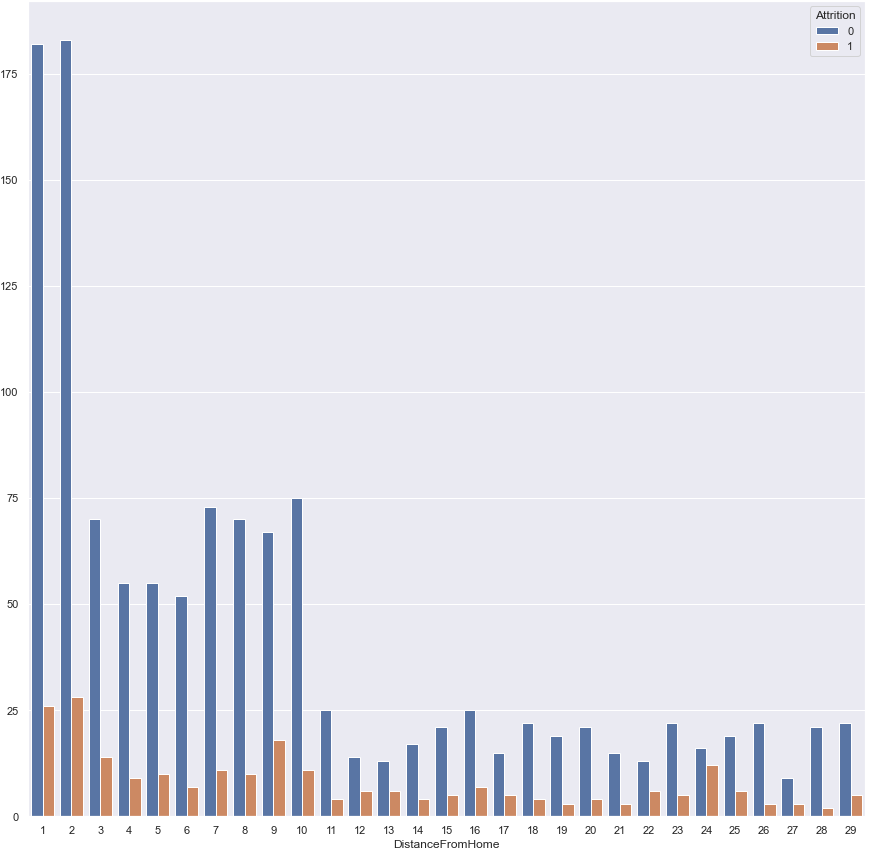


Most of the employees are between 26 to 45. We see the peak at 34 and 35 depicting the highest no of employees belonging to this age group

1. DistanceFromHome vs Attrition

sns**.**countplot(df['DistanceFromHome'],hue**=**df['Attrition'])

sns**.**set(rc **=** {'figure.figsize':(20,15)})

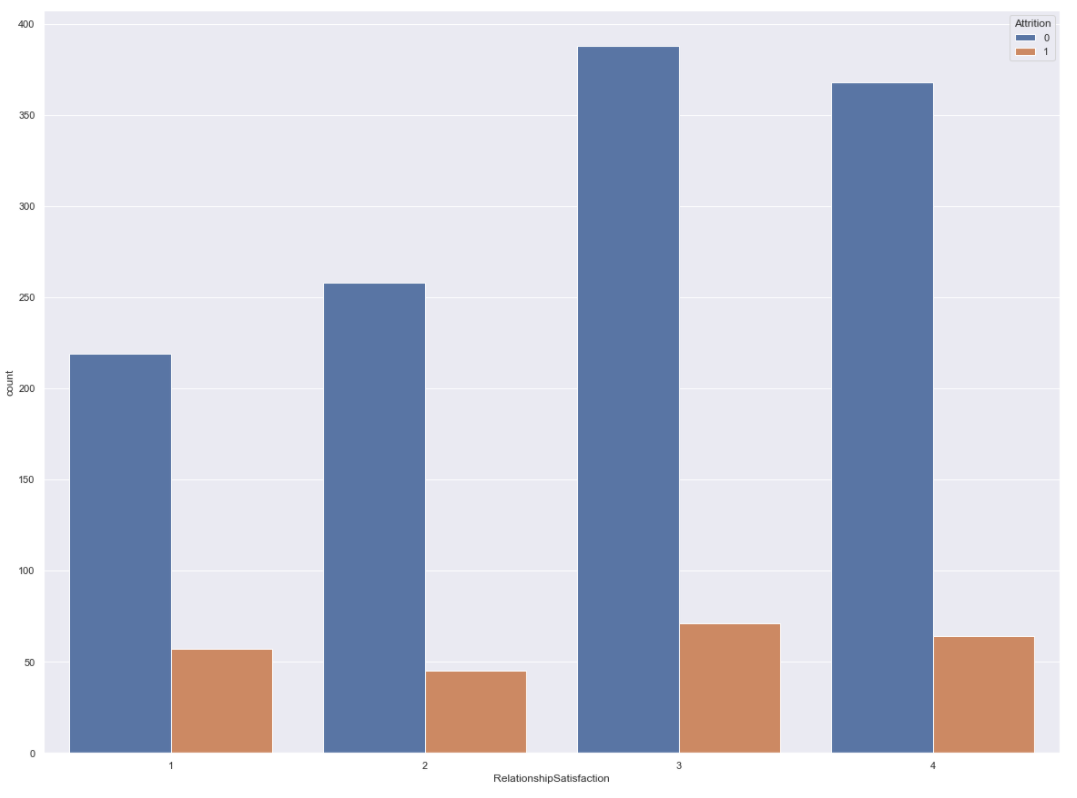


There is high chance of employee retention who stayed closer to the company. The people are more likely to leave if it is their hometown or the company at the nearby distance. It saves their commute time and need not need to leave in rented apartments or travel consuming their pockets

c)Relationship Satisfaction vs Attrition

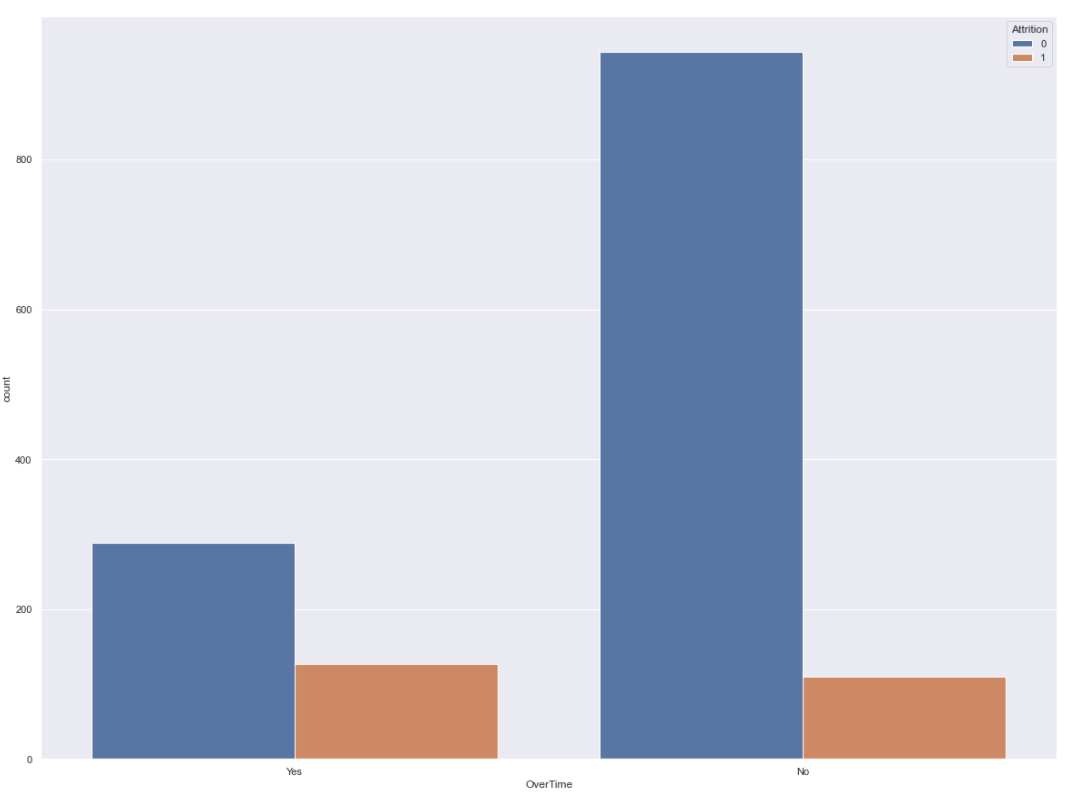
This doesn’t tell us much of the details. The attrition is same across all the satisfaction levels.

sns**.**countplot(df['RelationshipSatisfaction'],hue**=**df['Attrition'])



D)Overtime vs Attrition

sns**.**countplot(df['OverTime'],hue**=**df['Attrition'])

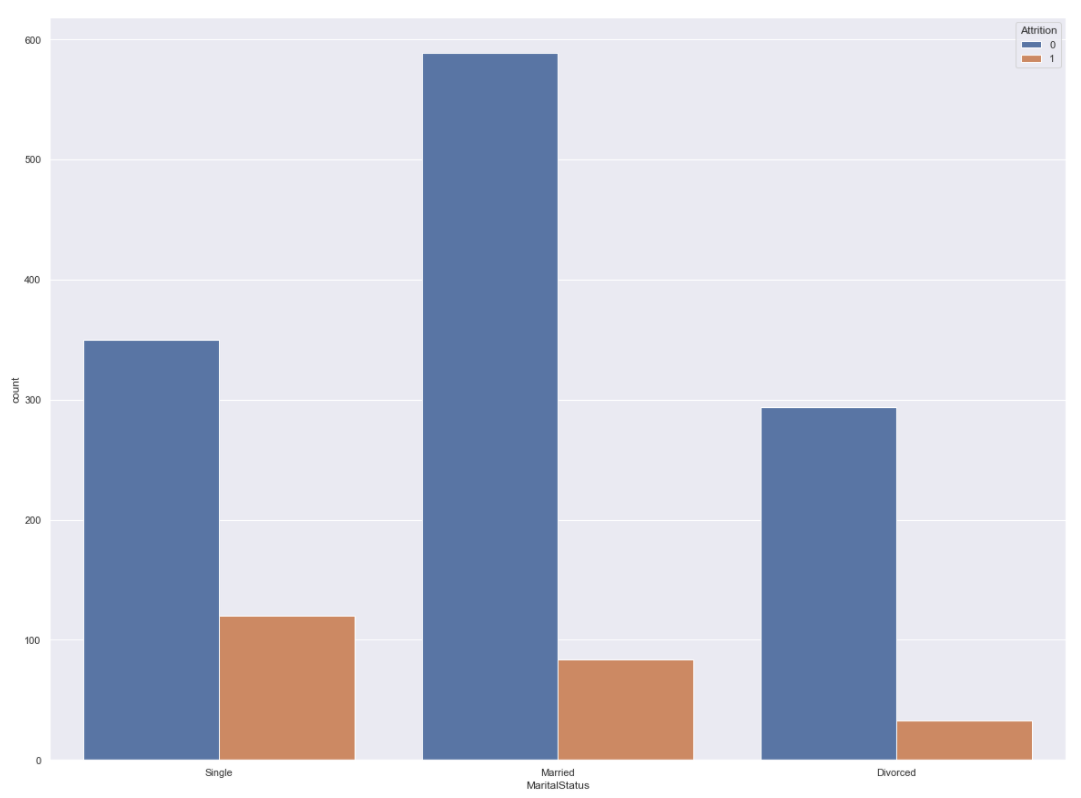


Same attrition rate of employees based on the overtime factor

E) Marital status vs Attrition

sns**.**countplot(df['MaritalStatus'],hue**=**df['Attrition'])

Out[60]:



Not much impact of marital status on the attrition. Surprisingly we see the low attrition for the divorced employees.

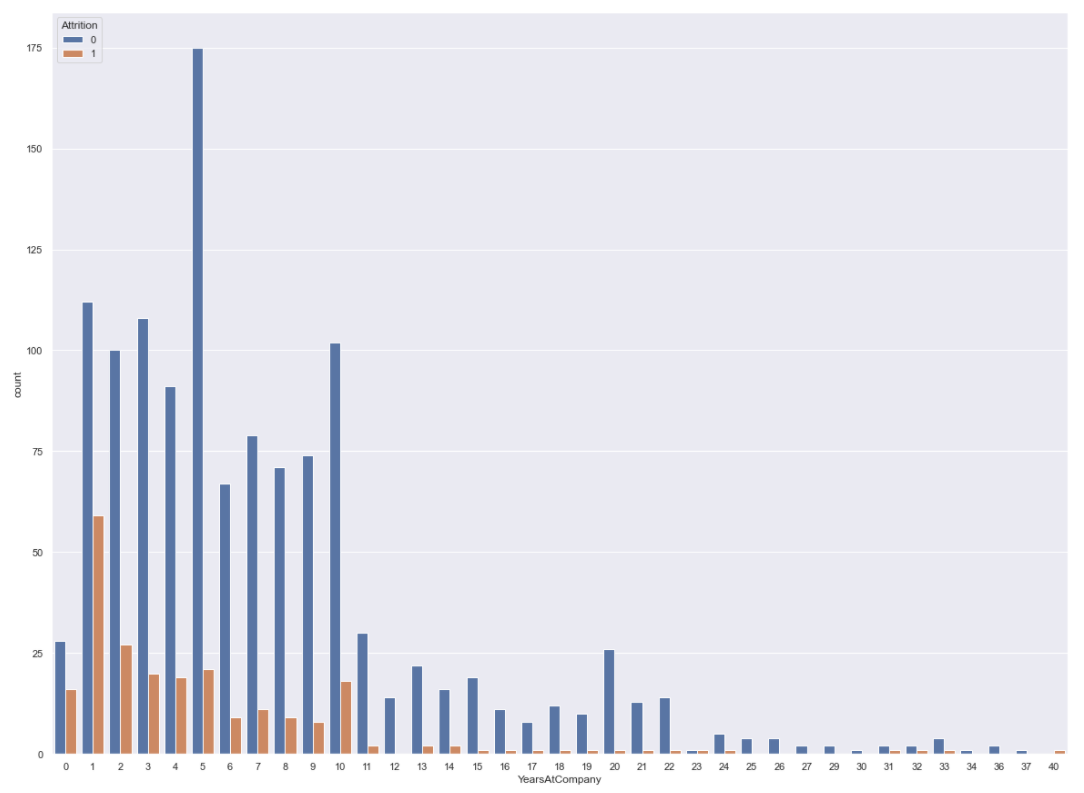
The attrition is high for single , usually at the beginning of their career people do switch and it decreases for the experiences employees which will be either married or divorced.

F) Years at company vs Attrition

The attrition rate was very low generally for the employees who stayed for the longer years. The attrition rate was high for the employee duration of 1 to 5 years the maximum for the 1 & 2 years.

When employees spent most of the years in the same company, they are less likely to switch the company as most of them will already be at some managerial position or director level and it gets difficult to switch in the same profile. Also, the openings for higher position are less. Most gets adjusted to the environment and are comfortable and already good at their job so wont be taking risk at the later stage of the career. Also, the network & connections need not be built up against the switch.

sns**.**countplot(df['YearsAtCompany'],hue**=**df['Attrition'])

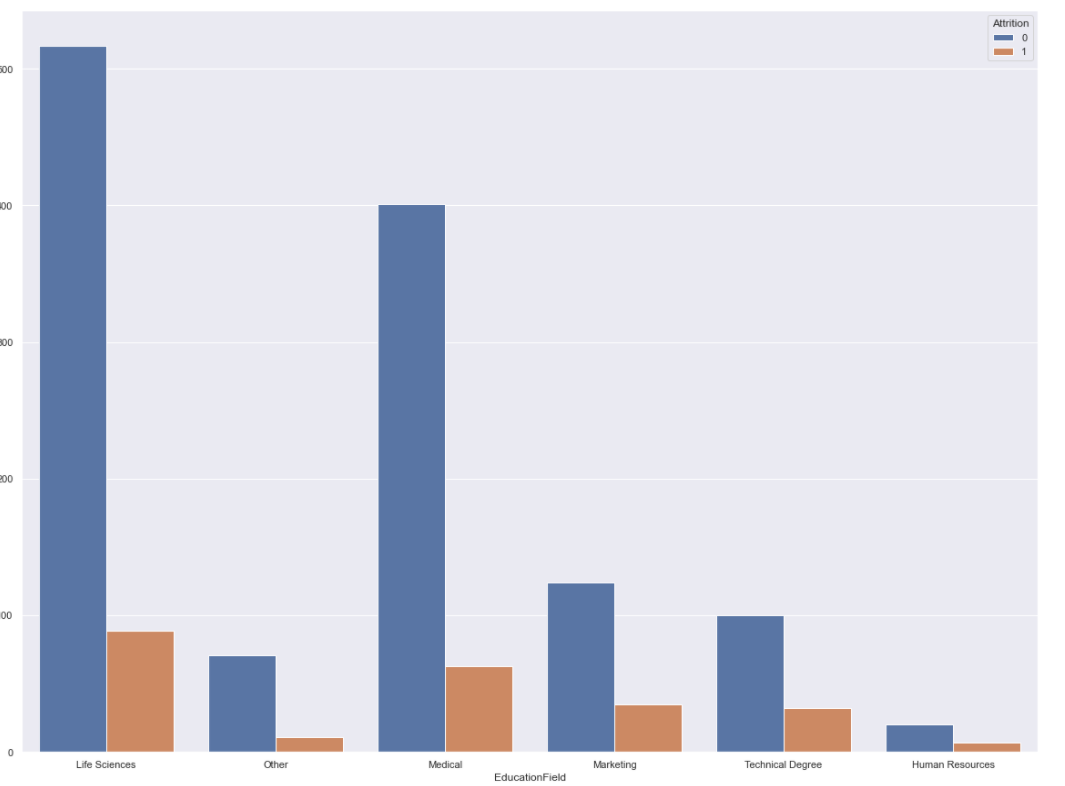


G) Education field vs Attrition

When we analyze the data considering the education background what we see is the lowest attrition is for Human Resource & others department. Generally HR holds most of the Confidenial information about the company and management.

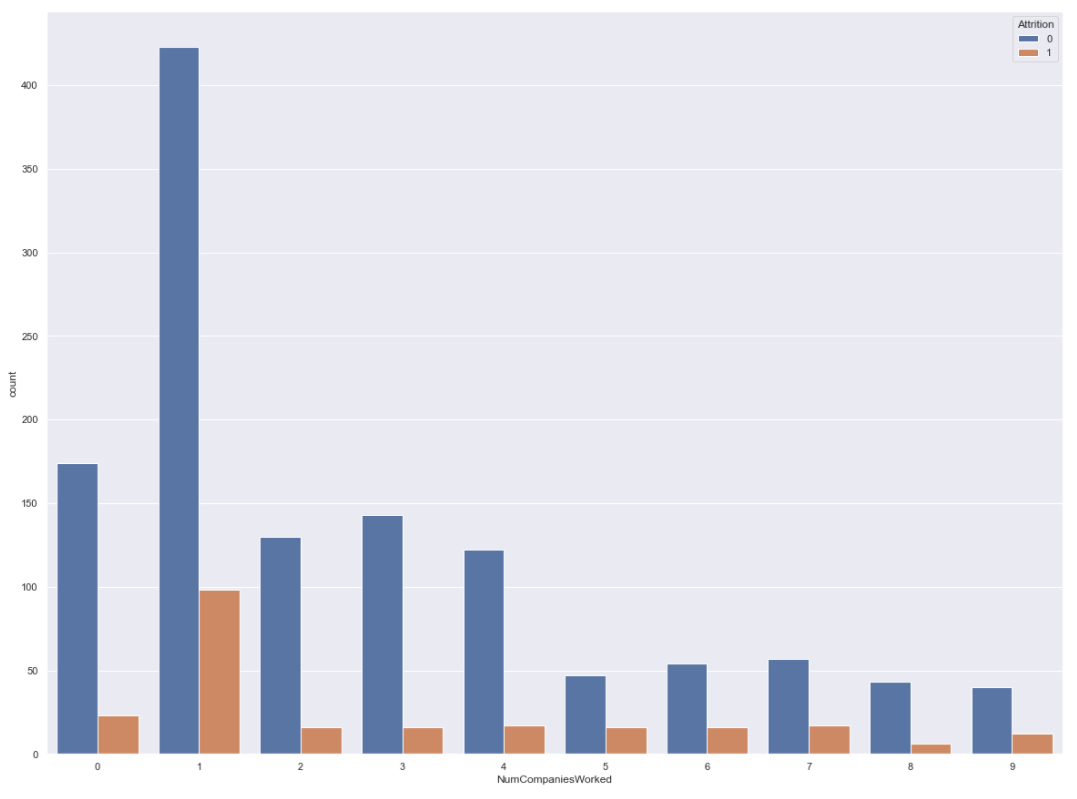
For other departments the attrition is almost same except for life sciences where the attrition is high.

sns**.**countplot(df['EducationField'],hue**=**df['Attrition'])



H) Numbr of companies worked vs Attrition

sns**.**countplot(df['NumCompaniesWorked'],hue**=**df['Attrition'])



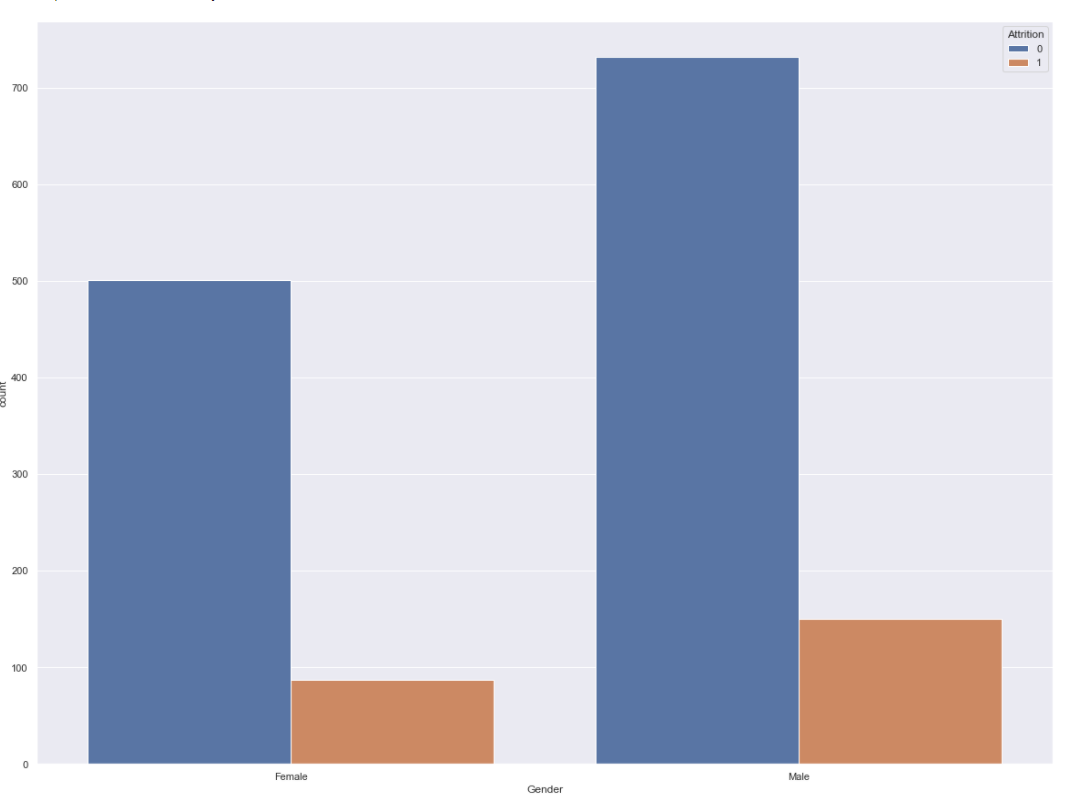
Employees who have worked in a single company are more likely to leave. They are mostly freshers who wants to experience new fields and companies that will match their skills and values.

Most of the beginners are in a search for change in their career field. This is also the best time for individuals to get exposure of different culture and work ethics.

I) Gender vs Attrition

The attrition is high for male employees

sns**.**countplot(df['Gender'],hue**=**df['Attrition'])



Corr

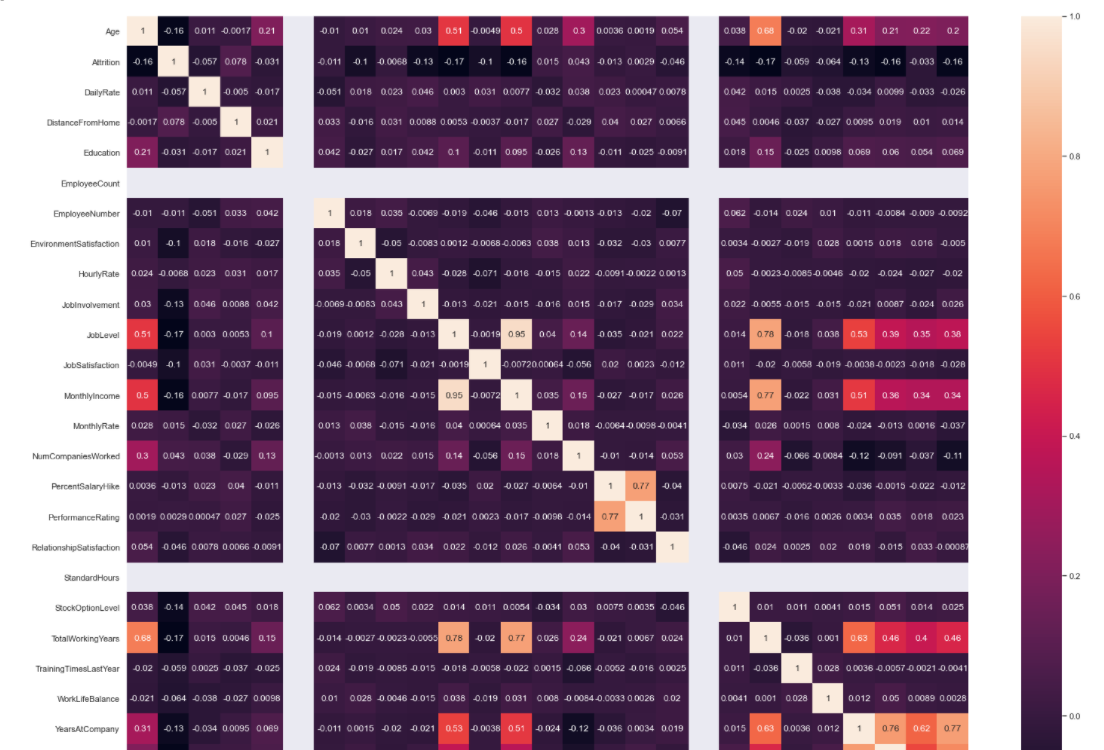
Here we have used df.corr function to find the pairwise correlation between the features.

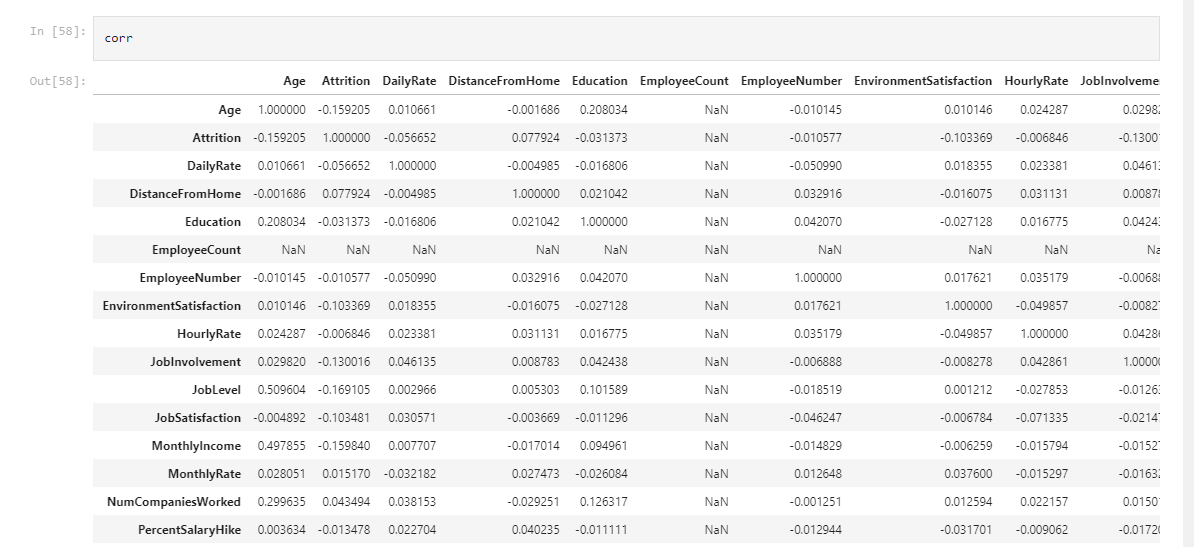
The lighter gradient represent strong correlational value.

corr**=**df**.**corr()

plt**.**figure(figsize**=**(25,20))

sns**.**heatmap(corr,annot**=True**,)





Removing Unwanted columns:

df**.**drop("EmployeeNumber",axis**=**1,inplace**=True**)

df**.**drop("EducationField",axis**=**1,inplace**=True**)

df**.**drop("OverTime",axis**=**1,inplace**=True**)

df**.**drop("MaritalStatus",axis**=**1,inplace**=True**)

df**.**drop("EmployeeCount",axis**=**1,inplace**=True**)

df**.**drop("NumCompaniesWorked",axis**=**1,inplace**=True**)

df**.**drop("BusinessTravel",axis**=**1,inplace**=True**)

df**.**drop("Department",axis**=**1,inplace**=True**)

1.Dropping the column which has less correlational value and based on the EDA analysis done above

2.Also the decision of removing fields cannot be made solely on the correlational value as the sample size is small viz

3.work life balance, relationship satisfaction, Hourly and daily rate etc .. thus keeping these columns which may have significant contribution for making decision on attrition

Dealing with Outliers:

In order to build the efficient model the outlier removal is a critical process. Outliers can shape the model in a different way and thus the prediction.

The black dot which we see beyond the max & min limits are the outliers which needs to be removed.

We have used the box plot to detect the outliers.

*# Checking the outliers for each column*

plt**.**figure(figsize**=**(20,25), facecolor **=**'white')

pltnumber**=**1

**for** column **in** df:

**if** pltnumber**<=**27:

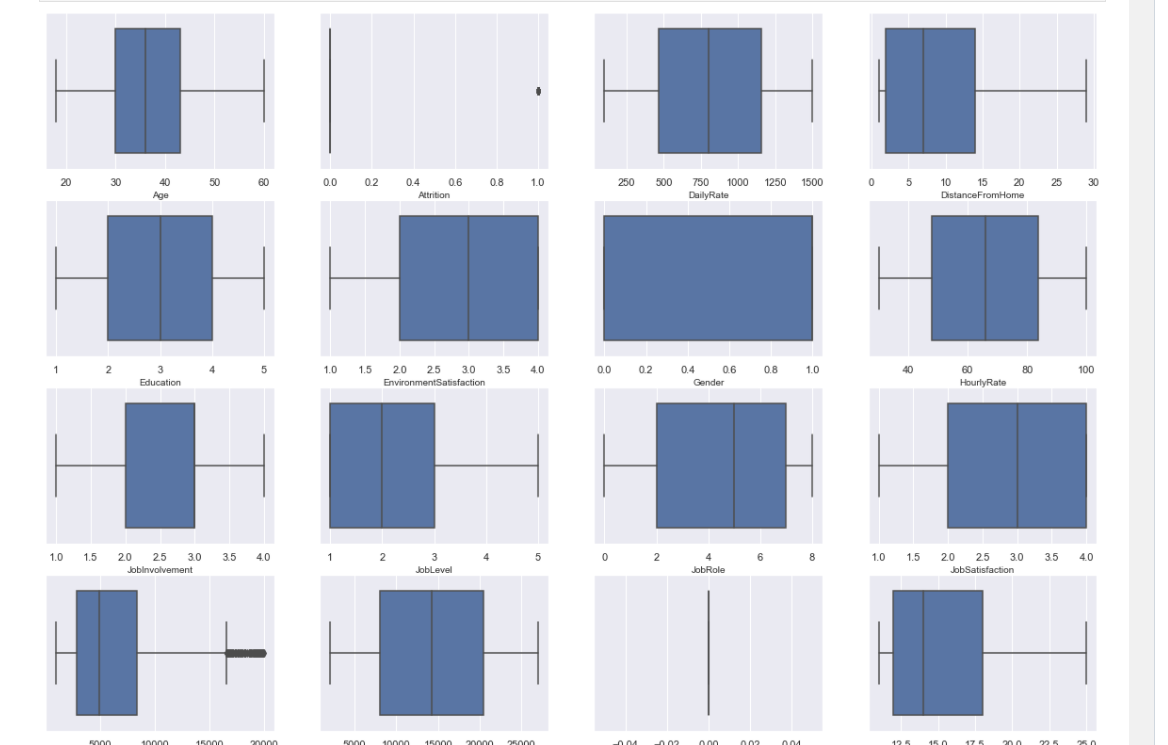
ax **=** plt**.**subplot(7,4,pltnumber)

sns**.**boxplot(df[column])

plt**.**xlabel(column,fontsize**=**10)

pltnumber **+=**1

plt**.**show() *#plt.show()*



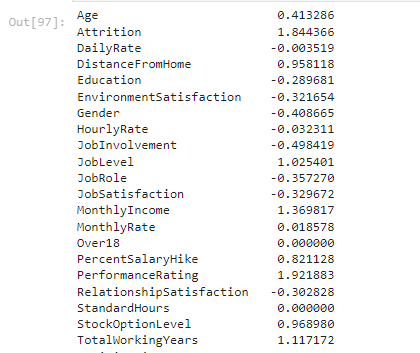
Checking for skewness :

Skewness defines the data distribution across the mean viz right skewed or left skewed.It show how much is our data deviates from the normal distribution.

It also tells about the direction of the outliers.

# Checking for the Skewness in the dataset

df**.**skew()



The skewnees is high in Attrition and years since last promotion followed by years at company

Removing Skewness:

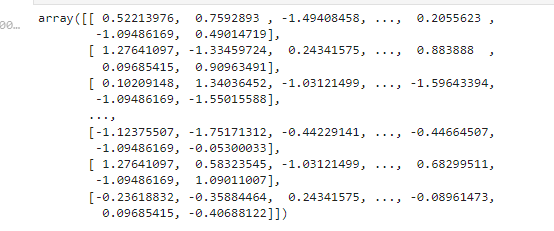
#Removing skewness in the dataset using power transform method

#Importing power transform from library

**from** sklearn.preprocessing **import** power\_transform

x**=**power\_transform(x,method**=**"yeo-johnson")

x



Feature Selection:

Here we select the target variable and the features by assigning them to individual variables

x**=**df**.**drop("Attrition",axis**=**1)

y**=**df["Attrition"]

Scaling the dataset: Usually the dataset will contain features of different units , which are differet in scale. Thus to standardize the dataset we have used Standard Scaler

#importing standard scaler from sklearn

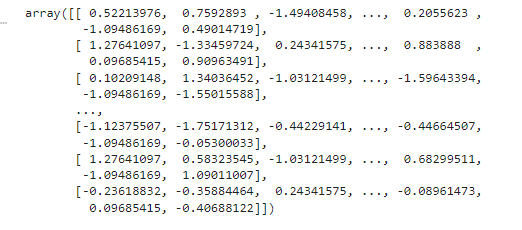
# Scaling the data using standard scaler

**from** sklearn.preprocessing **import** StandardScaler

sc**=**StandardScaler()

x\_scaled**=**sc**.**fit\_transform(x)

x\_scaled



Model Training:

This is a stage where our model is learning the values of all the attributes that we are feeding.

We feed the training data here. Quality of the data and the transformations used playes the important role.

#Training the model

**from** sklearn.model\_selection **import** train\_test\_split

x\_train,x\_test,y\_train,y\_test**=**train\_test\_split(x\_scaled,y,test\_size**=**.25,random\_state**=**56)

Model Selection:

This is the process of shortlisting the best model which gives the highest accuracy score and based on other key parameters like precision, recall, f1 score etc

#Importing ml libraries

*#*

**from** sklearn.model\_selection **import** train\_test\_split

**from** sklearn.linear\_model **import** LogisticRegression

**from** sklearn.metrics **import** r2\_score

**from** sklearn.metrics **import** mean\_squared\_error,mean\_absolute\_error

**from** sklearn.model\_selection **import** cross\_val\_score

**from** sklearn.metrics **import** accuracy\_score,confusion\_matrix,classification\_report

***#DecisionTreeClassifier***

**from** sklearn.tree **import** DecisionTreeClassifier

dtc**=**DecisionTreeClassifier()

dtc**.**fit(x\_train,y\_train)

predtc**=**dtc**.**predict(x\_test)

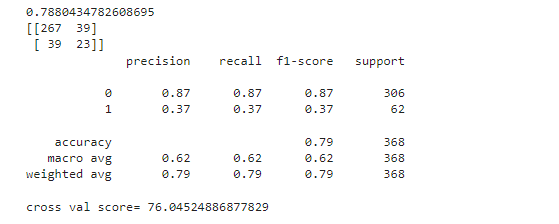
cvscore**=**cross\_val\_score(DecisionTreeClassifier(),x\_train,y\_train,cv**=**5)**.**mean()

print(accuracy\_score(y\_test,predtc))

print(confusion\_matrix(y\_test,predtc))

print(classification\_report(y\_test,predtc))

print("cross val score=",(cvscore**\***100))



#SVC

*#SVC*

**from** sklearn.svm **import** SVC

svc**=**SVC()

svc**.**fit(x\_train,y\_train)

predsvc**=**svc**.**predict(x\_test)

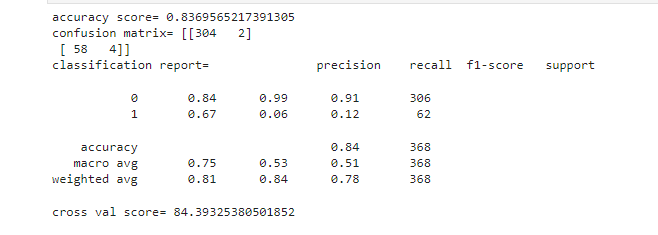
cvscore**=**cross\_val\_score(SVC(),x\_train,y\_train,cv**=**5)**.**mean()

print("accuracy score=",accuracy\_score(y\_test,predsvc))

print("confusion matrix=",confusion\_matrix(y\_test,predsvc))

print("classification report=",classification\_report(y\_test,predsvc))

print("cross val score=",(cvscore**\***100))



# The accuracy score is 83%

We are selecting Support Vector machine learning algorithm as the best model as it gives the highest accuracy score amongst all and also the difference between the Accuracy score and Cross val score is less

Saving the model:

*#saving the file*

**import** pickle

filename**=**"HRANALYTICS\_Attrition rate.pkl"

pickle**.**dump(svc,open(filename,"wb"))

AUC-ROC Curve:

ROC Curve is a performance measurement curve for the classification problems based on the various threshold values

*# importing roc curve from sklearn library*

**from** sklearn.metrics **import** roc\_curve,auc

fpr,tpr,threshold**=**roc\_curve(predsvc,y\_test)

roc\_auc**=**auc(fpr,tpr)

plt**.**figure()

plt**.**plot(fpr,tpr,color**=**"red",lw**=**10,label**=**"ROC\_CURVE (area-%0.2f)"**%roc\_auc**)

plt**.**plot([0,1],[0,1],color**=**"blue",lw**=**10,linestyle**=**'--')

plt**.**xlabel("False Positive Rate")

plt**.**ylabel("True Positive Rate")

plt**.**title("Reciving Operating Charaterictic")

plt**.**show()

