HR Analytics Project- Understanding the Attrition in HR

# Introduction:

**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.

**How does HR Analytics Help in analyzing attrition?**

For this we build ML Model , to get clear understanding. Following are the steps involved in the process:

1. Problem Statement
2. Data Source
3. Importing Necessary Libraries
4. Data Preprocessing
5. Finding categorical features and do cardinality check on them.
6. Exploratory Data Analysis
7. Encoding categorical features
8. Correlation
9. Skewness and Outlier Detection and removal
10. Scaling
11. VIF (Variance Factor Inflation) for Multicollinearity
12. Feature Selection Technique (SelectKBest)
13. Splitting Data into Training and Testing sets
14. Model Building and Evaluation
15. Dealing with data imbalance (SMOTE Technique)
16. Results and Conclusion
17. Save Model and Scaling object with Pickle

## Problem Statement:

Design a predictive model with the use of machine learning algorithms to predict weather an employee stays with the company or leave the company i.e. the attrition (Y/N).

## Data Source:

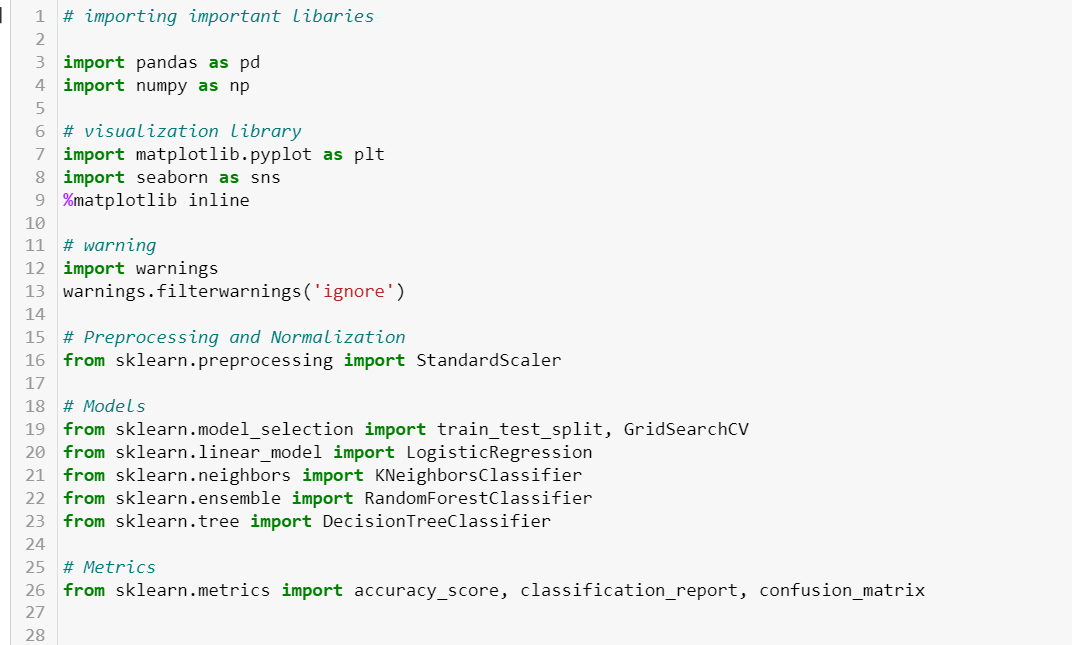
The data set is provided by the academy, and contain 35 columns including the label and 1470 rows. Consisting of details about the employees belonging to different department.

Data Description :

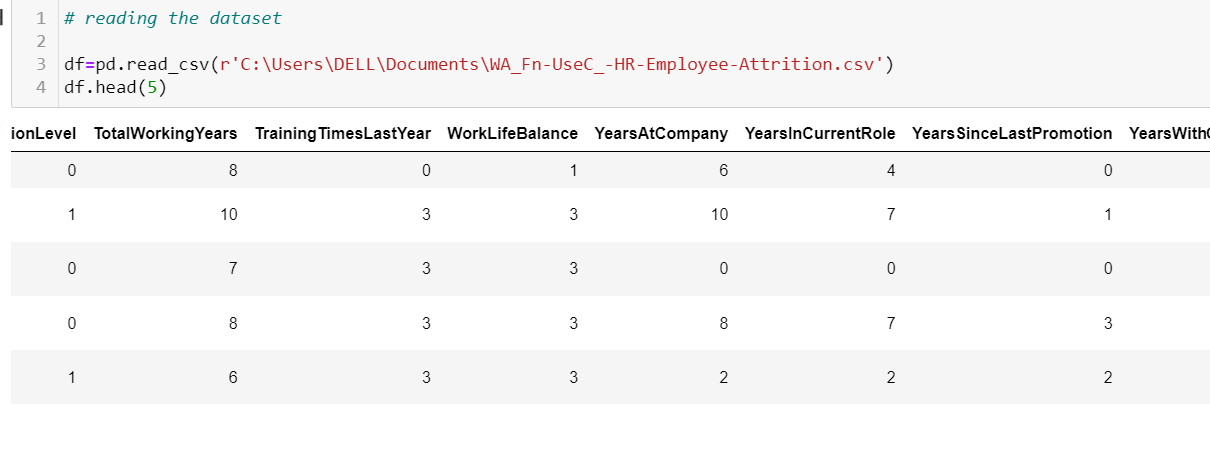
Number of columns: 35, Number of rows: 1470, Number of Independent Columns: 34 , Number of Dependent column: 1.

## Importing Libraries:

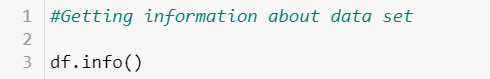
## The first step in Data Analysis is to import all the necessary libraries.

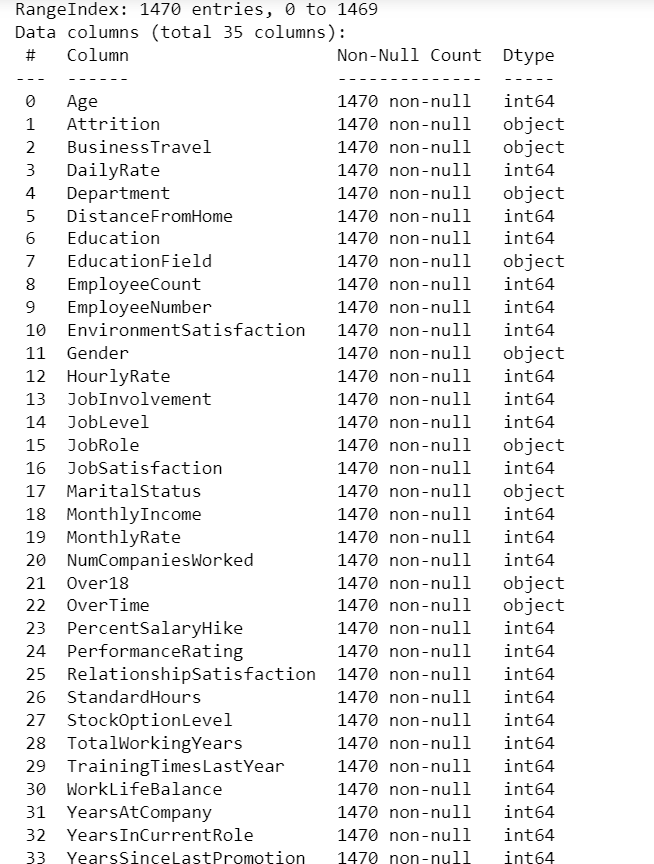


Now, Let’s go ahead and load the data, which we want to analyse.



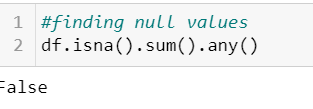
# **Checking the Information about the data,** this helps us to get complete info about dataset, its shape, datatypes of columns, Non Null values, etc. In short it gives concise summary of dataset.





We can even see the memory usage at the bottom of this list.

# **Checking for Null values in the dataset.**

****

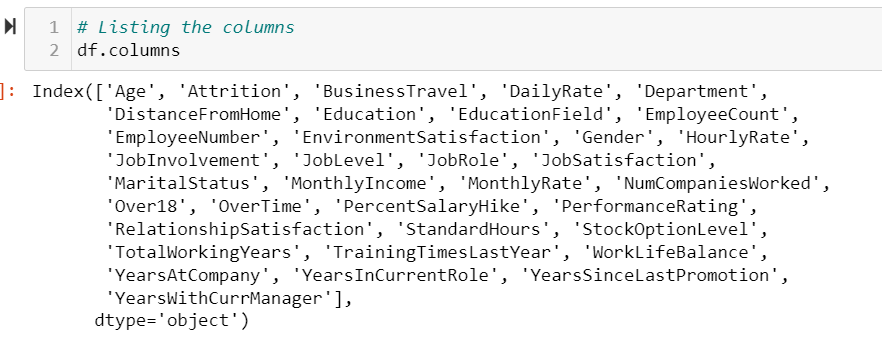
**Here** false denotes that no null values are present in the dataset.

## Data Pre-Processing:

Real-world data is often **messy**, **incomplete**, **unstructured**, **inconsistent**, **redundant**, sprinkled with wacky values. So, without deploying any Data Preprocessing techniques, it is almost impossible to gain insights from raw data.

Data preprocessing is a process of converting raw data to a suitable format to extract insights. It is the first and foremost step in the Data Science life cycle. Data Preprocessing makes sure that data is clean, organize and read-to-feed to the Machine Learning model.

# Checking all the columns in the dataset: column function is used



# Checking datatype of each column using .dtype function



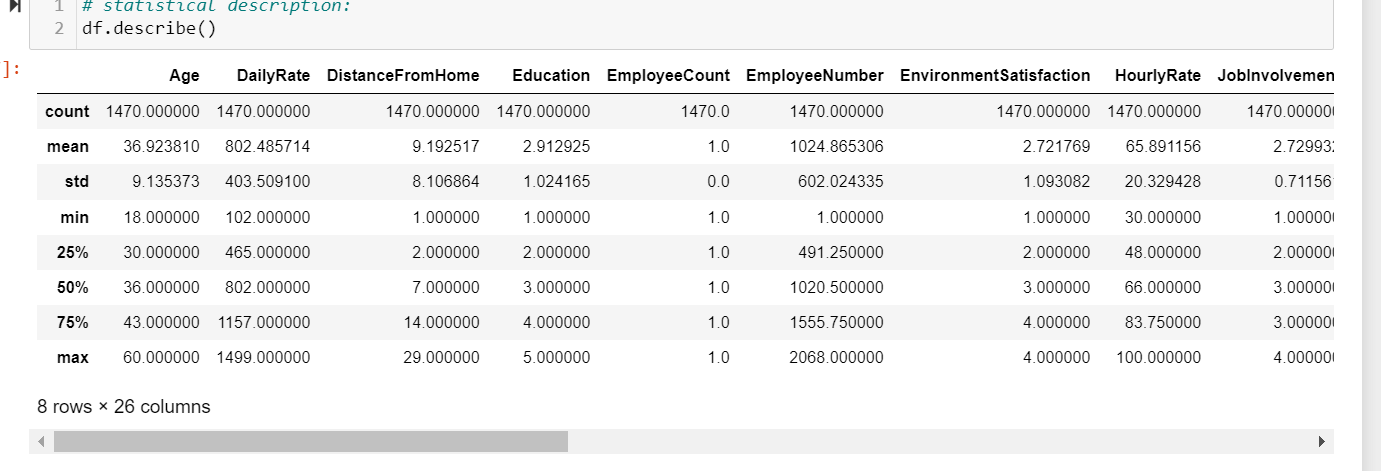
Observations:

Our dataset can roughly divide into following category:

1. Categorical/Nominal : Variables that can be divided into multiple categories but having no order or priority. Eg. Business Travel ('Travel\_Rarely', 'Travel\_Frequently', 'Non-travel').
2. Binary: A subtype of categorical features, where variable has only two values. Eg. Gender(M/F).
3. Continous: They can take up any value bewtween the minimum and maximum values in a column. Eg. Age, Monthly income, Total working years etc.
4. Useless: They don't contribute to the final outcome of a ML model. Here,as of now we can see that 'EmployeeCount' and 'StandardHours' is having same value through out the Column i.e '1' and '80' respectively.

# **Statistical description of dataset.**

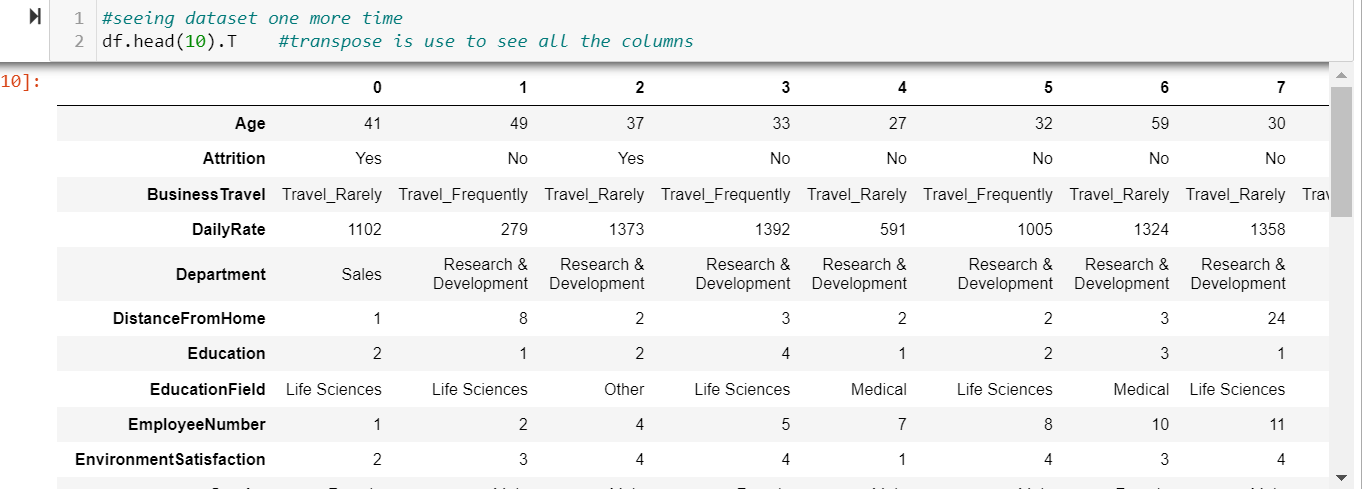
It is used to summarize and describe the features of data in a meaningful way to extract insights. We can perform Statistical analysis by using .describe() function of pandas. The describe() function computes a summary of statistics pertaining to the DataFrame columns. This function gives the **mean, std and IQR values**. And, function excludes the character columns and given summary about numeric columns.



**Observations**: From above also we can see that "EmployeeCount" and "StandardHours" possess same value throughout the column, which means if we drop them, it does not impact the output much. Let's Drop "EmployeeCount" and "StandardHours" Before proceeding further.

While analysing entire dataset one more time we come to know that Over18 column contain only one value in the entire dataset, so it is not of much use for Model building purpose, similarly EmployeeNumber is merely an ID given to employee, so we can drop that as well.

Since the number of rows are so much so, we use .T (Transpose) to see entire dataset.



# Dropping Unwanted columns:

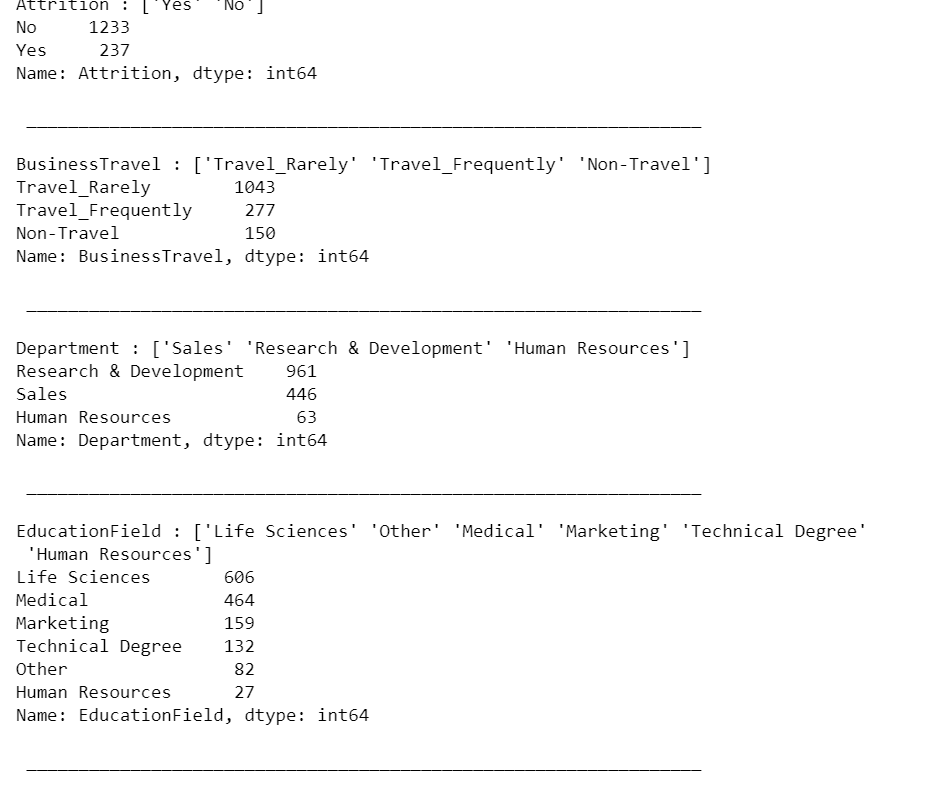


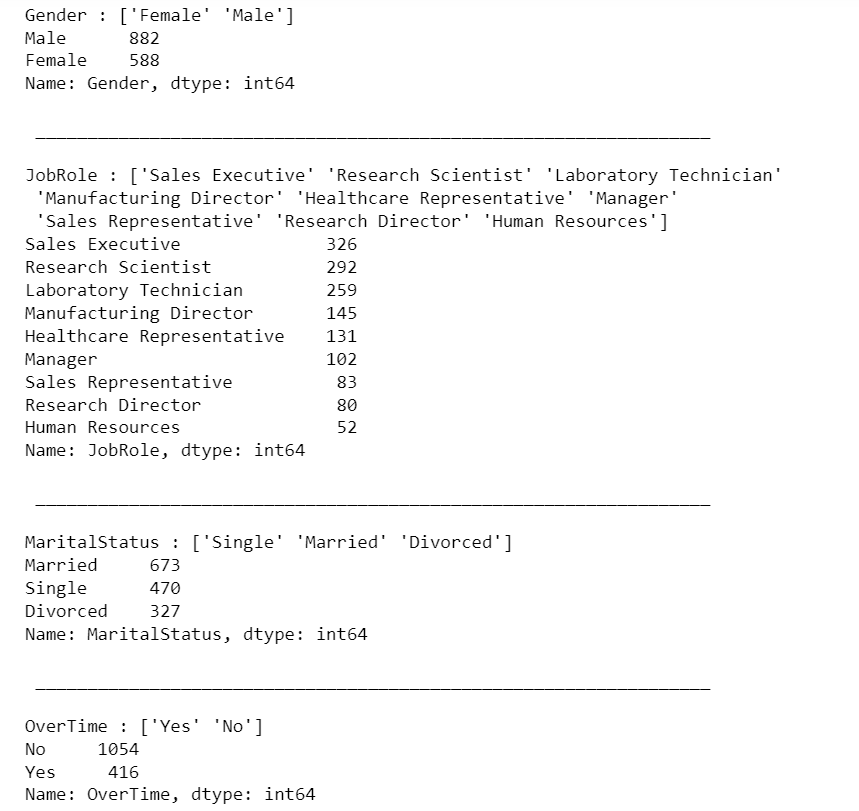


## Finding categorical features in the dataset and do cardinality check.

Categorical features are those which has Object datatype. And cardinality means the number of unique values in each categorical feature.







No higher cardinality is present in our dataset. Also We don’t have any null values in the dataset, so handling missing values is out from our to-do steps.

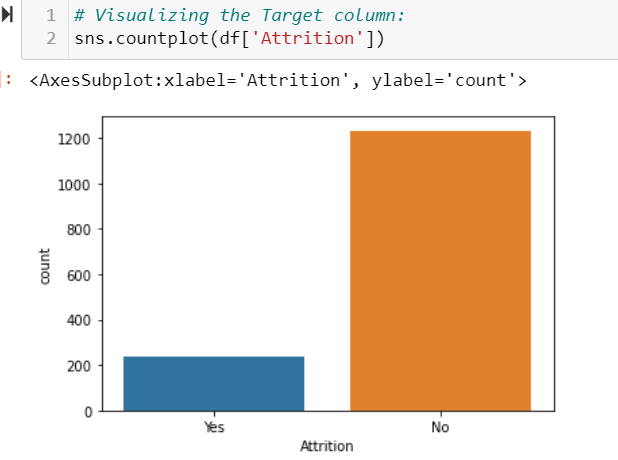
Let’s move ahead and perform some visualization. It comes under EDA.

## EDA (Exploratory Data Analysis)

Exploratory Data Analysis(EDA) is a technique used to analyze, visualize, investigate, interpret, discover and summarize data. It helps Data Scientists to extract trends, patterns, and relationships in data. We generally perform Univariate analysis and Multi variate analysis.

Univariate analysis:

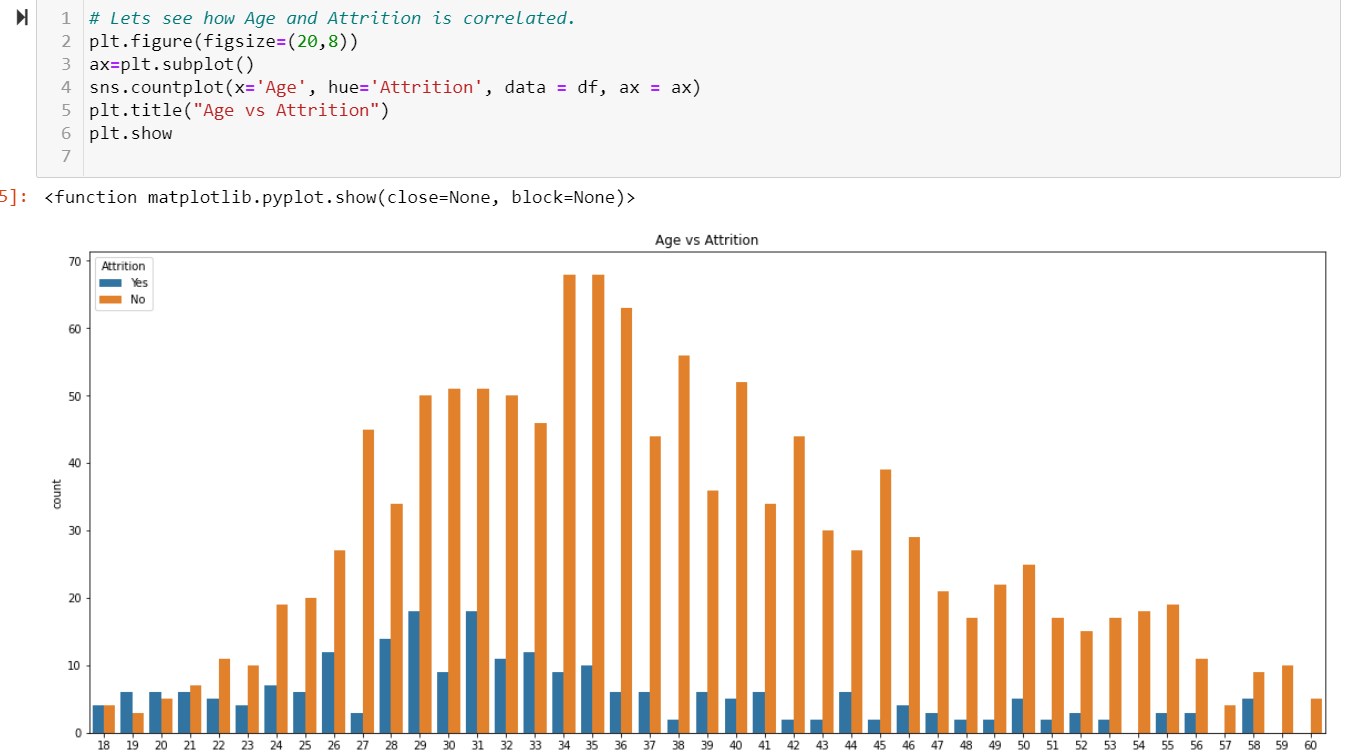
# Exploring target variable-



Looks like the target data is imbalanced , it has more number of No then Yes. If the data is imbalanced then it might affect the performance of the model. We might need to balance it before model building.

# **Bivariate analysis**

1. **Age - Attrition**

****

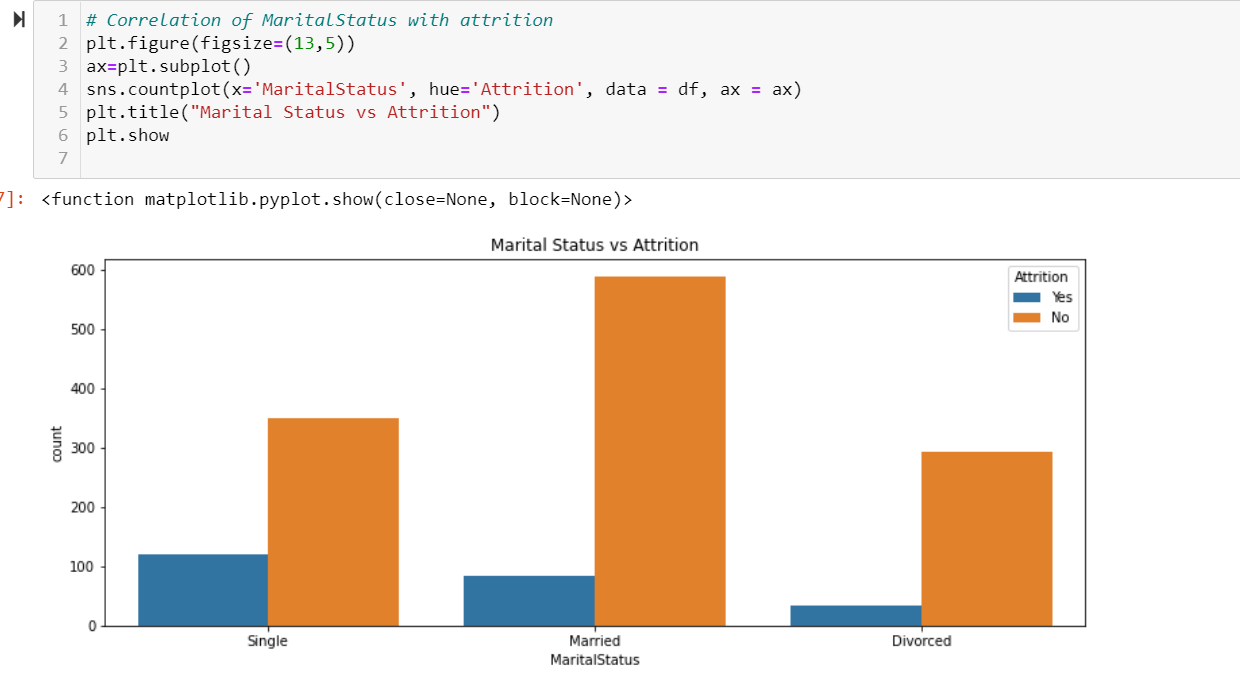
We can observe that the Age where is highest attrition is 29 and 31. While the age with high retention (i.e. Company retain the employee) is 34-35. We can also observe that people as they reached near their retirement tent to retain with the company as we can see 57-59-60 has zero attrition values.

1. Gender v/s attrition



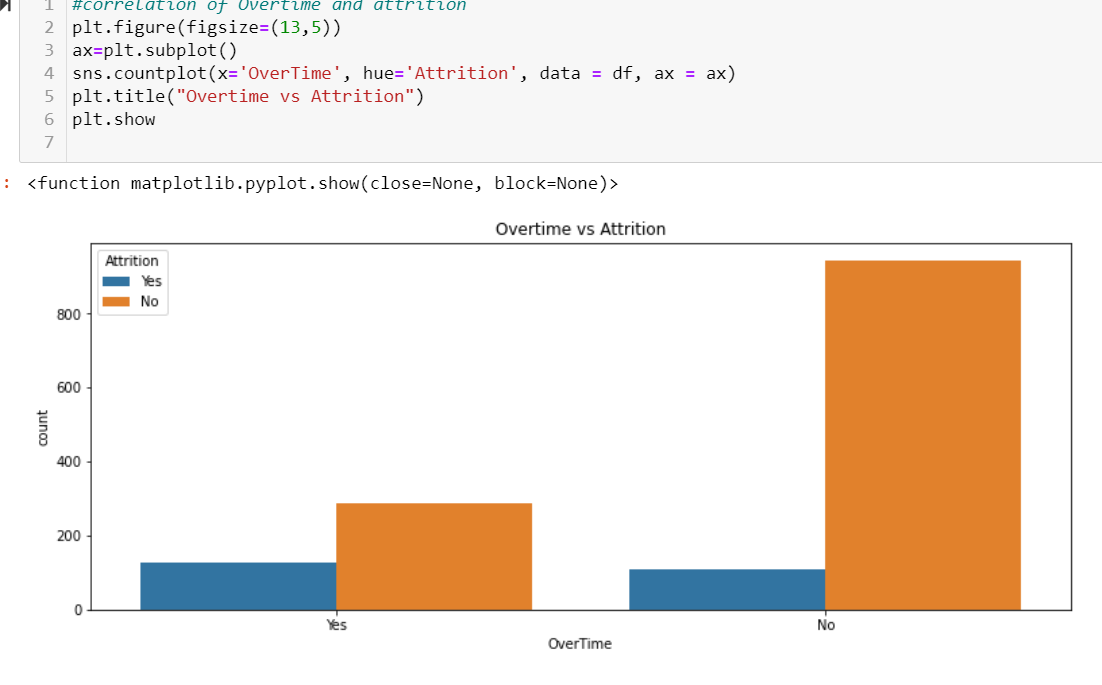
We can observe that the ratio is almost same for attrition, I don't think Gender contribute much in the attrition rate. Before concluding, We will cross check it.

1. Marital Status- attrition



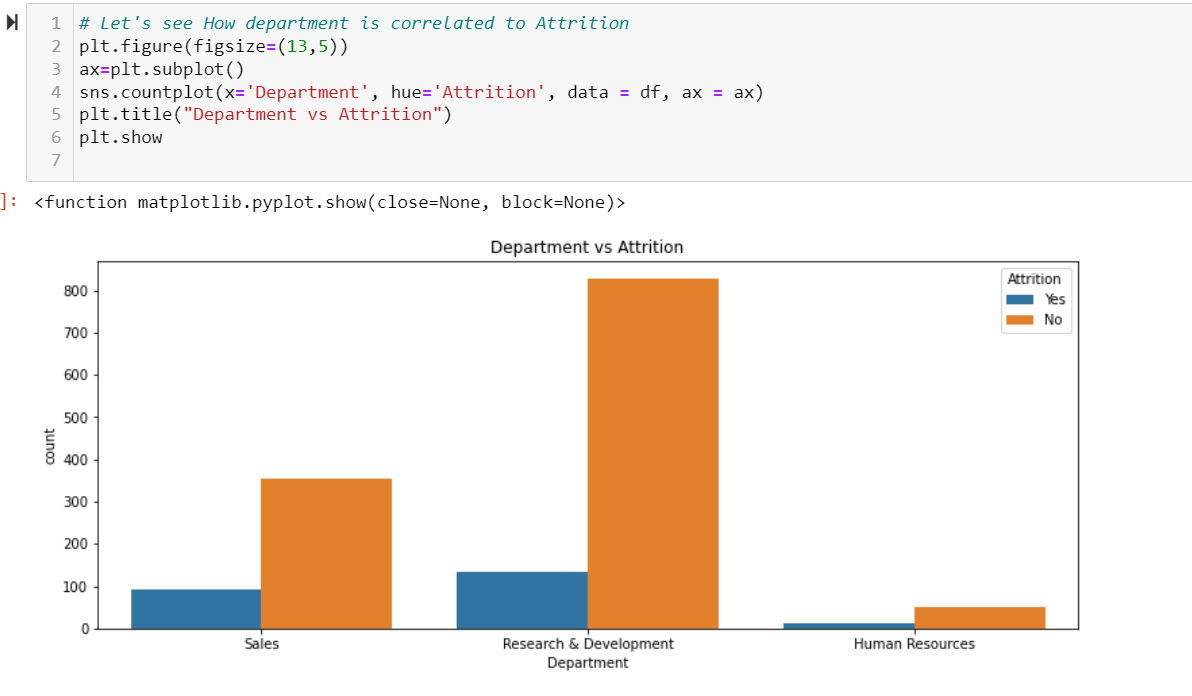
It's interesting to observe that Single person tend to left the company quite easily. So we can see a high attrition rate there.

1. OverTime- Attrition



It is interesting to observe that people retain with company where they don't need to do Overtime.

1. Department – Attrition

We can observe that the attrition and retention rate are high in Research & Development Dept, which is kind of obvious. while Attrition and retention rate is very low in Human Resources.

1. Jobrole – Attrition

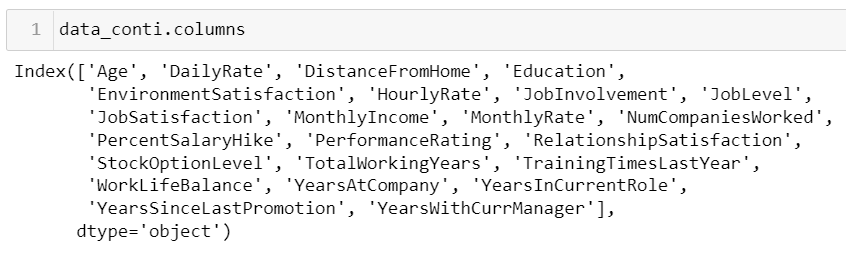


We can see that the attrition rate for higher positions like director and above is very low, employee and higher positions tents to stay with the company. Also, company tries to retain people with such experience and knowledge. Besides that we also observe that sales executive has both high attrition and retention rate, we know that these are the people bring business to the company so retaining them is really important, and they left as well when they see better opportunity somewhere else.

# Separating continuous columns and categorical columns in the dataset:

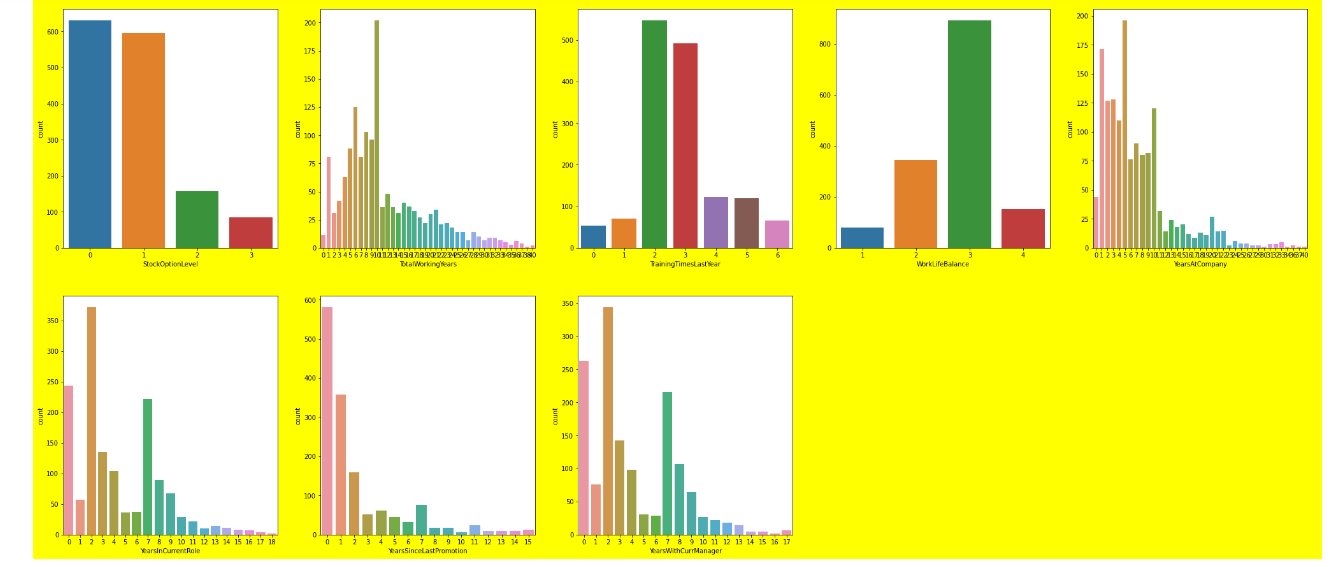
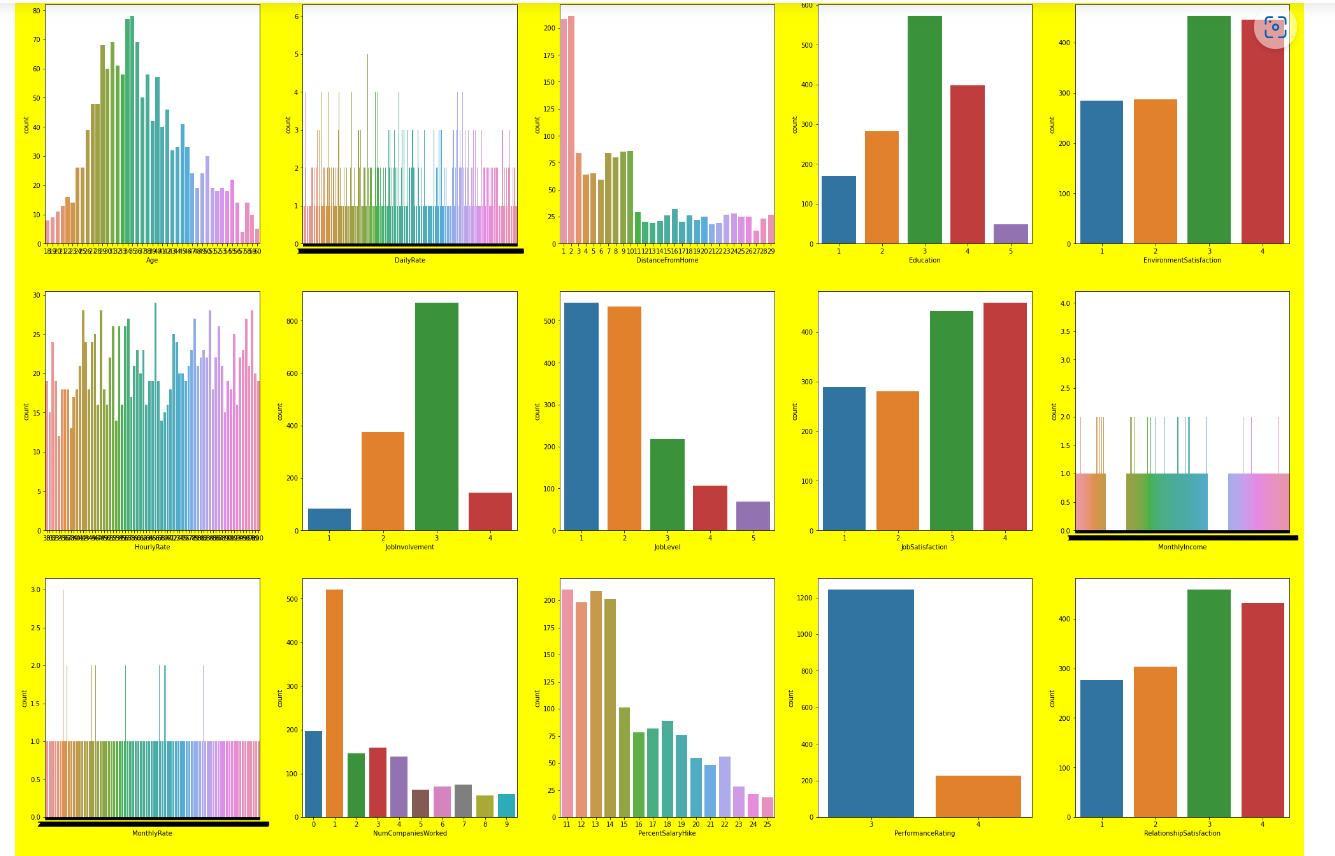


Now, as we get our continuous columns, let’s Visualise them as well.



For Visualization of continuous column Distplot is really helpful, we will be able to visualise proper distribution of data. We will call all the columns in a loop, able to see the entire continuous features at once.



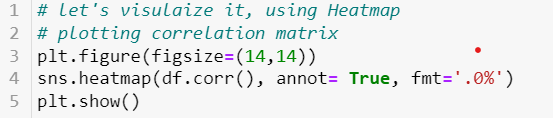


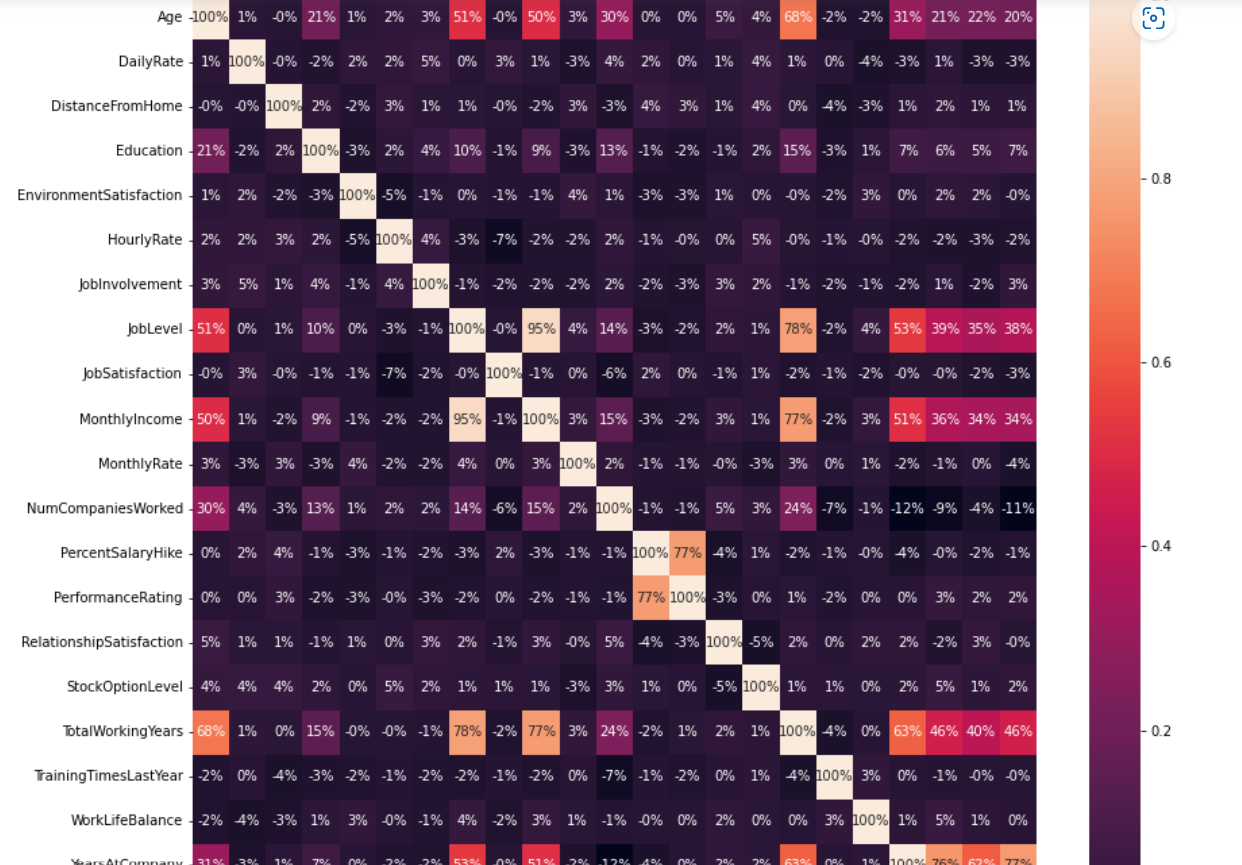
1. We can observe that here data is more or less discrete in nature.
2. Performance rating has only two values, 3 & 4. We can say that it may be less correlated data.
3. And also I think Education also don't place any vitol role in attrition. Let's do some more analysis to see how data is related with our target variable.
4. We can clearly see some skewness in the data, which we will treat further in the process.

# Let’s Check the correlation among these continuous features: We use .corr() to identify the correlation between features



Let’s visualise it using heatmap.

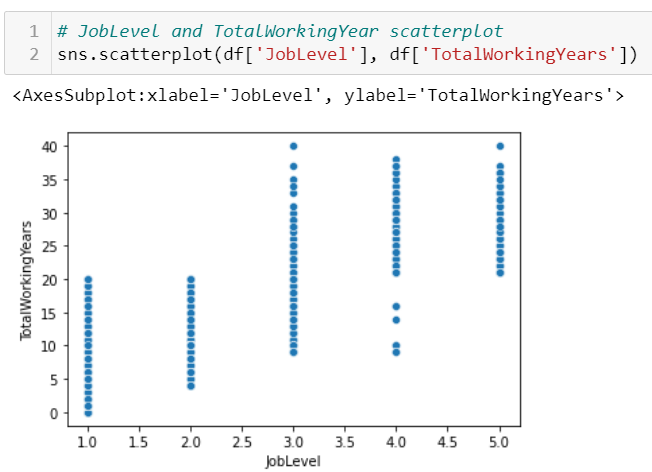
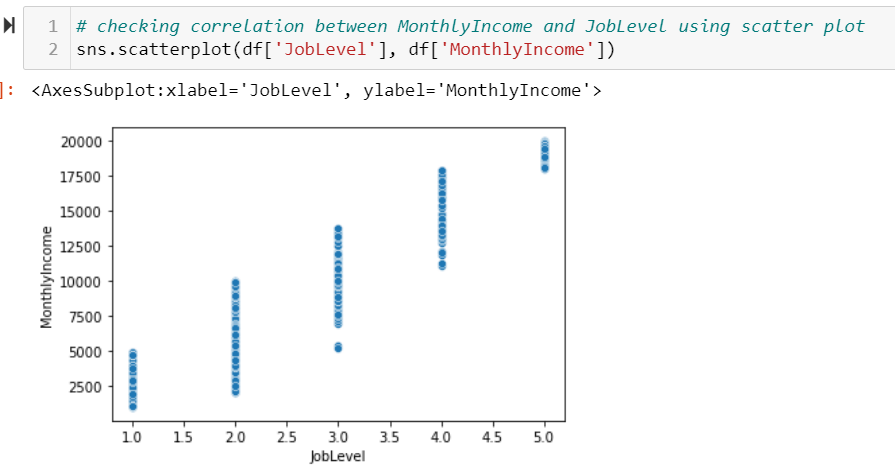




Seeing this it seems that certain features are correlated, Lets check for Multicollinearity. Also This data contain so many features, We will select top 80% of the features using selectK best.

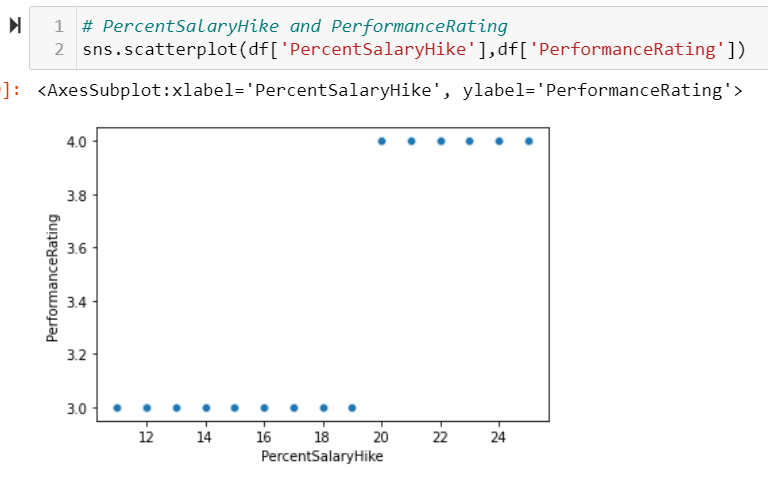
We can observe that there is remarkable correlation exist between MonthlyIncome and JobLevel let's confirm that using scatter plot and if the correlation is confirmed we can drop one of the column depending upon their correlation with attribution.

# Checking for correlation between features column.



From here we can observe that, our JobLevel and MonthlyIncome are highly correlated as One increases the value of other also increases, we can remove one of them, as it causes multicollinearity.

This shows some correlation but it’s not definite, as the value of one increases the value for other increases after certain point, it definitely has pattern but they don't seem much correlated to me at this point, We will check afterwards if the multicollinearity exist or not. For now, I can wait till further steps to make any conclusion.

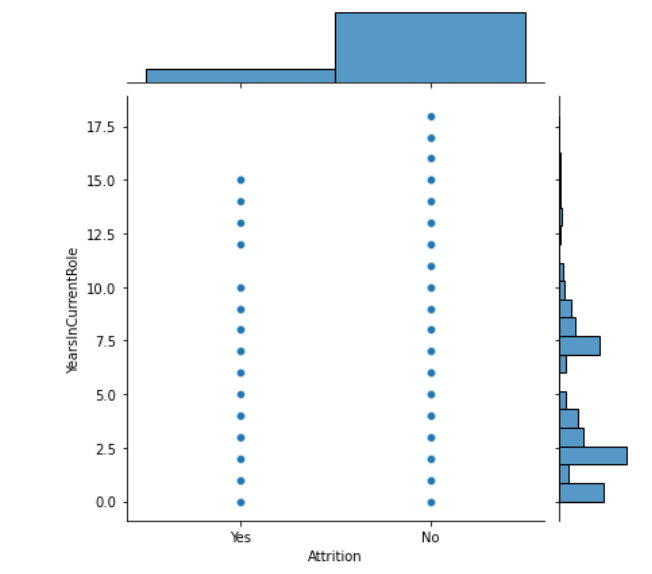
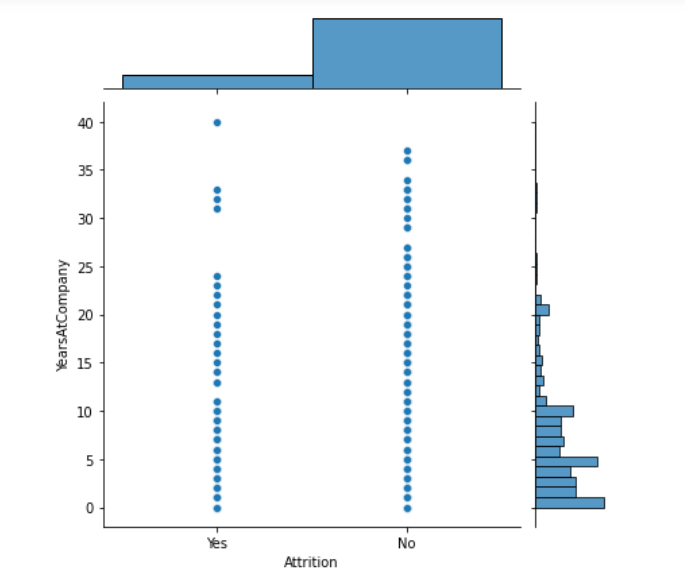


We can clearly observe that, if the value of one increases, the value for other also increases. Hence, we can say that they are correlated, And we can drop one of the column. For dropping the column we need to check how they are correlated with our label i.e. 'Attrition'.

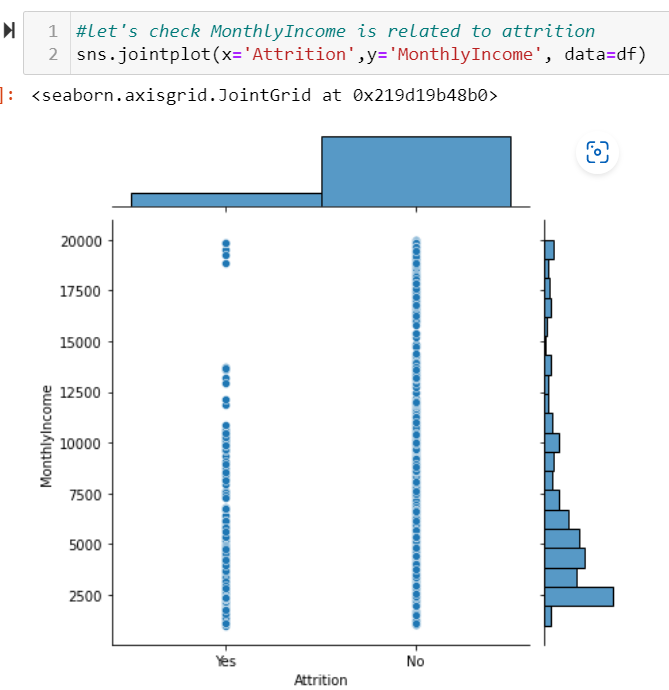
# **Multivariate Analysis.**

* **‘**YearsAtCompany’, ‘YearsWithCurrManager’**,**’YearInCurrentRole’ with Attrition

****

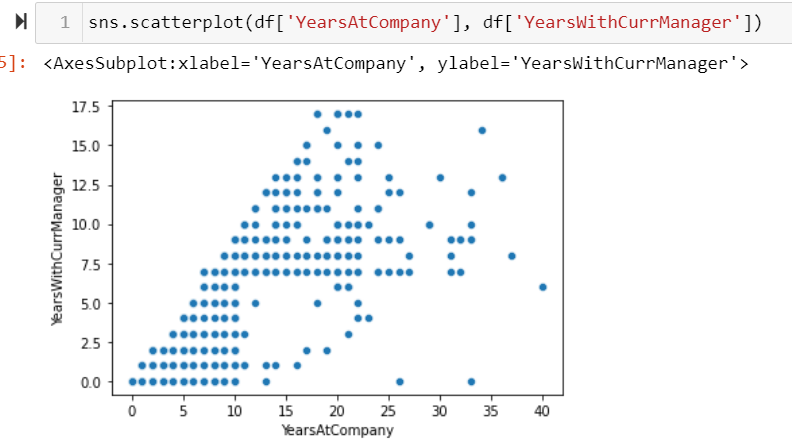


* MonthlyIncome -Attrition



We can say that employee with higher income will stay with the company while employee with monthly income under 10000 tends to left easily.

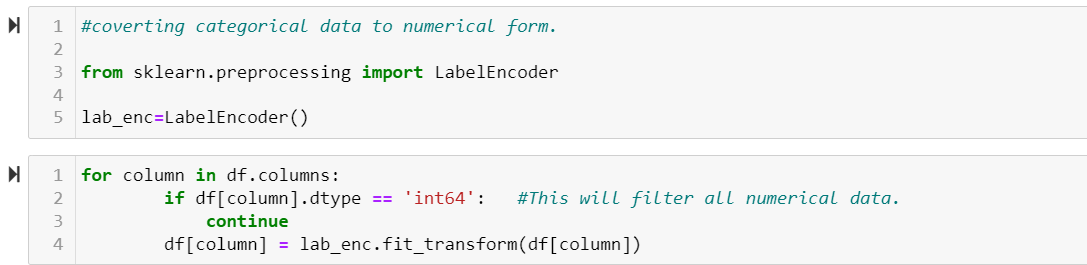
* **YearsAtCompany – YearsWithCurrManager**

****

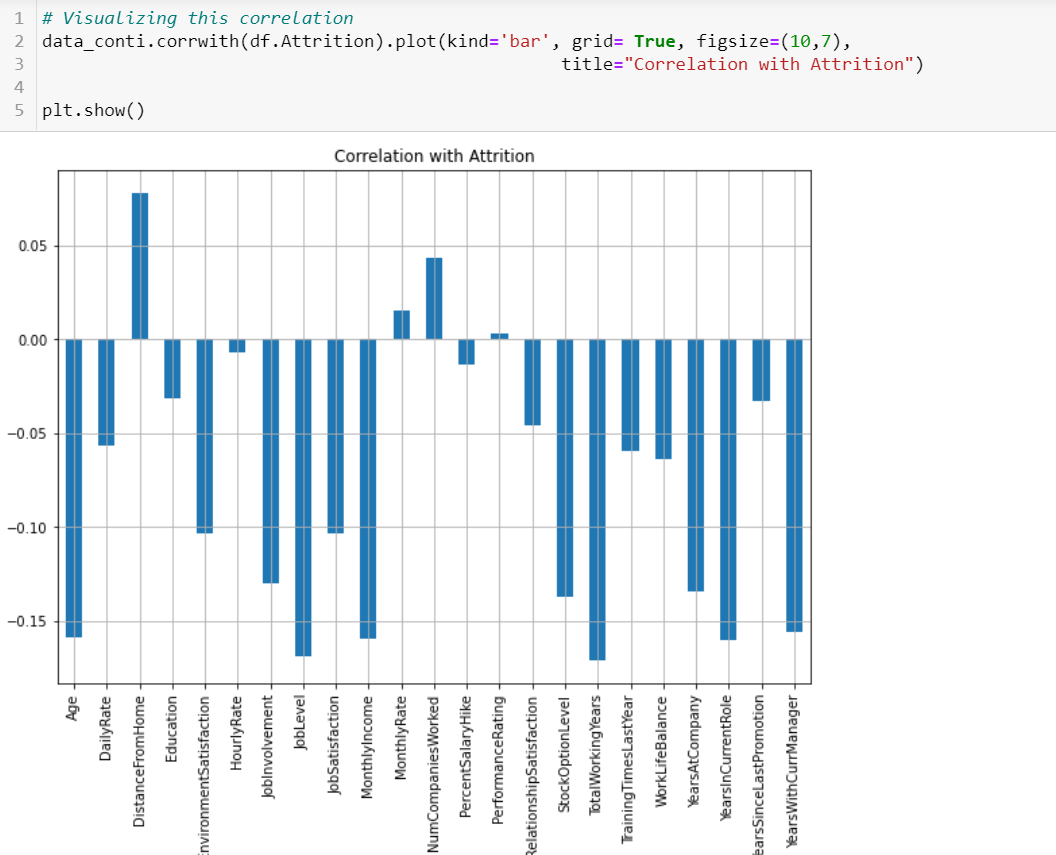
We can observe that there definitely exist some correlation, but we can drop one of them after visualizing there correlation with our Label(Attrition). The one with lowest correlation will be dropped.

## Encoding Categorical Features:

Most Machine Learning Algorithms like Logistic Regression, Support Vector Machines, K Nearest Neighbours, etc. can’t handle categorical data. Hence, these categorical data need to converted to numerical data for modeling, which is called  **Feature Encoding.** There are many feature encoding techniques like One code encoding, label encoding, Ordinal Encoder, replace(), mapping , etc. But in this particular blog, I will be using **Label Encoder** to encode categorical data to numerical data as all of our categorical features are Nominal data, so it will be the best fit. As, it won’t increase number of columns in the dataset, also reduce the chances of Multicollinearity which is drawback of One Hot Encoder.



Since our dataset is feature encoded, we don’t have any nulls. Now we can check the correlation of Continuous feature with Label for that we will use corrwith().

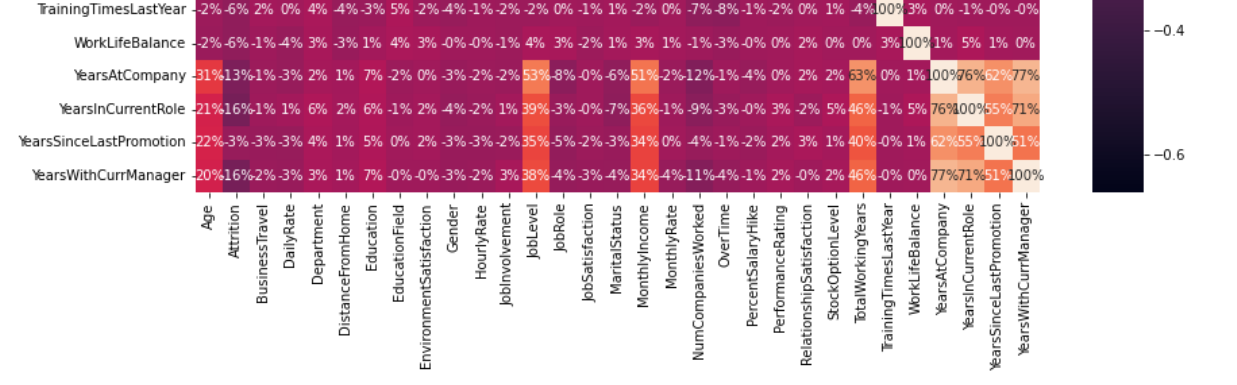
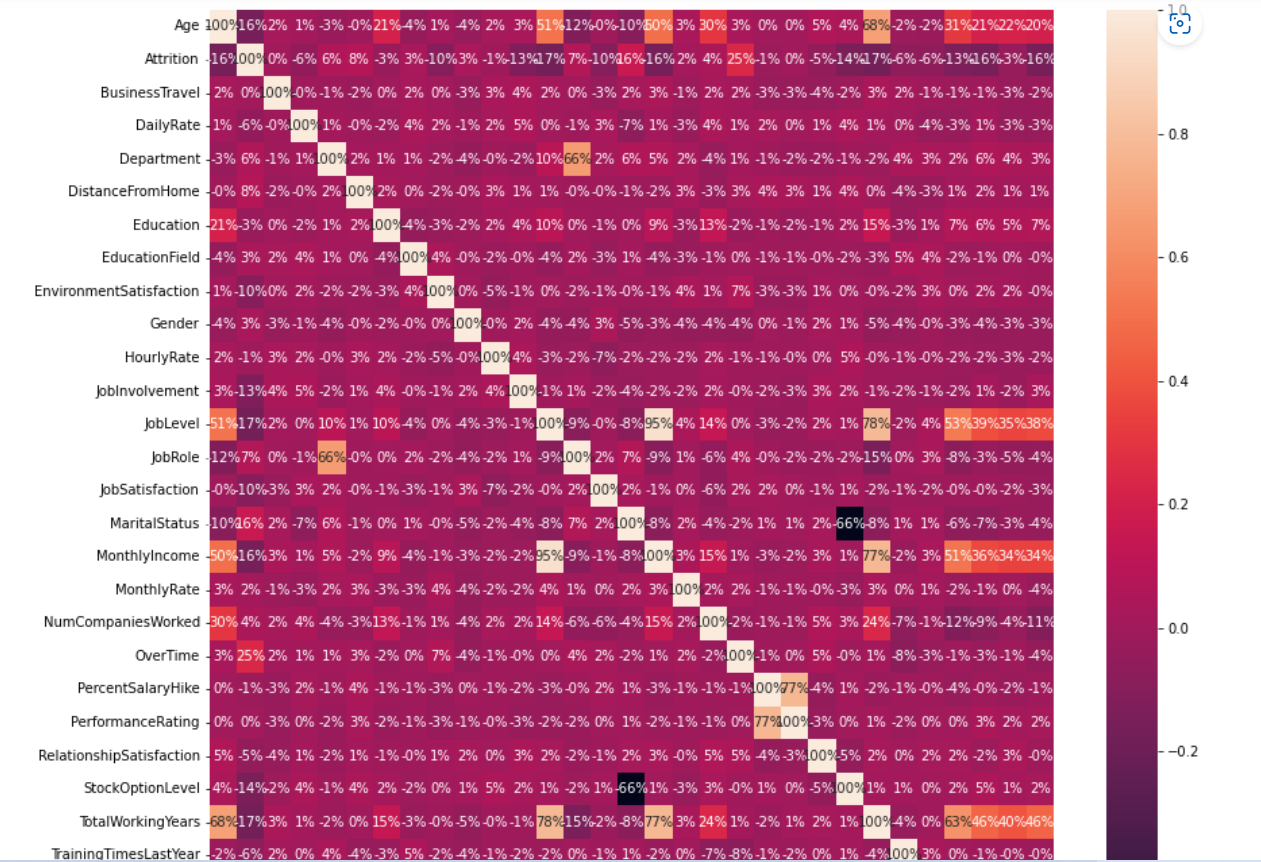


**Observation:**

* Data can be positively or negatively correlated. There correlation degree matters.
* We can observe that 'TotalWorkingYears', 'JobLevel','MonthlyIncome', 'Age','YearsInCurrentRole' these are the features which are highly correlated with attrition, while 'Performance Rating' and 'HourlyRate' shows least correlation with our target label i.e. attrition.
* From here we can decide which column to drop, as we have seen correlation between independent variables like MonthlyIncome and JobLevel, PerformanceRating and PercentSalaryHike.
* We can clearly drop PerformanceRating. And to decide among MonthlyIncome and JobLevel We can draw correaltion matrix, as the values are visible clearly there.

# Correlation Matrix:

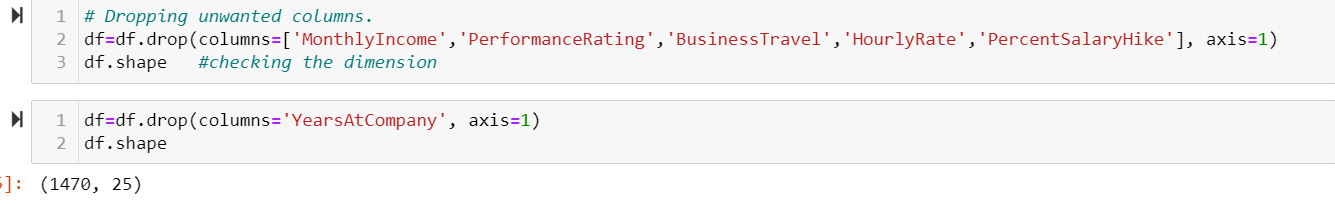
Correlation is a statistic that helps to measure the strength of the relationship between each column in the dataset. It is used in bivariate or linear analysis. Here we calculate correlation using corr() function in pandas.



We will keep JobLevel and drop MonthlyIncome, as Joblevel show strong relation compare to MonthlyIncome. And for PerformanceRating we can drop it as it shows 0 correlation with label. From here We can also observe that BusinessTravel also show 0% correlation with attrition, so we can drop that as well. We also See that HourlyRate and PercentSalaryHike shows only 1% correlation with Attrition. In my opinion we can drop that as well.

We can drop YearsAtCompany as well as it is correlated with YearsWithCurrManager.

# **Dropping Unwanted Columns:**

****

Dropping these columns will help us to reduce Multicollinearity, and hence reduce the noise. Which ultimately helps us to built a better machine learning model.

## Skewness and Outliers Detection and Removal.

Skewness is **asymmetry in a statistical distribution, in which the curve appears distorted or skewed either to the left or to the right**. Skewness can be quantified to define the extent to which a distribution differs from a normal distribution.

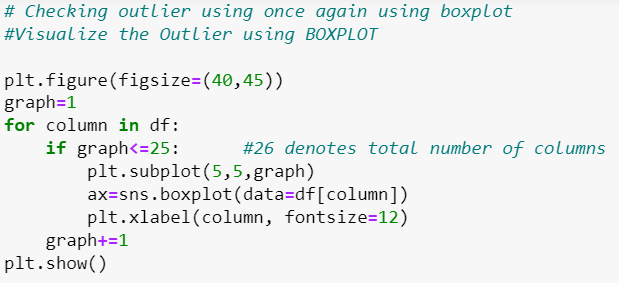
Both Skewness and Outlier detection and removal cannot be performed on categorical features and labels. So here we will check skewness on continuous features only. For that we will use .skew()

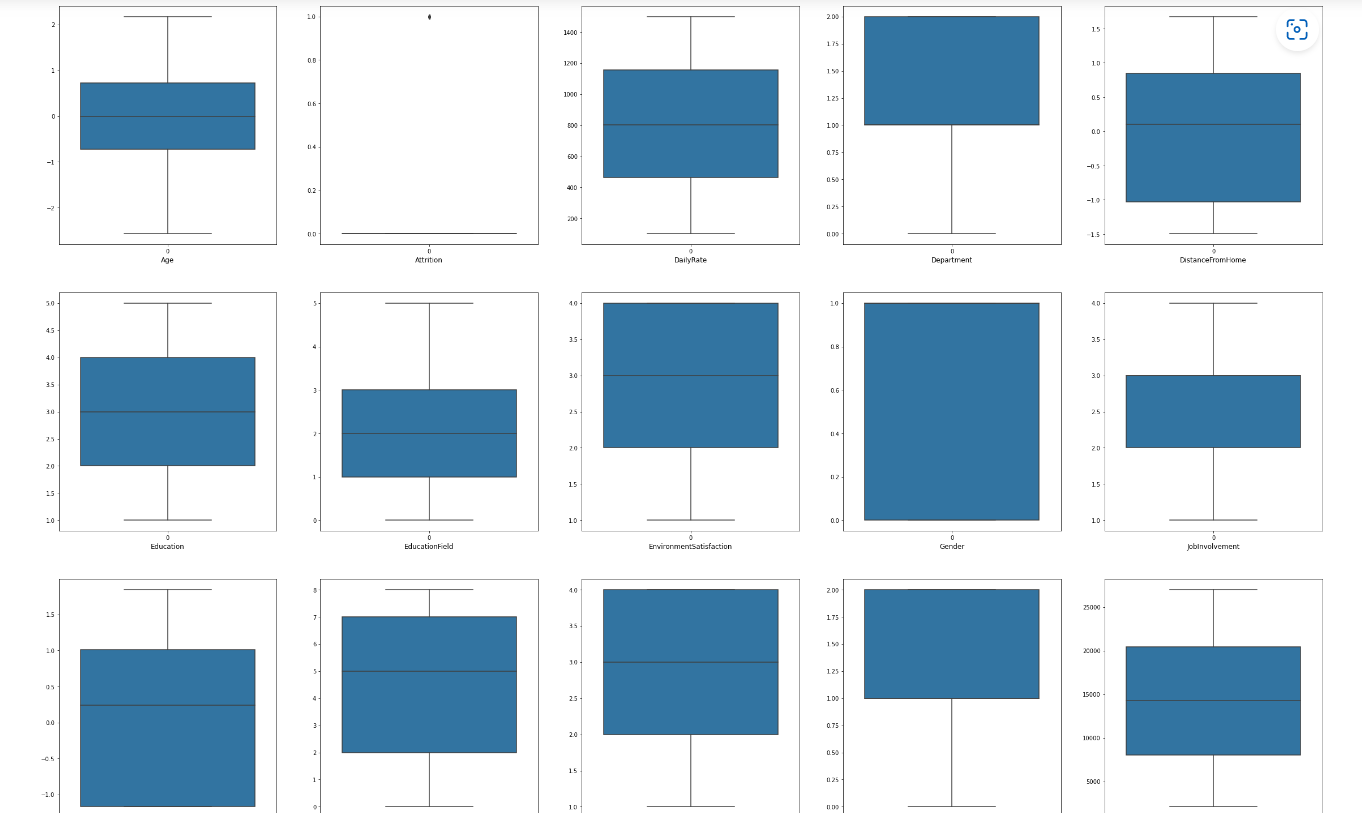
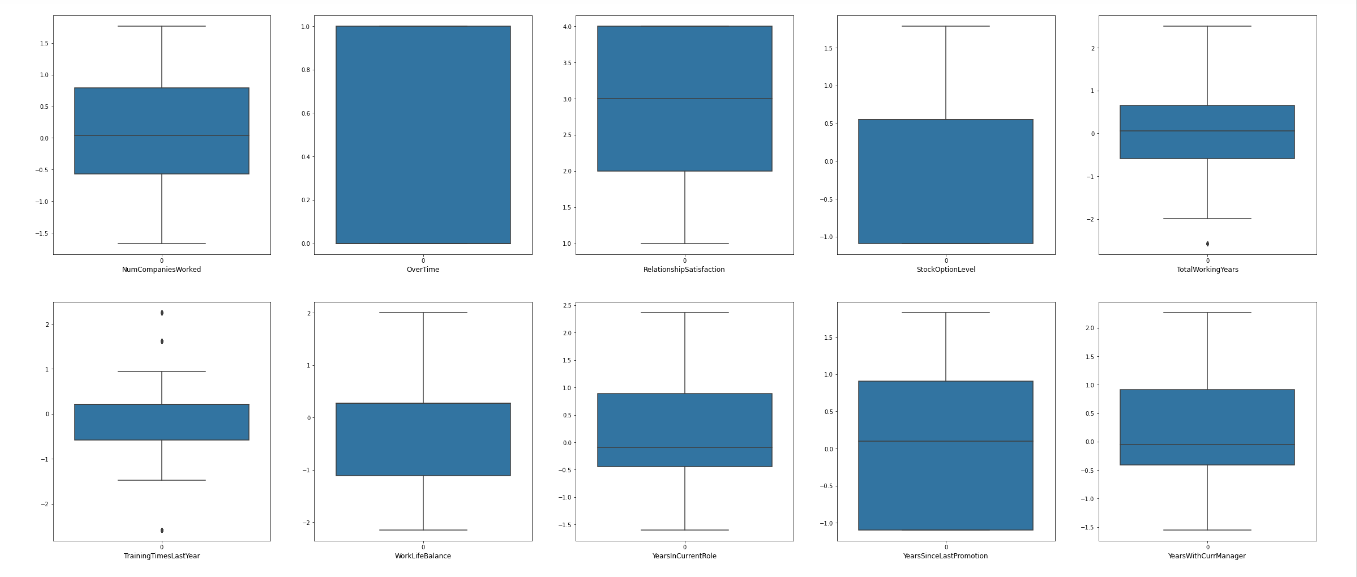


We can observe that some of our data is highly skewed even we take the threshold of +/-0.5, So let's remove the skewness using PowerTransformer.

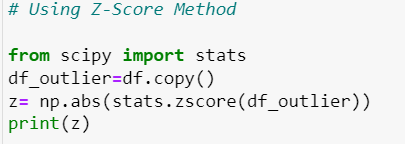


As skewness has been removed, now let’s visualise the data for outliers. We will plot boxplot for outlier visualization. An Outlier is basically **a data point in a data set that is distant from all other observations**. A data point that lies outside the overall distribution of the dataset.





We don’t see any outlier in our dataset, nut just to be sure we can perform z-score for outlier detection and removal.



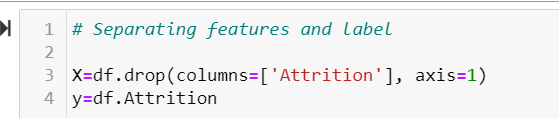


It is clear that we don’t have any outlier in the dataset, everything looks good so far, Now, let’s move ahead with feature Scaling.

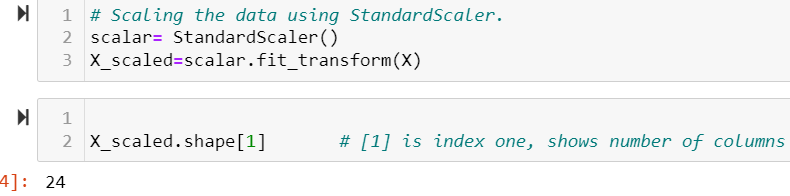
## Scaling

Feature Scaling or Standardization: It is **a step of Data Pre Processing that is applied to independent variables or features of data**. It basically helps to normalize the data within a particular range. Sometimes, it also helps in speeding up the calculations in an algorithm.

As we can see that Scaling should be done on features only, so let’s go ahead and separate Label and features.



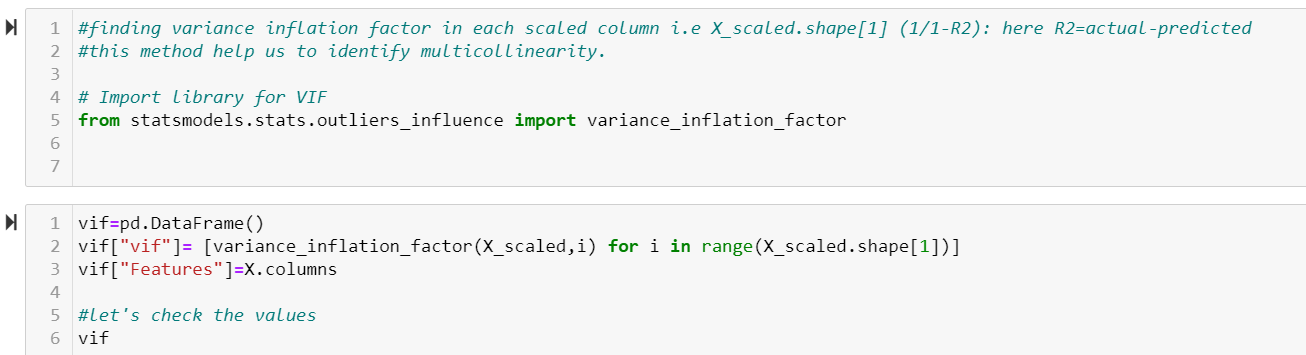
In the above code, X is our Features dataset, while y is our Label dataset.

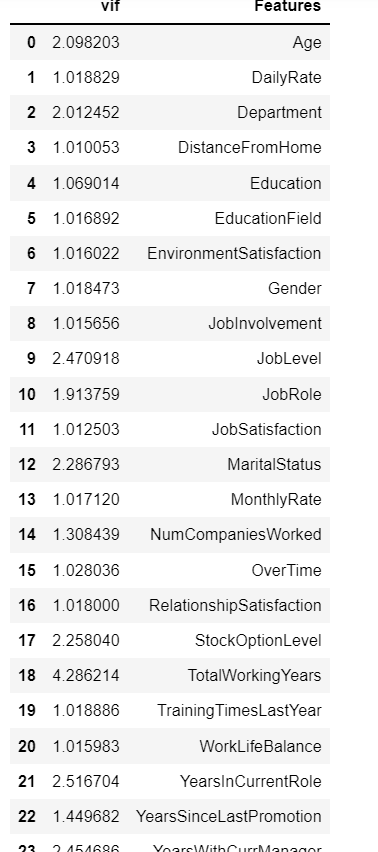
We will use standard scalar technique to scale our features.  


X\_scaled is our scaled features dataset. Which will be used further in the Model Building process.

## VIF (Variance Inflation Factor):

Variance inflation factor (VIF) is **a measure of the amount of multicollinearity in a set of multiple regression variables**. Mathematically, the VIF for a regression model variable is equal to the ratio of the overall model variance to the variance of a model that includes only that single independent variable.

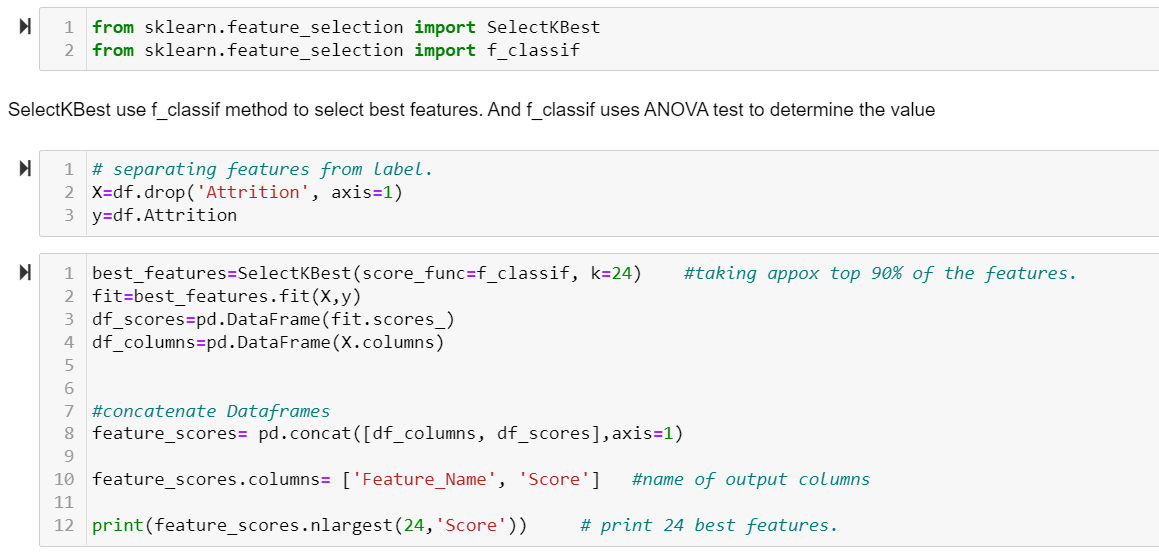




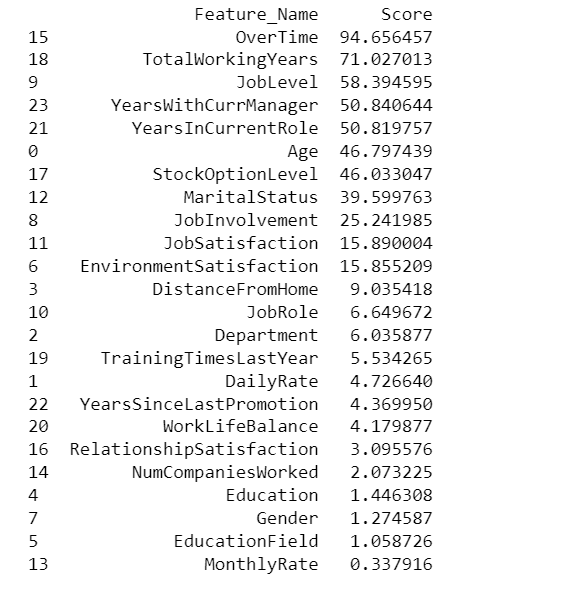
We can observe that for our dataset the VIF score is less than 5, which means these features are not correlated with each other, means no multicollinearity. Looks good, now we can go ahead with feature selection.

## Feature Selection Method (SelectKBest):

The SelectKBest method **selects the features according to the k highest score**. By changing the 'score\_func' parameter we can apply the method for both classification and regression data. Selecting best features is important process when we prepare a large dataset for training.

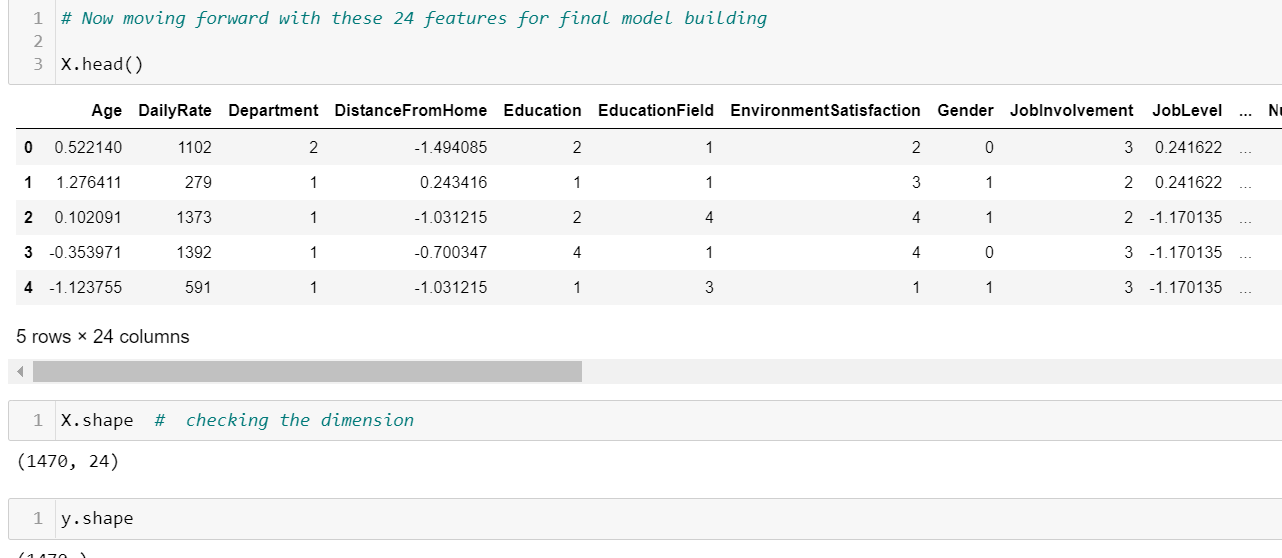


We are selecting top 24 features which shows highest correlation with our Label.



Above is the list of features which are highly correlated with our target variable, and going forward, we will built our model using these feature only.

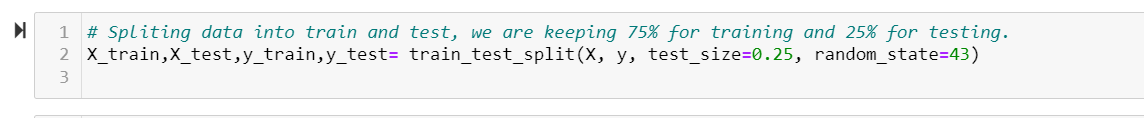
# Let’s check the dataset for feature and target column one last time before Model building



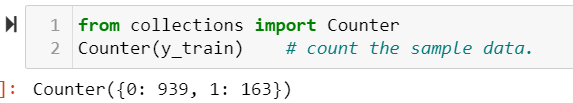
Everything lines up neatly, we can move ahead with splitting the dataset for test and train purpose.

## Splitting Data into Training and Testing set :

***train\_test\_split()*** is a method of model\_selection class used to split data into training and testing sets.



As we have observed before that our target column is imbalanced, let’s check how they are distributed in training set. For that we will use Counter ().



We can see that difference between label classes are huge, Our model may act biased or over fitted towards one class. so we need to balance this for dealing with this imbalanced data we will use over sampling method. But before applying that, how our data work with imbalance samples.

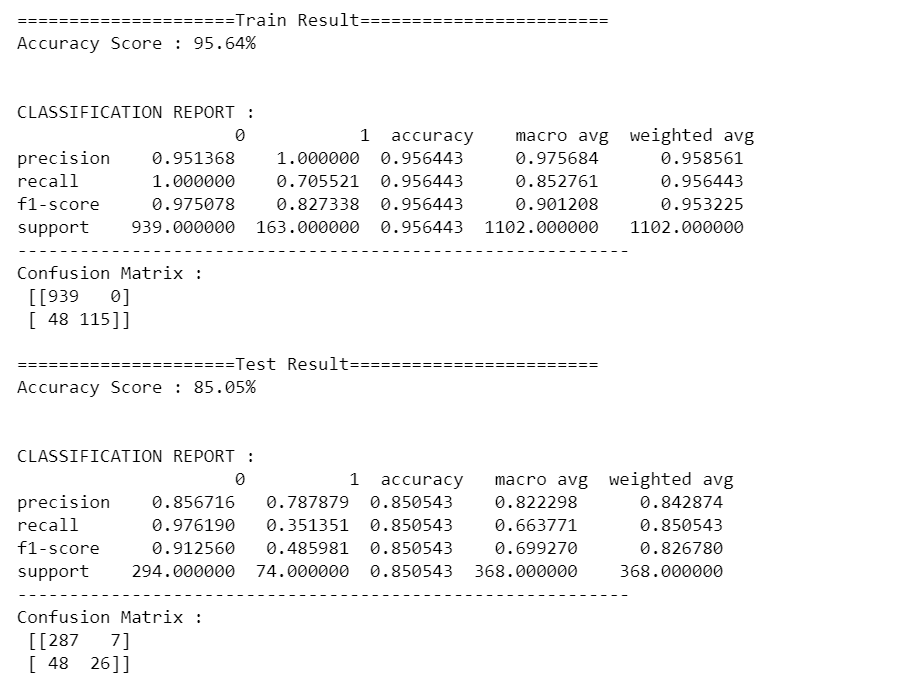
## Model Building

Let’s perform GradientBoost Classification algorithm in this imbalance dataset.

For any Model building, we did follow some common steps, like importing the necessary libraries, then calling the model, fitting the model, Predicting, determining scoring variables. Then finally print the final score.

After these basic steps we perform cross validation . And if it is our final model then we go ahead and do hyper parameter tuning, to improve the effectiveness of the model. Below code is an example of how Model Building is done for various algorithms.



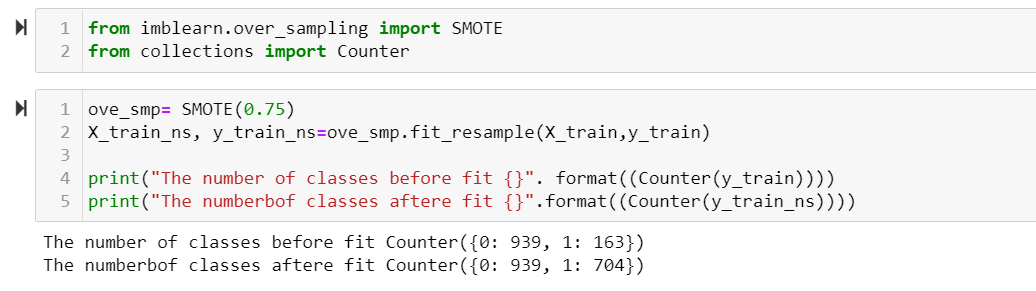


Although the scores looks good, as gradientBoost model is robust model, not get easily affected by imbalance dataset. However if we balance the dataset, the scores will further increases.

## Dealing with Data Imbalance:

We can see our model is giving pretty good prediction but that might be the case because there is huge difference between the label classes(Yes/No). So just to make sure our model is not overfitting towards higher number class or in other words show biasness in the prediction let's deal with this imbalance data. For that we will do over sampling by using SMOTE Technique as we don't want any loss of data.

# SMOTE Sampling Technique

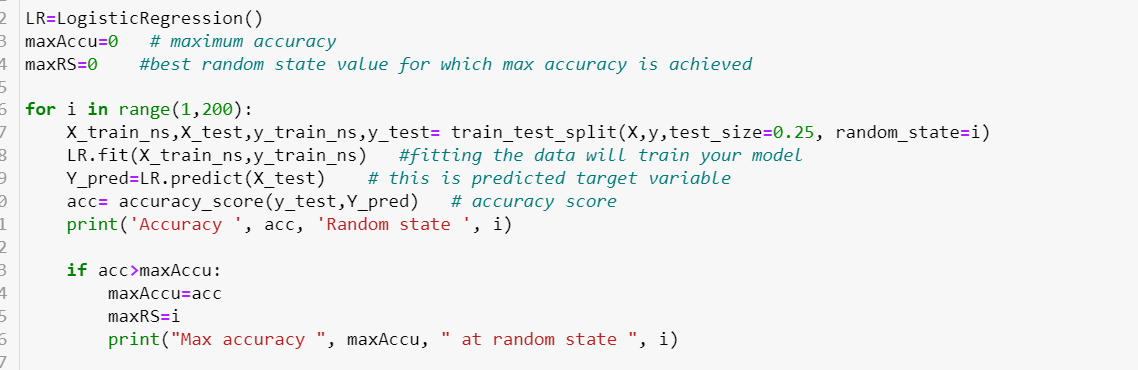


Now our data looks far better, over sampling is mostly recommend, as if we done under sampling there are chances of data lost. And we cannot afford it, so it’s a good option to do over sampling.

Now, let’s go ahead and Build Model with other algorithms.

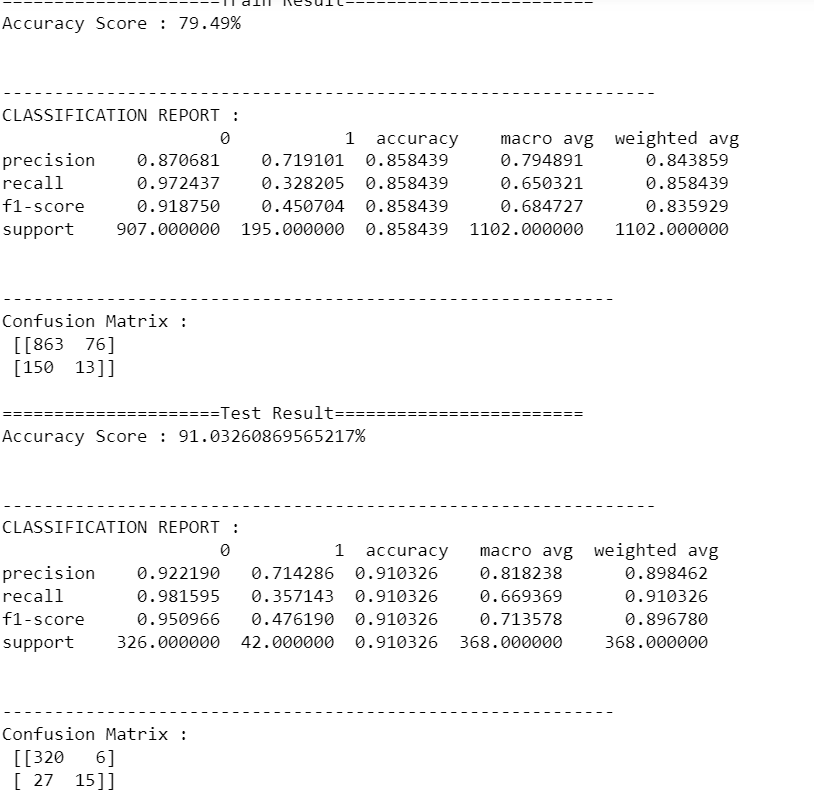
* **Logistic Regression:**

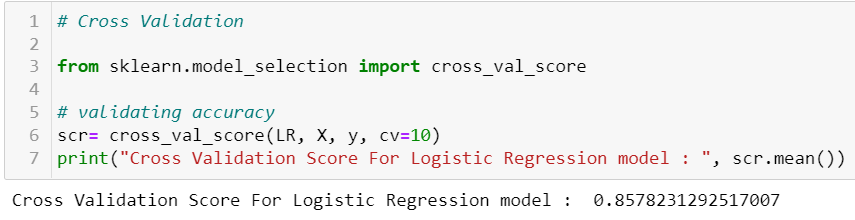
The main parameter of Logistic regression is random state, so let’s starts there to find best random state which gives highest accuracy.



Then with this random state we will build the model.

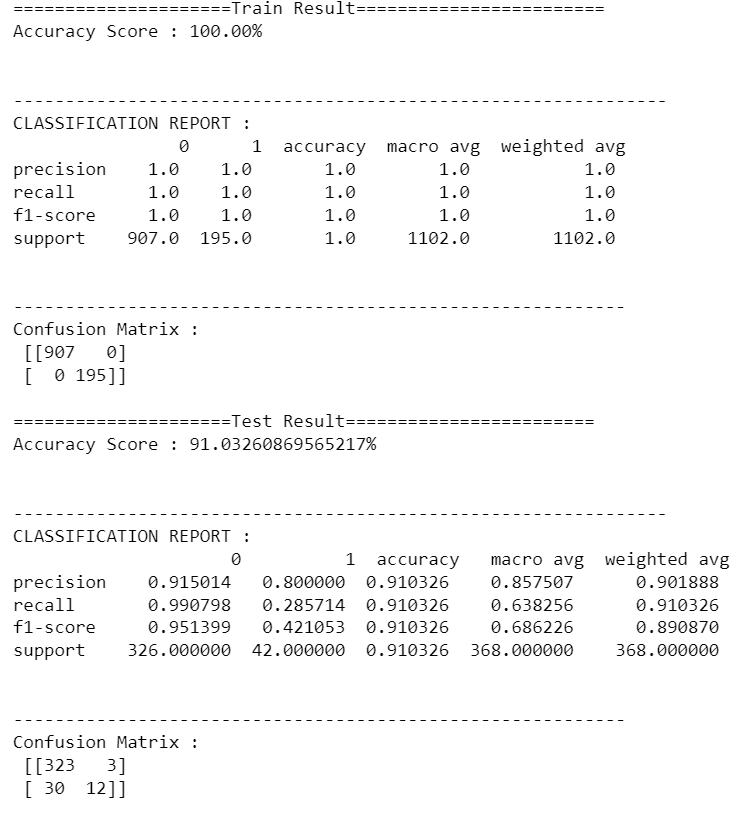






* **Ensemble Technique (RandomForest Classifier)**

**We** perform same steps here also and following is our score.



Cross validation Score of RandomForestClassifier model is : 0.8537414965986395

We can observe an improvement in score let’s perform Hyperparameter tuning.



We can observe that post tuning score is better than cross validation score, one thing to see here that the test score is 100% with means our model is overfitted for this dataset, and shows some biasness.

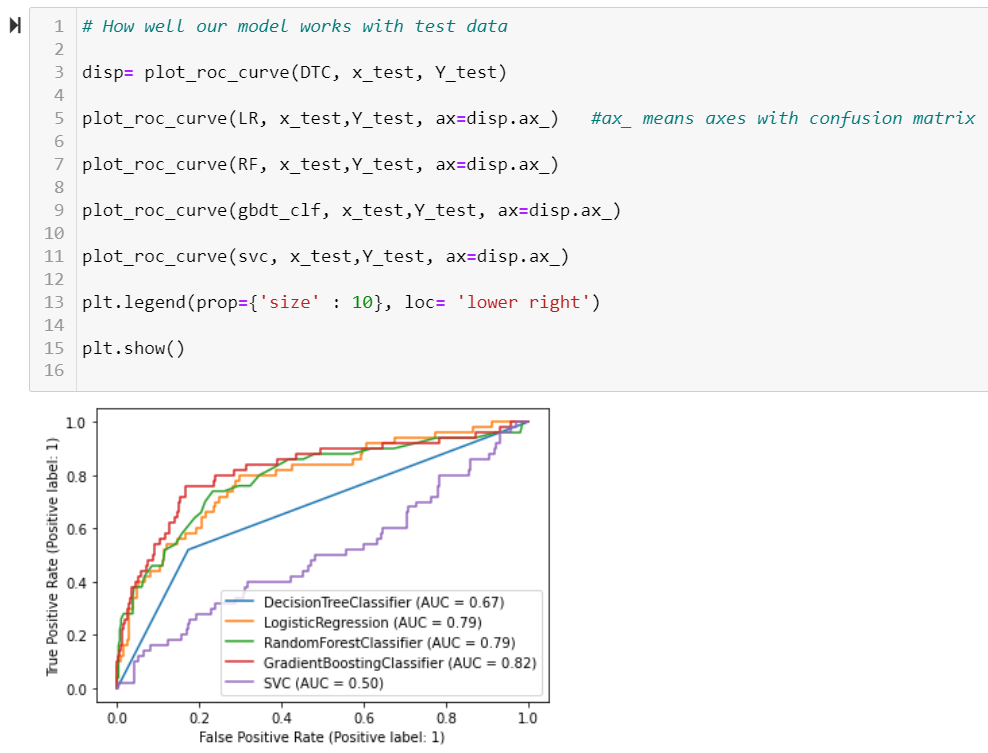
* We did perform same steps for column more classification algorithm like SVC, Decision Tree, Gradient boost again on sampled dataset.

The scores are more or less fall in the same line, so we plot roc auc curve in order to determine the best algorithm for our model.

## # Deciding Best Model Using ROC AUC Curve:

The Reciever operating characteristic curve **plots the true positive (TP) rate versus the false positive (FP) rate at different classification thresholds**.





## Results and Conclusion

* We can observe that Logistic Regression is better fit in our case, since its accuracy score for test and train model are almost on the same line. Although RandomForest and gradientboost shows higher result in train model but the difference between train and test model is huge comparitively. So we can pick Logistic Regression as our final model to save.
* The logistic Regression model accuracy score is 0.79. The model does a very good job of predicting.
* The model shows no sign of Underfitting or Overfitting. This means the model generalizing well for unseen data.
* We already perform cross validation , and determine that its accuracy can be increased by few percent only.

## Saving Model And Scaling Object using Pickle

**Pickle** is a python module used to serialize and deserialize objects. It is a standard way to store models in machine learning so that they can be used anytime for prediction by unpickling.

Here we use one of same model joblib to save the file.

