**HR Analytics Project- Understanding the Attrition in HR**

**Problem Statement:**

Every year a lot of companies hire a number of employees. The companies invest time and money in training those employees, not only that, 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 does HR Analytics fit in this and is it just about improving the performance of employees? Let us have a detailed look on that.

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 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.

The Problem here is, 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.

After understanding the problem description, let's jump into the approaching part of a project and understand the basic features of the data we are provided with.

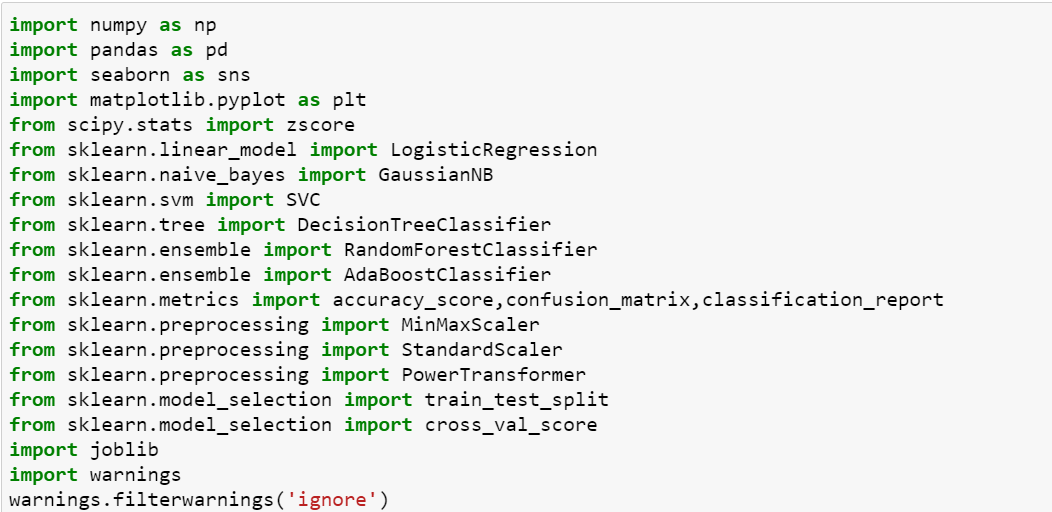
The dataset we have is small with 1470 rows and 35 unique columns (features).

**Feature\_columns =** ‘Age','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', 'TrainingTimesLastYear', 'WorkLifeBalance', 'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion', 'YearsWithCurrManager' **Label\_column/Target =**  Attrition

**Data Analysis:**

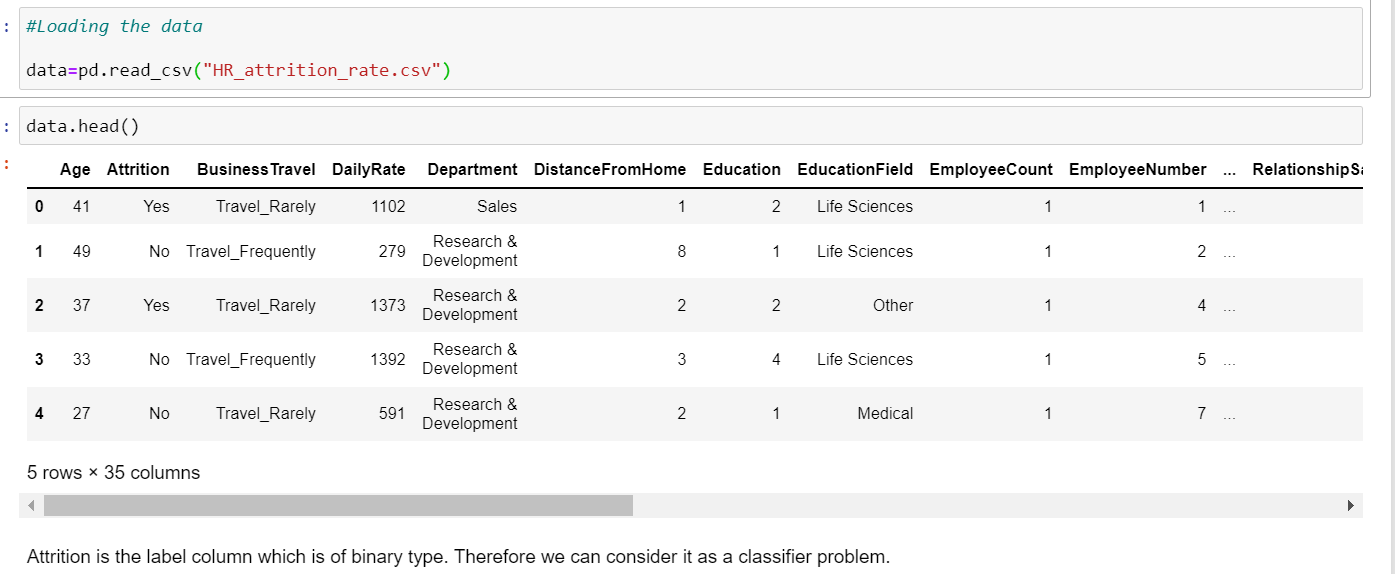
* **Import all required libraries**

We will be requiring NumPy and pandas for mathematical computation and data manipulation. And matplotlib with seaborn to visualize the data with interactive plots. Also, import preprocessing tools and models to deal with pre-processing and model building. We have imported libraries for categorical data as Target value is of binary type (Yes/No). Import warnings to ignore all the warnings.

****

* **Load Dataset**

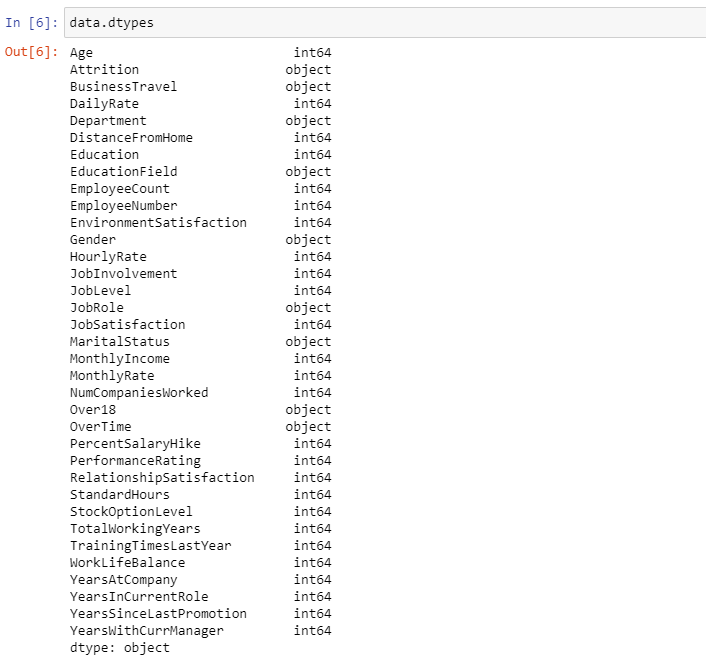
The dataset is in csv format so use the pandas read\_csv method to load the dataset. After loading just see its shape and head of the dataset to have a near view of the data we have.

****

* **Checking Datatypes**

We have int and object datatypes. Label column that is attrition is of object type which need to encoded using label encoder.

Business Travel, Department, Education Field, Gender, Job Role, Marital Status, Over18, Overtime are of object type. We have to check on this data and remove/encode as per need.



* **Dropping unwanted columns**

Checked for unwanted data. We found that 3 columns which are found to be unwanted as it contains same data for the entire dataset. We dropped Over18, Standard hours, Employee count.

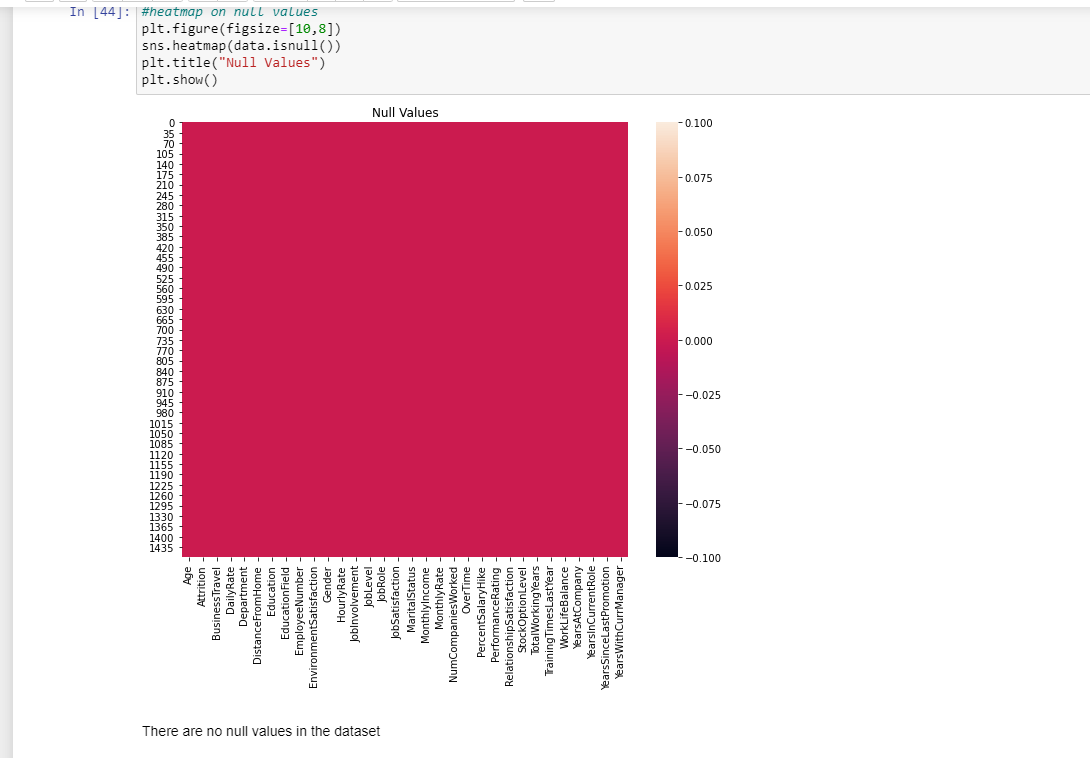


Now we have 1470 rows and 32 columns for Pre-Processing.

**Pre-Processing Pipeline**

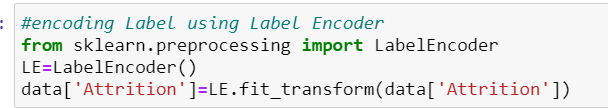
* **Missing Values:**

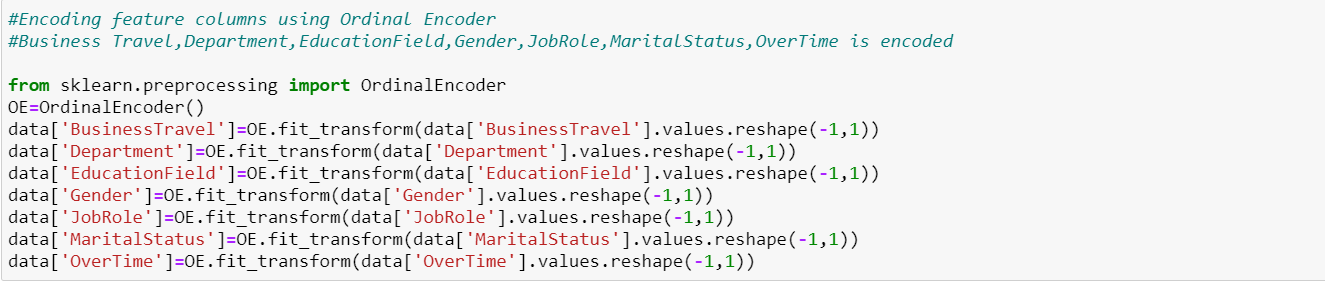
There are no Missing values/ Null values found in the dataset. The Presence of missing values can be found using data.isnull().



* **Encoding data**

Since we can see a lot of columns are of object data type. we need to encode it into numerical data using either label encoder for target columns or by Ordinal Encoder/one hot encoder for feature columns. We have identified columns to be encoded as shown in the figure.





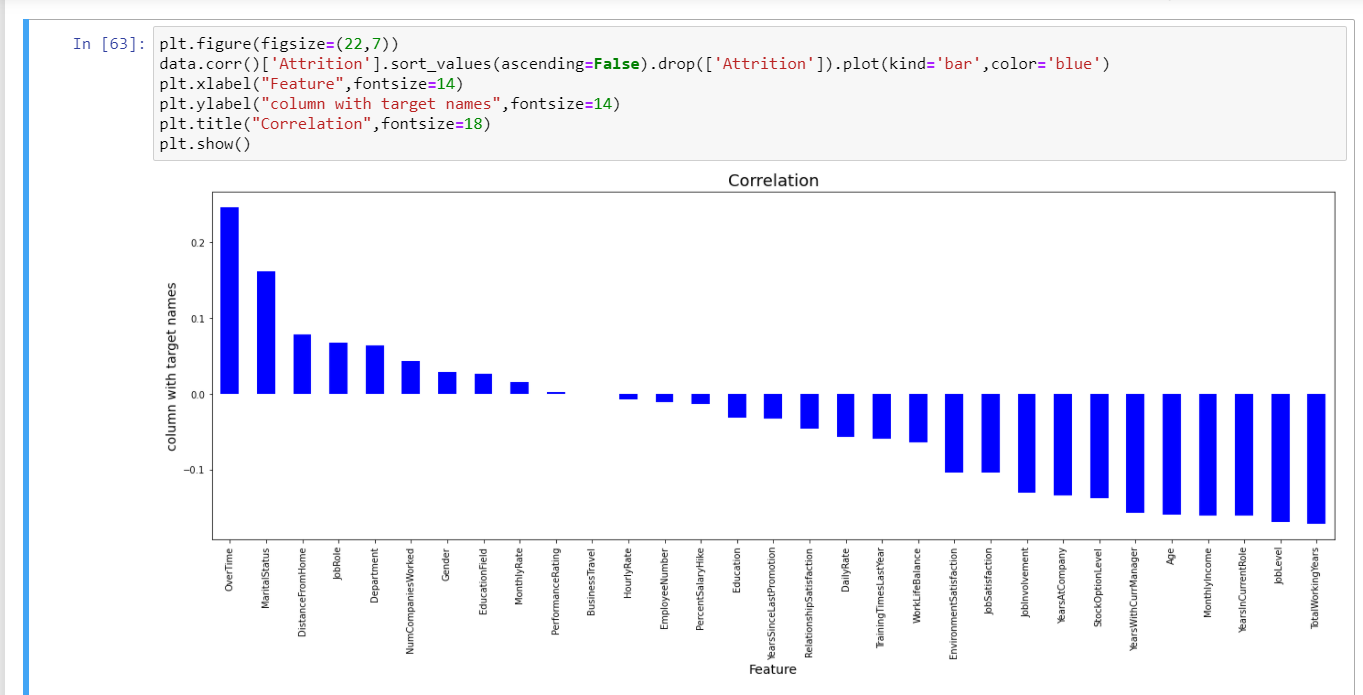
* **Statistical summary**

The statistical view of data is found from this summary. We can get the statistical view using data.describe() . The count is same in all data. There are possible marginal outliers in some of the columns. The mean and median look like in range apart from categorical data. The standard deviation of all data is seen clearly. We can further check for skewness and outliers through visualization and can treat efficiently.



# Correlation Check

Correlation between each column can be viewed using corr(). Business Travel, Performance Rating, Hourly Rate, Employee Number, Percent Salary Hike are nearly zero correlated which can be removed if needed. We can use PCA for feature selections also, OverTime is highly positively correlated to Attrition Total working years, job level are highly negatively correlated to Attrition. Multicollinearity possibility is there among columns which we can check and remove if needed.

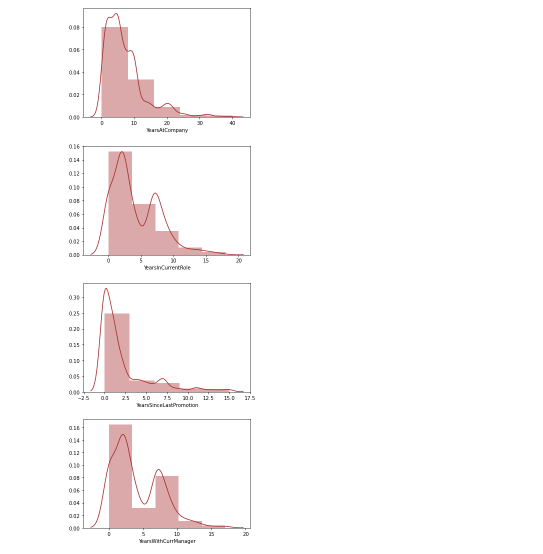


# Distribution of data:

# Skewness

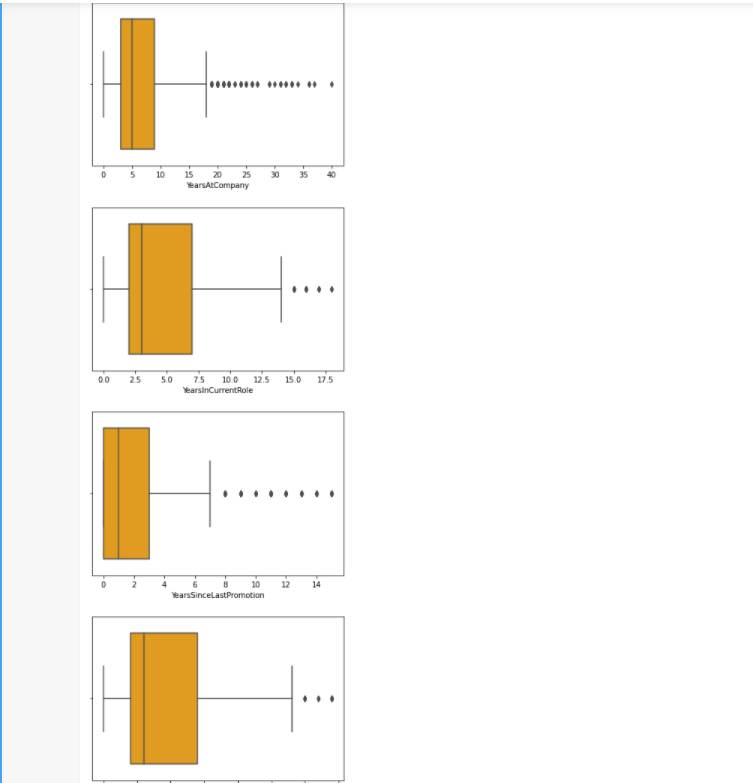
We can identify skewness using distplot. We ca interpret whether the data is normally distributed, rightly skewed or left skewed plotting distplot graph.

Here Years at Company, Years in Current Role, Years Since Last Promotion, Years with current manager are rightly skewed data.



* **Detecting Outliers:**

We can detect outliers using boxplot. It is found that there are outliers found in some of the columns like monthly income, Total working days, Training Times last year, Years at company, Years at current role, Years since last promotion, Years with current manager.



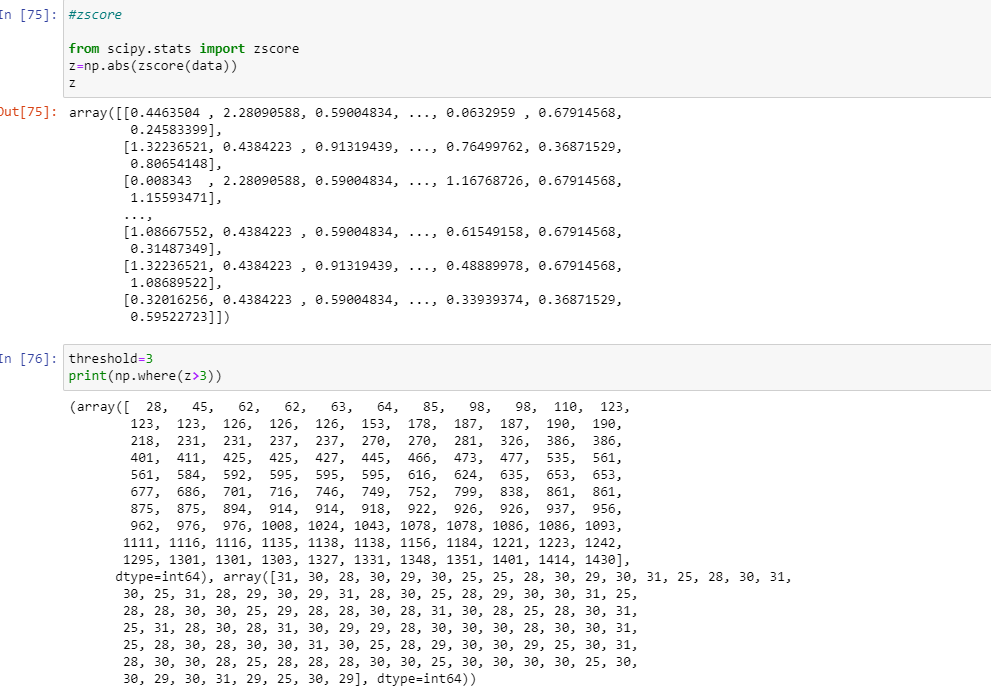
* **Distribution relationship:**

we can see here how the range of data in each column dominates the attrition data. This can be achieved using scatterplot. It is found that each data has either positive or negative relationship with the output data. The correlation value is shown using corr(). We have inferred that highly positive relationship with attrition is overtime and negative positive relationship is Total Working years.

We have identified skewness and outliers in the dataset. We can clean the data now

* **Data Cleaning:**
* **Removing Outliers:**

We are using z-score technique to remove outliers from the entire dataset



After removing outliers, it is found that we have a data loss of around 5 %.

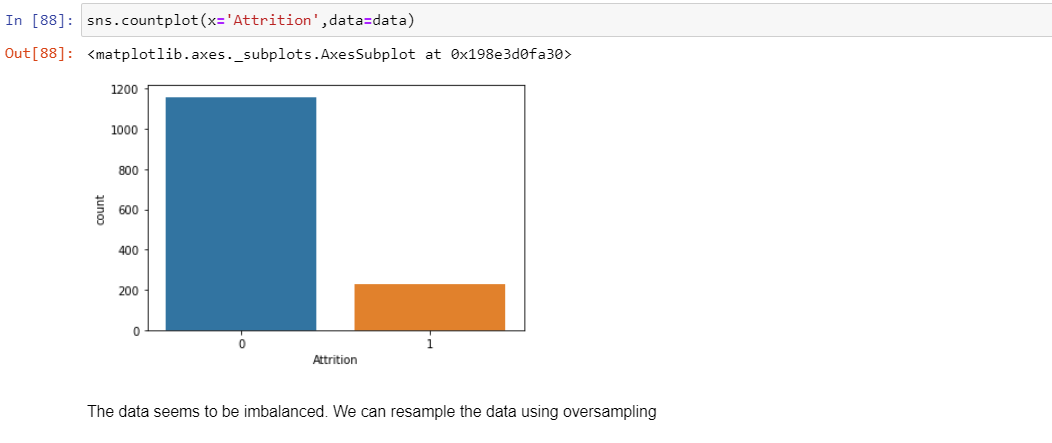
The new dataset consists of 1387 rows and 32 columns.

We have removed outliers from entire dataset. Now, data is clean with no outliers but we have to check skewness and correct it such the data become efficient for model building. We need target variable to be balanced for better result.

* **Imbalanced to balanced:**

Taking countplot of target variable, it is found that the data is imbalanced. We can resample the data using resample technique from sklearn.utils.

After resampling we got new dataset of 2316 rows and 32 columns where attrition column is balanced with equal number of binary data.

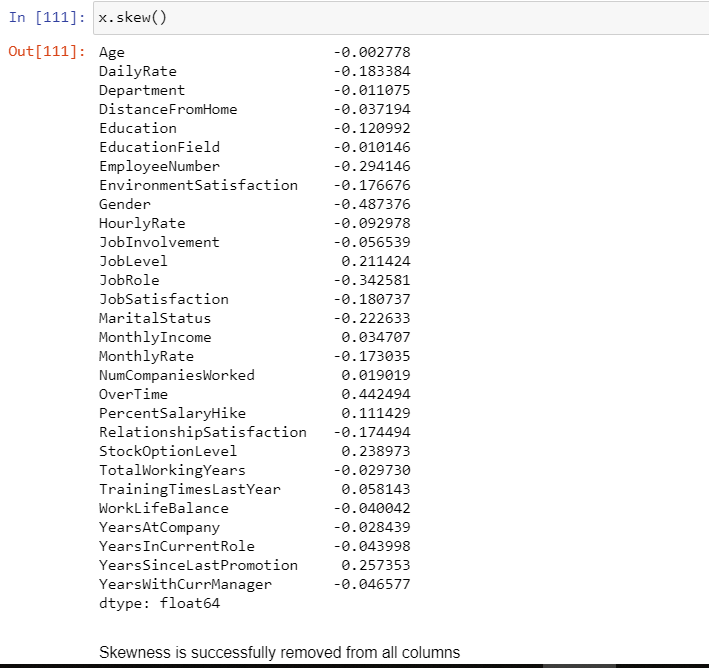




This is how we do resampling such that we have equal number of binary data for efficient model building. Now let us remove skewness.

* **Skewness Removal:**

As we have skewness present in many of the columns ignoring categorical columns, we can use power transformer to the entire dataset such that every data will be transformed efficiently. As we need to apply skewness only for feature columns we can classify the data into x and y where x is the feature columns and y is the target column. Skewness value can be determined using skew(). Power Transformer removes the skewness for the entire dataset and keeps the columns skewness value in acceptable range. We also found that Business Travel and Performance Rating has high skewness and nearly zero correlation with target column. So, we can directly drop it.

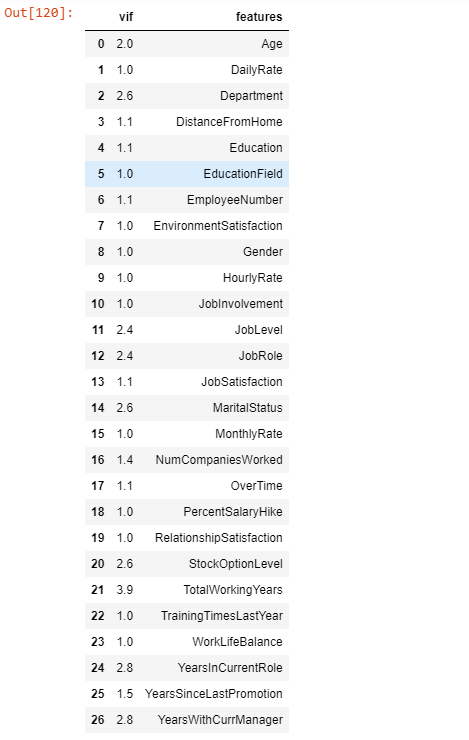


We can see that skewness are corrected and are in range.

The data is ready for model building but we have to check multi collinearity before proceeding.

* **Multicollinear column Removal:**

This helps in removing columns that have same importance to the output to be predicted. If we have variance inflation factor less than 5, it is considered to be a good column without multicollinearity. Total working years and years at company are multicorrelated. Dropping Years at company column as it has lowest p value. Job level and monthly income are multicorrelated. Dropping monthly income as it has lowest p value and low correlation.



We can see all the values below 5. That means multicollinearity of the columns removed.

***EDA CONCLUDING REMARKS:***

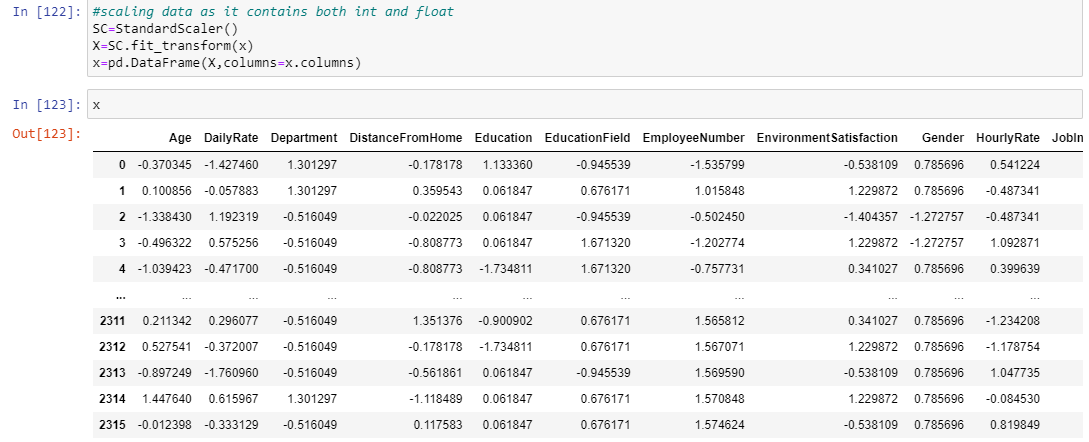
Through EDA we were able to analyse, visualize and clean data for model building, EDA helped in removing outliers and correcting skewness of the entire dataset. We have dropped useless data columns. It also helped in ensuring there is no multi correlation between columns. We were able to clean data efficiently dropping columns that may affect the performance of the model.

After analysing, visualization, computing and cleaning data. The dataset is ready for model building with 2316 rows and 28 columns including target variable.

***Building Machine Learning Models***

* Initializing x and y separately, where x contains of dataset of without target variable that is only feature\_columns and y contains only target variable data.
* **Scaling:**

We can scale the data using Standard Scaler before splitting it into train and predict dataset such that we have scaled data which will make easy for machine to learn . This is specifically done when we have different data type in a dataset.



* **Best Random\_State and train\_test\_split**

We can find best random state first such that we can apply that while splitting the dataset into training and prediction phase. As it is of binary output we can use logistic regression to find the best random state for the model as shown in the figure.

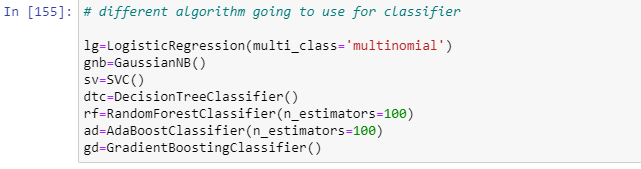


We can find random state as 7 with 82 % accuracy score which we will be using in train and test as shown. The random state is chosen from range 1-200

* **Different Algorithm used:**

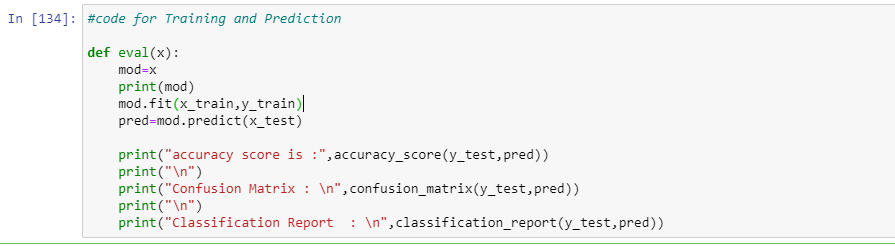
We have used different algorithm to determine the best model for the dataset. The algorithm used in this model are:

* Logistic Regression
* GaussianNB
* SVC
* DecisionTreeClassifier
* RandomForestClassifier
* AdaBoostClassifier
* GradientBoostingClassifier



* **Metrics:**

The metrics like accuracy score, confusion matrix and classification report are calculated for each algorithm and chosen the best algorithm for final model. The metrics can be found by,



Using this code, we can do training and prediction phase for each algorithm and can find the score through which we can have insights on how good each algorithm performs after data feeding.

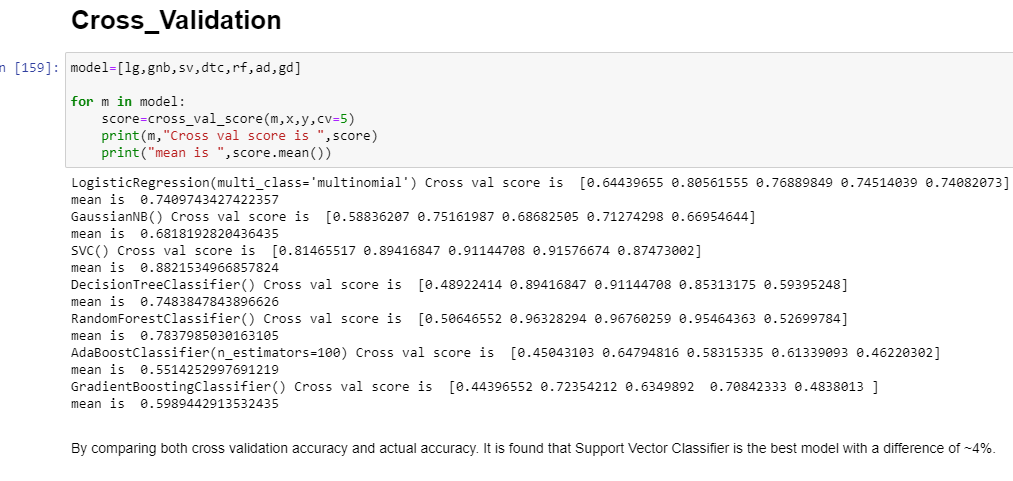
* **Accuracy Scores**

The accuracy score of each algorithm is as follows:

|  |  |
| --- | --- |
| **Algorithm** | **Score** |
| * Logistic Regression | 82.09 |
| * GaussianNB | 76.73 |
| * SVC | 92.41 |
| * DecisionTreeClassifier | 90.32 |
| * RandomForestClassifier | 96.73 |
| * AdaBoostClassifier | 87.82 |
| * GradientBoostingClassifier | 89.80 |

We can see that RandomForestClassifier and Support vector classifier (SVC) gives the top 2 score for the model. But we have to cross check with cross validation score to finalize the model.

* **Cross Validation Score**



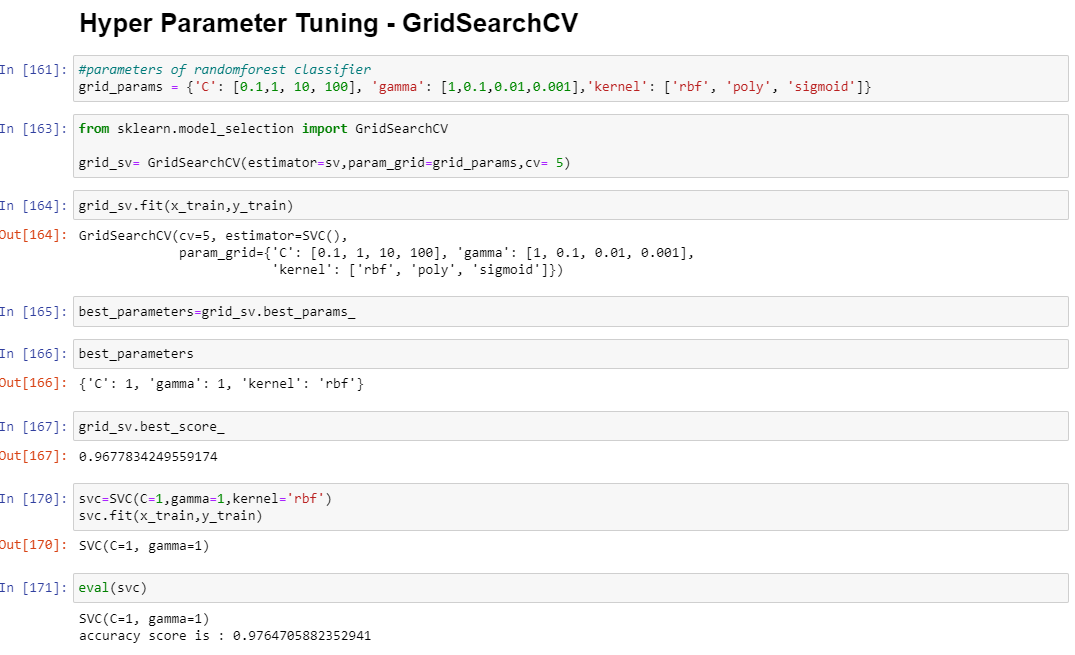
The minimum value difference between the accuracy score and cross validation score gives us the best model. By comparing both cross validation accuracy and actual accuracy. It is found that Support Vector Classifier is the best model with a difference of ~4%

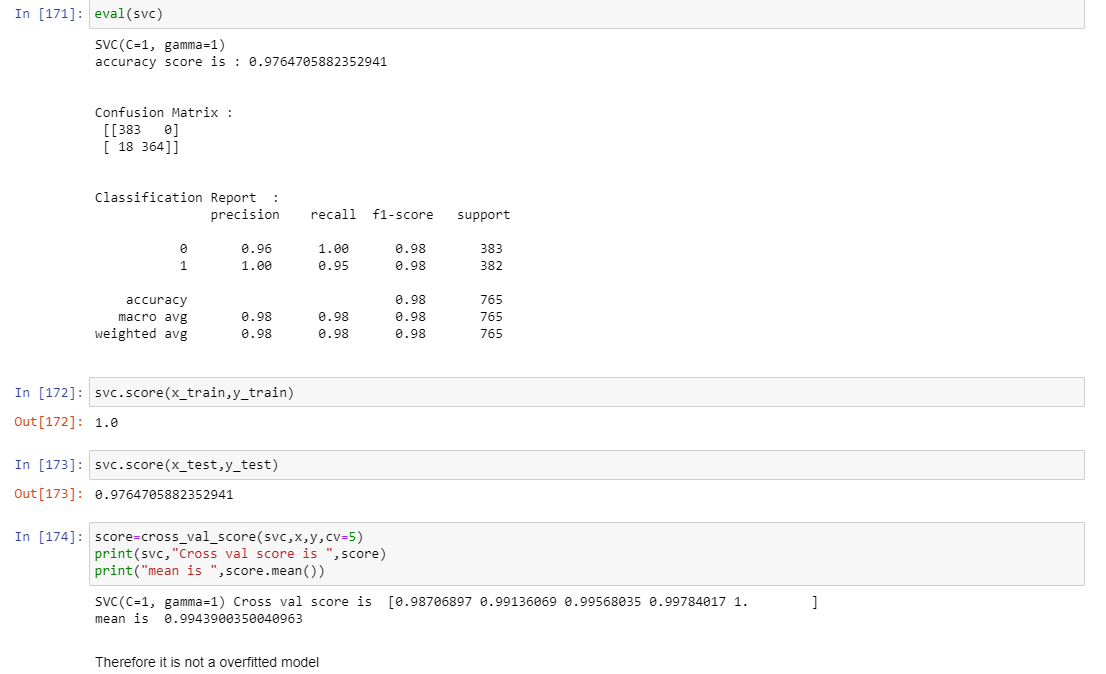
# Support Vector classifier model is the best model with 92.41% accuracy

We will try to improve accuracy by hyper tuning.

* **Hyper Tuning – GridSearchCV**

We can use different parameters for support vector classifier such as 'C': [0.1,1, 10, 100], 'gamma': [1,0.1,0.01,0.001],'kernel': ['rbf', 'poly', 'sigmoid'] such that we try to increase the accuracy score.





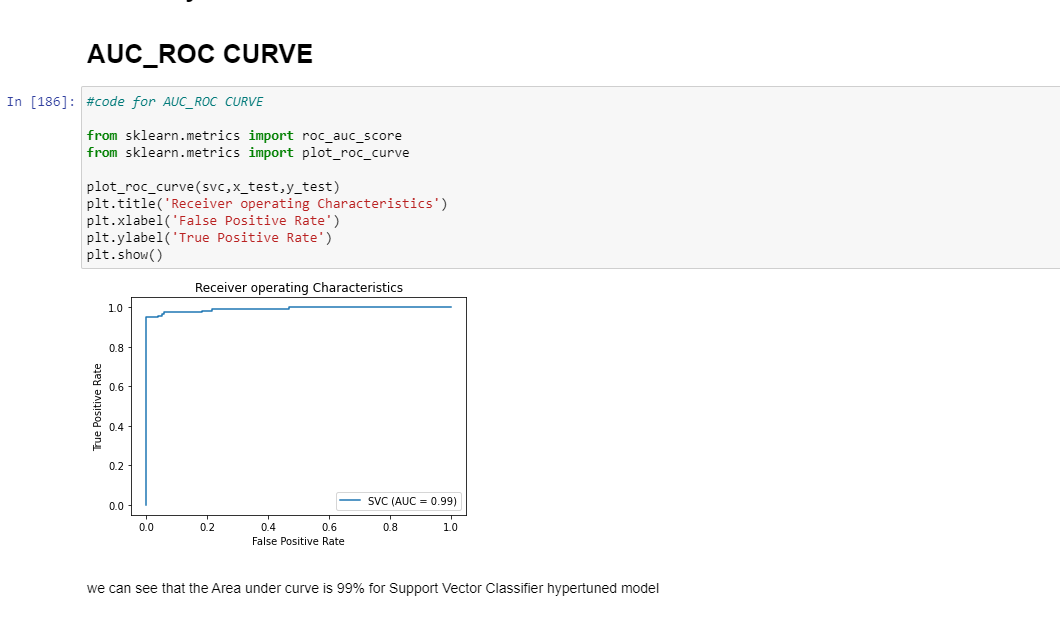
* ***Concluding Remarks***

The hyper parameter tuning of support vector classifier gives actual accuracy of 97.64 % and cross validation score of 99.43 %. Since there is an accuracy increase of 4% for the hyper tuned parameter, the hyper tuned model of Support Vector Classifier algorithm is selected for final output. Since Hyper parameter tuning taking a lot of time, it is performed only for top model to see whether it is improving the accuracy.

# Support Vector Classifier hypertuned model with true accuracy 97.64% and CV accuracy score of 99.43% is selected as final model for execution

We can check the same on the AUC\_ROC CURVE

* **AUC\_ROC CURVE**



The area under curve is high for Support Vector Classifier (hypertuned parameter) with AUC score of 0.99. That means 99% of area under curve is covered in this model.

# *Support Vector Classifier (svc) (hyper tuned) covers the maximum area. Therefore svc is selected as the final model with true accuracy of 97.64%*

# 