

**PROJECT REPORT ON**

**EMPLOYEE PERFORMANCE ANALYSIS**

**WORKFORCE**  
**ANALYSIS**

**USING**

**SUPERVISED MACHINE LEARNING TECHNIQUES**

**Submitted towards partial fulfilment of the criteria  
For award of PGPDSE by Great Lakes Institute of Management**

**SUBMITTED BY**  
**Group C [Batch: Dec 2018]**

**GROUP MEMBERS**  
**1. Yash Agarwal**  
**2. Aman Kumar**  
**3. Ashwin Yenigalla**  
**4. Sara Faruqui**

**RESEARCH SUPERVISOR**  
**Mr. Srikar Muppidi**



**Great Lakes Institute of Management**

**ABSTRACT**

- The performance of various employees in an organisation varies and so is the probability of each employee getting promoted. Not getting promoted could have a direct bearing against employee attrition and hence the HR department would like to know the probability that an employee will get promoted.  
The objective of this project is to predict whether an employee will get promoted or not and also understand the factors which impact the promotion. This helps HR team to plan for back up resources prior to rating cycle against the resources who have high chances of not getting promoted.
- Techniques:
  - Predictive Modelling
  - Supervised Machine Learning Models
  - SMOTE
- Tools:
  - Python
  - IDE - Jupyter Notebook

## ACKNOWLEDGEMENT

We hereby certify that the work done by us for the implementation and completion of this project is absolutely original and to the best of our knowledge. It is a team effort and each of the member has equally contributed in the project.

Date: 23-05-2019

Place: New Delhi

## CERTIFICATE OF COMPLETION

This is to certify that the project titled "<**Employee Performance Analysis**> – **Workforce Analysis**" for case resolution was undertaken and completed under the supervision of Mr. Srikar Muppidi for Post Graduate Program in Data Science and Engineering (PGP – DSE)

Mentor: Mr. Srikar Muppidi

## TABLE OF CONTENTS

S. No.	Topic	Page No.
1.	Executive Summary	6
2.	Chapter 1.	7-9
	• Project Flow	
	• Project Overview	
	• Explanation of Dataset	
3.	Chapter 2	10-26
	• Visualisation of the Target	10
	• Exploratory Data Analysis	11-26
	- Univariate Analysis	
	- Bivariate Analysis	
	- CrossTab Analysis	
4.	Chapter 3	
	• Modelling	27-37
	- Baseline Estimator	
	- Scaling the Dataset	
	- Model Preparation	
	- Random Forest Classifier	

## EXECUTIVE SUMMARY

The performance of various employees in an organisation varies and so is the probability of each employee getting promoted. Not getting promoted could have a direct bearing against employee attrition and hence the HR department would like to know the probability that an employee will get promoted.

The collected dataset represents variables that effect the promotion of the employees in the organisation. The variables are as follows:

Employee Id represents the identity in the organisation. Then in which department the employee is working with the Region mentioned. The qualification which is an essential variable that impacts the promotion of the employee. In an Organisation the ratio of gender that the company is following with the recruitment channel from where the company hire. The dataset also reveals the Age of the employees and their previous year ratings also their length of services they have served in the organisation. The Organisation has the tendency to give awards to their employees who works excellent. Employees are also provided with the training from which they get training Score for their upgradation. With all these, the promotion gets affected which is the target variable.

The Dataset has 54808 instances (rows) and 14 attributes (Columns). It has 3 Continuous variables and 11 Categorical.

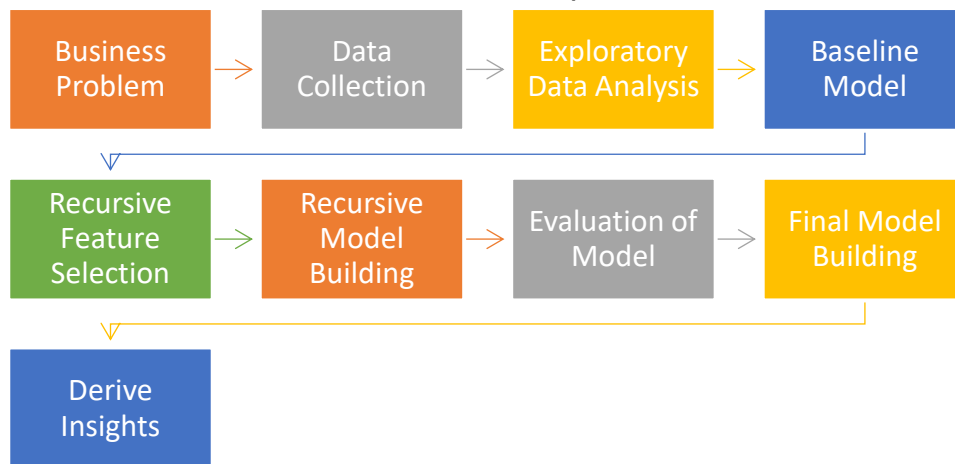
The project requires data pre-processing to treat for missing values and outliers and do exploratory data analysis to visualise the spread of each of the X variables and the relationship between the various X variables and the y variable.

The data set needs to be divided into train and test sets. After train test split, build a model to predict whether an employee will get promoted or not. Evaluate the model based on model performance measures for classification and recommend the most suitable model.

## CHAPTER 1

### 1. PROJECT FLOW

The process we chose to work on is an iterative process.



- We define the business problem.
- We collect data from different relevant sources.
- In Exploratory Data Analysis, we try to understand the structure of the data, the nature of the variables. The information present in the data. A special attention is paid to the target variable distribution.
- We build baseline models to get the idea of the minimum amount of performance that we can get from each of the models we have decided to use in the future.
- To start with the modelling, we found the correlation to start up with the baseline model. After seeing the result of baseline, we applied get dummies to have better inference and also, we have applied pre-processing on data.
- Then after seeing the baseline model, we started to apply machine learning models to get the better accuracy score.

## 2. PROCESS OVERVIEW

A stepwise process has been followed in implementation of the supervised learner.

1. The project starts with research work for the dataset.
2. Once the data is ready, it has been imported as a pandas data frame and the structure of the data is noted.
3. A recursive approach has been followed while checking the data for missing values, variable data types and outliers. We found that there were no missing values.
4. Exploratory data analysis has been performed including the univariate and bivariate analysis for the continuous variables and for the categorical variables as far as possible.
5. Data has been visualized using different charts including distance plots, histograms, bar charts, Factorplot, Violineplot and box plots.
6. The target variable has been completely analysed.
7. There were no missing values in the dataset.
8. Baseline model estimator for this model gives accuracy with 91.4%.
9. Based on conclusions on baseline model, we applied Dummies to extract the data which were categorical.
10. Then, from sklearn.model\_Selection we imported train\_test\_split to apply on the dataset.
11. Multiple classification models including Gaussian Naive Bayes, KNN, Decision tree, Random forest are implemented recursively in the process to compare the results of each.
12. To find the best model accuracy which is better than baseline we apply pre-processing on the dataset
13. Finally, we applied Kfold Cross Validation to find which model gives the best accuracy score, confusion matrix and classification report when compared.



### 3. EXPLANATION OF DATASET

DATASET HAS 54808 INSTANCES (ROWS) AND 14 ATTRIBUTES (COLUMNS).

#### **NUMBER OF ATTRIBUTES**

14 COLUMNS: 3 CONTINUOUS AND 11 CATEGORICAL.

#### **ATTRIBUTES INFORMATION**

ATTRIBUTE 1: EMPLOYEE\_ID WITH CONTINUOUS FIGURES.

ATTRIBUTE 2: DEPARTMENT (CATEGORICAL) WITH 9 DIFFERENT DEPARTMENT NAMES.

ATTRIBUTE 3: REGION (CATEGORICAL) WITH 34 DIFFERENT REGIONS.

ATTRIBUTE 4: EDUCATION (CATEGORICAL) WITH 4 DIFFERENT DEGREE NAMES.

ATTRIBUTE 5: GENDER (CATEGORICAL) WITH MALE AND FEMALE.

ATTRIBUTE 6: RECRUITMENT CHANNEL (CATEGORICAL) WITH 3 DIFFERENT CHANNELS.

ATTRIBUTE 7: NO\_OF\_TRAININGS (CATEGORICAL) WITH 10 DIFFERENT TRAINING.

ATTRIBUTE 8: AGE (CONTINUOUS) VARIES AS UPPER LIMIT – 60 AND LOWER LIMIT -20.

ATTRIBUTE 9: PREVIOUS\_YEAR\_RATING (CATEGORICAL) VARIES FROM 0-5.

ATTRIBUTE 10: LENGTH\_OF\_SERVICE (CATEGORICAL) TIME SPENT VARIES FROM 1-37 YEARS IN COMPANY.

ATTRIBUTE 11: KPIS\_MET >80% (CATEGORICAL) WITH 0 AND 1.

ATTRIBUTE 12: AWARDS\_WON (CATEGORICAL) WITH 0 AND 1.

ATTRIBUTE 13: AVG\_TRAINING\_SCORE (CONTINUOUS) WITH 61 DIFFERENT AVERAGES.

ATTRIBUTE 14: IS\_PROMOTED (CATEGORICAL) WITH 0 AND 1 AS TARGETS.

## CHAPTER - 2

### 1. Visualisation of the Target

- Performance Metrics with Variables



Inference:

- More people got promoted in case of KPI > 80%.
- Very few employees were able to win awards.



Inference:

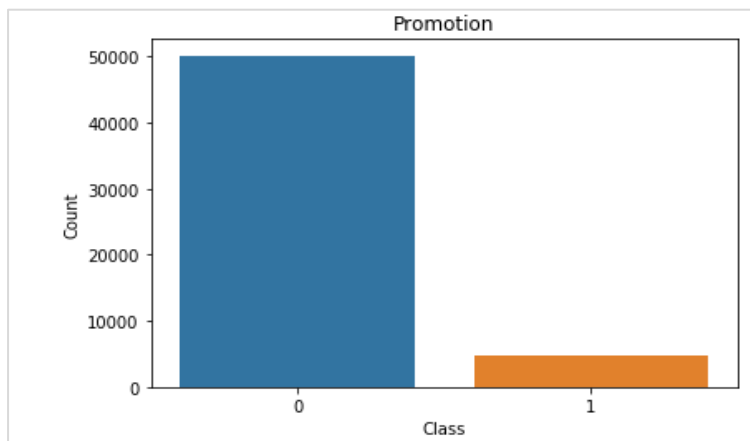
- Maximum number of promotions were given on single no. of training.
- Employees who scored more in Appraisal had more chances of promotion.



## 2. Exploratory Data Analysis

- Univariate Analysis (Categorical)

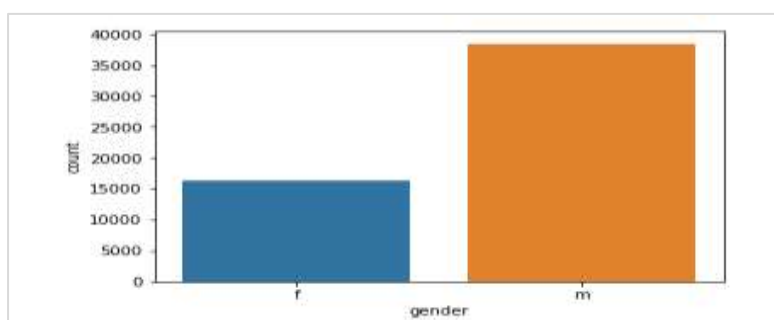
### Target Variable: is promoted



```
# Target Variable
grph = sns.countplot(employee.is_promoted)
grph.set(xlabel = "Class", ylabel = 'Count', title = 'Promotion')
plt.tight_layout()
plt.show()
```

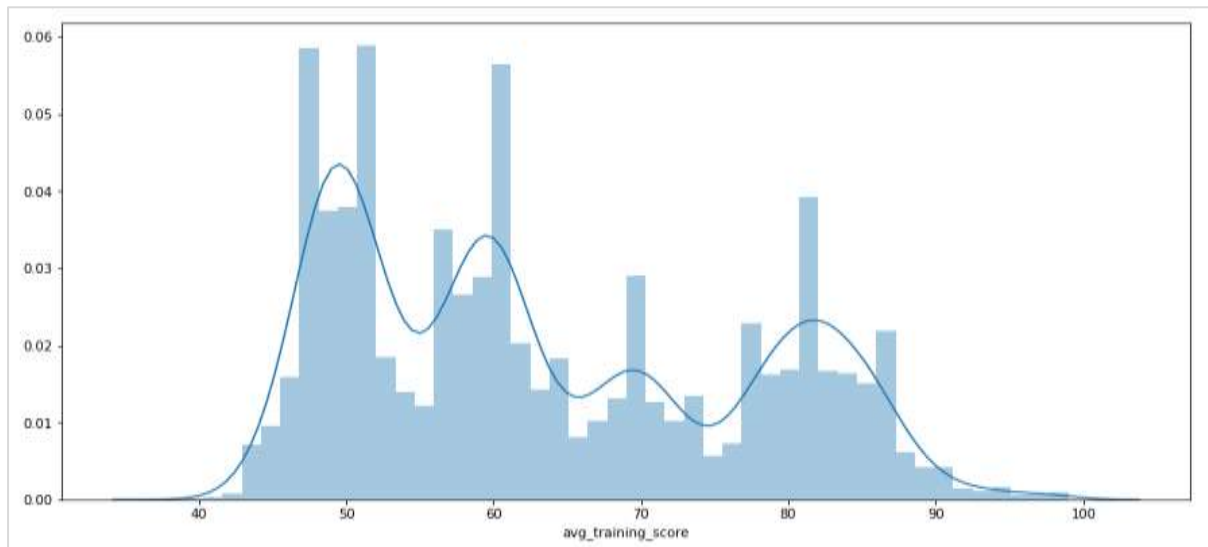
Inference: Target Variable has highly Imbalanced data, very few promotions are given.

### Attribute: Gender

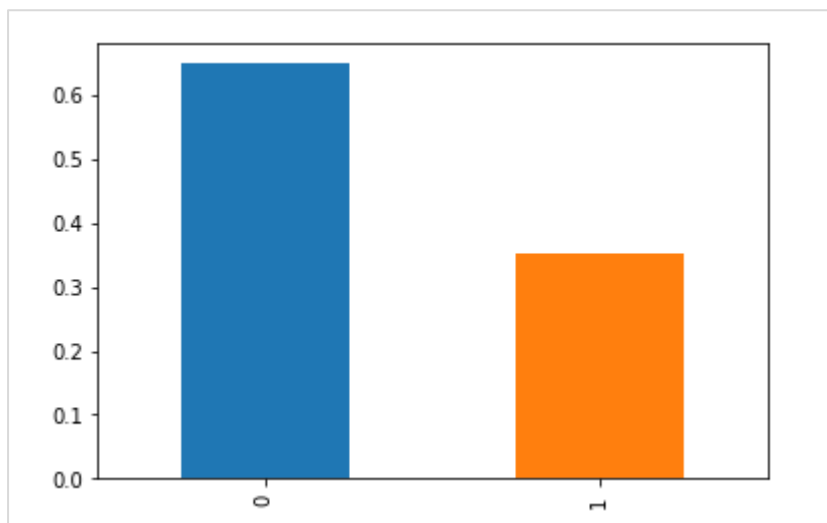


```
sns.countplot(x = 'gender', data = employee) # Male
plt.show()
```

Inference: The Data has been dominated by Males.

**Attribute : Average Training Score**

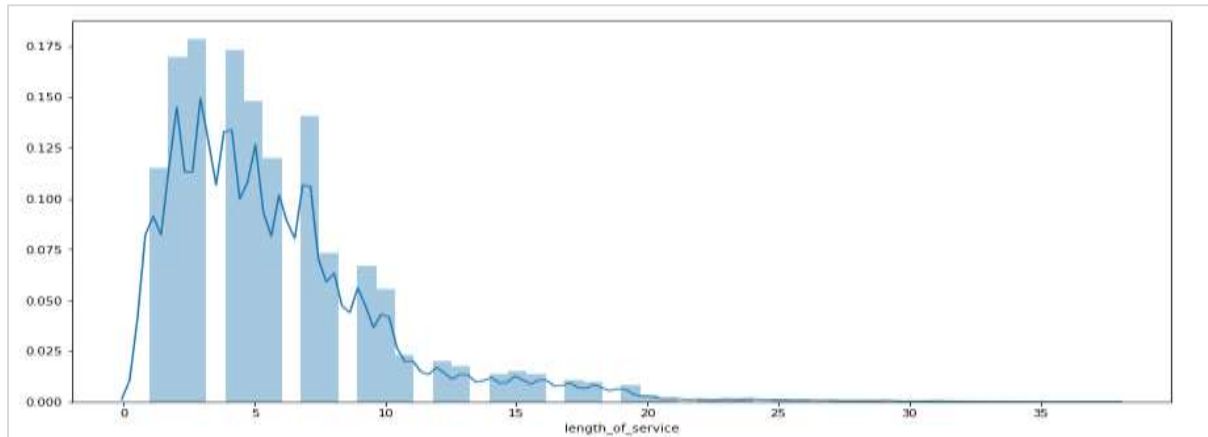
Inference: The highest Average Training Score majorly fall in between 40-55

**Attribute : KPIs met >80%**

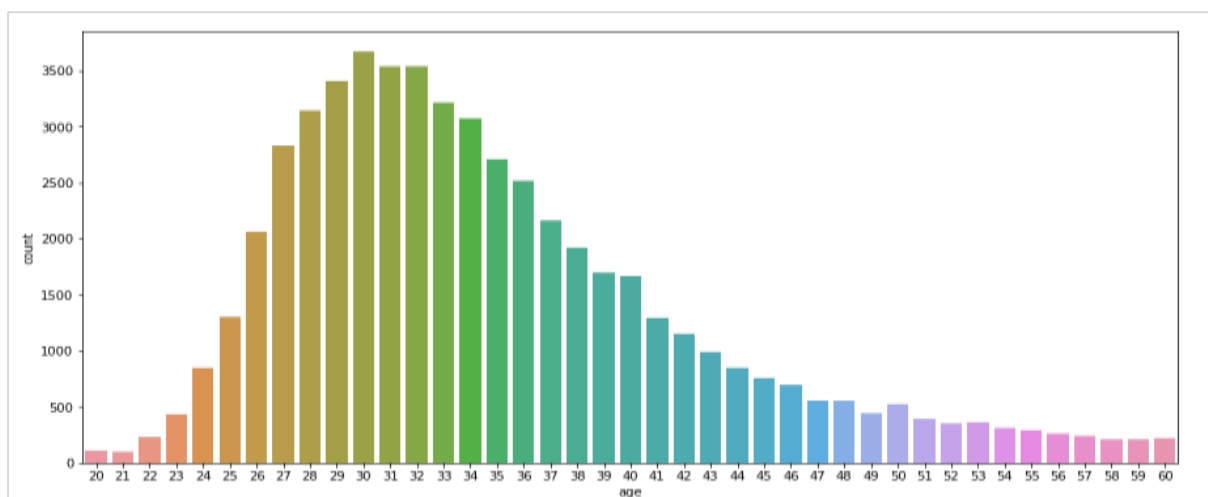
```
employee['KPIs_met >80%'].value_counts(normalize = True)
```

```
0    0.648026  
1    0.351974  
Name: KPIs_met >80%, dtype: float64
```

Inference: KPI with <80% are just double of KPI with >80%

**Attribute : Length of Service**

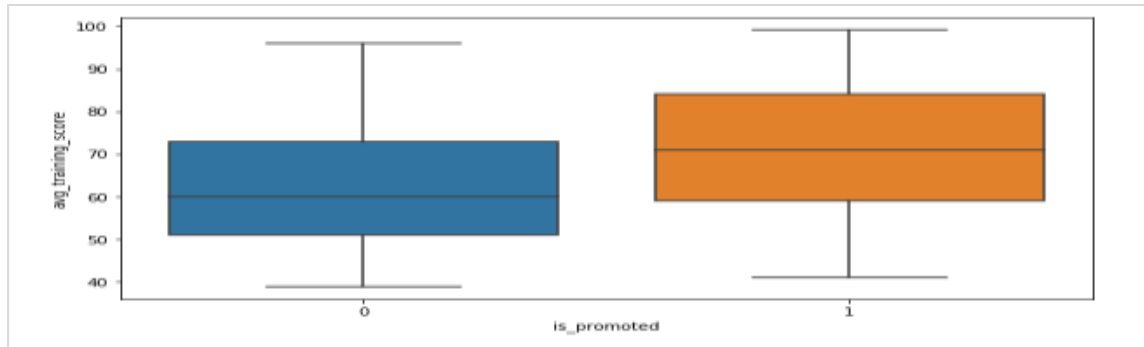
Inference: Services served by employees is majorly between 0-10 years

**Attribute : Age**

Inference: The Major Employees in the company are in between 27-36 years

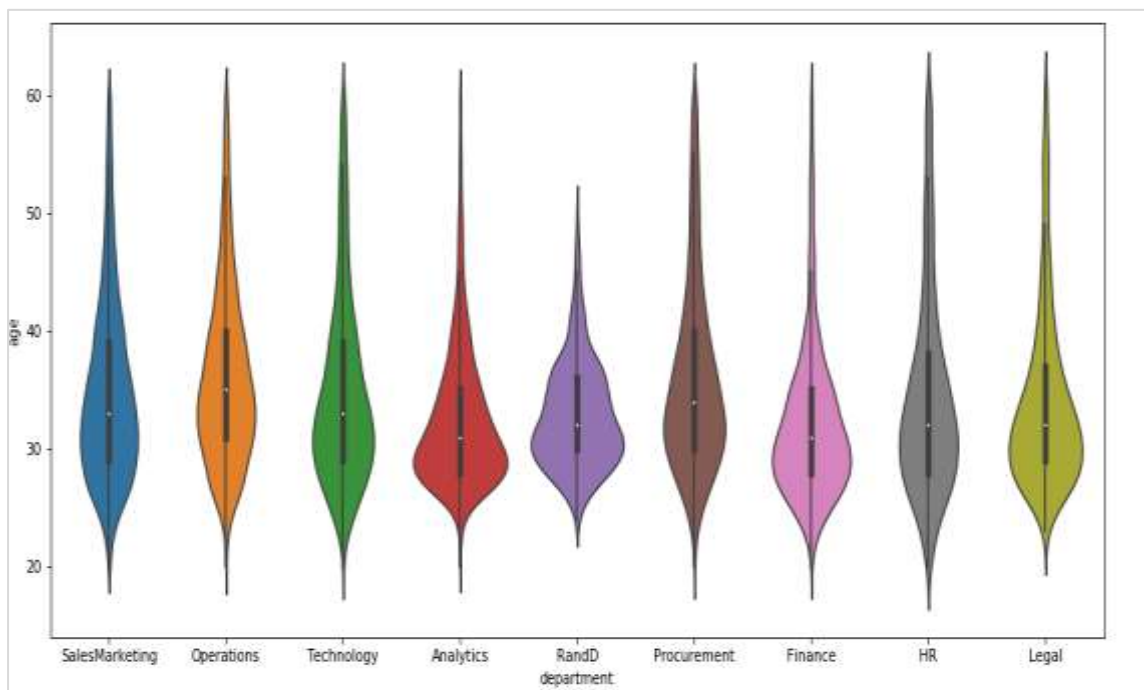
- **Bivariate Analysis**

**Attribute: x =[Promoted] y =[Average Training Score]**



Inference: The employee which are promoted have more Average Training Score than who are not promoted. This means More Training Score has more chance to get promote.

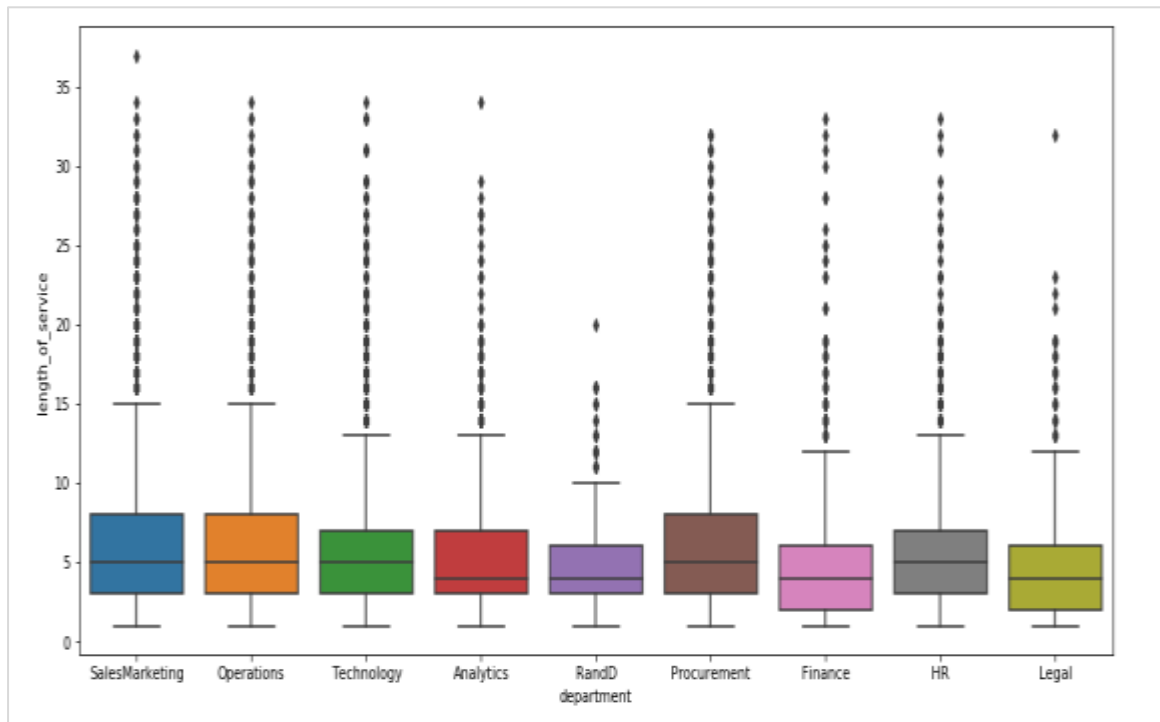
**Attribute : x =[Department] y =[Age]**



Inference:

- 1- Average age of people working in different departments are in range 32- 36.
- 2- Interesting part is R&D dept has quite younger people. (with analytics finance and legal.)
- 3- operations, procurement, technology, sales and marketing and HR have more people above 50. (especially compared to analytics and R&D).

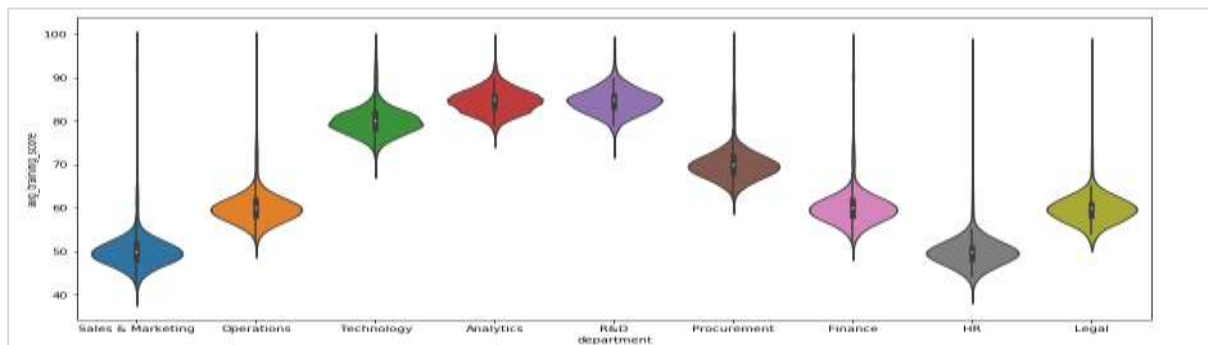
**Attribute :  $x$  =[Department]  $y$ =[Length of Service]**



Inference:

1. All departments have average length of service between 3-8 years.
2. Analytics and finance department has mostly less experienced (more young professionals) than compared to others.
3. Sales & marketing, Technology, Operations and HR has decent mix of experienced professionals (a good range of experience).

**Attribute :  $x$  =[Department]  $y$ =[Average Training Score]**

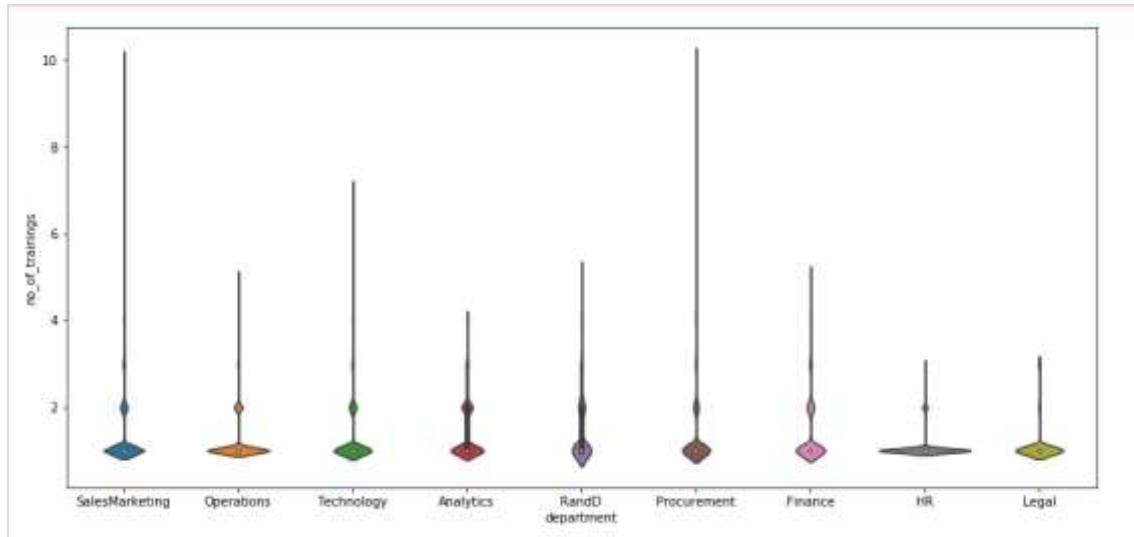


Inference:

1. Analytics and R&D dept scores highest in company training performance scores.

2. Sales & Marketing and HR dept who are huge in numbers scores the least.

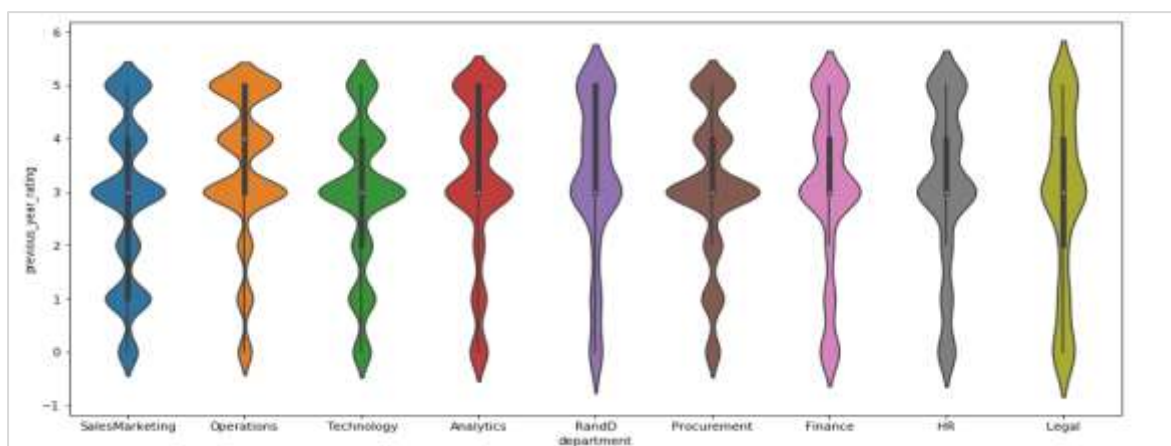
**Attribute x =[Department] y =[No of trainings]**



Inference:

1. Sales & Marketing, Procurement are good dept in terms of training as they are huge in numbers and they have done the maximum trainings.
2. HR is the worst dept in terms of training attended.
3. Analytics and R&D although less in number but have done good amount of 2 trainings than compared to every other who have done mostly just 1.

**Attribute : x =[Department] y=[Previous Year Rating]**



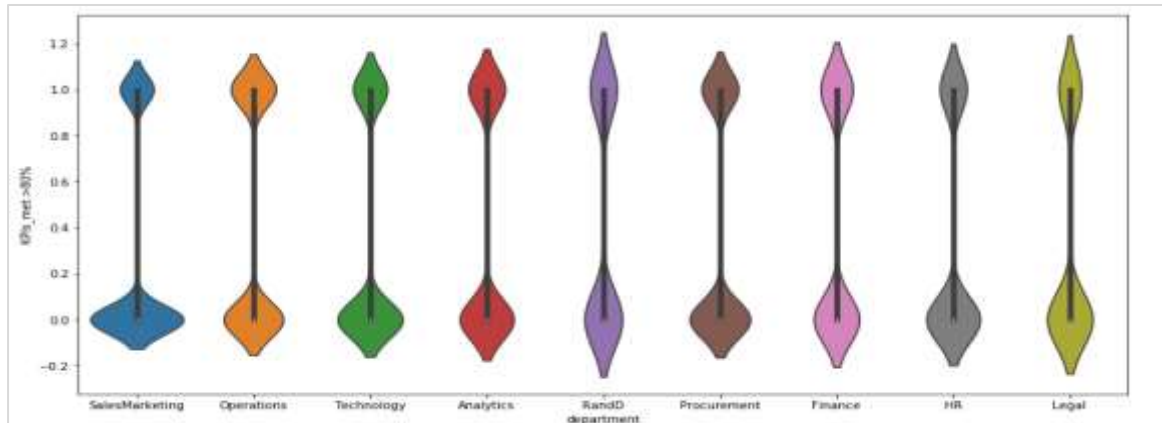
Inference:

1. Almost every dept has on an average previous year rating of 3.



2. Analytics, operations and R&D dept has good amount of 5 rating

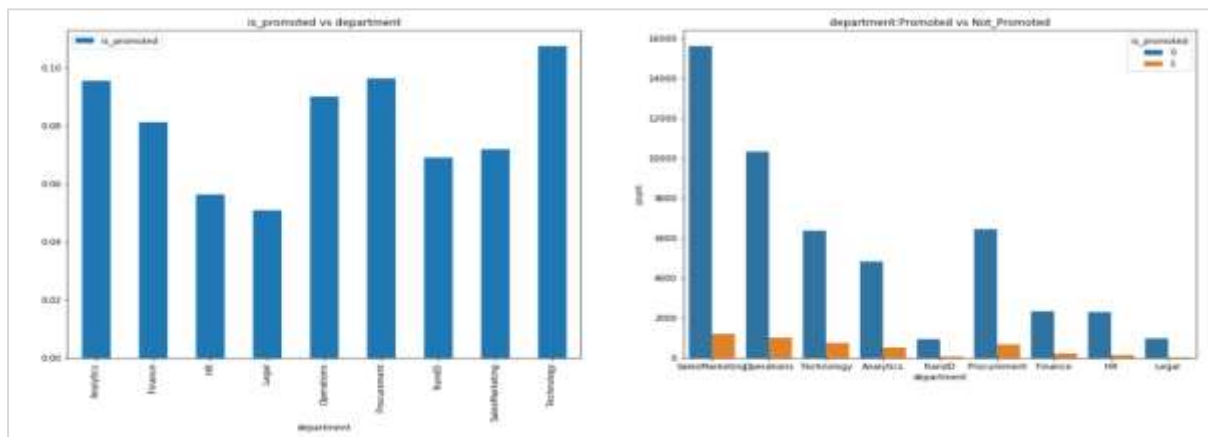
**Attribute : x =[Department] y =[KPIs met >80%]**



Inference:

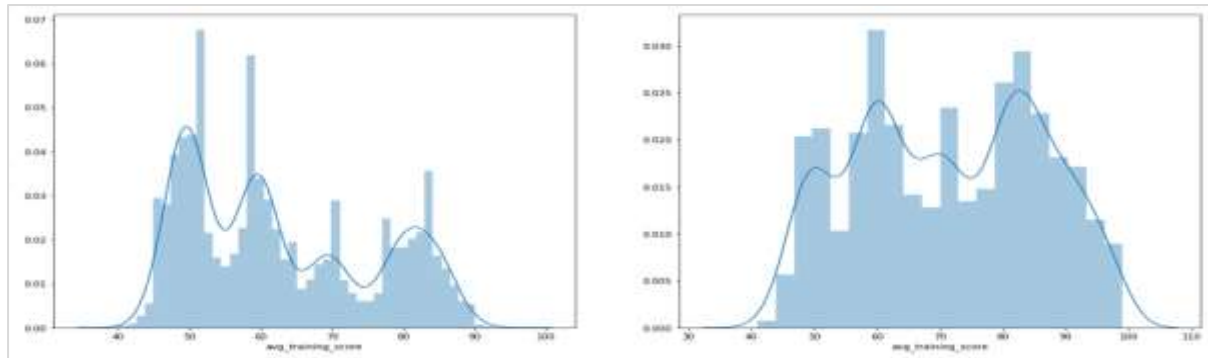
1. Every department has good mix blend of KPI scores.
2. Analytics can be seen much better among all of them.

## Department VS Promotion



Inference: From Sales & Marketing department, we have seen maximum employees get promoted.

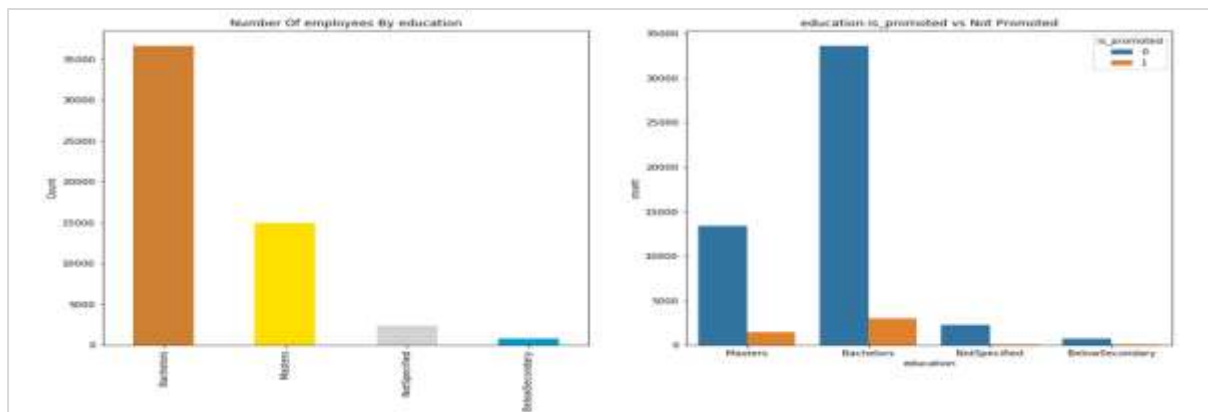
## Average Training Score VS Promotion



Inference:

1. Highest Average Training Score was: 99
2. Lowest Average Training Score was: 39
3. Average of all employee Average Training Score was: 63.38675010947307

## Education VS Promotion

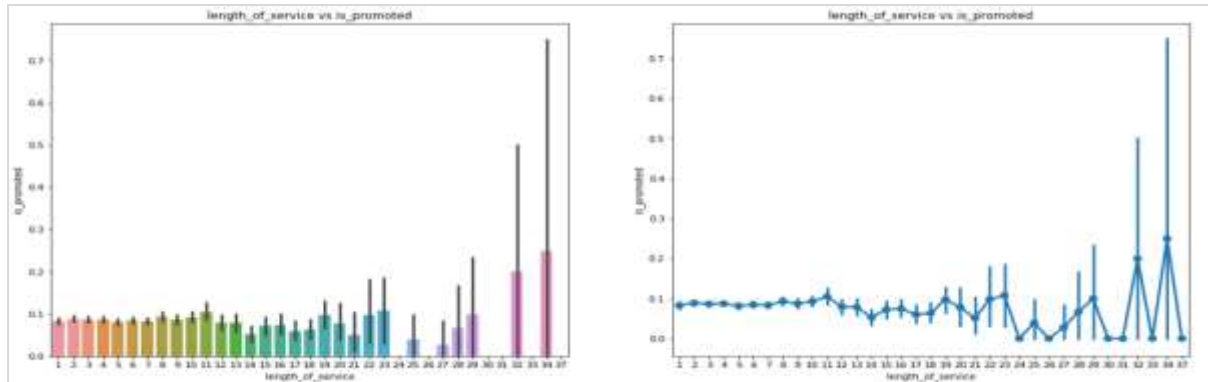


Inference:

As we can see that employees who are Post Graduate were given more priority than any other education.

For Masters, % promoted is around 10% while for Bachelors, is around 8%

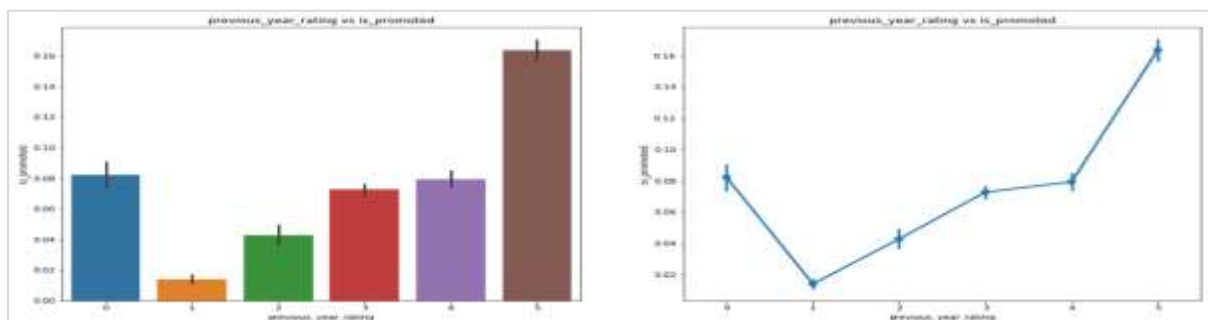
## Length Of Service VS Promotion



Inference:

1. Here too the results are quite similar. Employee with their more experience have greater chance of promotion.
2. Experience from 1 to 11 years there is constant rate of promotion.
3. It however increases after 26 years of experience as the number goes up.

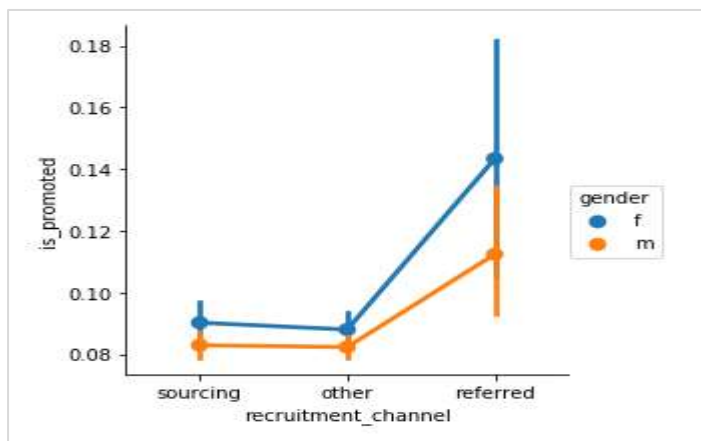
## Previous Year Rating VS Promotion



Inference: We can clearly observed from the Bar plot and Factor Plot that if Employee Previous Rating is High more chances employee get promoted

## Gender VS Promotion

recruitment_channel		other	referred	sourcing	All
gender	is_promoted				
f	0	8350	269	6226	14845
	1	805	45	617	1467
m	0	19540	735	15020	35295
	1	1751	93	1357	3201
All		30446	1142	23220	54808

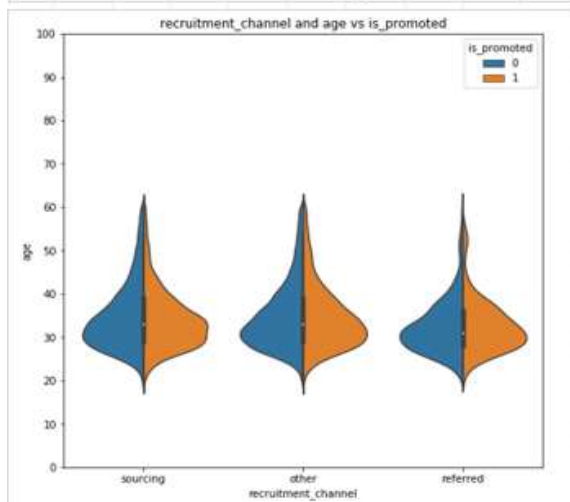


Inference: Looking at the Crosstab and the Factor Plot, we can easily infer that promotion for Women from Referred recruitment channel is about 14%.

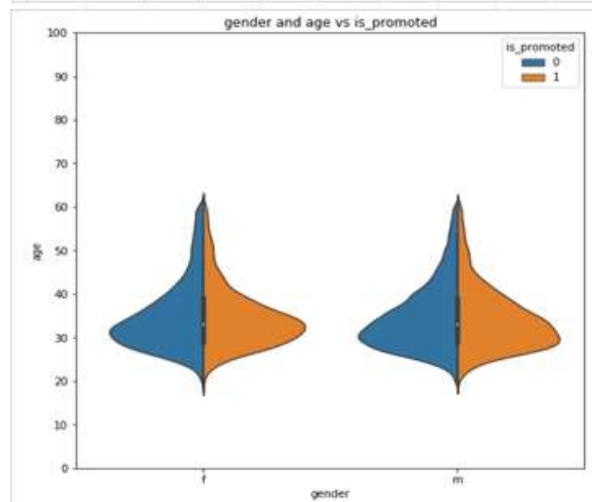
It is evident that irrespective of recruitment channel, Women were given first priority while promotion. Even Men from any recruitment channel have a less promotion chance.

## AGE VS Promotion

Recruitment\_channel and Age VS Promotion

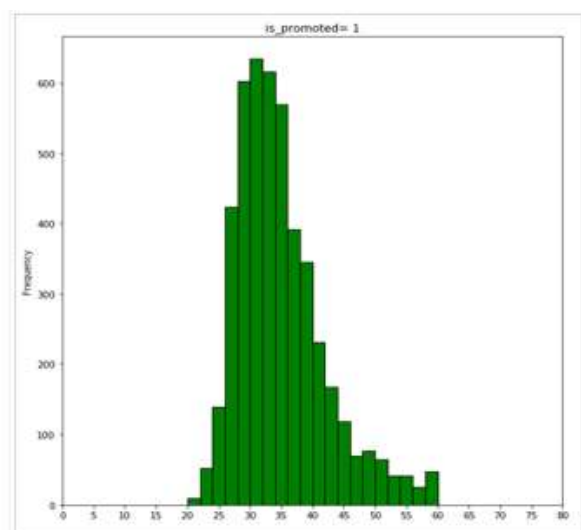
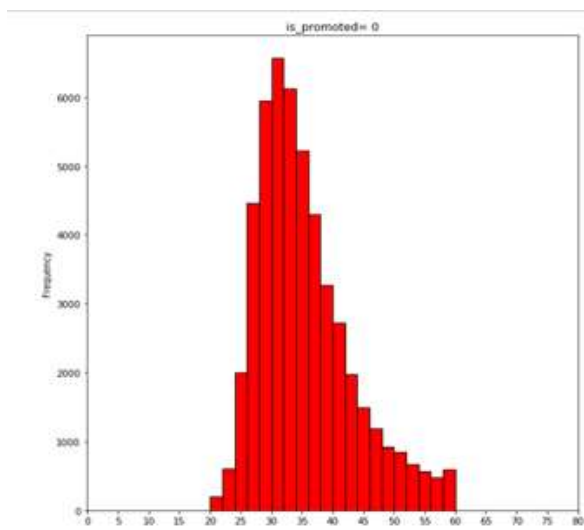


Gender and Age VS Promotion



Inference: 1. The oldest employee was promoted (60 years).

2. Maximum number of promotions were in the age group of 30-40.

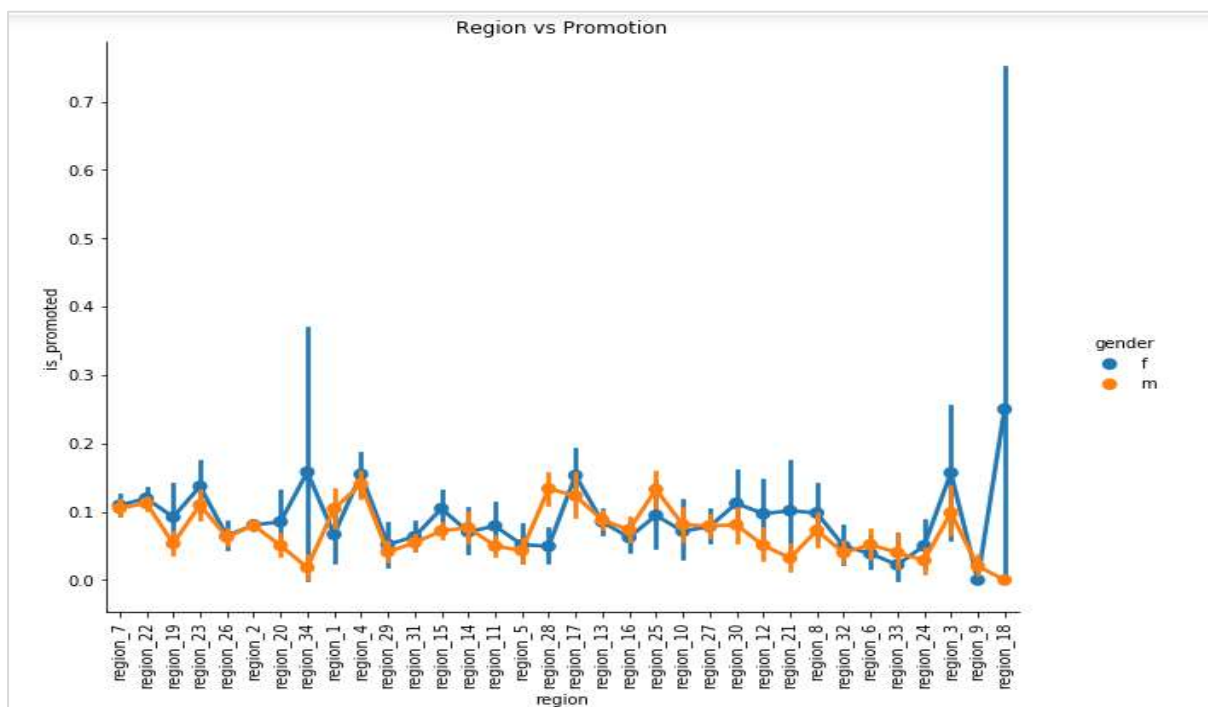
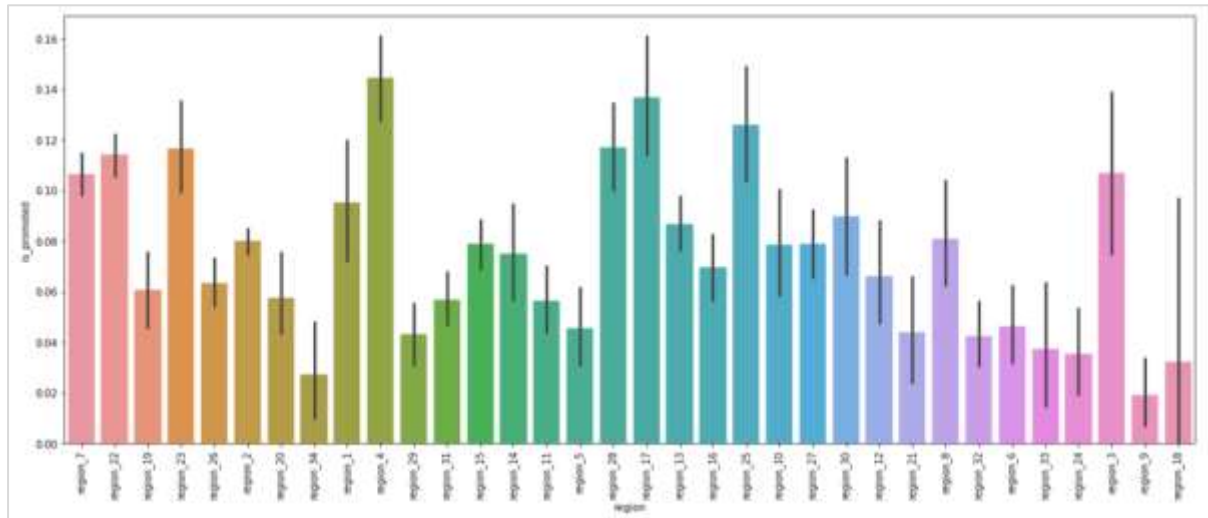


Oldest employee was of: 60 Years

Youngest employee was of: 20 Years

Average Age of employee: 34.8 Years

## Region VS Promotion.



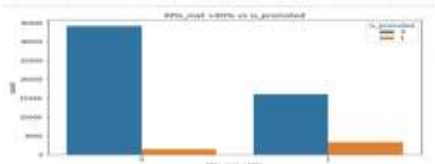
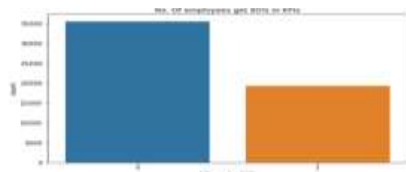
Inference:

1. From Bar plot and Factor Plot shows that region 4 has greater impact on Promotion. As average of Male and Female highest on region4.
2. We can clearly observe that from many of the regions female have higher promotion rate than male.

## KPIs\_met >80% VS Promotion

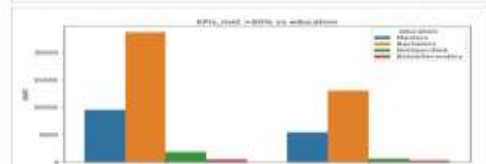
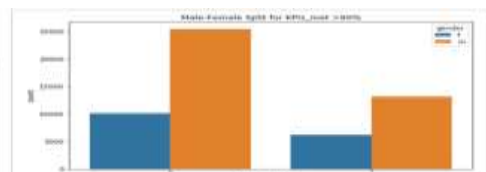
Crosstab Between KPIs\_met &gt;80%, Gender, Education VS promotion

		gender	f		m		All
		is_promoted	0	1	0	1	
KPIs_met >80%	education						
0	Bachelors	6469	272	16300	629	23670	
	BelowSecondary	166	9	295	13	483	
	Masters	2763	147	6340	300	9550	
	NotSpecified	294	3	1484	33	1814	
1	Bachelors	3455	658	7437	1449	12999	
	BelowSecondary	102	12	175	33	322	
	Masters	1513	355	2838	669	5375	
	NotSpecified	83	11	426	75	595	
All		14845	1467	35295	3201	54808	

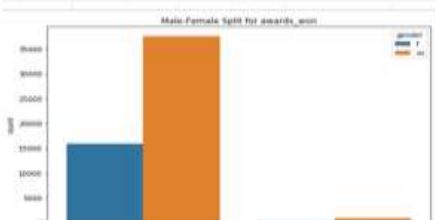
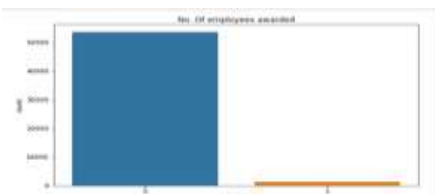


Insights :

1. More male met the KPIs score i.e. greater than 80%
2. Count of Bachelors employees who scored more than 80% and get promoted is higher than other education degree. But % of Masters degree has the highest among all four degrees.

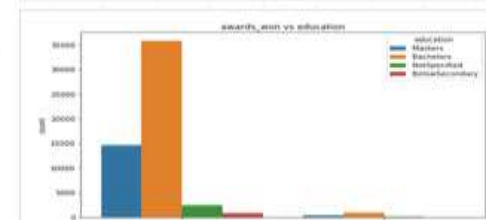
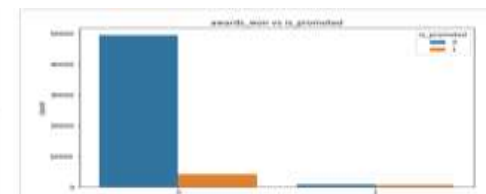


## Awards Won VS Promotion



Insights :

1. Male employee have awarded more than female employee.
2. Bachelors employee have awarded more than other education.





## • CROSSTAB BETWEEN NUMERICAL AND CATEGORICAL

### Education Backgrounds of Departments:

```
pd.crosstab(employee['education'],employee['department'])
```

department	Analytics	Finance	HR	Legal	Operations	Procurement	RandD	SalesMarketing	Technology
education									
Bachelors	3978	1895	1525	814	7781	4393	542	11099	4642
BelowSecondary	0	106	128	65	176	129	0	0	201
Masters	1037	499	733	156	3165	2544	429	4166	2196
NotSpecified	337	36	32	4	226	72	28	1575	99

Inference:

1. Shocking that dept like finance, HR, legal, operations, procurement and technology have education background below secondary.
2. R&D dept has bag of bachelors and masters.
3. Every dept has a greater number of bachelors.

### Recruitment Channels of Departments:

```
pd.crosstab(employee['recruitment_channel'],employee['department'])
```

department	Analytics	Finance	HR	Legal	Operations	Procurement	RandD	SalesMarketing	Technology
recruitment_channel									
other	2973	1463	1380	590	6279	4002	555	9290	3914
referred	83	5	103	14	238	79	19	259	342
sourcing	2296	1068	935	435	4831	3057	425	7291	2882

1. Technology has most referrals than any other dept.
2. Analytics and R&D has mixed blend of others and sourcing as recruitment channels.
3. The recruitment channel others is the most preferred for recruitment.



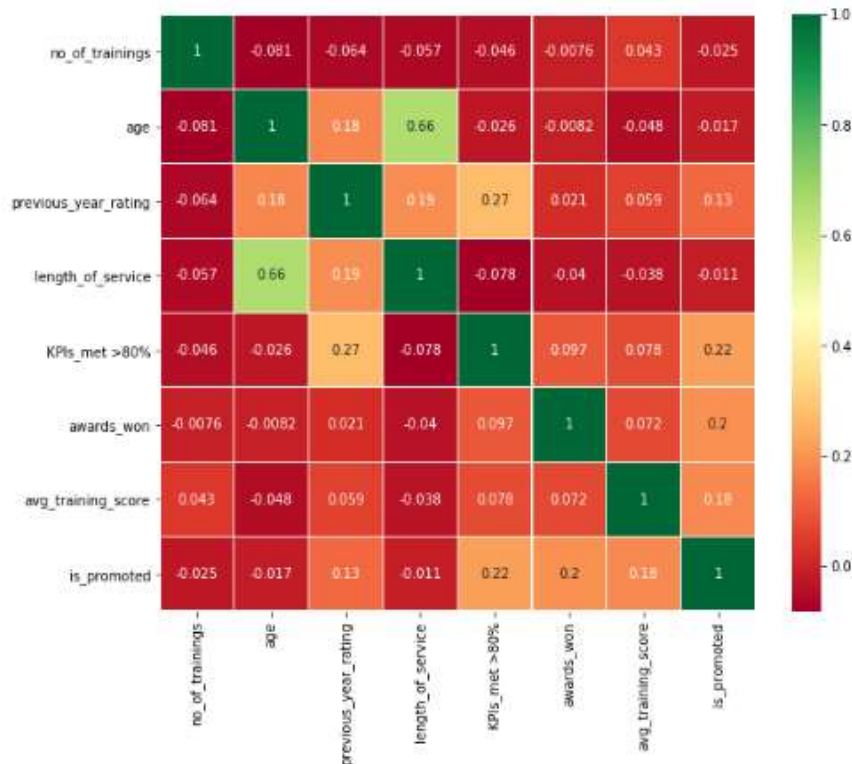
## Gender Ratio of Departments:

```
pd.crosstab(employee['gender'],employee['department'])
```

department	Analytics	Finance	HR	Legal	Operations	Procurement	RandD	SalesMarketing	Technology
gender									
f	513	681	1006	149	4677	3287	57	3154	2788
m	4839	1855	1412	890	6671	3851	942	13686	4350

1. Data is more tilted towards male by huge numbers.
2. Procurement department has the best gender equality(even when female crowd is less in huge numbers) and then HR.
3. The worst is sales & marketing dept., followed by analytics and R&D.

## Correlation Between The Features



### Insights :

Interpreting The Heatmap:

1. Only the **numeric features** are compared as it is obvious that we cannot correlate between alphabets or strings.

- **POSITIVE CORRELATION:** If an increase in feature A leads to increase in feature B, then they are positively correlated. A value 1 means perfect positive correlation.
- **NEGATIVE CORRELATION:** If an increase in feature A leads to decrease in feature B, then they are negatively correlated. A value -1 means perfect negative correlation.

2. Now lets say that two features are highly or perfectly correlated, so the increase in one leads to increase in the other. This means that both the features are containing highly similar information and there is very little or no variance in information. This is known as Multicollinearity as both of them contains almost the same information.

3. Now from the above heatmap, we can see that the features are not much correlated. The highest correlation is between Length of Service and Age i.e. 0.66. So we can carry on with all features

## CHAPTER -3

### 3. MODELLING ON DATASET

#### BaseLine Estimator for this model:

```
employee.is_promoted[employee.is_promoted==0].shape
(50140,)

employee.is_promoted[employee.is_promoted==1].shape
(4668,)

baseline = 50140/(50140+4668)
print(baseline)
```

**BaseLine Estimator is 91.4%**

#### Applying Dummies on the Dataset:

```
new_employee=pd.get_dummies(employee)
```

```
new_employee.columns
Index(['employee_id', 'no_of_trainings', 'age', 'previous_year_rating',
      'length_of_service', 'KPIs_met >80%', 'awards_won',
      'avg_training_score', 'is_promoted', 'department_Analytics',
      'department_Finance', 'department_HR', 'department_Legal',
      'department_Operations', 'department_Procurement', 'department_RandD',
      'department_SalesMarketing', 'department_Technology', 'region_region_1',
      'region_region_10', 'region_region_11', 'region_region_12',
      'region_region_13', 'region_region_14', 'region_region_15',
      'region_region_16', 'region_region_17', 'region_region_18',
      'region_region_19', 'region_region_2', 'region_region_20',
      'region_region_21', 'region_region_22', 'region_region_23',
      'region_region_24', 'region_region_25', 'region_region_26',
      'region_region_27', 'region_region_28', 'region_region_29',
      'region_region_3', 'region_region_30', 'region_region_31',
      'region_region_32', 'region_region_33', 'region_region_34',
      'region_region_4', 'region_region_5', 'region_region_6',
      'region_region_7', 'region_region_8', 'region_region_9',
      'education_Bachelors', 'education_BelowSecondary', 'education_Masters',
      'education_NotSpecified', 'gender_f', 'gender_m',
      'recruitment_channel_other', 'recruitment_channel_referred',
      'recruitment_channel_sourcing'],
      dtype='object')
```

## Applying Train Test Split on the independent variables and dependent variables:

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.linear_model import LogisticRegression
```

```
x = new_employee.drop(['employee_id','is_promoted'],axis=1)
```

```
y = new_employee['is_promoted']
```

```
x.columns
```

```
xtrain, xtest , ytrain , ytest = train_test_split(x,y,test_size = 0.3,random_state = 2)
```

```
Index(['no_of_trainings', 'age', 'previous_year_rating', 'length_of_service',
      'KPIs_met >80%', 'awards_won', 'avg_training_score',
      'department_Analytics', 'department_Finance', 'department_HR',
      'department_Legal', 'department_Operations', 'department_Procurement',
      'department_RandD', 'department_SalesMarketing',
      'department_Technology', 'region_region_1', 'region_region_10',
      'region_region_11', 'region_region_12', 'region_region_13',
      'region_region_14', 'region_region_15', 'region_region_16',
      'region_region_17', 'region_region_18', 'region_region_19',
      'region_region_2', 'region_region_20', 'region_region_21',
      'region_region_22', 'region_region_23', 'region_region_24',
      'region_region_25', 'region_region_26', 'region_region_27',
      'region_region_28', 'region_region_29', 'region_region_3',
      'region_region_30', 'region_region_31', 'region_region_32',
      'region_region_33', 'region_region_34', 'region_region_4',
      'region_region_5', 'region_region_6', 'region_region_7',
      'region_region_8', 'region_region_9', 'education_Bachelors',
      'education_BelowSecondary', 'education_Masters',
      'education_NotSpecified', 'gender_f', 'gender_m',
      'recruitment_channel_other', 'recruitment_channel_referred',
      'recruitment_channel_sourcing'],
      dtype='object')
```

```
model_1 = LogisticRegression()
```

```
model_1.fit(xtrain,ytrain)
```

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
verbose=0, warm_start=False)
```

```
from sklearn.metrics import accuracy_score,r2_score
from sklearn import metrics
```

```
ypred = model_1.predict(xtest)
metrics.accuracy_score(ytest,ypred)
```

```
0.9315818281335523
```

```
metrics.confusion_matrix(ytest,ypred)
```

```
array([[14975,   60],
       [ 1065,  343]], dtype=int64)
```

```
cr = metrics.classification_report(ytest,ypred)
print(cr)
```

	precision	recall	f1-score	support
0	0.93	1.00	0.96	15035
1	0.85	0.24	0.38	1408
avg / total	0.93	0.93	0.91	16443

**Inference :** Here we can see that our model accuracy is 93% which is better than baseline estimator

## Scaling the Dataset:

```
from sklearn import preprocessing

employee_std = new_employee.drop(['is_promoted'],axis=1)

a_scaled = preprocessing.scale(employee_std)
b = pd.DataFrame(a_scaled,columns= employee_std.columns)

Xtrain,Xtest,Ytrain,Ytest = train_test_split(X,Y,test_size = 0.3,random_state =2)

modelimp = LogisticRegression()
modelimp.fit(Xtrain,Ytrain)
```

↓

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
verbose=0, warm_start=False)
```

```
Y_Pred = modelimp.predict(Xtest)
```

```
Y_Pred
```

→ array([0, 0, 0, ..., 0, 0, 0], dtype=int64)

```
CR = metrics.classification_report(Ytest,Y_Pred)
print(CR)
```

↓

	precision	recall	f1-score	support
0	0.94	0.99	0.96	15035
1	0.81	0.26	0.40	1408
avg / total	0.92	0.93	0.92	16443

Now we will consider best model to our data not only with the basis of Accuracy. We will check the Bias and Variance Error in the model:

```
models = []
models.append(('DecisionTree', Dt_model))
models.append(('RandomForest', Rf_model))
models.append(('Base_KNN', knn))
models.append(('Base_LR', modelimp))
models.append(('Base_NB', NB))
```

```
from sklearn import model_selection
results = []
names = []
sho3 = {}
scoring = 'recall'
for name, model in models:
    kfold = model_selection.KFold(n_splits=10, random_state=2)
    cv_results = model_selection.cross_val_score(model, x, y, cv=kfold, scoring=scoring)
    results.append(cv_results)
    names.append(name)
    msg = "%s: %f (%f)" % (name, np.mean(1-cv_results), cv_results.var())
    print(msg)
    sho3.update({name+' Bias Error':round(np.mean(1-cv_results),6),name+' Variance Error':round(cv_results.var(),6)})
# boxplot algorithm comparison
fig = plt.figure()
fig.suptitle('Algorithm Comparison')
ax = fig.add_subplot(111)
plt.boxplot(results)
ax.set_xticklabels(names)
plt.show()
sorted([(value,key) for (key,value) in sho3.items()])
```

## Results

```
[(0.000142, 'RandomForest Variance Error'),
 (0.000163, 'Base_KNN Variance Error'),
 (0.000233, 'DecisionTree Variance Error'),
 (0.000277, 'Base_LR Variance Error'),
 (0.001013, 'Base_NB Variance Error'),
 (0.241024, 'Base_NB Bias Error'),
 (0.673646, 'DecisionTree Bias Error'),
 (0.751702, 'Base_LR Bias Error'),
 (0.858807, 'Base_KNN Bias Error'),
 (0.89591, 'RandomForest Bias Error')]
```

Decision Tree is performing well in the Employee Performance Prediction so predicting test data using Decision tree as trade off Bias and Variance Error is less than others



## Applying Label Encoding On Nominal Variables:

### Input :

```
nominal = ['department', 'region', 'gender', 'recruitment_channel']
```

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
# department
encode1 = pd.DataFrame(le.fit_transform(employee['department']))
encode1.head()
encode1.columns = ['department']
```

```
# region
encode2 = pd.DataFrame(le.fit_transform(employee['region']))
encode2.columns = ['region']
# gender
encode3 = pd.DataFrame(le.fit_transform(employee['gender']))
encode3.columns = ['gender']
# recruitment_channel
encode4 = pd.DataFrame(le.fit_transform(employee['recruitment_channel']))
encode4.columns = ['recruitment_channel']
# dummy variables for education variable
dum1 = pd.get_dummies(employee['education'])
#dum1 = dum1.drop(['recruitment_channel'],1)
# concatenate
end = pd.concat([encode1, encode2, encode3, encode4, dum1], 1)
end.head()
```

### Output :

	department	region	gender	recruitment_channel	Bachelors	BelowSecondary	Masters	NotSpecified
0	7	31	0	2	0	0	1	0
1	4	14	1	0	1	0	0	0
2	7	10	1	2	1	0	0	0
3	7	15	1	0	1	0	0	0
4	8	18	1	0	1	0	0	0



## Applying concatenate on the Variables:

### Input :

```
train2 = employee.drop(['department','region','gender','recruitment_channel','education'],1)
train2 = pd.concat([end,train2],1)
train2.info()
```

### Output :

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 54808 entries, 0 to 54807
Data columns (total 16 columns):
department      54808 non-null int32
region          54808 non-null int32
gender          54808 non-null int32
recruitment_channel  54808 non-null int32
Bachelors       54808 non-null uint8
BelowSecondary  54808 non-null uint8
Masters         54808 non-null uint8
NotSpecified    54808 non-null uint8
no_of_trainings 54808 non-null int64
age             54808 non-null int64
previous_year_rating 54808 non-null int64
length_of_service 54808 non-null int64
KPIs_met >80%    54808 non-null int64
awards_won      54808 non-null int64
avg_training_score 54808 non-null int64
is_promoted     54808 non-null int64
dtypes: int32(4), int64(8), uint8(4)
memory usage: 4.4 MB
```

## Model Preparation:

### Applying SMOTE Technique:

```
from imblearn import under_sampling, over_sampling
from imblearn.over_sampling import SMOTE
```

```
X = train2.drop(['is_promoted'],1)
y = train2['is_promoted']
```

## Input :

```
print('Shape of X: {}'.format(X.shape))
print('Shape of y: {}'.format(y.shape))
```

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=2)

print("Number transactions X_train dataset: ", X_train.shape)
print("Number transactions y_train dataset: ", y_train.shape)
print("Number transactions X_test dataset: ", X_test.shape)
print("Number transactions y_test dataset: ", y_test.shape)
```

## Output:

```
Shape of X: (54808, 15)
Shape of y: (54808,)
```

```
Number transactions X_train dataset: (38365, 15)
Number transactions y_train dataset: (38365,)
Number transactions X_test dataset: (16443, 15)
Number transactions y_test dataset: (16443,)
```

**Before and After Applying Oversampling:**

```
Before OverSampling, counts of label in train '1': 3260  
Before OverSampling, counts of label in train '0': 35105
```

```
Before OverSampling, counts of label in test '1': 1408  
Before OverSampling, counts of label i test '0': 15035
```

```
After OverSampling, the shape of train_X: (70210, 15)  
After OverSampling, the shape of train_y: (70210,)
```

```
After OverSampling, counts of label in train '1': 35105  
After OverSampling, counts of label in train '0': 35105
```

```
After OverSampling, the shape of test_X: (30070, 15)  
After OverSampling, the shape of test_y: (30070,)
```

```
After OverSampling, counts of label in test '1': 15035  
After OverSampling, counts of label in test '0': 15035
```

## Random Forest Classifier on Training Accuracy:

### Library to be exported

```
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
```

### INPUT :

```
est = range(100,200,20)
train_f1 = []
for i in est:
    rf = RandomForestClassifier(n_estimators = i,random_state = 3)
    rf.fit(X_train_res,y_train_res)
    y_pred = rf.predict(X_train_res)
    print('Training Accuracy Score:',metrics.accuracy_score(y_train_res,y_pred))
    print('Training F1 Score is:',metrics.f1_score(y_train_res,y_pred))
    train_f1.append(metrics.f1_score(y_train_res,y_pred))
```

### OUTPUT :

```
Training Accuracy Score: 0.9996296823814271
Training F1 Score is: 0.9996297983825038
Training Accuracy Score: 0.9996296823814271
Training F1 Score is: 0.9996297983825038
Training Accuracy Score: 0.9996296823814271
Training F1 Score is: 0.9996297983825038
Training Accuracy Score: 0.9996296823814271
Training F1 Score is: 0.9996297772968048
Training Accuracy Score: 0.9996296823814271
Training F1 Score is: 0.9996297772968048
```

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train_res)
X_test_scaled = scaler.fit_transform(X_test_res)

X_train_scaled = pd.DataFrame(X_train_scaled)
X_train_scaled.columns = X_train.columns

X_test_scaled = pd.DataFrame(X_test_scaled)
X_test_scaled.columns = X_test.columns
```

### Input :

```
rf2 = RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                             max_depth=10, max_features='auto', max_leaf_nodes=None,
                             min_impurity_decrease=0.0, min_impurity_split=None,
                             min_samples_leaf=1, min_samples_split=2,
                             min_weight_fraction_leaf=0.0, n_estimators=160, n_jobs=None,
                             oob_score=False, random_state=5, verbose=0, warm_start=False)
rf2.fit(X_train_scaled,y_train_res)
y_pred2 = rf2.predict(X_test_scaled)
print('F1 Score is:',metrics.f1_score(y_test_res,y_pred2))
```

### Output :

F1 Score is: 0.8258243390434696

#### Insights:

Scaling has not significant effect on accuracy and f1 score of the random forest.