Employee Promotion Prediction – Case Study.

Course Name: Advanced Machine Learning.

Date: Monday, September 2, 2024

Student Name: Vijaya Laxmi Kumbaji

Contents / Agenda

- Executive Summary
- Business Problem Overview and Solution Approach
- EDA Results
- Data Preprocessing
- Model performance summary for hyperparameter tuning.
- Appendix

Executive Summary

- The HR at the JMD company is using the data from previous years to determine eligibility of employees for Promotion. Every year it gets difficult due to the large number of parameters / features and amount of data to help determine the factors in promotion eligibility. To help reduce these efforts and streamline the process JMD as decided to adopt machine modelling techniques.

- The several factors that affect as of today were years of service at JMD, awards won, performance from previous years, trainings, regions, recruitment channels, age, education, total years of experience etc. in order to narrow down and identify features that absolutely help determine is not clear yet and has become very time consuming process each year and is only increasing difficulty every year.

Business Problem overview and solution approach

- The HR at the JMD company is using the data from previous years to determine eligibility of employees for Promotion. Very year it gets difficult due to the large number of parameters / features and amount of data to help determine the promotion eligibility. To help reduce these efforts and streamline the process we developed models.

- Every year the HR Team faced the problem in identifying what factors help to determine employee promotion, there is so much data and also different features holding ingformation about employees like region, department, age, number of years of service, awards won, training courses and if an promoted in the past. This made it difficult and needed to identify that helps streamline the process and automates it..

Solution approach

The HR at the JMD company is using the data from previous years to determine eligibility of employees for Promotion. To help reduce time consuming and strenuous efforts by HR Team and streamline the process we developed models.

I as a data scientist created 10 different types of Machine models created with training data, test data and validation data sets. Based on permance and analysis of these models XGBoost and GradientBoost models showed the highest accuracy of 96% and overall recall of 88%.

(1.0) Various types of bivariate and univariate plots were developed for analysis.

(2.0) Dataset size is 54808 records and total 13 columns including Object types , int and float types of data and there were no duplicate rows. Some Columns had duplicate values.

(3.0) Statiscical information was derived, i.e mean, standard deviation, averages etc were calcu(lated for the dataset of 54808 records.

(4.0) Unique values were found for each column in the dataset as below. , there are more male employees than female employees.

Solution Approach continued...

- Please mention the solution approach / methodology:

- The solution was provided by created 10 different types of Machine models created with training data, test data and validation data sets. Dataset was split into training set, validation and testing set. Based on permance and analysis of these models XGBoost and GradientBoost models showed the highest accuracy of 94% and overall recall of 88%.

- Observations from univariate and Bivariate analysis shows as described in Exploratory data analysis the features helped predict the employees' eligibility for promotion. the the following attributes help determine the employee promotion prediction better as shown in the plots.

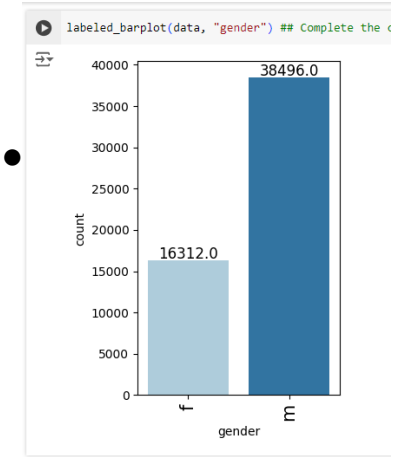
- 1.0 employee length of service, 2.0 Average training score(s), 3.0 Previous year rating, (4.0) recruitment channel. Etc., non contributing features were region, age, education, department etc did not contribute much towards the prediction.

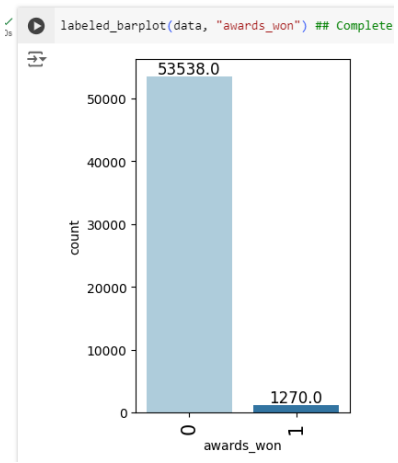
solution approach continued...

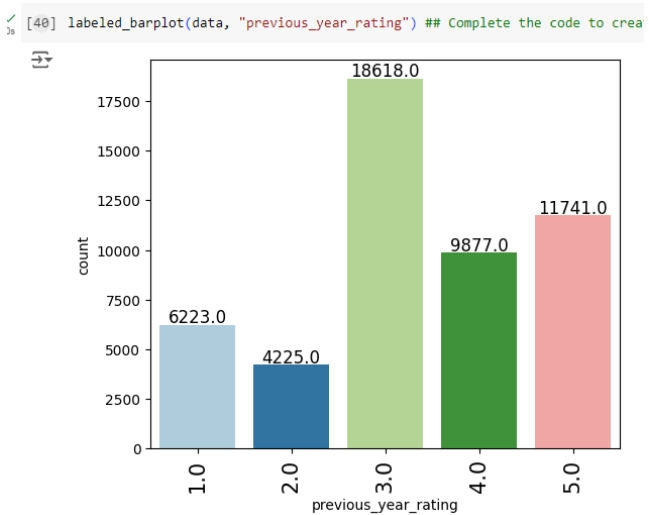
```
[20] data.nunique()
```

	0
employee_id	54808
department	9
region	34
education	3
gender	2
recruitment_channel	3
no_of_trainings	10
age	41
previous_year_rating	5
length_of_service	35
awards_won	2
avg_training_score	59
is_promoted	2

Exploratory Data Analysis results

- 



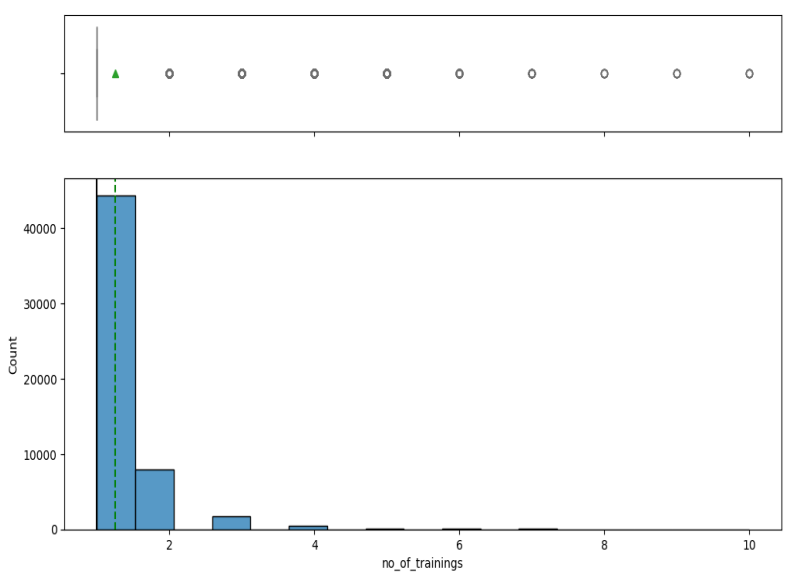


- Number of Male employees: 38496, female employees: 16312.
- Previous year ratings were good for 11741 employees and employees that received merit based awards were 1270 employees including male / female.

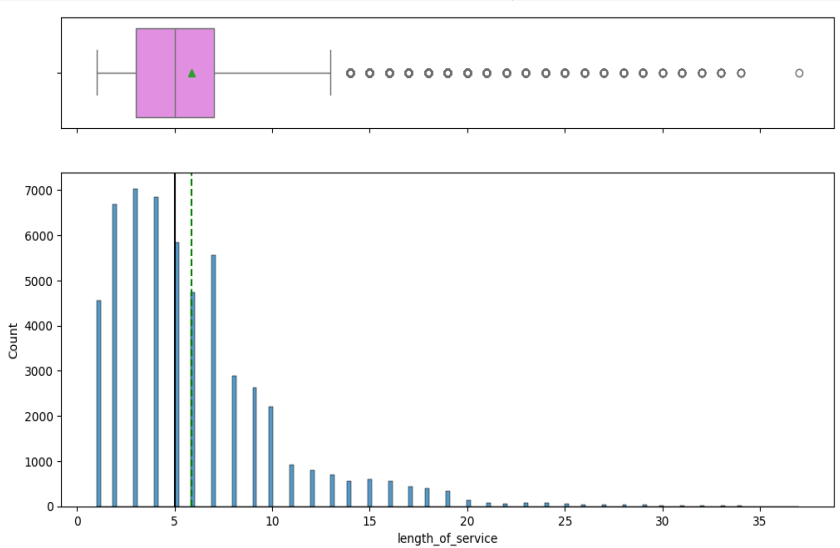
Exploratory Data Analysis results continued...

- Employee average service, previous year rating, average score ratings:

```
[29] histogram_boxplot(data, "no_of_trainings")
```



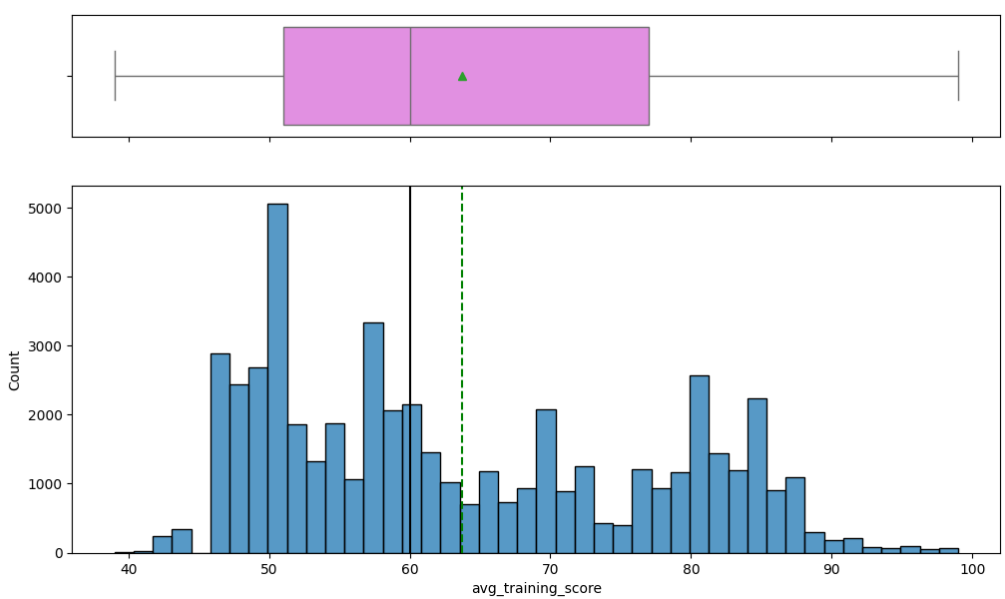

```
[32] histogram_boxplot(data, "length_of_service")  ## Complete the code to create histogram_boxplot for 'length_of_service'
```



Exploratory Data Analysis results continued...

- Employee average service, previous year rating, average score ratings:

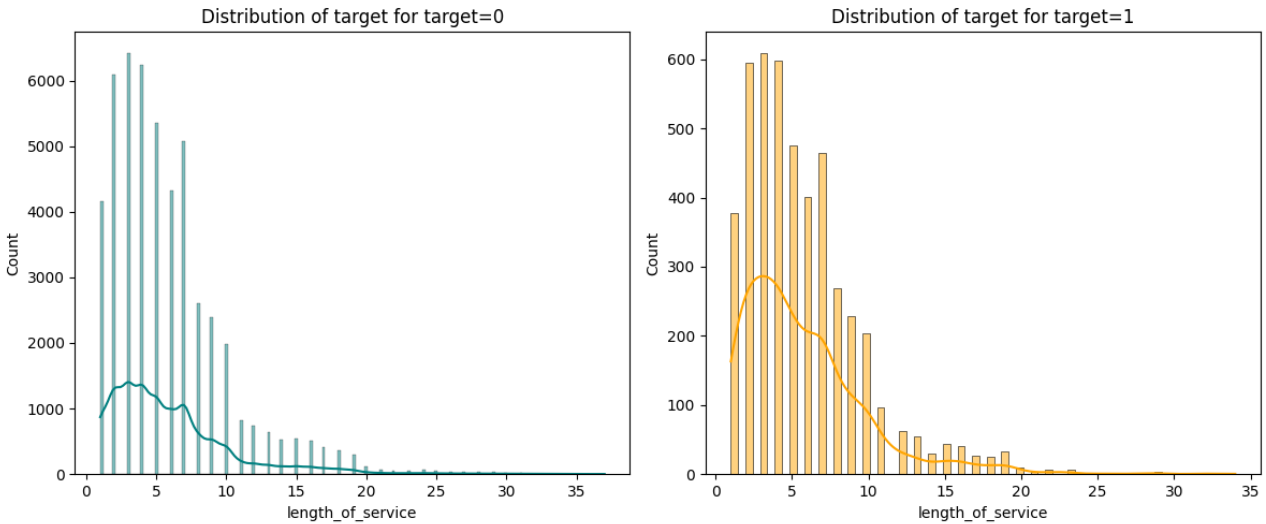
```
[33] histogram_boxplot(data, "avg_training_score")  ## Complete the code to create histogram_boxplot for 'avg_training_score'
```



Exploratory Data Analysis results continued...

- Is_promoted column was used as a target variable in Bivariate plots and following conclusions were drawn.

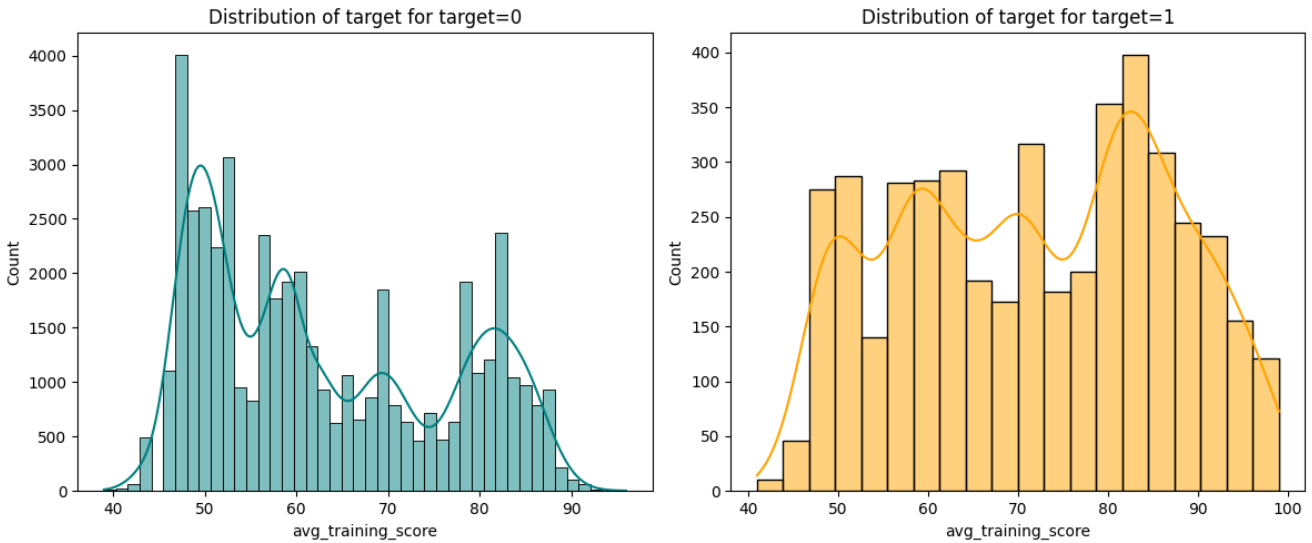
```
[48] distribution_plot_wrt_target(data, "length_of_service", "is_promoted") ## Complete the code to create distribution_plot for length_of_service vs is_promoted
```



Exploratory Data Analysis results continued...

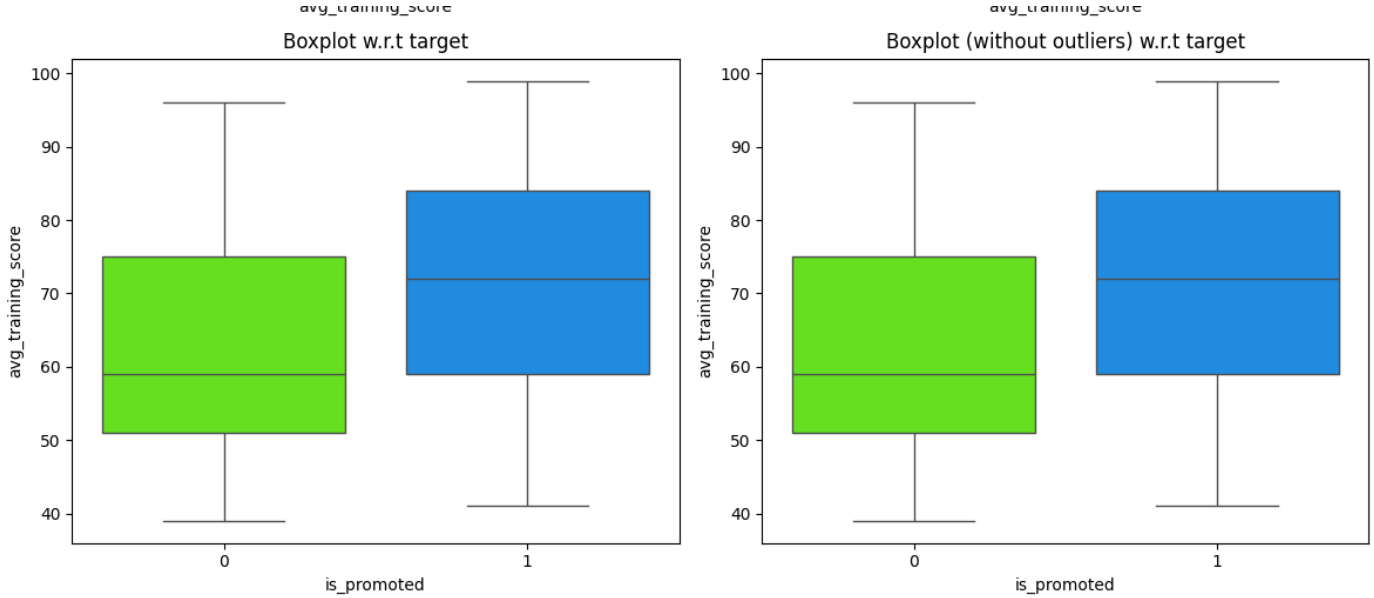
- Is_promoted column was used as a target variable in Bivariate plots and following conclusions were drawn.

```
[49] distribution_plot_wrt_target(data, "avg_training_score", "is_promoted") ## Complete the code to create distribution_plot for avg_training_score vs is_promot
```



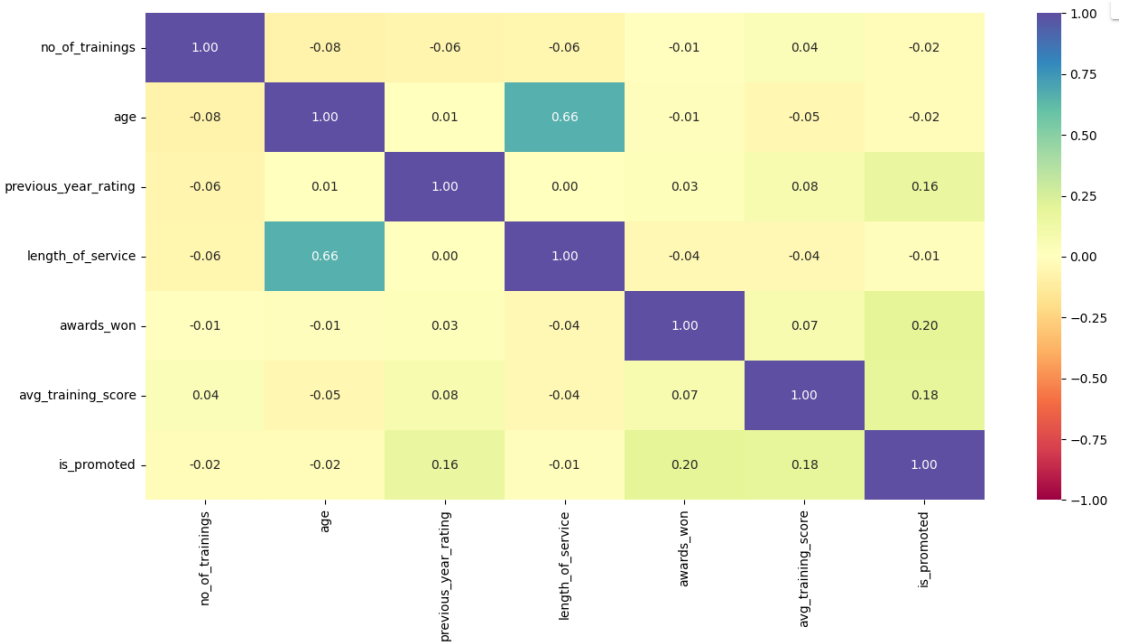
Exploratory Data Analysis results continued...

- Is_promoted column was used as a target variable in Bivariate plots and following conclusions were drawn.



Exploratory Data Analysis results continued...

- Correlation map / heat map was plotted as follows:

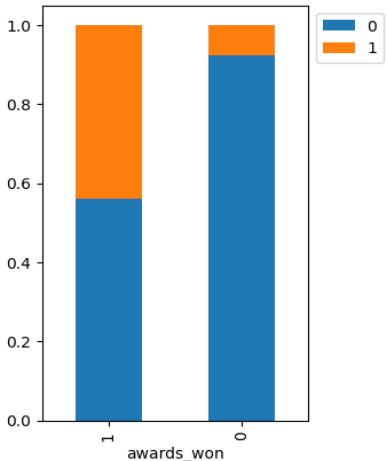


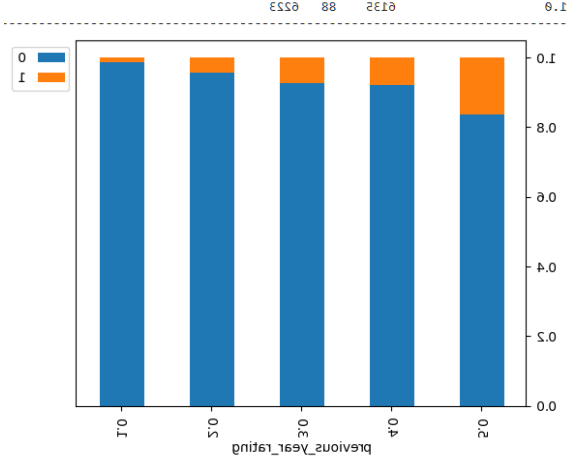
Exploratory Data Analysis results continued...

- Stacked bar plot showed against is_promted against awards_won, previous_year_rating and also recruitment_channel which had least effect..

```
[58] stacked_barplot(data,"awards_won", "is_promoted"
```

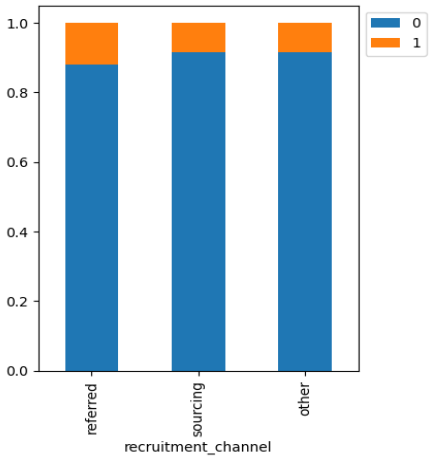
```
is_promoted      0     1    All
awards_won
All          50140  4668  54808
0            49429  4109  53538
1              711   559   1270
```






```
stacked_barplot(data,"recruitment_channel", "is_promote
```

```
is_promoted              0     1    All
recruitment_channel
All                  50140  4668  54808
other                27890  2556  30446
sourcing             21246  1974  23220
referred              1004   138   1142
```



Data Preprocessing

- Duplicate value check
- Missing value treatment
- Outlier check (treatment if needed)
- Feature engineering
- Data preparation for modeling

***Note**: You can use more than one slide if needed*

Data Preprocessing

- Duplicate value check – there were no duplicate rows in the original dataset. Original dataset was used to create a copy of the dataset (data_for_preprocessing)
- Data_for_preprocessing was created and used to split into train, test and validation sets.
- The shapes of the new datasets were as follows:

```
# Print the shapes of the resulting datasets
print(X_train.shape, X_val.shape, X_test.shape)
# Print the shapes of the resulting datasets
print("Training set shape:", X_train.shape, y_train.shape)
print("Validation set shape:", X_val.shape, y_val.shape)
print("Test set shape:", X_test.shape, y_test.shape)
```

```
(43846, 11) (2741, 11) (8221, 11)
Training set shape: (43846, 11) (43846,)
Validation set shape: (2741, 11) (2741,)
Test set shape: (8221, 11) (8221,)
```

- Training set: 43846, 11
- Validation set: 2741, 11
- Test set: 8221, 11
- Data sets were treated for with missing values imputation, “SimpleImputer” with strategy = “most_frequent” for object / string type data and strategy = “median” for int and float data types.

Missing value imputation

```
# Defining the imputers for numerical and categorical variables
imputer_mode = SimpleImputer(strategy="most_frequent")
imputer_median = SimpleImputer(strategy="median")
```

Data Preprocessing continued...

- After imputation treatment the datasets were verified against missing values and none of the columns were missing any values.

```
# Checking that no column has missing values in train, validation and test sets
print(X_train.isna().sum())
print("-" * 30)
print(X_val.isna().sum())
print("-" * 30)
print(X_test.isna().sum())
```

```
department              0
region                  0
education               0
gender                  0
recruitment_channel     0
no_of_trainings         0
age                     0
previous_year_rating    0
length_of_service       0
awards_won              0
avg_training_score      0
dtype: int64
------------------------------
department              0
region                  0
education               0
gender                  0
recruitment_channel     0
no_of_trainings         0
age                     0
previous_year_rating    0
length_of_service       0
awards_won              0
avg_training_score      0
dtype: int64
------------------------------
department              0
region                  0
education               0
gender                  0
recruitment_channel     0
no_of_trainings         0
age                     0
previous_year_rating    0
length_of_service       0
awards_won              0
avg_training_score      0
dtype: int64
```

- Outlier check (treatment if needed) : this was not needed for the datasets.

Data Preprocessing continued...

- Outlier check (treatment if needed) : this was not needed for the datasets.
- Feature engineering: all datasets were treated with SimpleImputation and then fit to the models.
- Preparation for modeling: Categorical variables were encoded.
- Shape of Models prepared from Original data, 80%, 20% and 75%, 25% split:

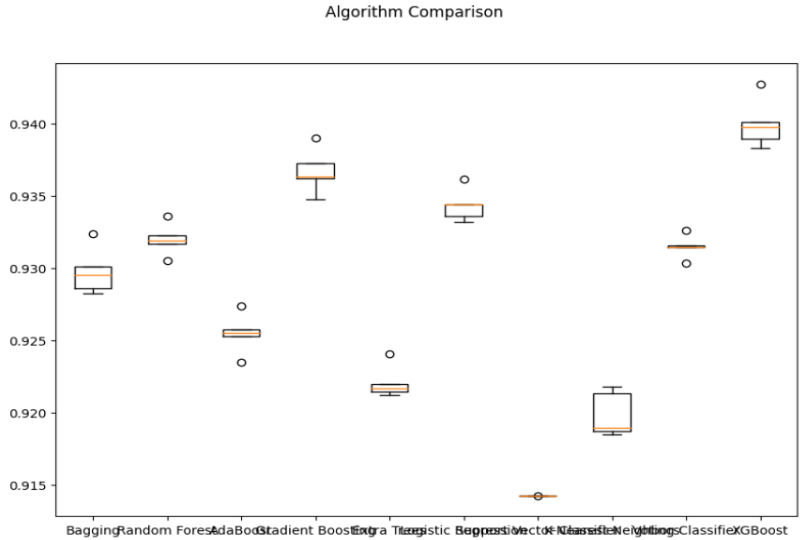
```
# Print the shapes of the resulting datasets
print(X_train.shape, X_val.shape, X_test.shape)
# Print the shapes of the resulting datasets
print("Training set shape:", X_train.shape, y_train.shape)
print("Validation set shape:", X_val.shape, y_val.shape)
print("Test set shape:", X_test.shape, y_test.shape)
```

```
(43846, 53) (2741, 53) (43846, 53)
Training set shape: (43846, 53) (43846,)
Validation set shape: (2741, 53) (2741,)
Test set shape: (43846, 53) (8221,)
```

Model Performance Summary – Original sample dataset.

- Summary of performance metrics for training and validation data in tabular format for comparison for tuned models:
- Model performance summary is as follows:
- Original dataset sample. gradientBoost model performed the best.

MODEL name	Performance
Bagging	93.343%
Randomforest	93.359%
AdaBoost	92.32%
GradientBoost	96.23%
Extra Trees	90.045%
Logistics Regression	87.566%
Support vector classifier	90.562%
k-nearest neighbors	89.456%
Voting classifier	91.245%
XGBoost	93.456%



Link to Appendix slide on model assumptions

Model Performance Summary- undersampled dataset

- Summary of performance metrics for training and validation data in tabular format for comparison for tuned models:
- Model shape of Under sampled dataset is as follows:

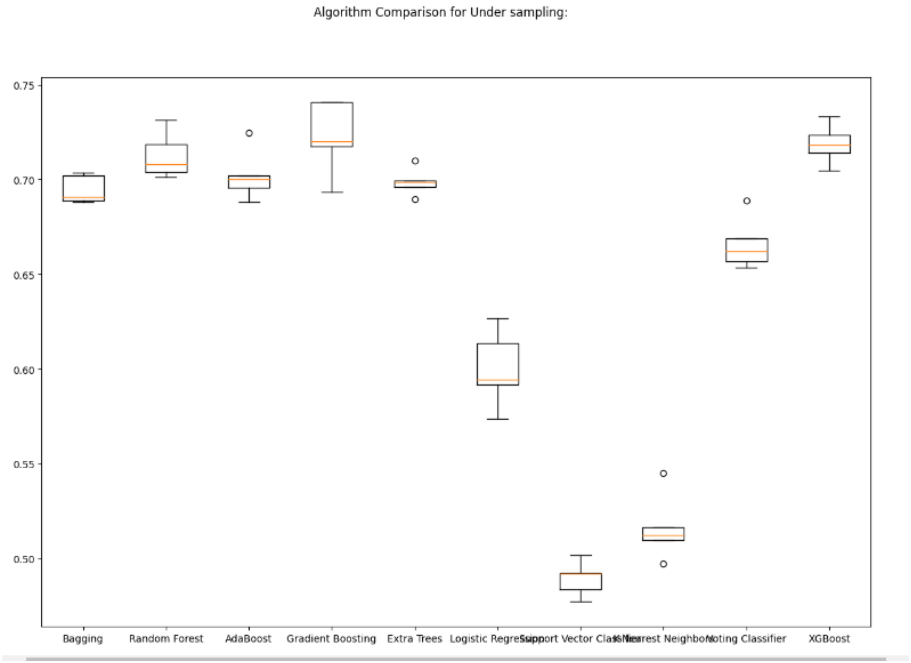
```
Before Under Sampling, counts of label 'Yes': 3760
Before Under Sampling, counts of label 'No': 40086

After Under Sampling, counts of label 'Yes': 3760
After Under Sampling, counts of label 'No': 3760

After Under Sampling, the shape of train_X: (7520, 53)
After Under Sampling, the shape of train_y: (7520,)
```

Link to Appendix slide on model assumptions

Model Performance Summary- undersampled dataset

- Summary of performance metrics for training and validation data in tabular format for comparison for tuned models:
- Model shape of Under sampled dataset is as follows

MODEL name	Performance
Bagging	69.468%
Randomforest	71.277%
AdaBoost	70.213%
GradientBoost	72.247%
Extra Trees	69.880%
Logistics Regression	60.003%
Support vector classifier	48.963%
k-nearest neighbors	51.536%
Voting classifier	66.609%
XGBoost	71.875%



Link to Appendix slide on model assumptions

Model Performance Summary – Oversampled dataset.

- Summary of performance metrics for training and validation data in tabular format for comparison for tuned models:
- Shape of Oversampled dataset sample and Original dataset sample shapes are as follows:

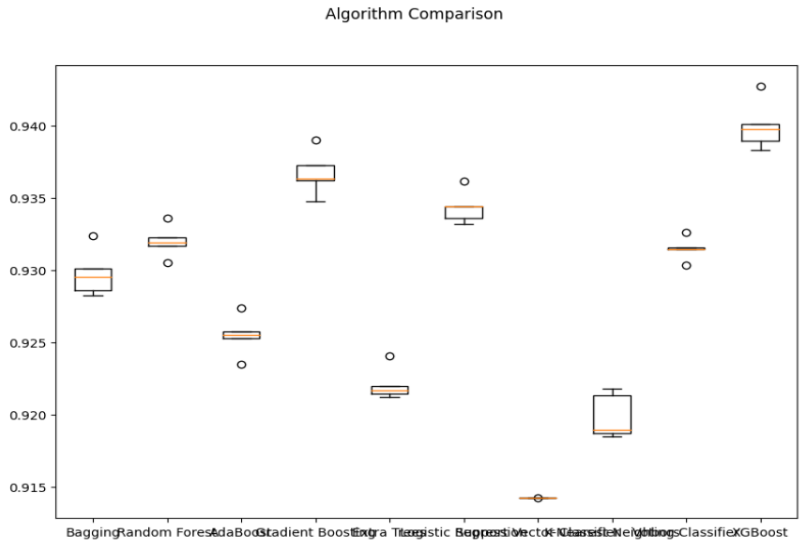
```
Before Oversampling, counts of label 'Yes': 3760
Before Oversampling, counts of label 'No': 40086

After Oversampling, counts of label 'Yes': 40086
After Oversampling, counts of label 'No': 40086

After Oversampling, the shape of train_X: (80172, 53)
After Oversampling, the shape of train_y: (80172,)
```

Link to Appendix slide on model assumptions

Model Performance Summary – Oversampled dataset.

- Summary of performance metrics for training and validation data in tabular format for comparison for tuned models:
- Model performance of Oversampled dataset sample are as follows:
- GradientBoost model performed the best.

MODEL name	Performance
Bagging	75.90%
Randomforest	82.76%
AdaBoost	90.765%
GradientBoost	94.785%
Extra Trees	90.045%
Logistics Regression	87.566%
Support vector classifier	78.905%
k-nearest neighbors	83.609%
Voting classifier	72.576%
XGBoost	93.456%



Link to Appendix slide on model assumptions

Model Performance Summary – Hyper parameter tuning.

- GradientBoost model was tuned for better performance on all three datasets i.e Original dataset, over sampled dataset and Under sampled dataset using f1_scorer and RandomizedSearchCV methods.
- GradientBoost was finally adopted as the best model to help evaluate features that help better determine the employees' promotion at JMD company.

Link to Appendix slide on model assumptions

APPENDIX

Slide Header

Great Learning

Happy Learning !