Data Collection and Preprocessing Phase

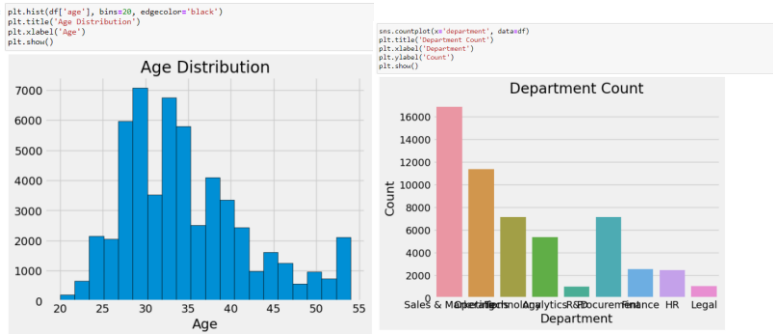
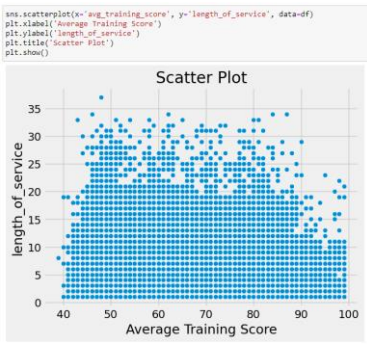
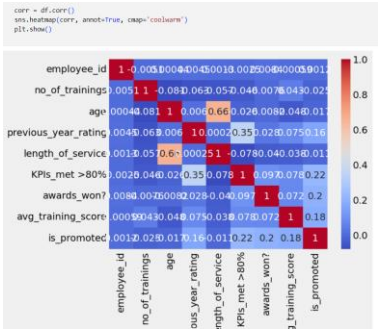
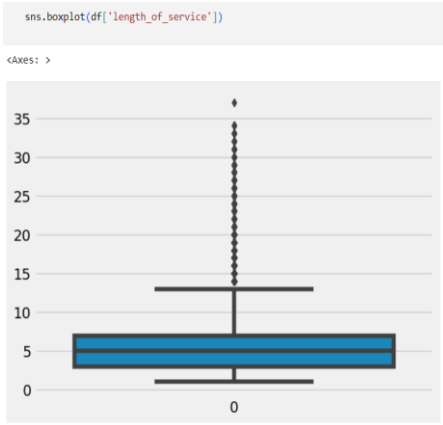
Date	09 July 2024
Team ID	SWTID1720455879
Project Title	Human Resource Management: Predicting Employee Promotions Using Machine Learning
Maximum Marks	6 Marks

Data Exploration and Preprocessing Template

Identifies data sources, assesses quality issues like missing values and duplicates, and implements resolution plans to ensure accurate and reliable analysis.

Section	Description
Data Overview	`df.describe()` |	employee_id	no_of_trainings	age	previous_year_rating	length_of_service	KPI
---	---	---	---	---	---	---
count	54808.000000	54808.000000	54808.000000	50684.000000	54808.000000	54808.(
mean	39195.830627	1.253011	34.803915	3.329256	5.865512	0.3
std	22586.581449	0.609264	7.660169	1.259993	4.265094	0.4
min	1.000000	1.000000	20.000000	1.000000	1.000000	0.(
25%	19669.750000	1.000000	29.000000	3.000000	3.000000	0.(
50%	39225.500000	1.000000	33.000000	3.000000	5.000000	0.(
75%	58730.500000	1.000000	39.000000	4.000000	7.000000	1.(

max	78298.000000	10.000000	60.000000	5.000000	37.000000	1.(	
							
							`df.shape`
							
							`(54808, 14)`

Univariate Analysis	
Bivariate Analysis	
Multivariate Analysis	
Outliers and Anomalies	

```
# Handle outliers with capping
numerical_cols = ['no_of_trainings', 'age', 'previous_year_rating', 'length_of_service', 'avg_training_score']
for col in numerical_cols:
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    df[col] = np.where(df[col] < lower_bound, lower_bound, df[col])
    df[col] = np.where(df[col] > upper_bound, upper_bound, df[col])
```

```
q1=np.quantile(df['length_of_service'],0.25)
q3=np.quantile(df['length_of_service'],0.75)
IQR= q3-q1
upper_bound=(1.5*IQR)+q3
lower_bound=(1.5*IQR)-q1
print("Skewed data:",len(df[df['length_of_service']>upper_bound]))
```

```
pd.crosstab([df['length_of_service']>upper_bound],df['is_promoted'])
```

```
df['length_of_service']=[upper_bound if x>upper_bound else x for x in df['length_of_service']]
```

```
pd.crosstab([df['length_of_service']<lower_bound],df['is_promoted'])
```

```
df['length_of_service']=[upper_bound if x<lower_bound else x for x in df['length_of_service']]
```

Data Preprocessing Code Screenshots

Loading Data

```
# Load your dataset
df = pd.read_csv("emp_promotion.csv")
df
```

Handling Missing Data

```
# Drop unwanted features
df = df.drop(['employee_id', 'region', 'gender', 'recruitment_channel'], axis=1)
df
```

```
print(df.isnull().sum())
```

```
print(df['education'].value_counts())
df['education'] = df['education'].fillna(df['education'].mode()[0])
```

```
print(df['previous_year_rating'].value_counts())
df['previous_year_rating'] = df['previous_year_rating'].fillna(df['previous_year_rating'].mode()[0])
```

```
print(df.isnull().sum())
```

Data Transformation

```
le = LabelEncoder()
df['department'] = le.fit_transform(df['department'])
df['education'] = le.fit_transform(df['education'])
df.head()
```

Handling Imbalanced Data

```
from imblearn.over_sampling import SMOTE
sm=SMOTE()
X_new,y_new=sm.fit_resample(X,y)
```

```
count_0 = np.count_nonzero(y_new == 0)
count_1 = np.count_nonzero(y_new== 1)

print(f"Number of 0s after sampling: {count_0}")
print(f"Number of 1s after sampling: {count_1}")
```

Splitting data

```
X_train,X_test,y_train,y_test=train_test_split(X_new,y_new,test_size=0.3,random_state=42)
```