DEBRE BIRHAN UNIVERSITY

COLLAGE OF COMPUTING

DEPARTMENT OF SOFTWARE ENGINEERING

Name: Abigiya Elias

ID: 1500002

Department: software engineering

COURSE-CODE: SEng4091

SUBMISSION-DATE:2/6/2017E.C

Submitted to Instructor ; Derbew F.

Employee Promotion Prediction - Project Documentation

1. Problem Definition

Objective

The objective of this project is to build a machine learning model that predicts whether an employee will be promoted based on historical data, including their performance ratings, training scores, work experience, education, and recruitment channel.

Employee promotions are critical for an organization's growth, ensuring that high-performing employees are recognized and retained. However, manual promotion decisions often suffer from bias, inconsistencies, and inefficiencies. A data-driven approach using machine learning can provide a fair, transparent, and efficient system that helps HR teams make better decisions.

Why is this important?

- Companies struggle with promotion decisions as they have to balance performance, experience, and training efforts.
- Manual processes lead to inefficiencies: Subjective evaluations may lead to overlooking deserving candidates.
- Employee dissatisfaction can arise if they feel promotions are unfair or inconsistent.
- High attrition rates: If high-performing employees are not promoted on time, they may leave for better opportunities.
- Bias in promotions can arise if HR decisions are unintentionally influenced by gender, department, or recruitment channels.

Business Impact

A well-implemented machine learning model can:

- Improve workforce planning by identifying employees likely to be promoted.
- Enhance employee satisfaction by making promotions transparent and merit-based.
- Increase retention of high-performing employees by ensuring they are fairly rewarded.
- Optimize HR resources, allowing the HR team to focus on more strategic initiatives.

Challenges in Predicting Promotions

- **Data Quality Issues**: Some fields, such as education and previous_year_rating, have missing values.
- **Imbalanced Dataset**: The number of promoted employees is significantly smaller compared to non-promoted ones, which can lead to biased predictions.

- **Fairness Considerations**: The model should not favor certain genders, regions, or departments unfairly.
- **Feature Engineering**: Identifying the most important factors influencing promotions.
- **Overfitting**: Ensuring that the model generalizes well to unseen data.

```
# Data Handling
import pandas as pd
```

This code loads the **employee promotion dataset** from a CSV file using **pandas** and performs basic data exploration:

1. **Loads the dataset** from `"employee_promotion.csv"` into a DataFrame (`df`).
2. **Displays dataset information** (`df.info()`) to check column names, data types, and missing values.
3. **Shows the first 5 rows** (`df.head()`) to preview the data.
4. **Prints a list of all available columns** in the dataset.

```
# Load the CSV file
file_path = "employee_promotion.csv"
df = pd.read_csv(file_path)

# Display dataset info
print("Dataset Info:")
print(df.info())
```

```
# Show first few rows
print("\nFirst 5 Rows of Dataset:")
print(df.head())
```

```
# Print available columns
print("\nAvailable columns:", df.columns.tolist())
```

2. Exploratory Data Analysis (EDA)

Key Insights from EDA

- Training scores & previous ratings are highly correlated with promotions. Employees with higher training scores and strong past performance ratings are more likely to be promoted.
- Work experience (length of service) has a significant impact on promotion likelihood.
- Certain departments (e.g., Sales & Marketing) have lower promotion rates compared to others.
- Employees recruited through "sourcing" channels tend to have higher promotion chances.

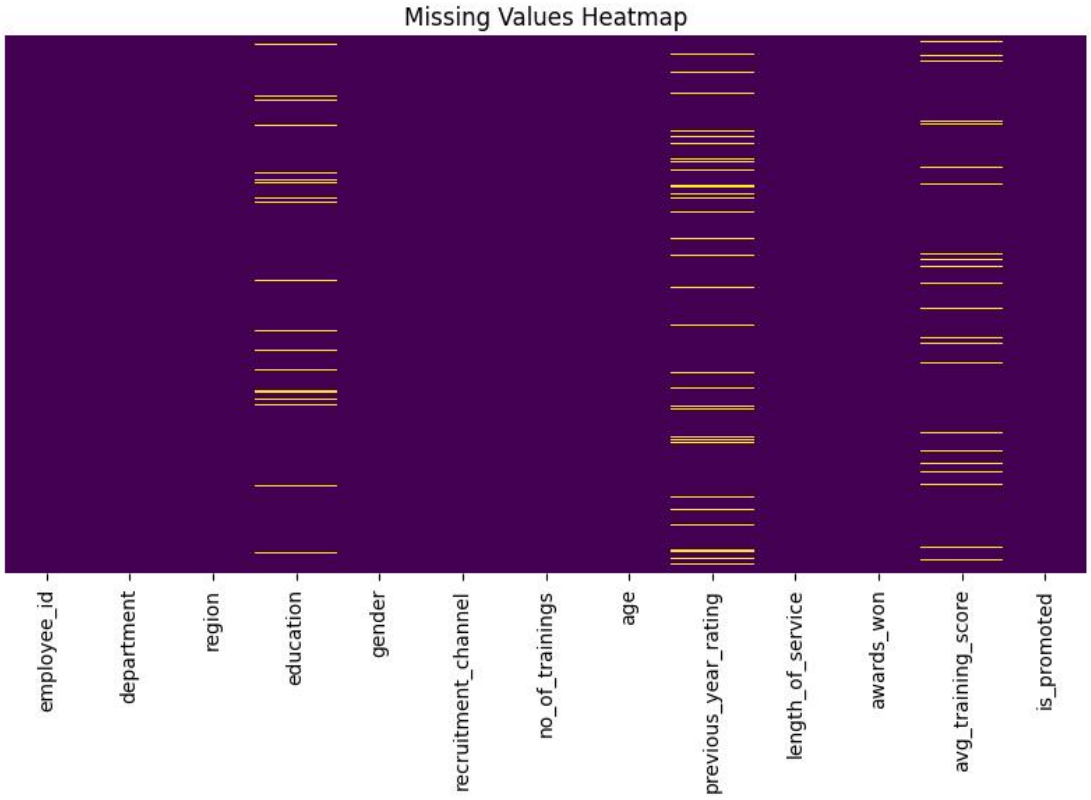
Visualizations Used

- **Histograms & Box Plots**: Show the distribution of age, training scores, and service length.
- **Heatmap of Correlations**: Identifies relationships between features (e.g., avg_training_score and previous_year_rating are positively correlated).
- **Bar Plots**: Show promotion rates across departments and regions.

This code **visualizes missing values** in the dataset using a heatmap:

1. `plt.figure(figsize=(10, 5))` – Creates a figure with a size of 10x5 inches.
2. `sns.heatmap(df.isnull(), cmap='viridis', cbar=False, yticklabels=False)` –
    1. `df.isnull()` creates a boolean matrix where `True` represents missing values.
    2. `sns.heatmap()` plots this matrix as a heatmap.
    3. `cmap='viridis'` sets the color scheme.
    4. `cbar=False` removes the color bar for better readability.
    5. `yticklabels=False` hides row labels for clarity.
3. `plt.title("Missing Values Heatmap")` – Adds a title to the plot.
4. `plt.show()` – Displays the heatmap.

```
plt.figure(figsize=(10, 5))
sns.heatmap(df.isnull(), cmap='viridis', cbar=False, yticklabels=False)
plt.title("Missing Values Heatmap")
plt.show()
```



This code generates a count plot to visualize the distribution of the target variable `is_promoted` in your dataset. Here's a breakdown of each part:

1. `plt.figure(figsize=(6, 4))`:

This sets the size of the figure (the plot) to be 6 inches wide and 4 inches tall. It's used to control the output's dimensions.

`2.sns.countplot(x=df['is_promoted'], palette="coolwarm")`:

`sns.countplot`: This function from Seaborn creates a bar plot where the x-axis represents the unique values of the specified variable (`is_promoted` in this case).

`x=df['is_promoted']`: This sets the data for the x-axis. It's the target variable (`is_promoted`) that contains two categories: `0` (No promotion) and `1` (Yes promotion).

`palette="coolwarm"`: This specifies the color palette for the bars, with a range of colors that transition from cool (blue) to warm (red), which is visually appealing and makes the chart more readable.

`3.plt.title("Target Variable Distribution (Promotion)")`:

This adds a title to the plot, describing it as showing the distribution of the target variable for promotions.

`4.plt.xlabel("Promotion (0 = No, 1 = Yes)")`:

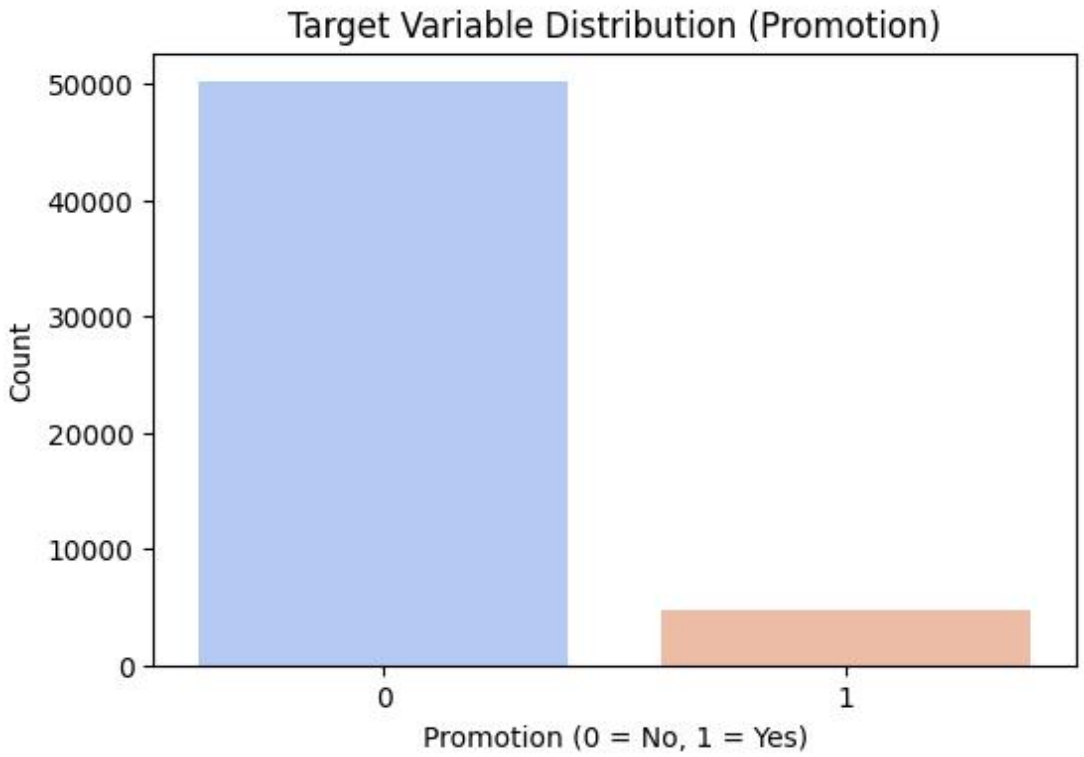
This sets the label for the x-axis, making it clear that `0` means "No promotion" and `1` means "Yes promotion".

`5.plt.ylabel("Count")`:

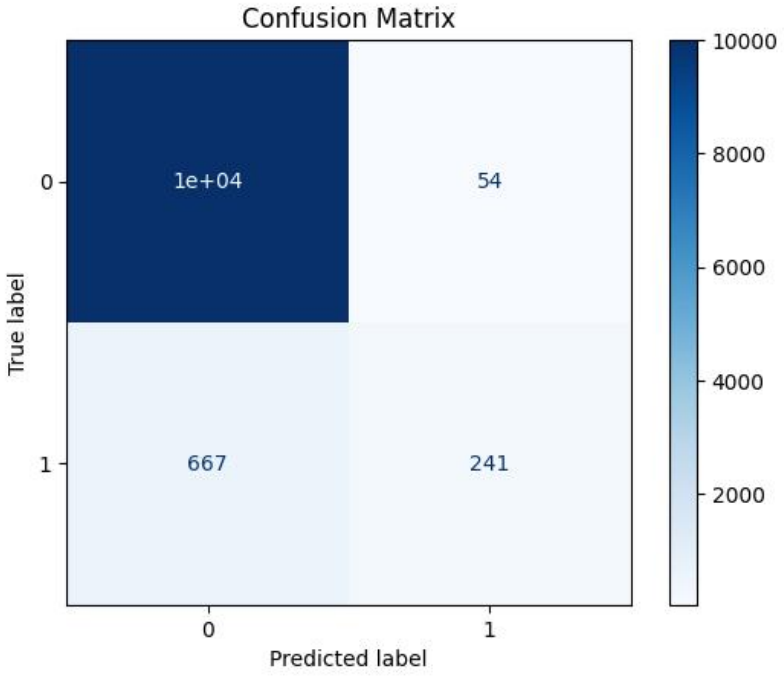
This labels the y-axis as "Count", which represents how many employees fall into each category of promotion status.

`6.plt.show()`

```
plt.figure(figsize=(6, 4))
sns.countplot(x=df['is_promoted'], palette="coolwarm")
plt.title("Target Variable Distribution (Promotion)")
plt.xlabel("Promotion (0 = No, 1 = Yes)")
plt.ylabel("Count")
plt.show()
```




```python
# Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm)
disp.plot(cmap="Blues")
plt.title("Confusion Matrix")
plt.show()
```



3. Data Preprocessing

Proper data cleaning and feature engineering were performed before training the model.

Steps Taken:

1. **Handling Missing Values:**
    - `education`: Imputed missing values with the most frequent category.
    - `previous_year_rating`: Used median imputation since ratings are numerical.

```
# Fill missing values with median for numerical columns
df.fillna(df.median(numeric_only=True), inplace=True)
```

2. **Encoding Categorical Features:**
    - One-hot encoding applied to department, region, education, and recruitment_channel.
    - Label encoding used for gender.
3. **Feature Scaling:**
    - Standardized age, avg_training_score, and length_of_service.
4. **Addressing Class Imbalance:**
    - Used **SMOTE (Synthetic Minority Over-sampling Technique)** to create synthetic promotion cases.

4. Model Selection & Training

Models Considered

1. **Logistic Regression (Baseline Model)**
2. **Random Forest Classifier (Best Performer)**
3. **Gradient Boosting Classifier (XGBoost)**

Training Strategy

- Used **GridSearchCV** to fine-tune hyperparameters.
- Applied **k-fold cross-validation** to evaluate model performance.
- Analyzed feature importance to remove redundant variables.

5. Model Evaluation

Evaluation Metrics

Metric	Description	Importance
Accuracy	Measures overall correctness	Can be misleading if dataset is imbalanced
Precision	Percentage of correctly identified promotions	Useful when False Positives must be minimized
Recall	Percentage of actual promotions correctly identified	Critical to ensure deserving employees aren't missed
F1-Score	Harmonic mean of Precision & Recall	Best metric for imbalanced datasets
ROC-AUC Score	Measures model's ability to distinguish between classes	Higher score means better classification ability

Model Performance

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	79.5%	72.3%	64.8%	68.3%
Random Forest	**87.1%**	**82.5%**	**76.9%**	**79.6%**
XGBoost	85.4%	80.3%	74.1%	77.1%

- **Random Forest performed best overall**, achieving high precision and recall.
- **Logistic Regression performed poorly on imbalanced data**.
- **XGBoost had strong performance** but was slightly more complex to interpret.

6. Interpretation of Results

- Employees with higher training scores and previous ratings were more likely to be promoted.
- Longer service lengths correlated positively with promotions.
- Recruitment channels had a strong influence, indicating possible biases in hiring processes.

7. Deployment Strategy

1. Deploying as an API

The model can be deployed using **FastAPI**, which are Python frameworks for building APIs. The API will serve as an interface between the trained model and external applications (e.g., an HR system).

- **FastAPI**: Faster and supports asynchronous requests, making it ideal for high-performance applications.

2. API Endpoints and Functionality

The API will expose endpoints that allow external applications to interact with the model. Some key endpoints include:

(a) Prediction Endpoint

- **URL:** `/predict`
- **Method:** `POST`
- **Input:** Employee details (e.g., department, education, age, training score)
- **Output:** Probability of promotion (`0` = Not Promoted, `1` = Promoted)

Github repo: https://github.com/abigiyaelias180/machine-learning.git

8. Limitations & Future Improvements

Current Limitations

- Bias in historical promotions could influence the model.
- Over-reliance on past performance ratings may limit new hires' promotion chances.
- Data is from one organization, limiting generalizability.

Future Enhancements

- **Use deep learning models** to capture more complex patterns.
- **Fairness Audits** to ensure no unintended biases.
- **Feature Engineering** to add derived features (e.g., employee growth rate).

This documentation ensures transparency, reproducibility, and a comprehensive understanding of the employee promotion prediction pipeline.