**Explanation of the Code:**

print("Name: Lokesh Dhoble")

print("Roll no.: TACO22131")

* This just prints out the name and roll number.

**Importing Required Libraries:**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import confusion\_matrix

import matplotlib.pyplot as plt

import seaborn as sns

* **pandas**: Used for data manipulation and analysis (loading and processing the dataset).
* **sklearn.model\_selection.train\_test\_split**: This function is used to split the dataset into training and testing sets.
* **sklearn.preprocessing.LabelEncoder**: Encodes categorical labels into numerical format.
* **sklearn.linear\_model.LogisticRegression**: For applying Logistic Regression to the dataset.
* **sklearn.metrics.confusion\_matrix**: Evaluates the accuracy of the classification model.
* **matplotlib.pyplot** and **seaborn**: For creating visualizations like boxplots and other graphs.

**Loading the Dataset:**

df = pd.read\_csv(r"C:\Users\Loki\Desktop\lp 2\end term\Extended\_Employee\_Performance\_and\_Productivity\_Data.csv")

df

* Loads the dataset from the specified path and stores it in the df DataFrame.
* Displays the contents of the DataFrame.

**Dropping Unnecessary Columns:**

df.drop(["Department", "Gender", "Job\_Title", "Hire\_Date", "Resigned"], axis=1, inplace=True)

df

* Drops irrelevant columns such as **Department**, **Gender**, **Job\_Title**, **Hire\_Date**, and **Resigned** from the DataFrame.

**Encoding Categorical Data:**

df["Education\_Level"].unique()

* Shows the unique values in the **Education\_Level** column.

le = LabelEncoder()

df['Education\_Level'] = le.fit\_transform(df['Education\_Level'])

df

* Uses **LabelEncoder** to convert categorical **Education\_Level** data into numerical values.

**Creating a New Column for High Performers:**

df['High\_Performer'] = (df['Performance\_Score'] >= 4).astype(int)

df

* Creates a new column **High\_Performer**: If **Performance\_Score** is greater than or equal to 4, it is marked as **1** (High Performer), otherwise **0** (Not High Performer).

**Boxplot Visualization:**

plt.figure(figsize=(8, 6))

sns.boxplot(x='High\_Performer', y='Monthly\_Salary', data=df, palette='Set2')

plt.title("Monthly Salary vs. High Performers")

plt.xlabel("High Performer (1 = Yes, 0 = No)")

plt.ylabel("Monthly Salary")

plt.show()

* Creates a **boxplot** to visualize the relationship between **High\_Performer** (1 or 0) and **Monthly\_Salary**.
* The boxplot shows the distribution of salaries for high performers vs non-high performers.

**Preparing Data for Training:**

X = df[['Performance\_Score', 'Education\_Level', 'Employee\_Satisfaction\_Score', 'Promotions']]

Y = df['High\_Performer']

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2, random\_state=42)

X\_test

* **X** contains the features (independent variables): **Performance\_Score**, **Education\_Level**, **Employee\_Satisfaction\_Score**, and **Promotions**.
* **Y** is the target variable: **High\_Performer**.
* Splits the data into training and test sets, with **80%** for training and **20%** for testing.

**Logistic Regression Model:**

lr = LogisticRegression()

lr.fit(X\_train, Y\_train)

LogisticRegression()

y\_pred = lr.predict(X\_test)

print("Confusion Matrix:\n", confusion\_matrix(Y\_test, y\_pred))

* Initializes the **Logistic Regression** model.
* **lr.fit(X\_train, Y\_train)** trains the model using the training data.
* **lr.predict(X\_test)** makes predictions on the test data.
* **confusion\_matrix** displays the confusion matrix to evaluate the model's performance.

**Accuracy Calculation:**

from sklearn.metrics import accuracy\_score

accuracy = accuracy\_score(Y\_test, y\_pred)

print("Accuracy: ", accuracy)

* Calculates and prints the **accuracy** of the model using **accuracy\_score**.

**Making a Prediction:**

prediction = lr.predict([[5, 1, 1.72, 2]])

print(prediction)

* Makes a prediction using the trained model for a new data point: **Performance\_Score = 5**, **Education\_Level = 1**, **Employee\_Satisfaction\_Score = 1.72**, **Promotions = 2**.
* Prints the predicted **High\_Performer** value.

**Questions & Answers:**

1. **Q: What is the purpose of train\_test\_split?**
   * **A**: It splits the dataset into training and testing sets, ensuring that the model can be trained on one part and evaluated on another.
2. **Q: Why are we using LabelEncoder for the "Education\_Level" column?**
   * **A**: LabelEncoder is used to convert the categorical data in the "Education\_Level" column into numerical values so that it can be processed by the machine learning model.
3. **Q: What does the boxplot visualize?**
   * **A**: The boxplot visualizes the distribution of monthly salaries between high performers and non-high performers.
4. **Q: What is the target variable in this model?**
   * **A**: The target variable is "High\_Performer," which indicates whether an employee is a high performer or not.
5. **Q: How does the Logistic Regression model work?**
   * **A**: Logistic Regression models the relationship between the independent variables and a binary target variable (high performer or not) and makes predictions based on learned patterns.
6. **Q: What does the confusion matrix show?**
   * **A**: The confusion matrix shows the number of true positives, true negatives, false positives, and false negatives, helping evaluate the model's accuracy.
7. **Q: How is accuracy calculated in this model?**
   * **A**: Accuracy is calculated by comparing the predicted labels with the actual labels from the test set and determining the percentage of correct predictions.
8. **Q: What does a prediction value of [0] or [1] represent?**
   * **A**: A prediction of [0] represents a non-high performer, while [1] represents a high performer.
9. **Q: Why do we drop certain columns like "Department" and "Gender"?**
   * **A**: These columns are dropped because they are not relevant to the prediction of high performance and may introduce noise or bias into the model.
10. **Q: What happens if you change the input values for a prediction?**
    * **A**: Changing the input values affects the prediction, and the model will output whether the employee is predicted to be a high performer or not based on the given features.