Mini Project for Internal II

HR Analytics to track employee Performance

1. Define problem statement.

The goal is to use HR data to track employee performance, identify key factors affecting productivity, and use data-driven insights to improve workforce management. The analysis will focus on understanding relationships between variables like training hours, job satisfaction, and performance scores.

2. Select suitable dataset.

```
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
  for filename in filenames:
    print(os.path.join(dirname, filename))
```

3. Implement Project using Python

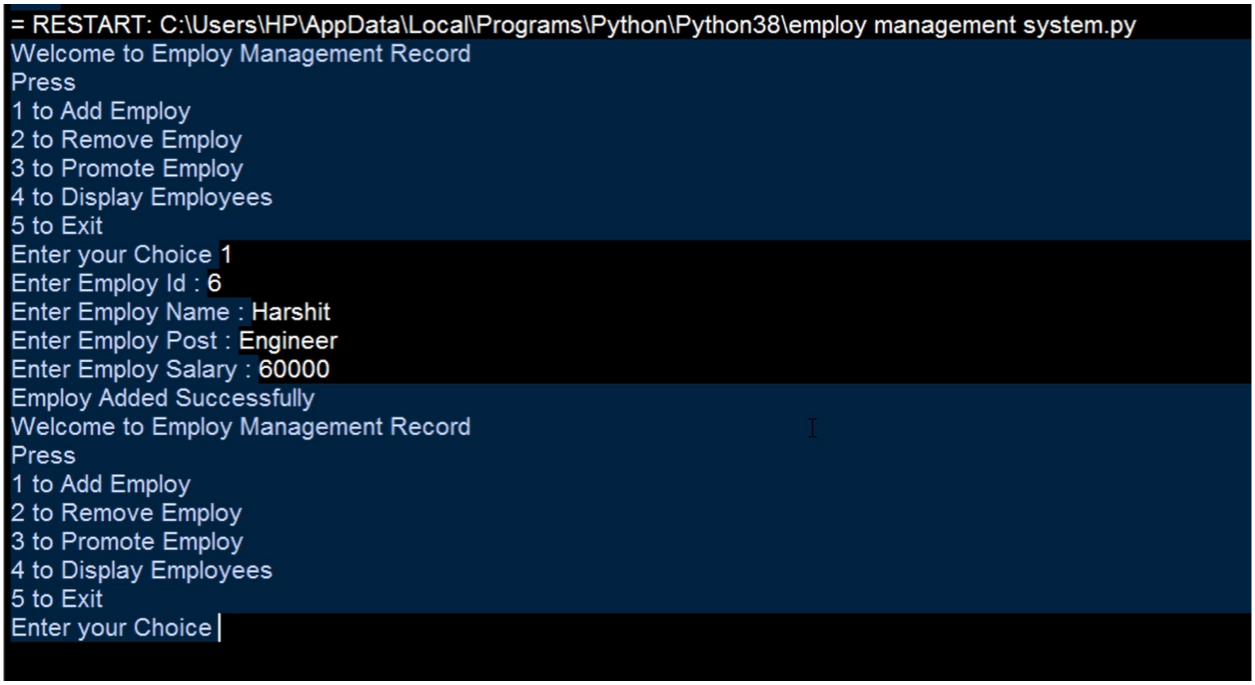
Code-

```
import mysql.connector

# Database connection
con = mysql.connector.connect(
     host="localhost",
     user="root",
     password="password",
     database="emp"
)

cursor = con.cursor()

# Function to check if an employee exists
def check_employee(employee_id):
     sql = 'SELECT * FROM employees WHERE id=%s'
     cursor.execute(sql, (employee_id,))
     return cursor.rowcount == 1

# Function to add an employee
def add_employee():
     Id = input("Enter Employee Id: ")
```

```
    if check_employee(Id):
        print("Employee already exists. Please try again.")
        return

    Name = input("Enter Employee Name: ")
    Post = input("Enter Employee Post: ")
    Salary = input("Enter Employee Salary: ")

    sql = 'INSERT INTO employees (id, name, position, salary) VALUES (%s, %s, %s, %s)'
    data = (Id, Name, Post, Salary)
    try:
        cursor.execute(sql, data)
        con.commit()
        print("Employee Added Successfully")
    except mysql.connector.Error as err:
        print(f"Error: {err}")
        con.rollback()

# Function to remove an employee
def remove_employee():
    Id = input("Enter Employee Id: ")
    if not check_employee(Id):
        print("Employee does not exist. Please try again.")
        return

    sql = 'DELETE FROM employees WHERE id=%s'
    data = (Id,)
    try:
        cursor.execute(sql, data)
        con.commit()
        print("Employee Removed Successfully")
    except mysql.connector.Error as err:
        print(f"Error: {err}")
        con.rollback()

# Function to promote an employee
def promote_employee():
    Id = input("Enter Employee's Id: ")
    if not check_employee(Id):
        print("Employee does not exist. Please try again.")
        return

    try:
```

```python
        Amount = float(input("Enter increase in Salary: "))

        sql_select = 'SELECT salary FROM employees WHERE id=%s'
        cursor.execute(sql_select, (Id,))
        current_salary = cursor.fetchone()[0]
        new_salary = current_salary + Amount

        sql_update = 'UPDATE employees SET salary=%s WHERE id=%s'
        cursor.execute(sql_update, (new_salary, Id))
        con.commit()
        print("Employee Promoted Successfully")

    except (ValueError, mysql.connector.Error) as e:
        print(f"Error: {e}")
        con.rollback()

# Function to display all employees
def display_employees():
    try:
        sql = 'SELECT * FROM employees'
        cursor.execute(sql)
        employees = cursor.fetchall()
        for employee in employees:
            print("Employee Id : ", employee[0])
            print("Employee Name : ", employee[1])
            print("Employee Post : ", employee[2])
            print("Employee Salary : ", employee[3])
            print("------------------------------------")

    except mysql.connector.Error as err:
        print(f"Error: {err}")

# Function to display the menu
def menu():
    while True:
        print("\nWelcome to Employee Management Record")
        print("Press:")
        print("1 to Add Employee")
        print("2 to Remove Employee")
        print("3 to Promote Employee")
        print("4 to Display Employees")
        print("5 to Exit")

        ch = input("Enter your Choice: ")
        if ch == '1':
            add_employee()
        elif ch == '2':
            remove_employee()
        elif ch == '3':
            promote_employee()
        elif ch == '4':
            display_employees()
        elif ch == '5':
            print("Exiting the program. Goodbye!")
            break
        else:
            print("Invalid Choice! Please try again.")

if __name__ == "__main__":
    menu()
```

Output-



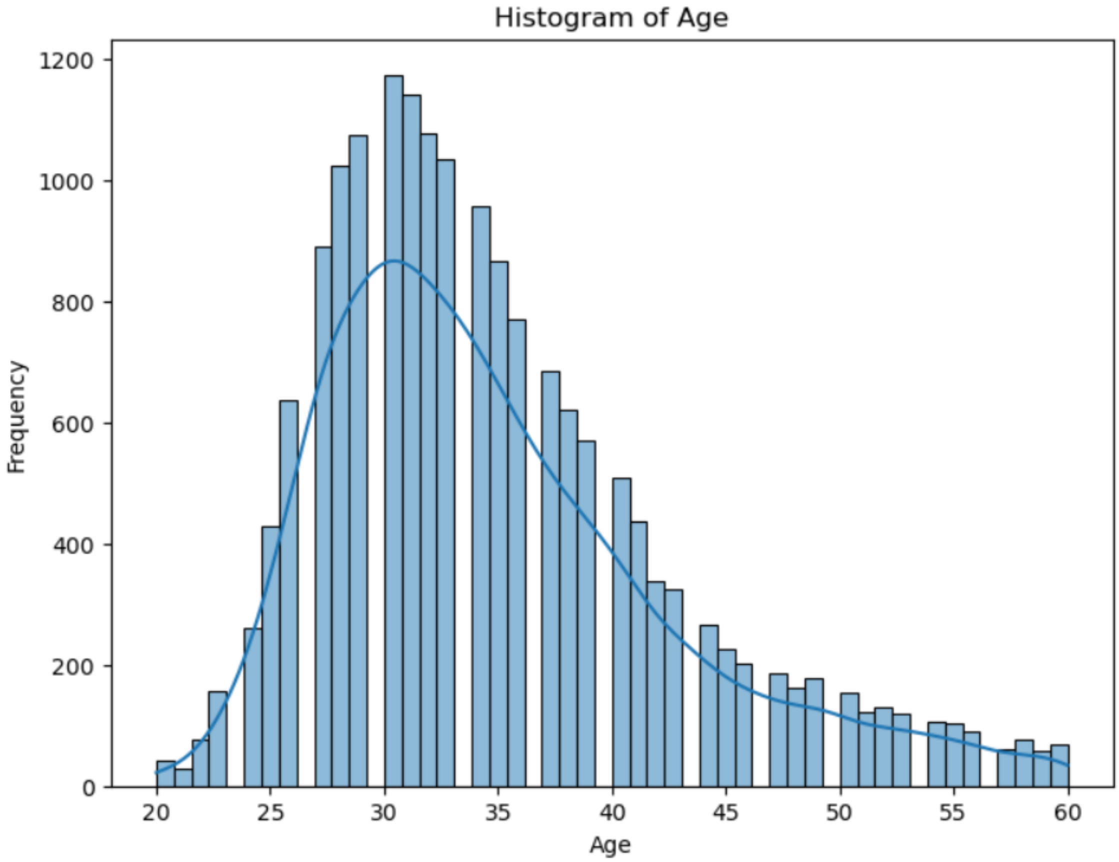
```
= RESTART: C:\Users\HP\AppData\Local\Programs\Python\Python38\employ management system.py
Welcome to Employ Management Record
Press
1 to Add Employ
2 to Remove Employ
3 to Promote Employ
4 to Display Employees
5 to Exit
Enter your Choice 1
Enter Employ Id : 6
Enter Employ Name : Harshit
Enter Employ Post : Engineer
Enter Employ Salary : 60000
Employ Added Successfully
Welcome to Employ Management Record
Press
1 to Add Employ
2 to Remove Employ
3 to Promote Employ
4 to Display Employees
5 to Exit
Enter your Choice
```

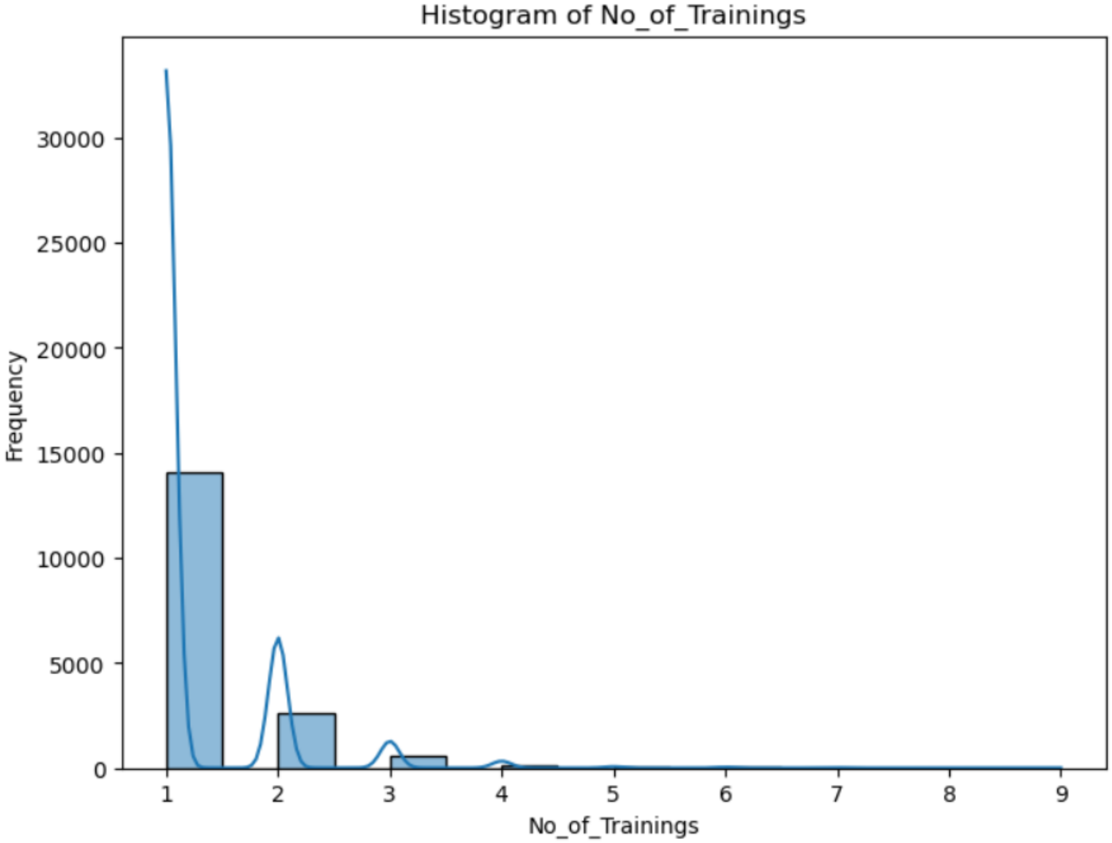
4. Visualize data with box plot, Histogram, Scatter Plot.

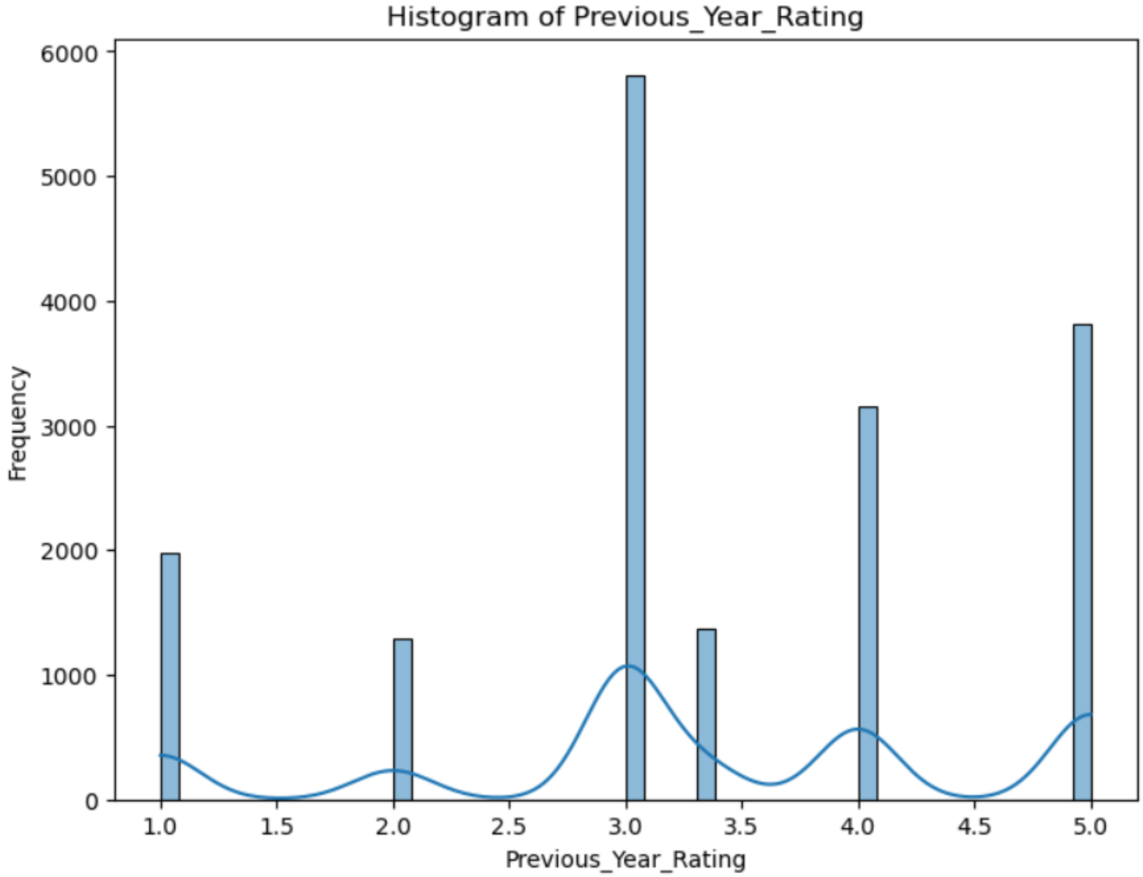
1. Plot Histograms for Numerical Variables

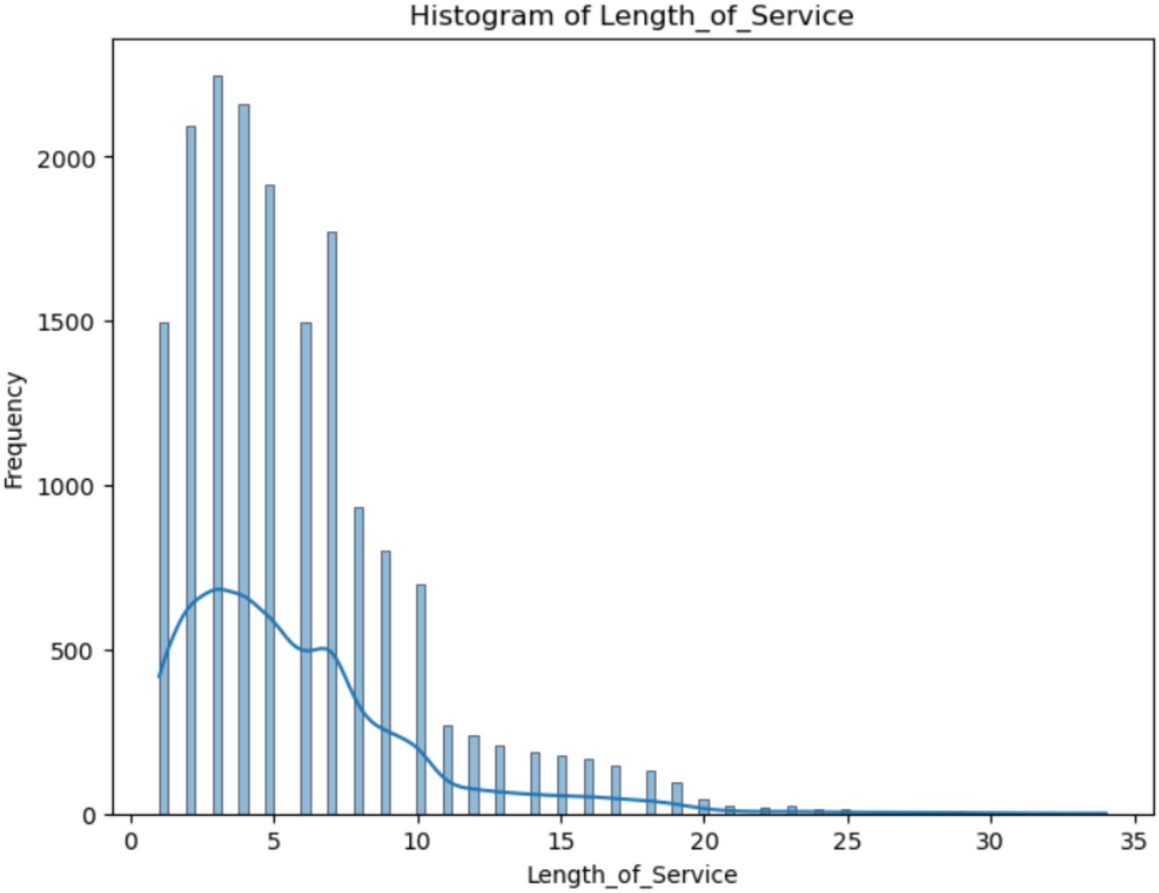
```
numerical_columns = ['Age', 'No_of_Trainings', 'Previous_Year_Rating', 'Length_of_Service', 'Avg_Training_Score']
```

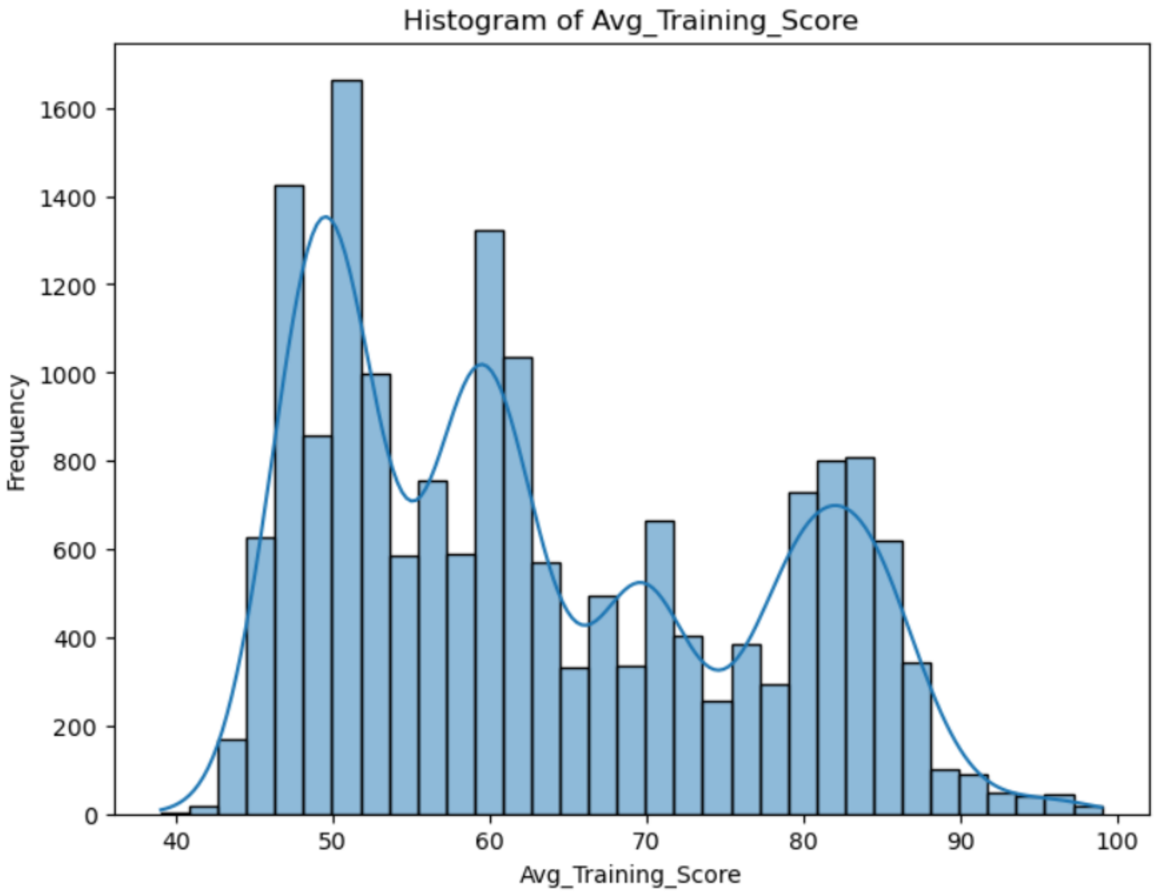
```
# Plot histograms for numerical variables
for column in numerical_columns:
    plt.figure(figsize=(8, 6))
    sns.histplot(hr[column], kde=True)
    plt.title(f'Histogram of {column}')
    plt.xlabel(column)
    plt.ylabel('Frequency')
    plt.show()
```




Histogram of No_of_Trainings
Frequency
30000
25000
20000
15000
10000
5000
0
1
2
3
4
5
6
7
8
9
No_of_Trainings

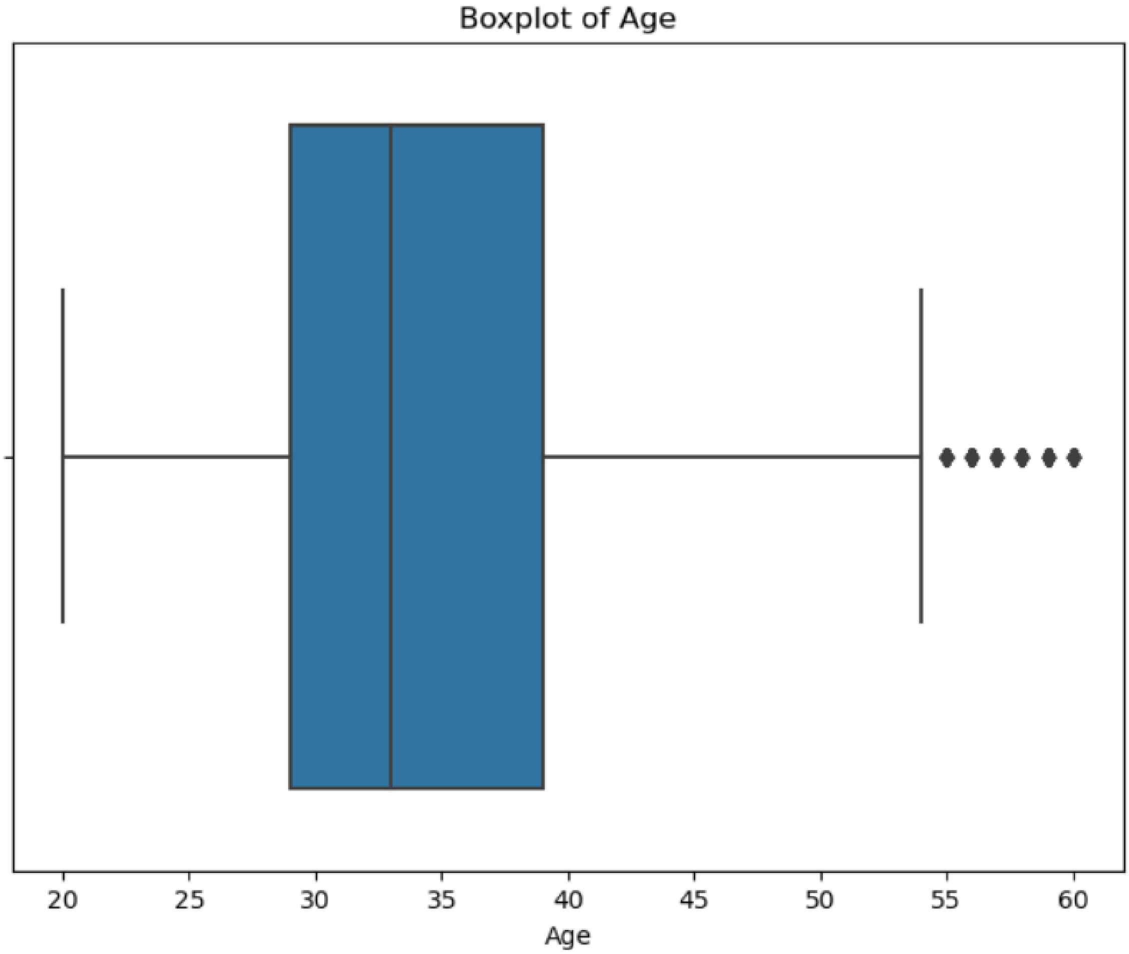

Histogram of Previous_Year_Rating
Frequency
0
1000
2000
3000
4000
5000
6000
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Previous_Year_Rating

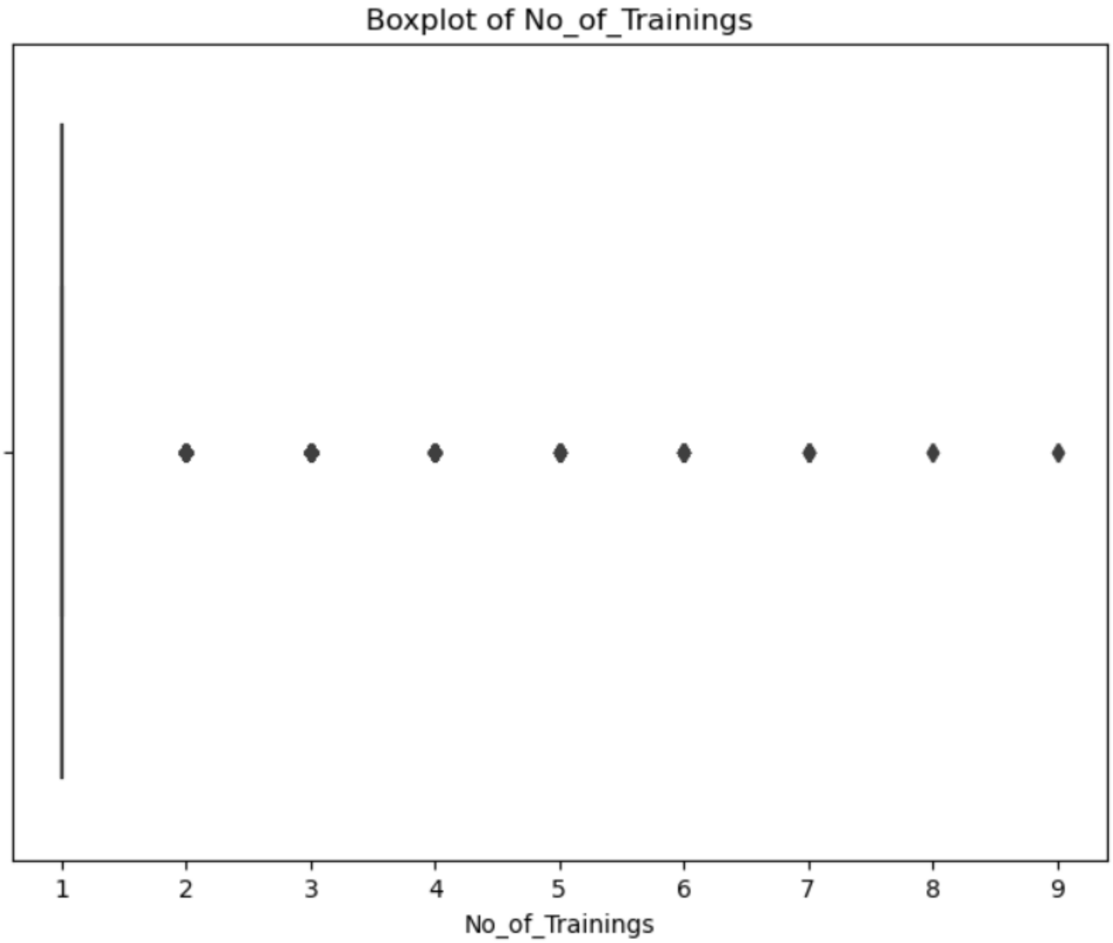

Histogram of Length_of_Service
Frequency
0
500
1000
1500
2000
0
5
10
15
20
25
30
35
Length_of_Service

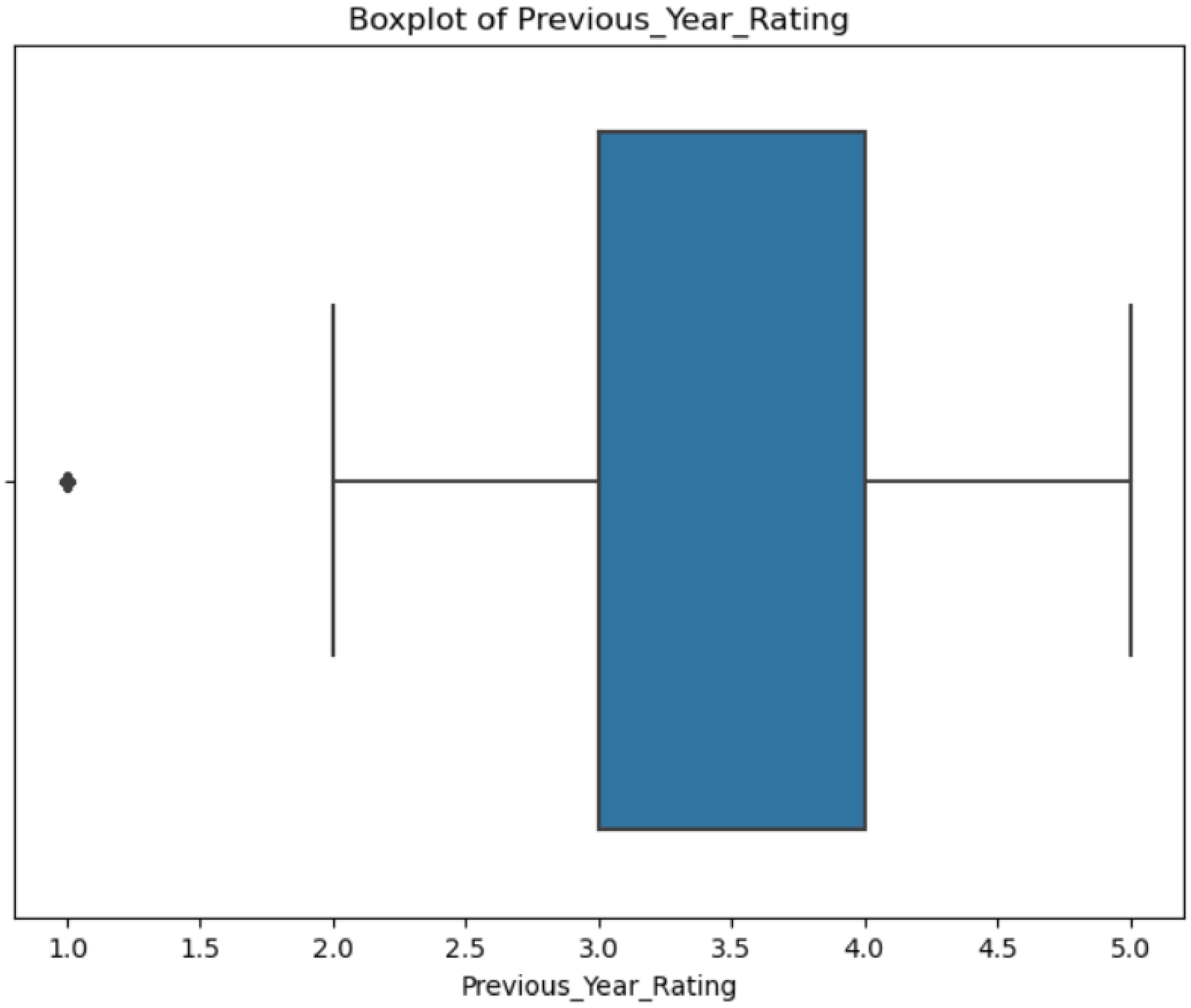


2. Plot boxplots for numerical variables

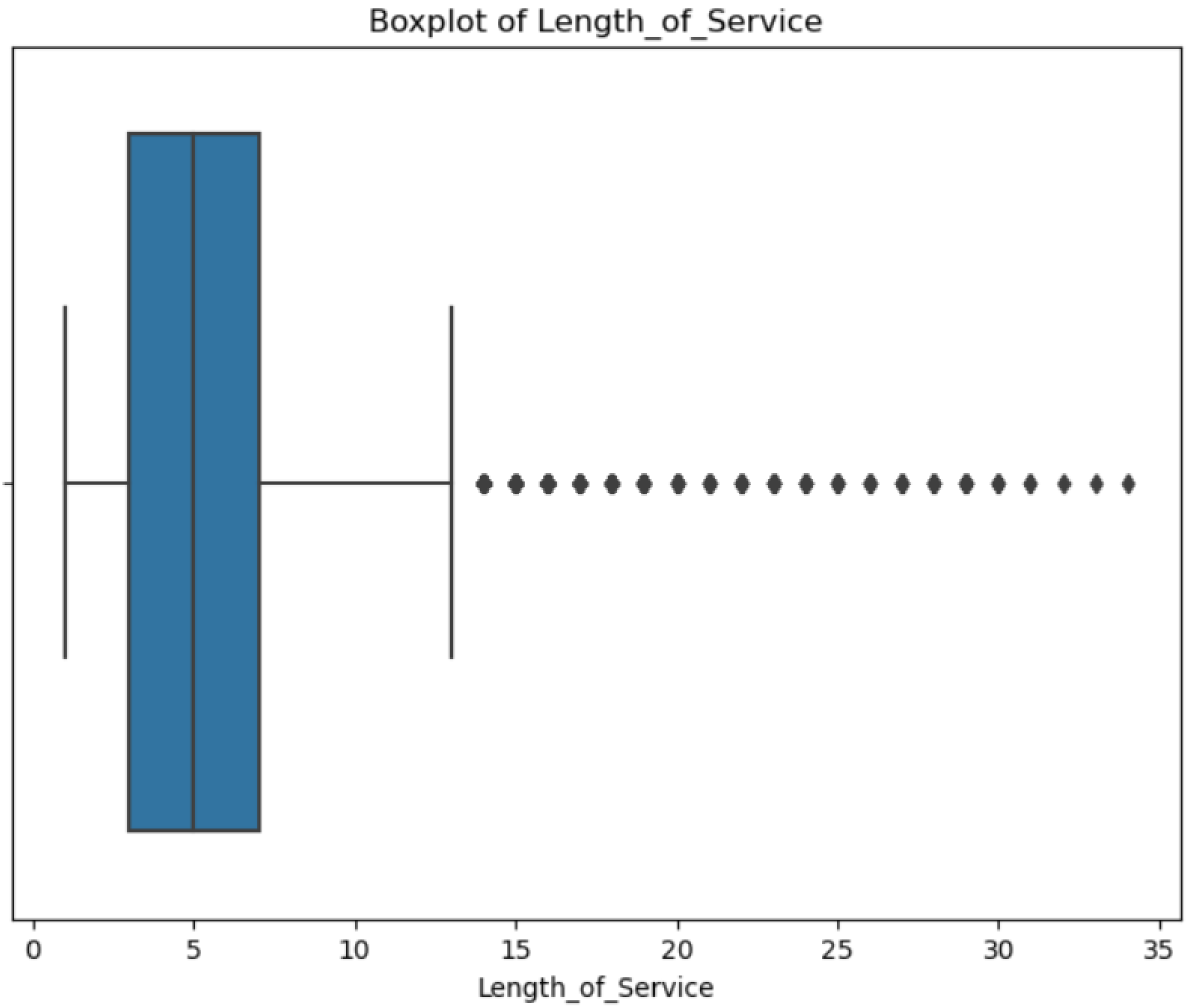
```
for column in numerical_columns:
    plt.figure(figsize=(8, 6))
    sns.boxplot(x=hr[column])
    plt.title(f'Boxplot of {column}')
    plt.xlabel(column)
    plt.show()
```

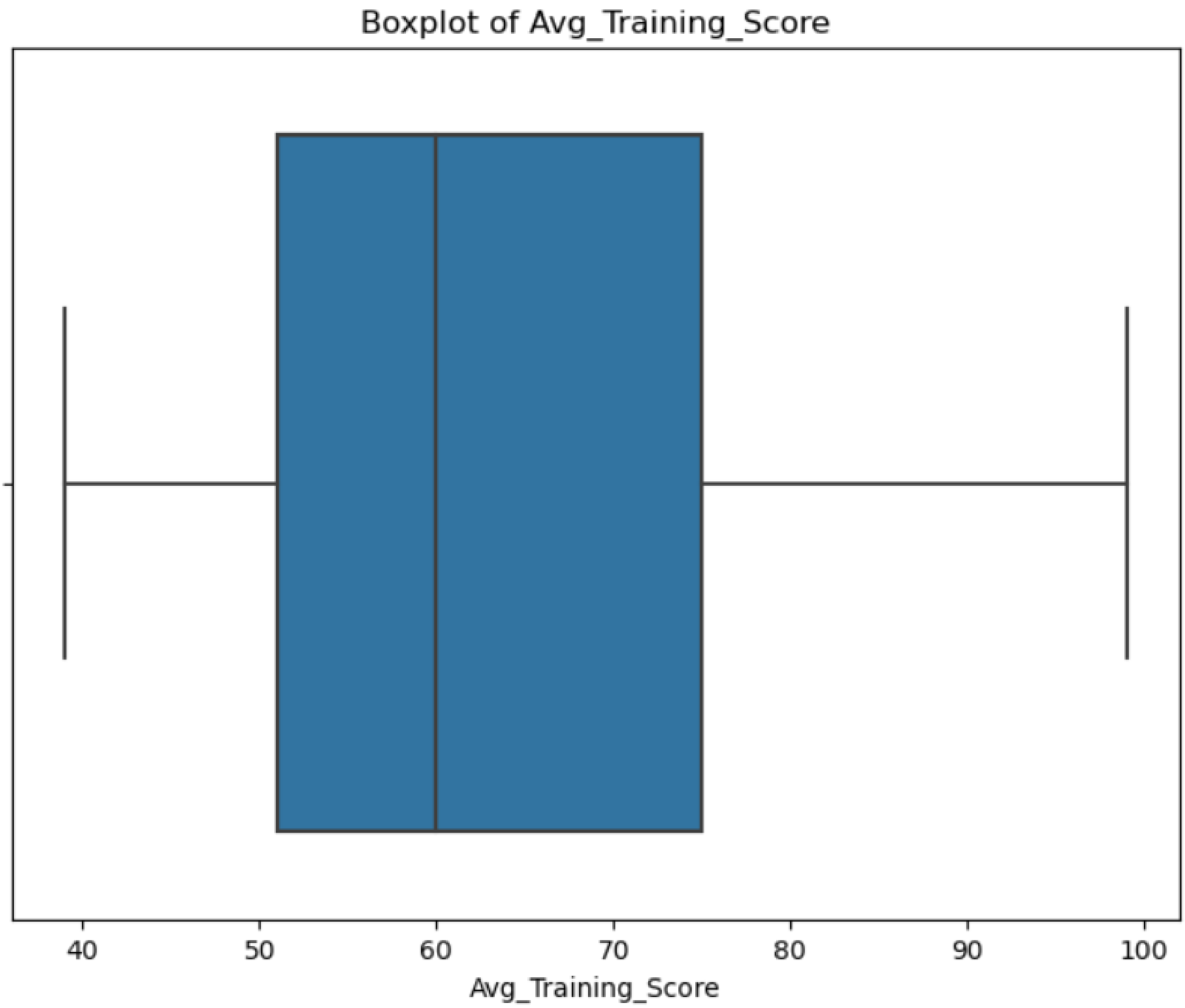

Boxplot of Age
20
25
30
35
40
45
50
55
60
Age


Boxplot of No_of_Trainings
1
2
3
4
5
6
7
8
9
No_of_Trainings



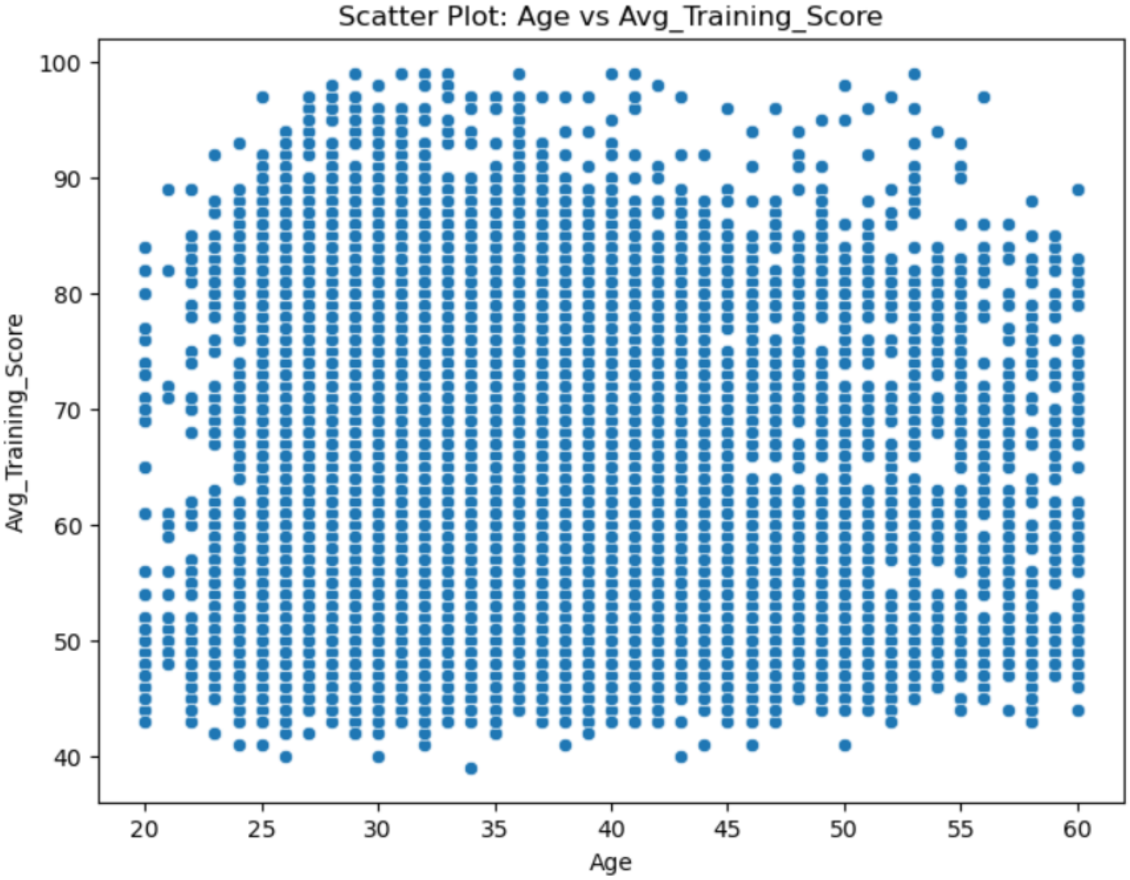
Boxplot of Previous_Year_Rating

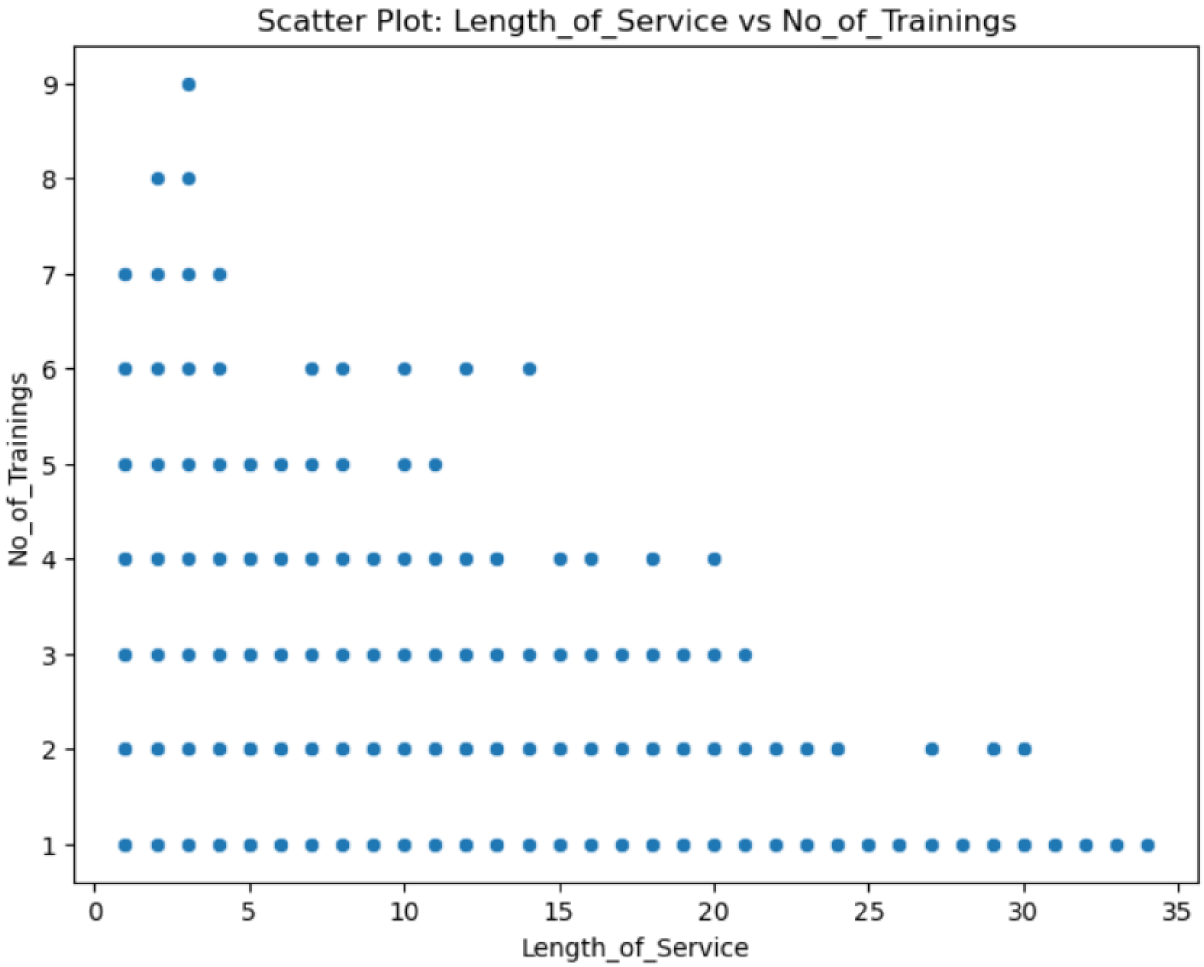

Boxplot of Length_of_Service
0
5
10
15
20
25
30
35
Length_of_Service



4. Create Scatter Plots or Line Plots for Interesting Relationships:

```
interesting_relationships = [('Age', 'Avg_Training_Score'),
('Length_of_Service', 'No_of_Trainings')]

# Create scatter plots
for x, y in interesting_relationships:
    plt.figure(figsize=(8, 6))
    sns.scatterplot(x=x, y=y, data=hr)
    plt.title(f'Scatter Plot: {x} vs {y}')
    plt.xlabel(x)
    plt.ylabel(y)
    plt.show()
```

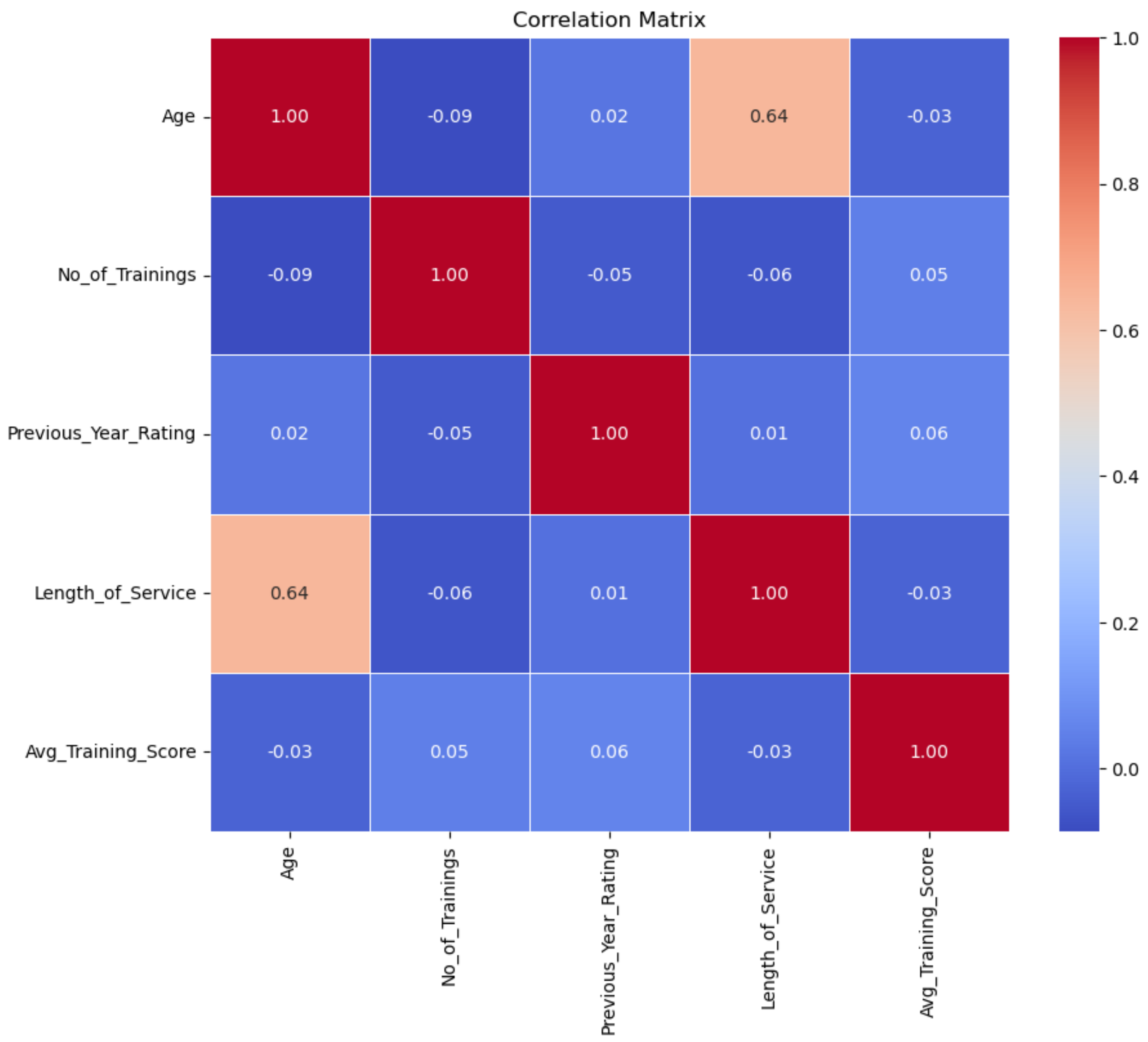

Scatter Plot: Age vs Avg_Training_Score
Avg_Training_Score
100
90
80
70
60
50
40
20
25
30
35
40
45
50
55
60
Age



5. Plot Pearson correlation and explain about relation.

Correlation analysis and visualize the correlation matrix using a heatmap

```
correlation_matrix = hr[numerical_columns].corr()

# Create a heatmap to visualize the correlation matrix
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f',
linewidths=0.5)
plt.title('Correlation Matrix')
plt.show()
```



6. Identify Dependent and Independent features.

Dependent Variable:

- PerformanceScore: This is the target variable we are trying to predict. It measures how well an employee is performing.

Independent Variables:

1. TrainingHours: The number of hours an employee has spent on training. Likely impacts performance as more training can improve skills.

2. JobSatisfaction: The satisfaction level of an employee with their job. Affects performance, as higher satisfaction often leads to better performance.
3. YearsAtCompany: The number of years an employee has worked at the company. Longer tenure may correlate with higher performance due to experience.
4. JobRole: The specific role or position of the employee. Different roles might have different performance expectations.
5. Age: Employee's age could influence work performance due to experience or generational differences in working styles.
6. Department: The department in which the employee works, as some departments may have different performance standards or challenges.
7. YearsSinceLastPromotion: Time since the last promotion, which can impact motivation and performance.

Each independent feature is used to predict how well an employee performs, with `PerformanceScore` being the outcome of interest.

7. Analyse /Predict as per problem statement.

To analyze and predict employee performance (`PerformanceScore`), follow these steps:

1. Data Preprocessing: Clean the data, handle missing values, and encode categorical variables like `JobRole` and `Department`. Normalize numerical features if necessary (e.g., `Age`, `YearsAtCompany`).
2. Exploratory Data Analysis (EDA): Visualize relationships between independent variables (e.g., `TrainingHours`, `JobSatisfaction`) and `PerformanceScore` using scatter plots, histograms, and box plots.
3. Correlation Analysis: Identify features with strong correlations with `PerformanceScore`, such as `TrainingHours` or `JobSatisfaction`.
4. Feature Selection: Choose relevant features based on correlation and domain knowledge. For example, `TrainingHours`, `JobSatisfaction`, `YearsAtCompany`, etc., can be key predictors.
5. Model Building: Train a machine learning model (e.g., Random Forest, Linear Regression) using the independent variables to predict `PerformanceScore`. Split the data into training and testing sets.
6. Model Evaluation: Evaluate the model using metrics like R^2 and Mean Squared Error (MSE) to assess prediction accuracy. High R^2 and low MSE indicate good predictive performance.
7. Insights: Based on the model's results, gain insights into factors affecting performance. For example, if `TrainingHours` and `JobSatisfaction` have high

feature importance, you can recommend more training programs or focus on improving job satisfaction to boost performance.

8. Final Predictions: Use the trained model to predict the performance of employees in the test set or on new data, providing actionable insights for HR decision-making.

This process helps HR teams understand how employee characteristics influence performance and how to take data-driven actions to improve workforce outcomes.

Name – Prachi Shelake Roll No – BTCS-351