```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

# Load dataset
df = pd.read_csv('/content/Employee_Salary_Dataset.csv')

# Display first few rows to understand the dataset structure
print("First few rows of the dataset:")
display(df.head())

# Check for missing values in each column
print("\nChecking for missing values in dataset:")
print(df.isnull().sum())

# Remove rows with missing values to avoid errors in model training
df.dropna(inplace=True)

# Remove the 'ID' column since it does not contribute to salary prediction
if 'ID' in df.columns:
    df.drop(columns=['ID'], inplace=True)
    print("\n'ID' column removed as it is not needed for prediction.")

# Convert 'Gender' into numerical format (Male = 0, Female = 1) for machine learning processing
df['Gender'] = df['Gender'].map({'Male': 0, 'Female': 1})
print("\nGender column encoded (Male = 0, Female = 1)")

# Define the independent variables (features) and the dependent variable (target)
X = df[['Experience_Years', 'Age', 'Gender']]  # Features used for prediction
y = df['Salary']  # Target variable (Salary)

# Splitting dataset into 80% training data and 20% testing data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
print("\nDataset split into training (80%) and testing (20%) sets.")

# Initialize the linear regression model
model = LinearRegression()
print("\nTraining the Linear Regression model...")
model.fit(X_train, y_train)
print("Model training completed.")

# Predict salaries on the test data
y_pred = model.predict(X_test)

# Evaluate the model performance
print("\nModel Evaluation:")
print("- Mean Absolute Error (MAE):", round(mean_absolute_error(y_test, y_pred), 2))  # Measures average absolute error in predictions
print("- Mean Squared Error (MSE):", round(mean_squared_error(y_test, y_pred), 2))  # Measures average squared difference between actual
print("- R² Score (Performance Indicator):", round(r2_score(y_test, y_pred), 4))  # Indicates how well the model explains the variation i

# User input for salary prediction
print("\nEnter details to predict salary:")
experience = float(input("- Years of Experience: "))  # Take experience as input
age = float(input("- Age: "))  # Take age as input
gender = input("- Gender (Male/Female): ").strip()  # Take gender as input
gender = 0 if gender.lower() == 'male' else 1  # Convert gender into numerical format

# Convert user input into a dataframe with proper column names to match training data
user_data = pd.DataFrame([[experience, age, gender]], columns=X.columns)

# Predict salary based on user input
your_salary = model.predict(user_data)
print("\nPredicted Salary: ₹", format(round(your_salary[0], 2), ','))  # Display predicted salary with formatting
```