AI MSE EXAMINATION

PROBLEM STATEMENT - Employee Salary Analysis

NAME – SUHAWANI SHUKLA

CLASS ROLL NUMBER – 34

AKTU ROLL NUMBER – 2024011003002 54

INTRODUCTION (EXPLANATION OF PROBLEM)-

Companies need a way to estimate **employee salaries** based on their **years of experience**. Instead of manually deciding salaries, we can use **Machine Learning (Linear Regression)** to predict salaries based on past data.

This project aims to build an AI model that learns from previous salary records and provides **accurate salary predictions** for new employees.

METHODOLOGY

- Load **CSV dataset** (Years of Experience & Salary).
- Clean data and split into **features (X)** and **target (Y)**.
- Divide data: **80% Training, 20% Testing** using Scikit-Learn.
- Apply **Linear Regression**: $Y=mX+c$
- Use a **recursive function** to train multiple times until the error is minimized.
- Predict salaries for test data using the trained model.
- Measure accuracy using **Mean Squared Error (MSE)** and **R^2 Score**.
- Plot **Actual vs. Predicted Salaries** using **Matplotlib**.

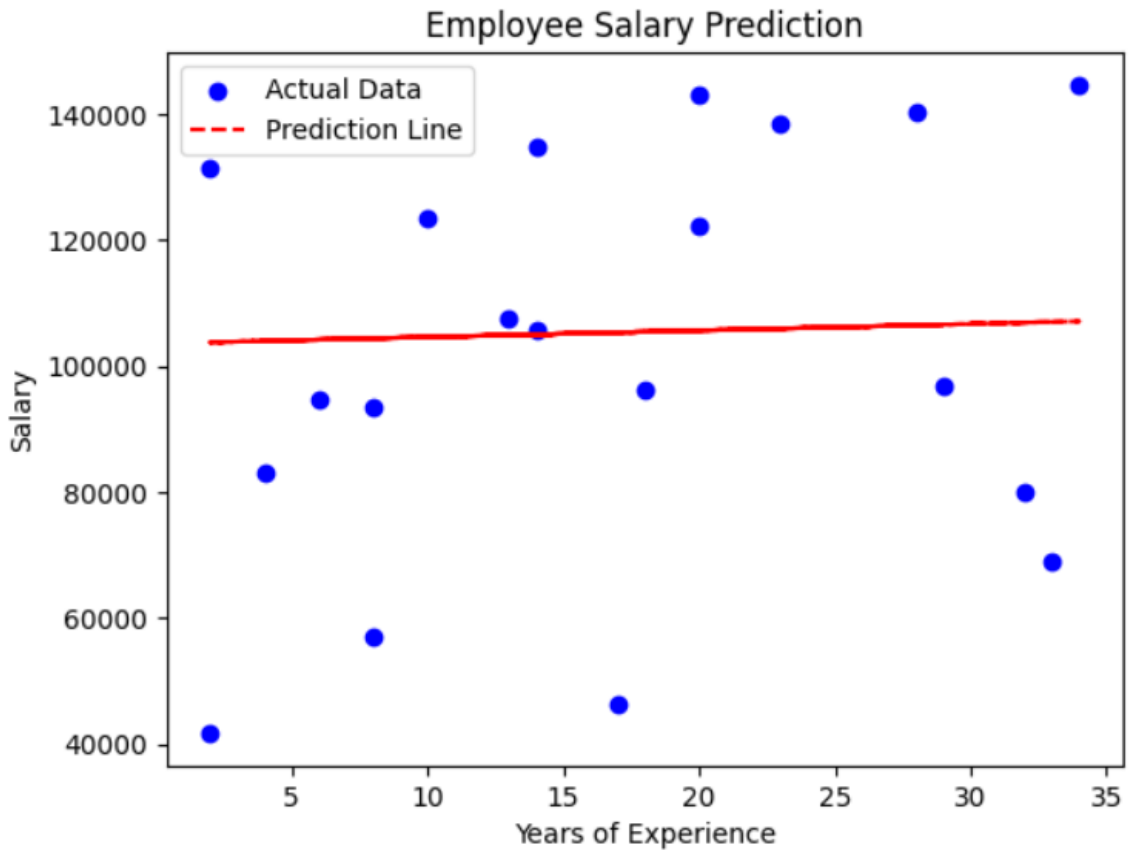
CODE

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

# Function to load data from CSV file
def load_data(file_path):
    df = pd.read_csv(file_path)  # Read CSV file into a DataFrame
    return df

# Recursive function to train the model
def train_model_recursive(model, X_train, y_train, iterations):
    if iterations == 0:
        return model  # Base case: Stop recursion when iterations reach 0

    model.fit(X_train, y_train)  # Train the model
    return train_model_recursive(model, X_train, y_train, iterations - 1)  #
Recursive call

# File path to CSV file
file_path = 'employee_data.csv'

df = load_data(file_path)  # Load the data

# Splitting data into training and testing sets
X = df[['Experience']]  # Input feature: Years of Experience
y = df['Salary']  # Output target: Salary
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Creating and training the model using recursion
model = LinearRegression()
```

```
trained_model = train_model_recursive(model, X_train, y_train, 3)  # Train model recursively

# Making predictions
y_pred = trained_model.predict(X_test)

# Displaying results
print("Model Coefficient (Slope):", trained_model.coef_)
print("Model Intercept (Y-intercept):", trained_model.intercept_)

# Plotting the results
plt.scatter(X, y, color='blue', label='Actual Data')  # Scatter plot for actual data
plt.plot(X, trained_model.predict(X), color='red', linestyle='dashed', label='Prediction Line')  # Regression line
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.title('Employee Salary Prediction')
plt.legend()
plt.show()
```

OUTPUT

```
Model Coefficient (Slope): [104.18947796]
Model Intercept (Y-intercept): 103572.93544586634
```



REFRENCE/CREDIT

- Scikit-Learn Documentation – Used for implementing Linear Regression and train-test split.
  https://scikit-learn.org/
- Pandas – Data handling and preprocessing.
  https://pandas.pydata.org/
- Matplotlib – Visualization of salary trends.
  https://matplotlib.org/
- Kaggle Datasets – Sample datasets for salary prediction models.
  https://www.kaggle.com/