```
import pandas as pd
import numpy as np
```

```
df=pd.read_csv('salary_data.csv')
df.head()
```

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29
Data columns (total 2 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   YearsExperience  30 non-null     float64
 1   Salary           30 non-null     float64
dtypes: float64(2)
memory usage: 612.0 bytes
```

```
X = df['YearsExperience'].values.reshape(-1, 1)
y = df['Salary'].values
```

```
X_mean, X_std = np.mean(X), np.std(X)
y_mean, y_std = np.mean(y), np.std(y)
X_scaled = (X - X_mean) / X_std
y_scaled = (y - y_mean) / y_std
```

```
def hypothesis(X, theta0, theta1):
    return theta0 + theta1 * X
```

```
def cost_function(X, y, theta0, theta1):
    m = len(y)
    return (1 / (2 * m)) * np.sum((hypothesis(X, theta0, theta1) - y) ** 2)
```

```
import matplotlib.pyplot as plt

def gradient_descent(X, y, lr, epochs):
    theta0, theta1 = 0, 0
    m = len(y)
    cost_history = []
    for epoch in range(epochs):
        predictions = hypothesis(X, theta0, theta1)
        error = predictions - y
        theta0 -= lr * (1/m) * np.sum(error)
        theta1 -= lr * (1/m) * np.sum(error * X)
        cost = cost_function(X, y, theta0, theta1)
        cost_history.append(cost)
        plt.figure(figsize=(6,4))
        plt.scatter(X, y, color='blue')
        plt.plot(X, hypothesis(X, theta0, theta1), color='red')
        plt.title(f'Epoch {epoch+1}')
        plt.xlabel('YearsExperience (scaled)')
        plt.ylabel('Salary (scaled)')
        plt.show()
    return theta0, theta1, cost_history
```

```
learning_rates = [0.001, 0.01, 0.05, 0.1]
epochs = 10
results = {}

for lr in learning_rates:
    theta0, theta1, cost_hist = gradient_descent(X_scaled, y_scaled, lr, epochs)
    results[lr] = cost_hist[-1]
    print(f'Learning Rate: {lr}, Final Cost: {cost_hist[-1]:.6f}')
```

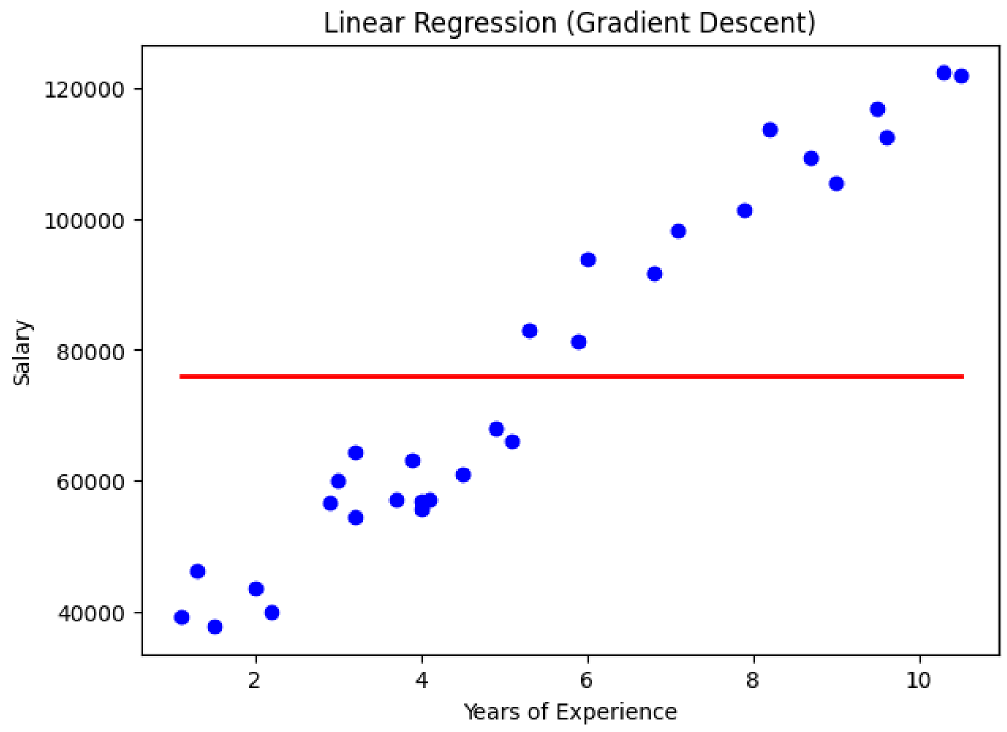
```
lr_best = 0.05
epochs_best = 10
theta0, theta1, cost_hist = gradient_descent(X_scaled, y_scaled, lr_best, epochs_best)
```

```
y_pred_scaled = hypothesis(X_scaled, theta0, theta1)
y_pred = y_pred_scaled * y_std + y_mean
ss_total = np.sum((y - np.mean(y)) ** 2)
```

```
ss_res = np.sum((y - y_pred) ** 2)
r2_score = 1 - (ss_res / ss_total)
r2_score
```

```
np.float64(-29.0)
```

```
plt.figure(figsize=(7,5))
plt.scatter(df['YearsExperience'], df['Salary'], color='blue')
plt.plot(df['YearsExperience'], y_pred, color='red', linewidth=2)
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.title('Linear Regression (Gradient Descent)')
plt.show()
```


Linear Regression (Gradient Descent)
Salary
Years of Experience