EXPERIMENT 8B

ADITHIYA.V

241501010

A PYTHON PROGRAM TO IMPLEMENT GRADIENT BOOSTING

AIM:

TO IMPLEMENT A PYTHON PROGRAM TO IMPLEMENT GRADIENT BOOSTING

CODE:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
np.random.seed(42)
X = np.random.rand(100, 1) - 0.5
y = 3*X[:, 0]**2 + 0.05 * np.random.randn(100)
df = pd.DataFrame()
df['X'] = X.reshape(100)
df['y'] = y
df
```

```
plt.scatter(df['X'],df['y'])
plt.title('X vs y')
```

```
df['pred1'] = df['y'].mean()
df
```

```
df['res1'] = df['y'] - df['pred1']
df

plt.scatter(df['X'],df['y'])
plt.plot(df['X'],df['pred1'],color='red')

from sklearn.tree import DecisionTreeRegressor
tree1 = DecisionTreeRegressor(max_leaf_nodes=8)
tree1.fit(df['X'].values.reshape(100,1),df['res1'].values)
DecisionTreeRegressor(max_leaf_nodes=8)
from sklearn.tree import plot_tree
plot_tree(tree1)
plt.show()


X_test=np.linspace(-0.5, 0.5, 500)
y_pred=0.265458 + tree1.predict(X_test.reshape(500, 1))
plt.figure(figsize=(14,4))
plt.subplot(121)
plt.plot(X_test, y_pred, linewidth=2, color='red')
plt.scatter(df['X'], df['y'])


df['pred2'] = 0.265458 + tree1.predict(df['X'].values.reshape(100,1))
df



df['res2'] = df['y'] - df['pred2']
df



from sklearn.tree import DecisionTreeRegressor
tree2 = DecisionTreeRegressor(max_leaf_nodes=8)
tree2.fit(df['X'].values.reshape(100,1),df['res2'].values)
y_pred = df['pred1'].iloc[0] + tree1.predict(X_test.reshape(-1,1)) +
```

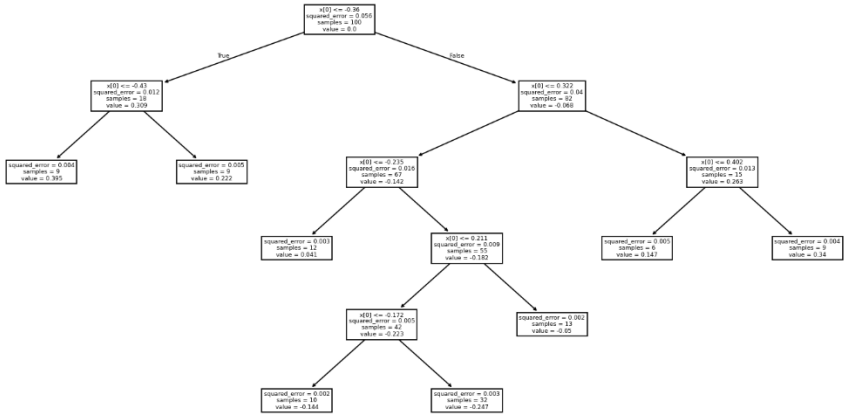
```
tree2.predict(X_test.reshape(-1,1))
plt.figure(figsize=(14,4))
plt.subplot(121)
plt.plot(X_test, y_pred, linewidth=2, color='red')
plt.scatter(df['X'], df['y'])
plt.title('X vs y')



def gradient_boost(X,y,number,lr,count=1,regs=[],foo=None):
  if number == 0:
    return
  else:
    # do gradient boosting
    if count > 1:
      y = y - regs[-1].predict(X)
    else:
      foo = y
    tree_reg = DecisionTreeRegressor(max_depth=5, random_state=42)
    tree_reg.fit(X, y)
    regs.append(tree_reg)
    x1 = np.linspace(-0.5, 0.5, 500)
    y_pred = sum(lr * regressor.predict(x1.reshape(-1, 1)) for regressor in
regs)
    print(number)
    plt.figure()
    plt.plot(x1, y_pred, linewidth=2)
    plt.plot(X[:, 0], foo,"r")
    plt.show()
    gradient_boost(X,y,number-1,lr,count+1,regs,foo=foo)
np.random.seed(42)
X = np.random.rand(100, 1) - 0.5
y = 3*X[:, 0]**2 + 0.05 * np.random.randn(100)
gradient_boost(X,y,5,lr=1)
```

OUTPUT:

Figure 1



`

Figure 3

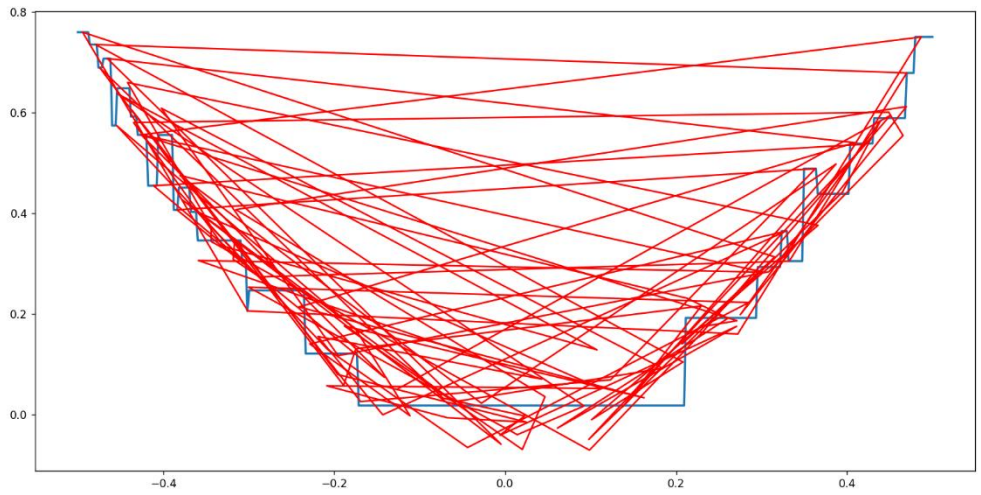
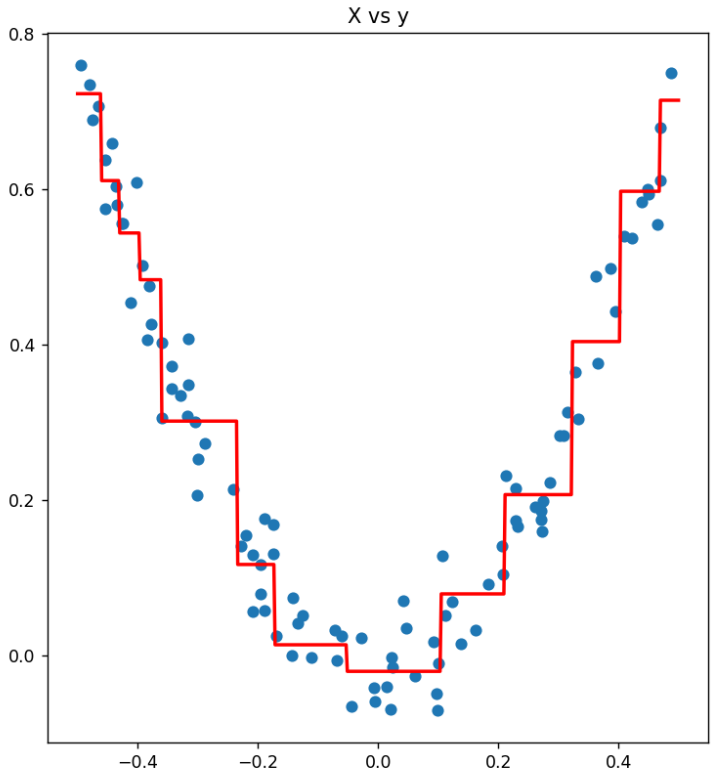
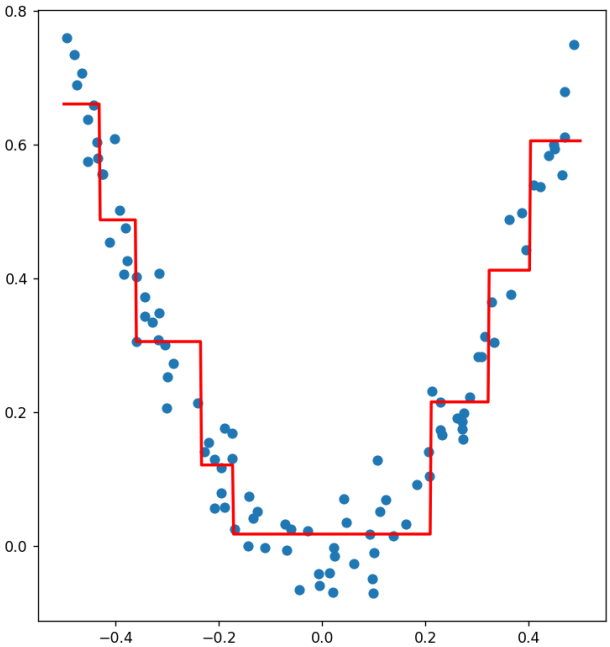
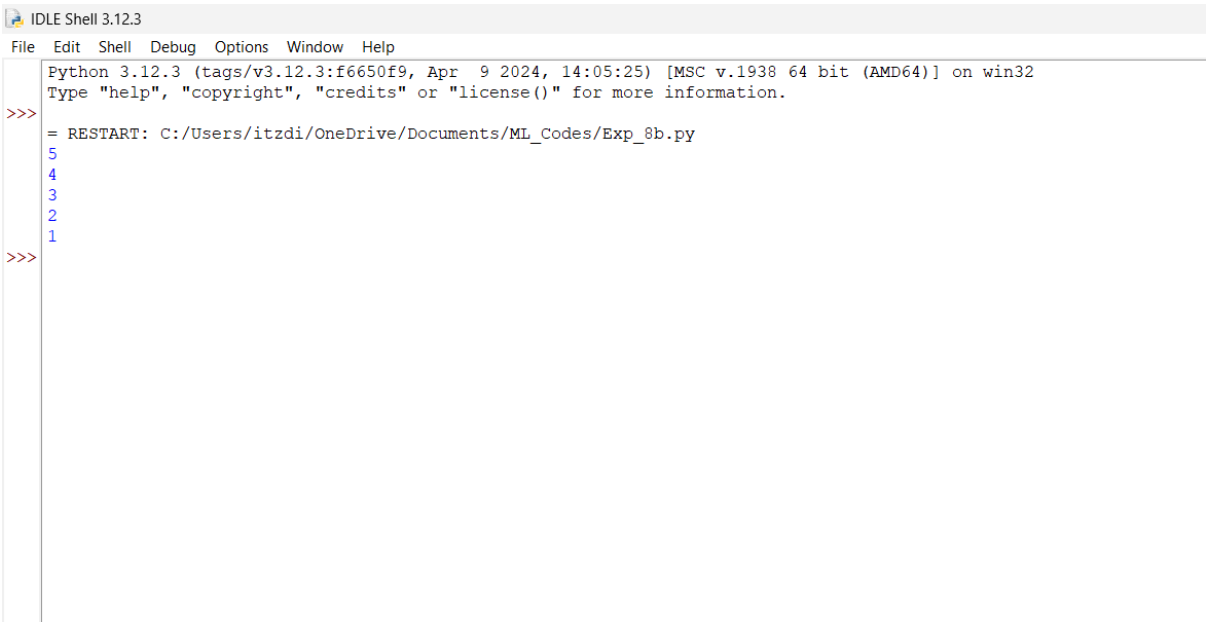


Figure 2







RESULT:

A PYTHON PROGRAM TO IM[PLEMENT GRADIENT BOOSTING AS BEEN ANALYSED AND VERIFEID