**Experiment 5 :**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# import warnings

import warnings

warnings.filterwarnings("ignore")

# We will use some methods from the sklearn module

from sklearn import linear\_model

from sklearn.linear\_model import LinearRegression

from sklearn import metrics

from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error

from sklearn.model\_selection import train\_test\_split, cross\_val\_score

df = pd.read\_csv("data.csv")

print(df.corr())

print(df.describe())

#Setting the value for X and Y

X = df[['Weight', 'Volume']]

y = df['CO2']

fig, axs = plt.subplots(2, figsize = (5,5))

plt1 = sns.boxplot(df['Weight'], ax = axs[0])

plt2 = sns.boxplot(df['Volume'], ax = axs[1])

plt.tight\_layout()

sns.distplot(df['CO2']);

sns.pairplot(df, x\_vars=['Weight', 'Volume'], y\_vars='CO2', height=4, aspect=1, kind='scatter')

plt.show()

# Create the correlation matrix and represent it as a heatmap.

sns.heatmap(df.corr(), annot = True, cmap = 'coolwarm')

plt.show()

X\_train,X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.3, random\_state = 100)

reg\_model = linear\_model.LinearRegression()

#Fitting the Multiple Linear Regression model

reg\_model = LinearRegression().fit(X\_train, y\_train)

#Printing the model coefficients

print('Intercept: ',reg\_model.intercept\_)

# pair the feature names with the coefficients

list(zip(X, reg\_model.coef\_))

#Predicting the Test and Train set result

y\_pred= reg\_model.predict(X\_test)

x\_pred= reg\_model.predict(X\_train)

print("Prediction for test set: {}".format(y\_pred))

#Actual value and the predicted value

reg\_model\_diff = pd.DataFrame({'Actual value': y\_test, 'Predicted value': y\_pred})

reg\_model\_diff

mae = metrics.mean\_absolute\_error(y\_test, y\_pred)

mse = metrics.mean\_squared\_error(y\_test, y\_pred)

r2 = np.sqrt(metrics.mean\_squared\_error(y\_test, y\_pred))

print('Mean Absolute Error:', mae)

print('Mean Square Error:', mse)

print('Root Mean Square Error:', r2)

**Program 2 :**

# Import necessary libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn import linear\_model

from sklearn.model\_selection import train\_test\_split

from sklearn import metrics

# Load the dataset

df = pd.read\_csv("data.csv")

# Prepare the features and target variable

X = df[['Weight', 'Volume']]

y = df['CO2']

# Perform exploratory data analysis

print(df.corr())

sns.pairplot(df, x\_vars=['Weight', 'Volume'], y\_vars='CO2', height=4, aspect=1, kind='scatter')

plt.show()

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# Create and fit the multivariable regression model

reg\_model = linear\_model.LinearRegression()

reg\_model.fit(X\_train, y\_train)

# Print model coefficients

print('Intercept:', reg\_model.intercept\_)

print('Coefficients:', reg\_model.coef\_)

# Make predictions

y\_pred = reg\_model.predict(X\_test)

# Evaluate the model

mae = metrics.mean\_absolute\_error(y\_test, y\_pred)

mse = metrics.mean\_squared\_error(y\_test, y\_pred)

r2 = metrics.r2\_score(y\_test, y\_pred)

print('Mean Absolute Error:', mae)

print('Mean Squared Error:', mse)

print('R-squared:', r2)

# Visualize actual vs predicted values

plt.scatter(y\_test, y\_pred)

plt.xlabel('Actual CO2')

plt.ylabel('Predicted CO2')

plt.title('Actual vs Predicted CO2')

plt.plot([y.min(), y.max()], [y.min(), y.max()], 'r--') # 45-degree line

plt.show()